Assignment Strategies for Spatial Reuse TDMA

Jimmi Grönkvist
Assignment Strategies for Spatial Reuse TDMA

Jimmi Grönkvist

Mars 2002

TRITA - S3 - RST – 0202
ISSN 1400-9137
ISRN KTH/RST/R--02/02--SE
Abstract

Spatial reuse TDMA has been proposed as an access scheme for multi-hop radio networks where real-time service guarantees are important. The idea is to increase capacity by letting several radio terminals use the same time slot when possible. A time slot can be shared when the radio units are geographically separated such that small interference is obtained. The transmission rights of the different users are described with a schedule. Unfortunately, due to the mobility of the users, this schedule must be constantly updated. In order to make this possible such updates must be made in parallel with only local information, i.e. distributed STDMA algorithms must be used.

In this report we investigate and list the properties a distributed STDMA algorithm must have to be efficient in a military scenario. Furthermore, we study the already existing algorithms and see what methods they use to fulfil the properties we seek. We can conclude that none of the existing algorithms can easily be used to create a sufficiently efficient distributed STDMA algorithm. Of the algorithms we have described USAP is the most interesting but not even this algorithm have the properties that are necessary for reliable and efficient communication on the battlefield.
Acknowledgments

I am grateful to many people for helping me carry this thesis to completion. First, I wish to thank my adviser Professor Jens Zander. His scientific guidance and support, especially in structuring my work, has been invaluable.

I also want to thank Eva Englund, my former project manager. Her encouragement and inspiring help in the beginning of my work really created the foundation of my research.

Many thanks also to Anders Hansson, who has always been there to listen to all my odd ideas. Many ideas have matured during our fruitful discussions.

Furthermore, I would like to thank Christian Jönsson, the Head of Department, and Jan Nilsson, my project manager, for giving me the opportunity to do this work.

Finally, I am grateful to all my colleagues at the Department of Communication Systems at the Swedish Defence Research Agency (FOI) for providing a pleasant work environment. In particular, I wish to thank the network research group.
Contents

1 Introduction 7
   1.1 Background ........................................ 7
   1.2 Previous Work ...................................... 12
   1.3 Contributions .................................... 13
   1.4 Thesis Outline ................................... 14

2 Network Model 17
   2.1 Data link layer .................................... 17
   2.2 Transport and Network Layer ..................... 19
   2.3 Performance Measures ........................... 23
   2.4 STDMA scheduling ............................... 25

3 Node and Link Assignment 29
   3.1 Link Assignment .................................. 29
   3.2 Node Assignment ................................. 31
   3.3 Analysis .......................................... 32
       3.3.1 Throughput for link assignment ............ 33
       3.3.2 Throughput in node assignment ............ 34
       3.3.3 An approximative formula for the network delay 35
   3.4 Evaluation and Results ......................... 37
       3.4.1 Unicast Traffic ............................ 37
       3.4.2 Broadcast Traffic ......................... 40

4 Link assignment with LET 45
   4.1 LET principle .................................... 45
   4.2 Basic Properties ................................. 49
4.3 Analysis ........................................ 52
4.4 Evaluation and Results ....................... 53
  4.4.1 Unicast Traffic ............................ 53
  4.4.2 Broadcast Traffic ......................... 55
  4.4.3 Conclusions .............................. 58

5 Algorithms designed for LET 61
  5.1 Double Priority .............................. 62
    5.1.1 Simulation results ....................... 63
    5.1.2 LET for broadcast traffic ............... 67
    5.1.3 Conclusions and Further work .......... 70

6 Model Comparison 71
  6.1 Graph-based Network Model ................. 71
  6.2 Evaluation and Results ..................... 74
  6.3 Conclusions and Comments ................... 77

7 Conclusions 79

A Simulation Model 81
  A.1 Simulation Setup .......................... 81
    A.1.1 Generation of networks ................. 81
    A.1.2 Link Gain ................................ 82
    A.1.3 Acknowledgment scheme .................. 82
    A.1.4 Traffic model - end to end .............. 83
Chapter 1

Introduction

1.1 Background

In future military operations, the armed forces must be able to operate against a variety of threats (from heavy armor to irregular forces) and in a variety of environments, including urban, mountainous and forested terrain, and under jamming threats, sometimes while remaining covert. This will require a very flexible command, control and communications system that can be adapted to the prevailing situation.

The system must dynamically tailor a configuration that connects members of smaller forces together, links these separate forces together and connects the forces to higher command authorities and sources of information.

The forces must be able to maintain communications capability over extended ranges and interoperate with strategic-, operational- and tactical-level information systems to support autonomous operations with highly dispersed organizational elements. The network must be self-forming, self-maintaining and scalable to permit use over small as well as large areas of operation, possibly with large number of users.

This will require a dynamic re-configurable network that can exploit all sensors and information sources available for maximum efficiency so that information can be quickly acquired and assimilated at all levels of the command’s hierarchy.
This means that much of all this information and its requirements must be available even at the level of the individual soldiers. The system must be able to provide its users with robust communication, situation awareness, planning, tasking and coordination, precise geolocation and navigation, and possibly other services not yet foreseen [1].

Since different parts of the network experience very different situations, these different parts must be able to autonomously adapt to the prevailing local situations and exploit these for maximum efficiency. These situations can vary from a merely portable Headquarter LAN to the rapid changes on the battlefield itself. In all situations there may be hostile jammers present, and the loss of any unit is possible.

Even under the most severe conditions, the network must provide each soldier with the ability to transmit and receive command and control data at a minimal information rate, regardless of combat situation, position or environment. The network must also function if divided into sub-segments.

Therefore, deployment should be rapid and no support of pre-installed infrastructure can be assumed. This means that the radio network should be able to be deployed in unknown terrain and with minimal or no need of network pre-planning.

The radio units may be distributed in the terrain in an ad hoc manner, and line-of-sight communications cannot be guaranteed. To provide coverage, robustness and flexibility in such networks, multihop functionality is important. In multihop networks, traffic can be relayed through intermediate units on its way from source to destination, thereby achieving coverage. Distributed multihop radio networks are often referred to as ad hoc networks.

This is also efficient for increasing the robustness of the system. Relaying permits the use of “shorter links”, which give better signal strength compared to noise and hostile jammers. Several alternative routes from a source to a destination can exist, thereby allowing multi-path routing for further flexibility.

These requirements differ considerably from most civilian networks, which usually are geared to low cost with pre-installed wireless infrastructures. Civilian networks are often hierarchical in the sense that mo-
bile units will communicate through a central, static node. The loss of this central node leads to network failure of all units in the surrounding area.

All services the network will provide must be simultaneously handled by the system, and each of them has different service demands. For example, voice transmissions put high demands on the delay. The human ear is especially sensitive to long delays, which can come from large delay variations (buffered in the end), large delay mean values or a combination of both. On the other hand, a rather high bit error rate can probably be accepted. Other information updates may have high demands on the avoidance of bit errors, while delay may be of lesser importance.

The upholding of these requirements is usually denoted as Quality-of-Service (QoS) guarantees. QoS can be seen as a performance contract between the network and the application. In a mobile radio network no absolute guarantees can be given since there is always the possibility that the network separates into more than one network. However, in this case QoS can be seen as best effort in the sense that the network will achieve the performance agreed upon as long as it is at all possible.

Although it is not possible to foresee all services that will be required in the future, some of the basic requested services today will most likely be relevant in the future. In [2], three services are described as particularly interesting: group calls, situation awareness, and intranet connections. Group calls are generally considered to be the most important of these services. Group calls require guaranteed low delay, i.e. an upper bound on the time it takes to transmit a message from the source to destination or an upper bound on the variance of the delay (jitter).

One of the most challenging problems today in ad hoc network research is these QoS guarantees.

Although multi-hop networks have great advantages in fulfilling military requirements, they also make many problems more difficult than in hierarchical networks. For example, although the relaying of traffic can enable communication between units further away than what would otherwise be possible, it also introduces the problem of finding the path from source to destination. This problem is generally referred to as routing, see e.g. [3] for an overview of different routing methods in ad hoc.
networks.

Another important design issue is Medium Access Control (MAC), i.e. how to avoid or resolve conflicts due to simultaneously transmitting radio units.

Traditionally, MAC protocols for ad hoc networks are based on contention-based access methods, i.e. a user attempts to access the channel only when it actually has packets to send. The user has no specific reservation of a channel and only tries to contend for or reserve the channel when it has packets to transmit. This has clear advantages when the traffic is unpredictable. More specifically, the most frequently used protocols are based on carrier sense multiple access (CSMA) [4], i.e. each user monitors the channel to see if it is used, and only if it is not will the user transmit. However, this is done in the transmitter while collisions appear in the receiver. This can lead to the so-called hidden terminal problem. A way around this is to first transmit a short request-to-send (RTS) and then only send the message if a clear-to-send (CTS) is received. This is the general principle of the IEEE 802.11 standard [5], which at present is the most investigated MAC protocol. However, several RTS can be lost in a row which makes delay guarantees difficult.

Efforts have been made to guarantee QoS in CSMA-based medium MAC, see e.g. [6], but contention based medium access methods are inherently inappropriate for providing QoS guarantees.

One of the most important QoS parameters in many applications that are specifically sensitive to the MAC is the delay guarantees previously mentioned.

One approach where delay bounds can be guaranteed is time division multiple access (TDMA), i.e. the time is divided into time slots and each user receives its own time slot.

Unfortunately, in sparsely connected networks this is usually inefficient. But, due to the multihop properties, the time slots can often be shared by more than one user without conflicts. This will automatically be the case with dynamic MAC protocols like CSMA, since a user’s access to the channel only will affect a local area.

However, to achieve both high capacity and delay guarantees one can use spatial reuse TDMA (STDMA) [7], which is an extension of TDMA.
1.1. Background

where the capacity is increased by spatial reuse of the time slots, i.e., a time slot can be shared by radio units geographically separated so that small interference is obtained.

Although STDMA has several properties that could make it useful in tactical military communication, much research and development is still necessary if we want this MAC protocol to reach its full potential, especially in mobile environments.

The problem is to design STDMA schedules that fulfill required properties, e.g. minimizing delay or being able to update the schedules in a distributed fashion. An STDMA schedule describes the transmission rights for each time slot.

In figure 1.1 we show an example of a graph representation of a 9-node network. In this we can see that communication between nodes 1 and 9 must be relayed by nodes 2 and 3. An STDMA schedule could assign link 1, 2, and 3 to transmit simultaneously, since they are sufficiently far from each other. Another set could be 4, 5 and 6, but not 1, 5, and 6 since, at least in this example, we are using omni-directional antennas and the transmission of node 3 would interfere with the reception of node 2.

In the above example, all transmission rights are assigned to the links, i.e. both transmitting and receiving nodes are determined in advance when the schedule is created. This is called link assignment. An alternative would be to assign transmission rights to the nodes instead. In this case only the node is scheduled to transmit in the time slot. Any of its neighbors, or all, can be chosen to be the receiving node. This is called...
node assignment.

Generally, node assignment is used for broadcast traffic, and link assignment is used for unicast traffic. However, in the multipurpose networks of the future, the network must be able to handle both these types of traffic simultaneously, which means that it is preferable if one assignment method can be used for all traffic types.

All of these choices can be seen as different assignment strategies for STDMA scheduling. In this thesis we will investigate the properties of different assignment strategies that are useful for STDMA scheduling.

1.2 Previous Work

The problem of designing STDMA schedules is well addressed in the literature. Centralized algorithms [8, 9] as well as distributed algorithms [10, 11, 12, 13, 14] for mobile ad hoc networks have been proposed. Few of these have been implemented into functional systems, but one of these, USAP [14], is used for the generation of multi-channel STDMA schedules in the soldier phone radio [15], which is designed as an ad hoc radio for military use in mobile environments.

Previous work on STDMA has generally investigated the two types of assignment strategies previously mentioned, i.e. node assignment and link assignment.

Examples of node assignment algorithms can be found in [16, 17, 8, 13], and link assignment algorithms in [10, 12, 9].

Furthermore, it has been shown that finding a minimum length schedule is NP-complete for both link and node scheduling [18, 19]. In [20] algorithms for both link and node scheduling are described that focus on generation on short schedules, i.e. a schedule where all nodes or links are given a slot with as few time slots as possible. In [21] a more general description of the assignment problem is given. The different assignment methods are seen as constraints in a unified algorithm for the assignment problem given in the paper.

The network models used by these algorithms have varied in complexity. The oldest algorithms in this area generally used a simple graph model of the network, see e.g. [7, 16]. In this an edge between two nodes
represents that they can communicate with each other, and the lack of an edge represents that they cannot affect each other even as interferences.

A more complex network model uses a two-level graph model, see e.g. [22]. Here interference edges are added, meaning that there is not sufficient signal power to receive the packet without error, but it is strong enough to interfere with reception from other users. This model gives a better description of the network and is still useful for a mobile scenario where the schedule must be updated often.

A more realistic, although much more complex, model is the use of the signal-to-interference ratio. In this case, a node is assumed to be able to receive a packet without error if the received signal strength is sufficient compared to the noise and all interfering signals (from simultaneously transmitting nodes in the network). This model was suggested for STDMA scheduling in [23]. Although the model gives the best description of a radio network it is probably very difficult to use in a mobile scenario.

Since a two-level graph model probably has to be used in a mobile scenario, it is important to know how much capacity is lost compared with the full interference information.

### 1.3 Contributions

We will study the behavior of STDMA in static, or low mobility, networks, rather than the effects and problems that the mobility creates. The investigation can therefore be seen as an upper bound on the efficiency of STDMA.

Furthermore, for the same reason, we will not study the practical issues of how to make an STDMA scheme work. Things that can affect this include, slot synchronization, distribution of schedules in a mobile scenario, exact functionality of data link layer, effects of TX/RX turnaround time, hostile jammers and similar effects which do affect the performance of the network.

The original contributions to this thesis are mainly the following;

1. A comparison between the two most common assignment methods
used today, i.e. node and link assignment. This comparison is performed both in an analytical manner and via simulations. This is done in order to determine when node or link assignment should be used. Results indicate that only the size and the connectivity of the network is necessary to determine this.

This has been published in [24].

2. A novel assignment strategy achieving the advantages of both link assignment and node assignment. The strategy proposed is based on a link schedule, but where transmission rights are extended. This strategy is evaluated in comparison with the other two methods by using approximations as well as using simulations.

Most of this is published in [25] and [26].

3. Furthermore, the design of algorithms that generate the link schedules is studied. We also suggest how to expand a basic link assignment algorithm in order to take better advantage of the new assignment strategy so as to decrease the packet delay. Most of this will be published in [27].

4. We include an investigation of the loss of efficiency (in terms of throughput and delay) when the STDMA algorithm only has knowledge about the two-level graph model of the network compared to having full knowledge of the attenuation between all pairs of nodes.

This is published in [28].

1.4 Thesis Outline

In chapter 2 we describe the network model we have used. We also describe the layers, according to the OSI model, that are of interest to us, i.e. data link layer, network layer and transport layer. The data link layer describes the functionality of the links and when a link can be used without conflicts.
1.4. Thesis Outline

The transport layer basically gives us a model of the external traffic of the network, whereas the network layer includes the routing of this traffic. This main purpose of this is to calculate the traffic on the links and nodes in the network. We also define the evaluation parameters, which are the average end-to-end packet delay and the maximum throughput, and describe the algorithm we assume are used.

In chapter 3 we define and exemplify node assignment and link assignment. We also provide approximations for the maximum throughput and the average packet delay for these assignment methods. These are then used to compare the efficiency of the algorithms. We conclude the chapter with simulations of delay and throughput to determine how well the approximations work.

In chapter 4 we describe a novel assignment method LET. Here, too, we give an approximate formula for the delay and use this in comparison with simulations. These results are then compared to node and link assignment showing the advantages of LET.

In chapter 5, we develop more advanced link assignment algorithms that are able to take advantage of the LET property. We show that this algorithm gives a lower average delay than LET based on the basic algorithm used in the previous chapter.

In chapter 6, we make a comparison between using the traditional graph model when designing an STDMA schedule and using the signal-to-interference ratio. Finally, in chapter 7 we present some general conclusions on the effect of using different assignment strategies.

In the appendix we present a more specified description of how simulations is performed.
Chapter 2

Network Model

This chapter introduces the network model we use and the assumptions required. These can be divided into two parts: first, the assumptions on the data link layer and then the assumptions from the network and transport layers. The data link layer is described in the first section. The assumptions on the higher layers describes how the traffic is generated and routed.

Furthermore, we also describe how the performance is evaluated.

2.1 Data link layer

In this section we describe our model for the data link layer. In essence, it is an interference-based model of the radio network, which is represented by a set of nodes $V$ and the link gain $G(i, j)$ between any two distinct nodes $v_i$ and $v_j$, $i \neq j$.

For the link level we will make the following assumptions:

- All antennas are isotropic.
- All nodes use equal transmission power.
- The data rate is equal on all links in the network.
- There is only one fixed required BER on the links.
• Slot synchronization is perfect.

• All packets are of equal length.

• A node cannot transmit more than one packet in a time slot and a node cannot receive and transmit simultaneously in a time slot.

The assumption on isotropic antennas and equal transmission power is mainly for simplicity, but in section 4.2 we will present a brief discussion of the consequences of directional antennas and varying transmission power specified on LET, since this assignment strategy will be affected most by these assumptions.

For any two nodes, \( v_i \) and \( v_j \) where \( v_i \) is the transmitting node and \( v_j \neq v_i \), we define the signal-to-noise ratio (SNR), \( \Gamma_{ij} \), as

\[
\Gamma_{ij} = \frac{P_i G(i, j)}{N_t},
\]

where \( P_i \) denotes the power of the transmitting node \( v_i \), \( G(i, j) \) is the link gain between nodes \( v_i \) and \( v_j \), and \( N_t \) is the noise power in the receiver. For convenience, we define \( \Gamma_{ii} = 0 \) corresponding to the physical situations of a node not being able to transmit to itself.

We say that a pair of nodes \( v_i \) and \( v_j \) form a link \((i, j)\), if the signal-to-noise ratio (SNR) is not less than a communication threshold, \( \gamma_C \). That is, the set of links in the network, \( \mathcal{L} \), is defined:

\[
\mathcal{L} = \{ (i, j) : \Gamma_{ij} \geq \gamma_C \}
\]

For a set of links, \( L \subseteq \mathcal{L} \), we define the transmitting nodes:

\[
V_T(L) = \{ v_i : (i, j) \in L \}
\]

For any link, \( (i, j) \in L \), we define the interference as follows

\[
I_L(i, j) = \sum_{v_k \in V_T(L) \setminus v_i} P_k G(k, j).
\]
Furthermore, we define the signal-to-interference ratio (SIR):

$$\Pi_L(i, j) = \frac{P_i G(i, j)}{(N_r + I_L(i, j))}.$$  (2.4)

We assume that any two radio units can communicate a packet without error if the SIR is not less than a reliable communication threshold, $\gamma_R$. The choice of these thresholds is of course dependent on several factors, such as the actual modulation method of the signal, properties of the receiver noise, data rate and required BER.

The threshold $\gamma_R$ will be determined by the factors described above. However, these factors only decide the lowest possible $\gamma_C$. That is, we can choose a higher $\gamma_C$, thereby excluding some node pairs from communicating with each other. By doing this we can create an interference margin so that all links can handle some interferences. However, this comes at the price of longer routes and the risk that the network will divide into sub-segments. For simplicity we will assume that $\gamma_R = \gamma_C$.

### 2.2 Transport and Network Layer

The traffic arriving at the network can be separated into two types. The first is unicast traffic, with a single source and destination. This type of traffic can be the carrier of many types of information, e.g. file transfer or telephone conversations. The second type of traffic is multicast or broadcast, i.e. a packet has one source but many destinations. With broadcast we mean the entire network. Broadcast traffic is very usual in military networks, e.g. group calls or situation awareness data.

Since all nodes cannot directly communicate with all other nodes in the network, due to limited transmission power, obstacles and large distances, all nodes in the network are assumed to be able to relay packets. In order to do this, each node is assumed to have a routing table with entry’s for all other nodes in the network. We will not elaborate on how this information is obtained but merely note that the routing will have an effect on the traffic in the network. We assume that all networks are connected, i.e. there is always a path between any pair of nodes.
In the following we first discuss the effects of this for unicast traffic and then the effects for broadcast traffic.

Unicast traffic assumes that a packet entering the network has only one destination. Packets enter the network at entry nodes according to a probability function, \( p(v), v \in V \), and packets exit the network at exit nodes. When a packet enters the network, it has a destination, i.e. an exit node from the network. The destination of a packet is modeled as a conditional probability function, \( q(w|v), (w,v) \in V \times V \), i.e. given that a packet has entry node \( v \), the probability that the packet’s destination is \( w \) is \( q(w|v) \). For simplicity we will assume a uniform traffic model, i.e. \( p(v) = 1/N \), and \( q(w|v) = 1/(N - 1) \), where \( N \) is the number of nodes, \( N = |V| \). This assumption will not affect our results since we use traffic controlled schedules, thereby compensating for variations caused by the input traffic model.

Let \( \lambda \) be the total traffic load of the network, i.e. the average number of packets per time slot arriving at the network as a whole. Then, \( \lambda/N(N - 1) \) is the total average of traffic load entering the network in node \( v_i \) with destination node \( v_j \). As the network is not necessarily fully connected, some packets must be relayed by other nodes. In such a case, the traffic load on each link can be calculated only when the traffic has been routed.

For unicast traffic we use the shortest route counted in the number of hops, i.e. packets sent between two nodes will always use the path which requires the least number of transmissions. If several routes of the same length exist, all packets between two specific nodes will always use the same route.

The reason why this routing strategy is used, except for its simplicity, is that this minimizes the number of retransmissions needed before a packet reaches the destination. Since we are assuming a fixed data rate on the links, it would be difficult to take advantage of a strategy which uses a longer route, since we would have to generate schedules with a much better spatial reuse to compensate for this.

Now, let \( R_u \) denote the routing table for unicast traffic, where the list entry \( R_u(v, w) \) at \( v, w \) is a path \( p_{vw} \) from entry node \( v \) to exit node \( w \), where \( p_{vw} \) is given by the routing algorithm described above. Let the
number of paths in \( R_u \) containing the directed link \((i, j)\) be equal to \( \Lambda_{ij} \).

From now on, we will call this parameter the relative traffic on link \((i, j)\).

Further, let \( \lambda_{ij} \) be the average traffic load on link \((i, j)\). Then \( \lambda_{ij} \) is given by

\[
\lambda_{ij} = \frac{\lambda}{N(N-1)} \Lambda_{ij}.
\]

Moreover, we have, \( \lambda_i \) as the average traffic load on node \( v_i \). Here \( \lambda_i \) is given by:

\[
\lambda_i = \frac{\lambda}{N(N-1)} \sum_{j: (i,j) \in L} \Lambda_{ij} = \frac{\lambda}{N(N-1)} \Lambda_i,
\]

where \( \Lambda_i \) is denoted as the relative traffic of node \( v_i \).

This can be described in a similar way for broadcast traffic, which assumes that a packet entering the network has all other nodes as destination. Packets enter the network at entry nodes according to a probability function, \( p(v) \). We will also assume a uniform traffic model, i.e. \( p(v) = \frac{1}{N} \).

Again, let \( \lambda \) be the total traffic load of the network, i.e. the average number of packets per time slot arriving at the network as a whole. Then \( \lambda/N \) is the total average of traffic load entering the network in node \( v_i \) destined to all other nodes.

As the network is not necessarily fully connected, some packets must be relayed by other nodes. In such a case, the traffic load on each link can be calculated only when the traffic has been routed.

Radio is an inherent broadcast medium, especially when we are using omnidirectional antennas. This means that if we are sending a packet, all nodes within range can receive the packet if nothing else interferes with them. Since the interference allowed in the nodes is determined by the assignment strategy, this means that depending on what assignment strategy that is used this determines the actual traffic that must be transmitted by the nodes. For example, node assignment guarantees that all neighbors are collision-free, meaning that all neighboring nodes that should receive the packet will do so by a single transmission. Link assignment,
on the other hand, is a single transmission over a link. If several neighbors should receive the packet, they have to receive a transmission each.

We will therefore define the average traffic of a node as the average number of different packets that are to be transmitted, regardless of whether they have one or several destinations. This means that the actual number of transmissions of the node will be at least this high, which is the case if node assignment is used since all packets will reach all their destinations with a single transmission each. Other transmission strategies may require a larger number of transmissions if more than one is required for a transmission, e.g. link assignment.

For broadcast traffic, a more advanced routing method must be used than for unicast traffic. As mentioned, radio is an inherent broadcast medium. Dependent on which assignment strategy that is used, we can use this to our advantage in a more or less efficient manner. One way of doing this is to minimize the number of retransmissions needed for a packet to reach all destinations, i.e. we want as many neighboring nodes as possible to be reached by each transmission. This can also be described as maximizing the number of leafs in the routing tree.

However, this is a very complex problem, so for simplicity we will use a heuristic algorithm in an attempt to achieve this.

The following creates a routing tree for each node.

- Initiate by choosing the node as root. Find the node \( v_i \) with the highest number of neighboring nodes that is not included in the tree. Include all these neighboring nodes and the edges from \( v_i \) to these nodes. This is repeated until all nodes are included in the tree.

Now, let \( R_b \) denote the routing table for broadcast traffic, where the list entry \( R_b(v) \) at \( v \) is a tree with entry node \( v \) as root. Let the number of trees in \( R_b \) containing the directed link \( (i, j) \) be equal to \( \Lambda_{ij} \) and let the number of trees containing vertex \( v_i \) as a non-leaf in the tree be equal to \( \Lambda_i \).

Further, let \( \lambda_{ij} \) be the average traffic load on link \( (i, j) \) and \( \lambda_i \) be the average traffic load on node \( v_i \). Then \( \lambda_{ij} \) is given by

\[
\lambda_{ij} = \frac{\lambda}{N} \Lambda_{ij},
\]
2.3 Performance Measures

One important parameter which will affect the usefulness of a radio network is the delay of information from source to destination. We will start by discussing delay for unicast traffic and then discuss how this differs from broadcast traffic.

As seen from a network level, this can be translated to the network delay. Network delay is the expected time, in time slots, from the arrival of a packet at the buffer of the entry node to the arrival of the packet at the exit node, averaged over all origin-destination pairs. This is the first parameter we will investigate.

To determine the network delay we need the following definitions and notations.

The stochastic variable path delay $D_{kl}$ for a path $p_{kl}$ is the time, in time slots, from the arrival of a packet at the buffer of the arrival node $v_k$ to the arrival of the packet at the destination node $v_l$. The edge delay $d_{kl}(i, j)$ is the time, in time slots, from the packet arrives at the buffer of node $v_i$ until it is received by node $v_j$, given that the packet is relayed on path $p_{kl}$. This path is deterministically given by the routing algorithm as described in the previous section.

The path delay is thus the sum of the edge delays of that path,

$$D_{kl} = \sum_{(i, j) \in p_{kl}} d_{kl}(i, j). \quad (2.5)$$

The average path delay is a stochastic variable defined as an average over all origin-destination pairs:

$$\frac{1}{N(N-1)} \sum_{(k,l) \in N^2} D_{kl}. \quad (2.6)$$

The network delay $D$ is the expected value of the average path delay.
Due to the relaying of packets the statistical properties of this variable are complicated, and an exact analytical analysis of the network delay is difficult [7]. Instead we will use computer simulations to determine this parameter. How this is done is described in the appendix.

We will now compare the above-described with broadcast traffic. When we have broadcast traffic, each arriving packet has all other nodes as destination. We can now use several different definitions of the packet delay since a packet will arrive at its destinations at different times. One example would be the maximum delay, i.e. the delay of a packet would be the delay until the arrival at the last node. However, most nodes will experience a much lower delay. So instead we will use the average delay of a packet. The delay of a packet arriving at node $v_i$ can be written as

$$D_i = \frac{1}{N - 1} \sum_{j: (i,j) \in \mathcal{N}^2} D_{ij}$$

where $D_{ij}$ can be described as the path delay between node $v_i$ and node $v_j$. This is the path in the tree with root $v_i$ given by the routing algorithm for broadcast routing described in the previous section.

The network delay, $D$, can thus be described as the expected value of the following expression

$$\frac{1}{N} \sum_{i \in \mathcal{N}} D_i = \frac{1}{N(N - 1)} \sum_{(i,j) \in \mathcal{N}^2} D_{ij},$$

which is similar to the expression for unicast traffic. However, the actual traffic on the nodes and links will differ, which will result in different delays for the different traffic types.

The demand for higher data rates can often lead to highly loaded networks. One of the advantages of STDMA is that it can function well even under high traffic loads.

In particular, the largest admissible traffic load yielding a finite network delay is highly interesting. This maximum traffic load is commonly referred to as the throughput of the network. We define the maximum throughput as the number $\lambda^*$ for which the following expressions hold
for all traffic loads $\lambda$,

$$\begin{cases} \lambda < \lambda^* & \text{yields bounded } D \\ \lambda > \lambda^* & \text{yields unbounded } D \end{cases}$$

Throughput is the second parameter we study.

For the specific assignment methods we study in this thesis, we present a good estimation of the maximum throughput given the schedule and a specific routing. Except for LET with broadcast traffic, in which case simulations will be performed, this is done in chapter 3.

We define connectivity as the fraction of nodes in the network that can be reached by a node, in one hop, on average, i.e. $M/(N(N-1))$, where $M$ is the number of directed links in the network.

### 2.4 STDMA scheduling

In this section we describe and motivate the choice of STDMA algorithm we assume is used. This algorithm is described in such a way that it can be used independently on the specific assignment strategy.

In all assignment strategies we effectively assign time slots to sets of links. In link assignment, these sets consist of single links, and node assignment can be described as all outgoing links from the assigned node. We will use the notation $x_i$ for one of these link sets and the notation $X$ for the union of all link sets. One thing to notice is that a link $(i, j)$ can belong to more than one link set, although this is not the case for node or link assignment.

A schedule $S$ is defined as the sets $Y_t$, for $t = 1, 2, \ldots, T$, where $T$ is the period of the schedule. The sets $Y_t$ contain the link sets assigned time slot $t$.

We will use the notation transmit simultaneously to denote that all possible receiving nodes of the link sets assigned the time slot have SIR above the reliable communication threshold. A schedule is called conflict-free if this is the case for all time slots in the schedule.

The algorithm assumes full knowledge of the interference environment. It needs as an input the basic path-loss, or estimates, from all nodes to all other nodes.
This is a description of a basic algorithm.

For each new loop a time slot, numbered \( t \), is created. In step two of the algorithm we first check to see if the link sets that have not yet received a time slot can be assigned to this slot, i.e. transmit simultaneously with the assigned link sets. In the third step the same check is performed on the rest of the link sets. For some \( t = T \leq |\mathcal{V}| \), all link sets will have received at their time slots and the algorithm will terminate. The output of the algorithm are the sets \( Y_t \) for \( t = 1, 2, \ldots, T \), where \( T \) is the period of the schedule and the length of the frame. When the algorithm terminates the sets \( Y_t \) will contain the link sets \( x_i \) that are assigned to time slot \( t \).

In multi-hop networks the traffic load on the various nodes will differ considerably. This will cause “bottleneck” effects at busy nodes with long packet delays as a result. This can be compensated for by assigning the more heavily loaded link sets several time slots.

That is, the algorithm works so that each link set is guaranteed a certain number of slots in each period or frame. This number is based on the relative traffic of each link set. Furthermore, the link sets are assigned time slots according to a priority list. The priority of a link set is based both on the relative traffic load and on the number of time slots passed since the link set previously was assigned a slot. With this procedure, the slots assigned to each link set will be spread out evenly over the period, resulting in a decreased network delay.

In the following we will use the notation \( \Lambda_{x_i}^\tau \) to denote the relative traffic of link set \( x_i \). If we use link assignment, the link set is actually a link \((i, j)\) and \( \Lambda_{x_i}^\tau = \Lambda_{ij} \). Node assignment, on the other hand, means that \( x_i = \{(i, k) : (i, k) \in \mathcal{L}\} \) and \( \Lambda_{x_i}^\tau = \Lambda_i \).

In the algorithm, link set \( x_i \) will be guaranteed \( \Lambda_{x_i}^\tau \) number of slots per frame.

In the final schedule all link sets will have at least this many time slots, and some may have more. In general the algorithm will work for any fixed routing.

With the above procedure, link sets with a high traffic load will obtain several slots per frame. In this case the network delay will also depend on how evenly over the frame, these slots are arranged. To spread out
the slots over the frame the link sets are ordered in a list of priority. The
link set priority is set to \( \tau_i \lambda_i^x \), where \( \tau_i \) is the number of slots that have
passed since the link set was previously allocated a time slot. The link
set allocation is then performed in the order described by the priority list,
highest priority first, etc.

In the following we describe the traffic sensitive algorithm obtained
with the above ideas. In addition to the inputs to the basic algorithm, this
algorithm needs the knowledge about the relative traffic of each link set.

**Step 1** Initialize:

1.1 Enumerate the link sets

1.2 Create a list, \( A \), containing all of the link sets and an empty
list \( B \).

1.3 Set \( t \) to zero.

1.4 Calculate the number of time slots each link set is to be guar-
anteed and set \( h_i = \lambda_i^x \).

1.5 Set \( \tau_i \) to zero for all link sets.

**Step 2** Repeat until list \( A \) is empty:

Step 2.1 Set \( t \leftarrow t + 1 \) and \( Y_t \leftarrow \emptyset \).

Step 2.2 For each link set \( x_i \) in list \( A \):

2.2.1 Set \( Y_t \leftarrow Y_t \cup x_i \).

2.2.2 If the link sets in \( Y_t \) can transmit simultaneously:

- If \( h_i = 1 \), remove the link set from list \( A \) and add to
list \( B \).
- Set \( h_i \leftarrow h_i - 1 \), and set \( \tau_i \) to zero.

2.2.3 If the set of link sets in \( Y_t \) cannot transmit simultane-
ously, set

\[
Y_t \leftarrow Y_t \setminus x_i.
\]

Set \( \tau_i \leftarrow \tau_i + 1 \).
Step 2.3 For each link set $x_i$ in list $B$ but not in $Y_t$:

2.3.1 Set $Y_t \leftarrow Y_t \cup x_i$.

2.3.2 If the link sets in $Y_t$ can transmit simultaneously, set $\tau_i$ to zero.

2.3.3 If the link sets in $Y_t$ cannot transmit simultaneously, set

$$Y_t \leftarrow Y_t \setminus x_i$$

and set $\tau_i \leftarrow \tau_i + 1$.

Step 2.4 Reorder lists $A$ and $B$ according to link set priority,

$$\tau_i \Lambda_i^x,$$

highest priority first.
Chapter 3

Node and Link Assignment

This chapter describes the two most frequently used assignment methods so far, node assignment and link assignment, and describes the advantages of both of them.

3.1 Link Assignment

In link-oriented assignment, the directed link is assigned a slot. A node can thus only use this slot for transmission to a specific neighbor. In general this knowledge can be used to achieve a higher degree of spatial reuse. The effect is higher throughput.

Below, we describe the criteria for a set of links to be able to transmit simultaneously with sufficiently low interference level at the receiving nodes.

We say that a link \((k, l)\) is adjacent to any other link \((i, j)\) \(\in L\) iff \(\{i, j\} \cap \{k, l\} \neq \emptyset\) \((i, j) \neq (k, l)\). Furthermore we define \(\Psi(L)\) as the union of all adjacent links to the links in \(L\). We assume that a node cannot transmit more than one packet in a time slot and that a node cannot receive and transmit simultaneously in a time slot. Alternatively, we say that a set of links \(L\) and the set of its adjacent links \(\Psi(K)\) must be disjoint:
We also require that the SIR value is sufficiently high for reliable communication, see section 2.1,

\[ \Pi_L(i, j) \geq \gamma_R \quad \forall \quad (i, j) \in L. \]  

(3.2)

If the above two conditions, (3.1) and (3.2), hold for a set of links \( L \in \mathcal{L} \), we say that the links in \( L \) can transmit simultaneously.

Figure 3.1 shows a small network. Assuming that interference between nodes without communication links is small, we can see that links 1, 2 and 3 can transmit simultaneously.
One problem with this assignment method is that it does not take advantage of the inherent broadcast properties of the radio medium. Each transmitted packet will only be received by the assigned receiver despite being sent in all directions (omnidirectional antennas). This is no problem with unicast traffic, but for broadcast traffic, where each packet should reach several destinations, it is inefficient. In these cases the packet has to be retransmitted for each of these receivers.

For example, if a broadcast packet has node $B$ as source, it must be transmitted three times from node $B$. Link 1, 5 and 6 must transmit the packet at different time slots. Then to reach the last destination, the packet must be transmitted on links 2 and 3. As these two links can transmit simultaneously, it is possible to use only four time slots in order to reach all destinations.

### 3.2 Node Assignment

In a node-assigned schedule, a node is allowed to transmit to any of its neighbors in its slot. If the schedule is to be conflict-free, this means that we have to guarantee that we will not have a conflict in any of the neighboring nodes.

The advantage is that if the packet should reach more than one of the neighbors to a node, one transmission of the packet is sufficient to reach them all. This makes node assignment very efficient for broadcast traffic.

There are two necessary conditions for the situation when all the nodes in a set $V$ are allowed to transmit packages simultaneously. Let the neighbors $\Omega(v)$ to a node $v \in V$ be the set of all nodes that have a link from $v$ to itself. The neighbors are the nodes that $v$ possibly can transmit a packet to. Similarly, let $\Omega(V)$ denote the union of all neighbors of all nodes in $V$.

The first condition is that two neighbors cannot transmit at the same time. Another way to say this is that the sets $V$ and $\Omega(V)$ must be disjoint:

$$V \cap \Omega(V) = \emptyset.$$  (3.3)
Let $L(V)$ be the set of all links from the nodes in $V$ to their neighbors in $\Omega(V)$. Since it must be possible to use all the links in $L(V)$ for transmission simultaneously, we state the second condition:

$$\Pi_{L(V)}(i, j) \geq \gamma_R \text{ for all } (i, j) \in L(V). \tag{3.4}$$

If the above two conditions, (3.3) and (3.4), hold for a set of nodes $V \in \mathcal{V}$, we say that the set of nodes can transmit simultaneously.

If we return to the example network and compare it with link assignment, we see that the transmitting nodes to the three links that could transmit simultaneously cannot be permitted to transmit simultaneously when we use the node-assignment strategy, since both node $A$ and $C$ may choose to transmit to $B$.

In the broadcast traffic example, nodes $A$, $C$ and $E$ can be reached by one transmission of $B$, thereby decreasing the number of necessary transmissions. However, to reach the last two nodes we must transmit on $A$ and $C$, and these nodes cannot transmit simultaneously. This results in the use of three time slots in order to reach all destinations, which is one time slot lower than for link assignment.

### 3.3 Analysis

In this section we present some analytical results that are useful when evaluating the properties of node and link assignment. We first discuss throughput and then continue with an approximation of the network delay at low traffic arrival rate.

In [29] the throughput in a network with fixed capacities on the links and fixed routing is determined. Here, we will do the same specifically for a link-assigned schedule and a node-assigned schedule.

A link-assigned schedule contains, for each time slot in the frame, the set of links $(i, j)$ that are allowed to transmit in that time slot. A node-assigned schedule contains the set of nodes $v_i$ that are allowed to transmit in each time slot. We assume that the number of time slots in a frame, $T$. We do not consider packet failures and retransmissions.

Each link may be assigned several time slots per frame. Let $h_{ij}(S_L)$ denote the number of slots in schedule $S_L$ where link $(i, j)$ can access a
time slot and similarly let $h_i(S_N)$ denote the number of slots in schedule $S_N$ where node $v_i$ can access a time slot.

3.3.1 Throughput for link assignment

The following is done for link assignment. Later we will do the same for node assignment. As was previously defined, the throughput is the largest admissible traffic load yielding a finite network delay. Studying equations (2.5) and (2.6), it is easy to realize that the network delay is bounded as long as the edge delay of all links is bounded; the network delay is unbounded if the edge delay of any of the links is unbounded.

So for a good estimation of the throughput, we only need to find the smallest value of $\lambda$ so that at least one of the link queues are saturated. This will happen when the offered traffic rate on the link $\lambda_{ij}$ equals the rate at which it can transmit packets, i.e. $h_{ij}/T_L$ packets per time slot, where $T_L$ is the frame length of the link-assigned schedule. This gives us the following equation for each link in the network for unicast traffic

$$\lambda_{ij} = \frac{\lambda}{N(N-1)} \Lambda_{ij} = \frac{h_{ij}}{T_L},$$

thus resulting in the following saturation traffic arrival rate for the link

$$\lambda = \frac{h_{ij}N(N-1)}{T_L \Lambda_{ij}}.$$

The throughput of the network $\lambda^*(S_L)$ can now simply be calculated as

$$\lambda^*(S_L) = \min_{(i,j) \in L^*} \frac{h_{ij}(S_L)N(N-1)}{T_L \Lambda_{ij}}. \quad (3.5)$$

Furthermore, if we use schedules that are fully compensated for traffic, i.e. $h_{ij} = \Lambda_{ij}$, the formula for link assignment can be simplified to

$$\lambda^*_L = \frac{N(N-1)}{T_L}. \quad (3.6)$$
Figure 3.2: The figure shows the ratio between estimated values of the maximum throughput and simulated values for the maximum throughput for networks of different connectivity. The ratio is plotted for 100 networks of size 20 nodes with unicast traffic.

For broadcast traffic the formula for the throughput will be

$$\lambda^*_L(S_L) = \min_{(i,j) \in L^*} \frac{h_{ij}(S_L)N}{T_L \Lambda_{ij}},$$

and for a schedule that is fully compensated for traffic

$$\lambda^* = \frac{N}{T_L}.$$  \hfill (3.8)

In figure 3.2 we show a comparison of this estimation with simulation results. As can be seen this estimation is sufficiently good for our purposes. The largest part of the deviation between simulated and estimated results is probably due to the high delay variance of the simulation results close to the maximum throughput.

### 3.3.2 Throughput in node assignment

The corresponding notation for node assignment is $h_i(S_N)$ as the number of time slots assigned to node $v_i$, and $T_N$ as the frame length. The
throughput for node assignment in the case of unicast traffic is

$$\lambda^*(S_N) = \min_{i \in V} \frac{h_i(S_N)N(N-1)}{T_N \Lambda_i},$$

(3.9)

and the corresponding result for broadcast traffic can be written as

$$\lambda^*(S_N) = \min_{i \in V} \frac{Nh_i(S_N)}{T_N \Lambda_i}.$$  

(3.10)

For schedules that are fully compensated for traffic, i.e. $h_i = \Lambda_i$ the formula for node assignment with unicast traffic can be simplified to

$$\lambda^*_N = \frac{N(N-1)}{T_N}.$$  

(3.11)

and, similarly, for broadcast traffic

$$\lambda^*_N = \frac{N}{T_N}.$$  

(3.12)

The ratio between the throughput of a link-assigned schedule and the throughput of a node-assigned schedule can then be written as:

$$\frac{\lambda^*_L}{\lambda^*_N} = \frac{T_L}{T_N}.$$  

(3.13)

Notice, this is valid for both unicast and broadcast traffic.

**3.3.3 An approximative formula for the network delay**

We now derive an expression for the network delay $D_L$ for a link-assigned schedule with unicast traffic.

$$D_L = \sum_{(i,j) \in L} \frac{\Lambda_{ij}}{N(N-1)} d_{ij},$$

(3.14)

where $d_{ij}$ is the average expected edge delay on link $(i, j)$. 
In order to determine the network delay, we make two extra assumptions. Even if they are not fulfilled, it is still a useful approximation of the network delay.

First, the slots in the schedules assigned to a node are perfectly spread over the frame, i.e. the distance between two assigned slots is equal. The algorithm used in the simulations attempts to do this, but the algorithm is not optimal and it is not always possible to spread the slots evenly.

Second, the relay traffic can be described as a Poisson process. This is certainly not the case, since relay packets can only arrive in specific time slots. However, it is normally a good approximation. This is an attempt to use the same principle as the independence assumption [29], but for a TDMA network with fixed packet size.

With these assumptions, the average expected edge delay for a link can at low traffic arrival rate be written as:

\[
d_{ij} = \frac{T_L}{2h_{ij}},
\]

which inserted in (3.14) gives the network delay of a link-assigned schedule

\[
D_L = \sum_{(i,j) \in L} \frac{\Lambda_{ij}}{N(N - 1)} \frac{T_L}{2h_{ij}}.
\]

Furthermore, if the schedule is fully compensated for traffic, we have

\[
D_L = \frac{MT_L}{2N(N - 1)},
\]

where \(M\) is the number of links in the network.

The corresponding result for a node-assigned schedule is

\[
D_N = \sum_{i \in V} \frac{\Lambda_i}{N(N - 1)} \frac{T_N}{2h_i},
\]

and for a schedule fully compensated for traffic

\[
D_N = \frac{NT_N}{2N(N - 1)}.
\]
3.4 Evaluation and Results

The ratio between the network delay of link assignment and the network delay of node assignment can then be written as:

$$\frac{D_L}{D_N} = \frac{MT_L}{N\bar{T}_N} = \frac{M\lambda_N^*}{N\lambda_L^*}$$

(3.15)

Simulations in the next section will show that this is a good approximation except for low connectivity.

### 3.4 Evaluation and Results

In this section we compare the delays and throughput for node assignment and link assignment. The results for network delay are generated by network simulation as described in chapter 2. Furthermore, for unicast traffic we compare these simulated results with the approximative results given in section 3.3. The comparison begins with unicast and concludes with broadcast traffic.

#### 3.4.1 Unicast Traffic

We start the investigation by studying the networks at high traffic loads. The first parameter we study here is the ratio between maximum throughput $\lambda_L^*$ of link assignment and maximum throughput $\lambda_N^*$ of node assignment, i.e.

$$\frac{\lambda_L^*}{\lambda_N^*},$$

which we know from equation (3.13) equals $\bar{T}_N/T_L$.

As can be seen in figure 3.3, this ratio exhibits considerable variations over the networks studied. One conclusion in these simulations is that link assignment provides higher throughput. This is not so surprising since the degree of spatial reuse is higher for link assignment.

To determine how much better link assignment can be, we plot in figure 3.4 the ratio $\lambda_L^*/\lambda_N^*$ averaged over connectivity for networks of different sizes. As can be seen, $\lambda_L^*/\lambda_N^*$ increases with the size of the network and decreases if connectivity is increased.
Figure 3.3: The figure shows the ratio between maximum throughput for link assignment and node assignment for networks of different connectivity. The ratio is plotted for 500 networks of size 20 nodes.

Figure 3.4: The figure shows the average ratio between maximum throughput for link assignment and node assignment for networks of different connectivity. Ratio between throughput for networks of different size 10, 20, and 40 nodes.
We continue by studying the network delay at low traffic loads. The second parameter studied is the ratio between network delay of link assignment and node assignment at low traffic loads. In figure 3.5 this parameter can be studied. The variance is rather low and a linear relationship between delay and connectivity can be detected.

To see how well equation (3.15) approximates the delay we plot $D_L/D_N \ast \lambda_N/\lambda_L$, which should be approximately $M/N$. As can be seen in figure 3.6, this works fairly well. $D_L/D_N$ is slightly lower than predicted independent of connectivity.

The last parameter we study for unicast traffic is the input traffic load of the network which gives equal network delay for node and link assignment. This parameter is interesting since it determines for what traffic loads link/node assignment is preferable. That is, for traffic loads higher than this parameter, link assignment is preferable, and at lower traffic loads node assignment is preferable.

As can be seen in figure 3.7 the variance over the simulated networks of this parameter is less than for the other parameters, which means that the average value is highly interesting. This average is shown in figure
3.4.2 Broadcast Traffic

We start the investigation on broadcast traffic by returning to the case of networks at high traffic loads. The parameter studied is the ratio between maximum throughput $\lambda^*_L$ of link assignment and maximum throughput $\lambda^*_N$ of node assignment, i.e.

$$\frac{\lambda^*_L}{\lambda^*_N},$$

which equals $T_N/T_L$ for broadcast traffic, too.

This ratio can be studied for networks of size 20 nodes in figure 3.9. Similar to unicast traffic, this parameter exhibits considerable variations.

Figure 3.6: The figure shows the ratio between delay for link assignment and delay for node assignment multiplied by the ratio between the throughput for link assignment and throughput for node assignment. This should approximately be equal to the connectivity of the network multiplied by $N - 1$, which is the line plotted in the figure. The ratio is plotted for 500 networks of size 20 nodes.

3.8. From this we can conclude that for unicast traffic the preferable assignment method can be determined with knowledge only of the connectivity of, and input traffic to, the network.
Figure 3.7: The figure shows the input traffic level giving equal network delay for different network connectivity. This is plotted for 500 networks of size 20 nodes.

Figure 3.8: The figure shows the traffic load which gives equal network delay for 500 networks of sizes 10, 20, and 40 nodes.
Figure 3.9: The figure shows the ratio between maximum throughput for link assignment and node assignment for broadcast traffic. The ratio is plotted for 500 networks of different connectivity of size 20 nodes.

over the networks studied. However, unlike for unicast, the decrease in variation with connectivity is significant. For high connectivity there is very little variation left.

Not very surprisingly, node assignment behaves better than link assignment for nearly all networks, especially for high connectivity, although there are networks where link assignment achieves the higher throughput.

To make a more complete comparison, we plot, in figure 3.10, the ratio $\lambda^*_L / \lambda^*_N$ averaged over connectivity for networks of different sizes. As can be seen, $\lambda^*_L / \lambda^*_N$ decreases with both network size and connectivity. This is because the number of neighbors a node has increases when any of these parameters are increased, which node assignment can take better advantage of.

We continue by studying the network delay at low traffic loads. In figure 3.11 we see that node assignment always performs better that link assignment, especially for networks of high connectivity. In figure 3.12 this is averaged over the networks. We see that this effect increases with network size and connectivity in the same way as for throughput.
Figure 3.10: The figure shows the average ratio between maximum throughput for link assignment and node assignment for broadcast traffic. The ratio between throughput for networks of different size 10, 20, and 40 nodes.

From this we can conclude that link assignment can never compete with node assignment for pure broadcast traffic.
Chapter 3. Node and Link Assignment

Figure 3.11: The figure shows the ratio between delay for link assignment and delay for node assignment with broadcast traffic. The ratio is plotted for 500 networks of size 20 nodes.

Figure 3.12: The figure shows the average ratio between delay for link assignment and delay for node assignment for networks of different connectivity with broadcast traffic. The ratio between delay for networks of different size 10, 20, and 40 nodes.
Chapter 4

Link assignment with Extended Transmission rights (LET)

In this chapter we present a novel assignment strategy which is based on link assignment.

4.1 LET principle

Notice that the interference term in (2.3) depends only on which nodes that are transmitting and not on the nodes that are receiving packets. Assume that a node is assigned as a transmitter in a slot, i.e. an outgoing link of the node is assigned the slot. If this node redirects the transmission to a node, other than the assigned receiving node, the inequality in (3.2) still holds for all links originally assigned to the slot. This means that the interference level of the other simultaneously receiving nodes will not change. (Recall the assumption of omnidirectional antennas.) The redirected transmission in itself cannot always be guaranteed to be conflict-free.

Based on these observations, we suggest the following scheme for extending transmission rights for any given link-assigned schedule. When a link is assigned a time slot, the node first checks whether there is a packet to transmit on that link. If there is no such packet, any other link with the same transmitting node might be used if the node has a packet
to transmit. Preferably links that are conflict-free should have priority, so as to avoid unnecessary packet loss.

We call this strategy Link assignment with Extended Transmission rights (LET).

To illustrate how it works we present a simple example. Assume links 1, 2, and 3 in figure 4.1 have be scheduled to transmit in the same slot. Let us study node B in more detail. If node B does not have any packets to transmit to E, it is permitted to transmit on either of the links 5 or 6 in the slot assigned to link 1. Now, if both links 2 and 3 are used (or if these nodes also use the LET property), neither transmission on 5 or 6 will be successful. However, for low traffic the probability that this would happen is small. If none of the other two nodes use their slot, the redirected transmission will be successful; and if only one of
them transmits, we still have 50 percent probability of a success. This is because node B cannot know which one (if any) of the others will transmit.

We now continue by proving that by redirecting the transmissions the nodes in the network will not cause any conflict at any node which has not redirected their transmission. Assume that $L$ is a set of links such that they can transmit simultaneously according to equations (3.1) and (3.2), i.e.

$$\Pi_{L}(i, j) \geq \gamma_{R} \forall (i, j) \in L.$$ 

Furthermore, assume that the transmitting nodes of $L_{R} \subseteq L$ redirect their transmissions to other receiving nodes than scheduled in the initial link schedule and that $L_{NR}$ is the rest of the links, i.e.

$$L_{NR} = L \setminus L_{R}.$$ 

Let $L_U$ be the set of links used by the redirecting nodes, therefore,

$$V_{T}(L_{U}) = V_{T}(L_{R}).$$ 

If $L_{NR}$ is to be conflict-free, the following inequality must be valid

$$\Pi_{L_{NR} \cup L_{U}}(i, j) \geq \gamma_1 \forall (i, j) \in L_{NR}.$$ 

For any $(i, j) \in L_{NR}$, we can write

$$\Pi_{L_{NR} \cup L_{U}}(i, j) = \frac{P_{i}G(i, j)}{N_{t} + I_{L_{NR} \cup L_{U}}(i, j)}.$$

and

$$I_{L_{NR} \cup L_{U}}(i, j) = \sum_{v_{k} \in V_{T}(L_{NR} \cup L_{U}) \setminus v_{i}} \frac{P_{k}}{L_{b}(k, j)}.$$ 

However,

$$V_{T}(L_{NR} \cup L_{U}) = V_{T}(L_{NR}) \cup V_{T}(L_{U}) = V_{T}(L_{NR}) \cup V_{T}(L_{R}) = V_{T}(L),$$
resulting in, \( I_{LR\cup LA}(i, j) = I_L(i, j) \) and \( \Pi_{LR\cup LA}(i, j) = \Pi_L(i, j) \), which of course fulfills (3.2).

For broadcast traffic, the situation will be a little more complicated. Each packet transmitted may have more than one receiver. This means that some, but not all, packets may be received correctly. A worst case scenario would be that only transmissions over scheduled links would be received, and all of the rest of the links must be retransmitted. LET can then behave exactly as link assignment for high traffic loads. On the other hand, for low traffic loads, the behavior will be that of node assignment, due to the low interference in the network. (The interference may be higher for broadcast traffic than for unicast traffic since a packet will be split and then might be relayed simultaneously by two nodes.)

However, there is also a possibility that more links than scheduled happen to be collision-free. This can easily be the case if two nodes lie close each other. Any transmission with sufficiently high SIR to one of them will also reach the other with sufficiently high SIR.

The variable distance between the nodes in this type of network would be another reason for an improvement compared to link assignment. If the distance between transmitter and receiver is small enough, the signal strength will be sufficient to handle very high interference, which can often be the case if the link is not scheduled to the time slot.

We conclude this section by giving a more formal description of the LET algorithm that we use in our simulations.

This is done in each node:

**Step 1** If there is a packet in the scheduled link’s queue:

1.1 Transmit this packet on all links that it is supposed to be transmitted on.

1.2 Remove the packet from all link queues where this transmission was successful.

**Step 2** Otherwise if there is no such packet. Choose one of the node’s other outgoing links that can be exchanged conflict-free in the schedule that has a packet in queue:
2.1 Transmit this packet on all links that it is supposed to be transmitted on.

2.2 Remove the packet from all link queues where this transmission was successful.

Step 3 Otherwise if there is no such packet. Choose one of the rest of the node’s outgoing links that has a packet in queue:

3.1 Transmit this packet on all links that it is supposed to be transmitted on.

3.2 Remove the packet from all link queues where this transmission was successful.

Step 4 Otherwise, do nothing.

4.2 Basic Properties

In figure 4.2, network delay for different $\lambda$ is shown schedules using for three different assignment strategies in a network of 30 nodes with
unicast traffic. The assignment strategies are node assignment, link assignment and LET.

We see from this figure that in this network, link assignment is preferable to node assignment for high traffic loads. For low traffic loads, node assignment achieves a smaller delay. The LET method combines the advantages of the two methods and in this case achieves a smaller delay for all traffic loads.

In this figure some areas of interest can be seen; at low traffic loads, the ratio between delay of node assignment and delay of LET. Moreover, the ratio between delay of link assignment and delay of LET is interesting.

At high traffic loads, the ratio between throughput of node and link assignment is interesting. The ratio between the throughput of LET and link assignment is also interesting, but at very high traffic levels most of the links in the network will have packets in queue. For this case, LET will appear mainly as link assignment. LET will achieve the same throughput as the link-assigned schedule it is based on if the link-assigned schedule is fully traffic compensated.

This is because at very high traffic loads, the probability that a link will have a packet to transmit in its time slot will be close to one. The LET property will not be used, and the network will appear exactly as a link-assigned schedule. No conflicts will appear, resulting in the same throughput as the link-assigned schedule.

If the link-assigned schedule is not fully compensated for varying network traffic, LET will give at least as high throughput as the link-assigned schedule. This is because no packet transmitted on a link assigned in the time slot is lost in LET, and a highly loaded link can use one of the lower loaded outgoing links from that node.

Although any link-assigned schedule may be used to extend transmission rights, some link schedules may give LET more or less desirable properties. Here we discuss what effect the link schedule has on delay.

At low traffic load, the nodes will normally only have at most one packet in queue at a time. In this case, LET behaves as node assignment, with the node-assigned schedule as the transmitting nodes in the link-assigned schedule. A problem when generating STDMA schedules
is how to determine which slots to give a node or link that is going to receive more than one slot, since delay through the node depends on which specific time slots the node is given.

For example, assume that a node has received two time slots and these are spread in such a way that the distance between them is approximately half the frame length. Then for low traffic loads, the delay will be at most half the frame length. If the node is given two consecutive time slots, the maximum delay might be the entire frame length. That is, it is usually efficient to spread a node’s time slots evenly over the frame. This problem gets worse if nodes receive many time slots, especially since a large part of the traffic usually flows through these nodes.

In some algorithms, see e.g. [30], an effort is made to spread the time slots a node or link is given equally over the frame. However, even if the link schedule has perfectly spread time slots, this may not be the case for the transmitting nodes. Therefore, LET might give considerably higher delay than a node-assigned schedule if the link schedule tends to give the transmitting nodes consecutive time slots.

Any assignment algorithm, especially if it is of a greedy type, has a set of rules to determine which link to assign to a time slot. One method used is node ID, i.e. the lower the ID number, the higher the priority. To assign links, the pair of node IDs of the transmitting and receiving nodes can be used. A sorted list according to link priority would then give the outgoing links from a node consecutive places in the list. Even if another system for link priority is used, node ID is eventually used if priority is equal for several links.

Now, assume we have a fully connected network, i.e. all nodes can communicate with all other nodes without relaying. Furthermore, in this case we can assume that there is no spatial reuse, although in a link schedule using an interference-based model, this might be possible due to capture. In this case the assigned schedule would be the sorted list described above, which gives high network delay.

One way to avoid this problem is to give the links a link ID which is random, although different, for each link.

The assumptions used in this paper is the use of omnidirectional antennas and equal transmission power of all nodes in the network. If this
is not the case, a node cannot redirect its transmission to any other of its neighbors since this might require an increase of signal power or redirection of the antennas. Any change of the outgoing power strength and direction can ruin the conflict-free properties of the other receiving nodes.

However, some of the nodes may still be reached without such a change, and the LET properties can still be used with these nodes, although this is less efficient than if all nodes could be reached.

An alternative way of designing a LET schedule would be to base it on a node-assigned schedule instead and replace each node in the schedule with one of its outgoing links. Using LET on this schedule would result in a schedule that would behave as a node-assigned schedule for all traffic loads and traffic types, thereby achieving the advantages of node assignment for broadcast traffic. The actual order in which packets are transmitted can be different, though.

Unfortunately, this schedule would not have the positive properties of link assignment for unicast traffic.

4.3 Analysis

The same assumptions as we made for node and link assignment in previous chapter can be made for LET as well. However, since LET behaves like a node-assigned schedule at low traffic loads, the link-assigned schedule would have to try to spread the time slots for the transmitting nodes of the links evenly over the frame instead of the link time slots. However, the algorithm used in the simulations does not attempt to do this. This results in a larger discrepancy than for node assignment or link assignment when compared with simulations.

The network delay for unicast traffic using LET for low traffic loads under these assumptions can be written as:

\[ D_{LET} = \sum_{i \in V} \frac{\Lambda_i}{N(N-1)} \frac{T_L}{2h_i} \]

(4.1)

For a schedule fully compensated for unicast traffic \( h_i = \Lambda_i \), this
4.4. Evaluation and Results

results in,

$$D_{\text{LET}} = \frac{NT_l}{2N(N - 1)}.$$  

The ratio between network delay of link assignment and network delay of LET for unicast traffic can be written as

$$\frac{D_l}{D_{\text{LET}}} = \frac{M}{N}.$$  

(4.2)

and the ratio between network delay of node assignment and network delay of LET can be written as

$$\frac{D_N}{D_{\text{LET}}} = \frac{T_N}{T_l} = \frac{\lambda^*_L}{\lambda^*_N}.$$  

(4.3)

4.4 Evaluation and Results

In this section we compare the delays and throughput for LET with node assignment and link assignment. The results for network delay are generated by network simulation as described in the appendix. No comparison for throughput in the case of unicast traffic is included since LET achieves the same throughput as link assignment. For broadcast traffic such a comparison is included, using simulations to determine the maximum throughput for LET.

Furthermore, we compare these simulated results with the approximative results for unicast traffic given in section 4.3.

In the simulations of LET we have the extra simulation assumption that the link ID is random, as described in section 4.2.

4.4.1 Unicast Traffic

The first comparison is node assignment and LET. Therefore, the parameter studied is the ratio between network delay of node assignment and network delay of LET at low traffic loads, i.e.

$$D_N/D_{\text{LET}}.$$
Figure 4.3: The figure shows the ratio between the delay of node assignment and the delay of LET. The ratio is plotted for 500 networks of size 20 nodes.

If this parameter is greater than one, LET is always preferable. If it is less than one, node assignment is preferable for low traffic loads.

In figure 4.3, this parameter can be studied for networks of size 20 nodes. As can be seen there are some variations over the different networks. In figure 4.4 we plot the ratio $D_{LET}/D_N$ averaged over connectivity for networks of different sizes. As can be seen, $D_{LET}/D_N$ decreases with the connectivity. This is because the gain in spatial reuse of link assignment compared with node assignment decreases with connectivity. Its increase with network size is consistent with the approximation in equation (4.3), since it should be close to $\lambda_L/\lambda_N$.

It can be concluded that for the chosen assignment algorithms, except for high connectivity, a link-assigned schedule with extended transmission rights gives lower delay than a node-assigned schedule. For very high connectivity, LET can give a higher delay than node assignment because the link-assigned schedule our method is based on does not attempt to spread the time slots a node is assigned evenly over the frame.

We conclude the study of network delay for unicast traffic by examining the ratio between network delay of link assignment and network
delay of LET at low traffic loads $D_{LET}/D_l$. In figure 4.5, this ratio is averaged for networks of different sizes. It can be concluded that LET decreases the network delay considerably compared with link assignment. This effect increases with network size and connectivity. This is not a surprising result, since increasing network size or connectivity increases the number of outgoing links of a network node, thereby giving LET more opportunities.

From these simulations and the knowledge that LET always achieves at least as high throughput as link assignment, we can see that for unicast traffic LET is preferable to both link and node assignment except for networks of very high connectivity and low traffic.

### 4.4.2 Broadcast Traffic

We will now study LET in the case of broadcast traffic. In figure 4.6 we plot the ratio between delay for LET and the delay for node assignment. As can be seen there is a considerable variation over the different net-
Figure 4.5: The figure shows the average ratio between delay for link assignment and delay for LET for networks of different connectivity. The ratio between delay for networks of different size: 10, 20, and 40 nodes.

Figure 4.6: The figure shows the ratio between delay for LET and delay for node assignment for networks of different connectivity. This ratio is plotted for 500 networks of size 20 nodes with broadcast traffic.
works. However, for all except the highest connectivities, LET achieves the lower delay.

In figure 4.7, this ratio is averaged for networks of different sizes. It can be concluded that for high connectivity, LET achieves approximately the same result as node assignment with small dependence on network size. However, for low connectivity we see a considerable improvement when using LET for increasing network size.

We conclude the investigation of broadcast traffic by returning to the case of networks at high traffic loads. The parameter studied then is the ratio between maximum throughput $\lambda^*_{\text{LET}}$ of LET and maximum throughput $\lambda^*_N$ of node assignment, i.e.

$$\frac{\lambda^*_{\text{LET}}}{\lambda^*_N}.$$  

This ratio can be studied for networks of size 20 nodes in figure 4.8. Except for very low connectivity, in most networks node assignment gives considerably much higher throughput than LET; a factor two or more. It is interesting to see that for very low connectivity we have networks where LET outperforms node assignment.
Figure 4.8: The figure shows the ratio between maximum throughput for LET and maximum throughput for node assignment for networks of different connectivity. This ratio is plotted for 500 networks of size 20 nodes with broadcast traffic.

We can study this parameter averaged over size in figure 4.9 and see that this effect increases with the network size.

4.4.3 Conclusions

In this chapter we have presented a novel assignment strategy (LET), which is based on link assignment. We have shown that for unicast traffic LET is preferable to both link and node assignment except for networks of very high connectivity and low traffic.

For broadcast traffic we see that LET outperforms node assignment for low traffic arrival rates. However, this comes at a considerable cost in terms of throughput. One exception is low connectivity, where LET can give the same throughput as node assignment.

We can therefore conclude that LET is very useful for low connectivity networks independent of the traffic type. For a higher network connectivity we have to consider what kind of traffic the network is designed to handle when choosing the assignment strategy.
Figure 4.9: The figure shows the ratio between maximum throughput for LET and maximum throughput for node assignment for networks of different connectivity. This is plotted for networks of different size 10, 20, and 40 nodes with broadcast traffic.
Chapter 5

Algorithms designed for LET

As could be seen in the previous chapter, LET can achieve very poor performance results if the link schedule which it is based on is poorly adapted to the properties of LET. In this chapter we will study the performance enhancement in terms of delay that can be achieved if we take into consideration that the LET property will be used when the link schedule is generated.

The main reason for this poor behavior is that LET behaves as node assignment for low traffic loads, but the algorithm designs the schedule for link assignment. In order to achieve a low delay for low traffic arrival rates we use a priority system to spread a link’s slots evenly over the frame. However, this does not necessarily correspond to the transmitting nodes in the link-assigned schedule being spread evenly over the frame.

In this chapter we will see the effect when we use an algorithm that attempts to do this. This will be done mainly for unicast traffic, but some examples of the effect for broadcast traffic will also be included.

All results in chapter 4 indicate that although LET can achieve network delays as low as node assignment for broadcast traffic, we will lose a considerable amount of the capacity. This will also be the case for this modified algorithm.
5.1 Double Priority

We will expand the algorithm described in chapter 2 so that it uses double priority. The links are here assigned time slots according to two priority lists. The primary priority of a link is the number of time slots passed since the transmitting node was previously assigned a slot multiplied by the relative traffic of the transmitting node. The secondary priority of a link is the number of time slots passed since the link was previously assigned a slot multiplied by the relative traffic of the link. When we determine which link to test next, we first check which link has the highest primary priority. If two or more links have equal priority, the secondary priority is used to determine which of these to test first.

With this procedure, the slots assigned to each node will be spread evenly over the period, resulting in a decreased network delay when using LET.

In the following, we describe the traffic sensitive algorithm obtained with the above ideas.

Step 1 Initialize:

1.1 Enumerate the links
1.2 Create a list, $A$, containing all of the links and an empty list $B$.
1.3 Set $t$ to zero.
1.4 Calculate the number of slots each link is to be guaranteed and set $h_{ij}$ to $A_{ij}$.
1.5 Set $\tau^1_i$ to zero for all nodes.
1.6 Set $\tau^2_{ij}$ to zero for all links.

Step 2 Repeat until list $A$ is empty:

Step 2.1 Set $t \leftarrow t + 1$ and $V_t \leftarrow \emptyset$.
Step 2.2 For each link $(i, j)$ in list $A$:
   2.2.1 Set $L_t \leftarrow L_t \cup (i, j)$.
5.1. Double Priority

2.2.2 If the set of links in \( L_t \) can transmit simultaneously:
- If \( h_{ij} = 1 \), remove the link from list \( A \) and add to list \( B \).
- Set \( h_{ij} \leftarrow h_{ij} - 1 \), and set \( \tau^1_i \) and \( \tau^2_{ij} \) to zero.

2.2.3 If the set of links in \( L_t \) cannot transmit simultaneously: set

\[
L_t \leftarrow L_t \setminus (i, j).
\]

Set \( \tau^2_{ij} \leftarrow \tau^2_{ij} + 1 \).

Step 2.3 For each link \((i, j)\) in list \( B \) but not in \( L_t \):

2.3.1 Set \( L_t \leftarrow L_t \cup (i, j) \).

2.3.2 If the links in \( L_t \) can transmit simultaneously, set \( \tau^1_i \) and \( \tau^2_{ij} \) to zero.

2.3.3 If the links in \( L_t \) cannot transmit simultaneously, set

\[
L_t \leftarrow L_t \setminus (i, j)
\]

and set \( \tau^2_{ij} \leftarrow \tau^2_{ij} + 1 \).

Step 2.5 Set \( \tau^1_i \leftarrow \tau^1_i + 1 \) for nodes not in \( V_T(L_t) \).

Step 2.6 Reorder lists \( A \) and \( B \) according to priorities,

\[
\tau^1_i \Lambda_i \text{ and } \tau^2_{ij} \Lambda_{ij}
\]

with the highest priority first.

We will now compare the algorithm described above with the previously described algorithm for LET. To separate them, we need some form of notation. We will use the term modified LET to denote LET results based on the above algorithm.

5.1.1 Simulation results

We now compare the delay of link assignment to the delay of LET for unicast traffic, i.e. \( D_L / D_{LET} \). According to equation (4.2), this ratio
Figure 5.1: The figure shows the ratio between network delay for link assignment and network delay for the modified LET with unicast traffic. This should be approximately equal to the connectivity of the network multiplied by \( N - 1 \), which is the line also plotted in the figure. The ratio is plotted for 500 networks of size 20 nodes.

should be approximately equal to \( N - 1 \). The results, given in figure 5.1, with this modification are close to the expected result, but with an increase in variance with connectivity. This double priority system does not seem to spread the time slots as well as a single priority system and this becomes more complicated for larger numbers of links.

A possible explanation for this is the following: The links which are most loaded in the network are often rather centralized links in the network and are normally not especially short, due to the routing principal of using the least number of hops. These links are thereby rather difficult to schedule simultaneously. However, with the single priority system these links will be tested before the rest of the links in the network. This results in a fairly efficient way of finding schedules permitting these links to transmit simultaneously.

A problem of using node priority first is that most nodes have at least one outgoing link that is short and thereby simpler to add to the schedule. These links can be sorted for testing much earlier than before, thus
5.1. Double Priority

Figure 5.2: The figure shows the ratio between network delay for LET and network delay for the modified LET with unicast traffic. The ratio is plotted for 500 networks of size 20 nodes.

blocking the more difficult links from being assigned. This can result in the case where few of the more difficult links to assign receive time slots in the first part of the schedule, but get virtually all their time slots in the last part of the schedule, which is not a good solution for spreading the time slots evenly over the schedule.

In figure 5.2, we plot the ratio between network delay for the modified LET and network delay for ordinary LET. We see in this figure that for most of the networks the modification improves the performance of LET. This is especially the case for high connectivity. For low connectivity it is less efficient, probably due to rather inefficient distribution of time slots in these cases. In figure 5.3 we plot this ratio averaged over connectivity for networks of different size.

We see here that the modified LET performs better for all but low connectivity. This is probably because the normal link assignment.

We now return to the approximation given in equation (4.3). In order to see how well this works we plot $D_{\text{LET}}/D_N * \lambda_L/\lambda_N$, which should give results close to one for all connectivities. This can be studied in figure 5.4 averaged over connectivity. The approximation works rather well,
Figure 5.3: The figure shows the ratio between network delay for LET and network delay for the modified LET. The ratio is plotted for networks of different size 10 and 20 nodes.

Figure 5.4: The figure shows the ratio between network delay for the modified LET and network delay for node assignment multiplied by the ratio between the throughput of link assignment and the throughput of node assignment. This should be approximately equal to one. This is plotted for networks of different size; 10 and 20 nodes.
5.1. Double Priority

Figure 5.5: The figure shows the ratio between delay for the modified LET and delay for node assignment with unicast traffic. The ratio is plotted for 500 networks of size 20 nodes.

especially for the 10-node networks.

Finally, for unicast traffic we compare the modified LET with node assignment. In figure 5.5 we can study this for networks of size 20 nodes. As can be seen, there is a considerable variance over the networks, but most networks have lower delay when we are using LET. In figure 5.6, we average $D_{\text{LET}}/D_N$ over connectivity. This average stays below one for all connectivities and network sizes, i.e. for unicast traffic LET always performs better on average than node assignment.

5.1.2 LET for broadcast traffic

For broadcast traffic, the variance of traffic over the outgoing links of a node is smaller, making the method of spreading the time slots of a link easier.

Since the double priority method for LET was developed for unicast traffic, it is interesting to see how it behaves compared to the normal LET. Therefore the first parameter studied here is the ratio between delay of the modified LET and normal LET.
In figure 5.7 this is plotted for networks of size 20 nodes. We can see that for most networks the modification gives a considerable improvement, especially for medium connectivity networks. We can conclude that the modification works well also for broadcast traffic. (Even better than for unicast)

We continue by studying the most interesting parameter, i.e a comparison between LET and node assignment. In figure 5.8 $D_{\text{LET}}/D_N$ is plotted for 500 networks of size 20 nodes. As can be seen, LET gives a lower delay for nearly all networks, except for the full connectivity networks.

In figure 5.9, this is averaged over connectivity for networks of different size. It can be concluded that also for broadcast traffic, LET gives, in average over the simulated networks, lower network delay than node assignment, at least for low traffic loads.
Figure 5.7: The figure shows the ratio between delay for the modified LET and delay for LET with broadcast traffic. The ratio is plotted for 500 networks of size 20 nodes.

Figure 5.8: The figure shows the ratio between delay for the modified LET and delay for node assignment with broadcast traffic. The ratio is plotted for 500 networks of size 20 nodes.
5.1.3 Conclusions and Further work

In this chapter we have briefly introduced how network performance is affected by the link-assignment algorithm we base LET on. The use of double priority gives very efficient schedules for a fully connected network. However, for low connectivity it is a less efficient way of spreading time slots than the single priority method used in previous chapters. More research is necessary if we want to fully exploit the advantages of LET.

The double priority method is a good start, though. For most networks it gives an improvement compared to node assignment in terms of delay both for unicast traffic and broadcast traffic. On the average over the simulated networks, the modified LET gives lower network delay than node assignment.

The result of this is that LET is the preferable assignment method for all networks except for networks with a high traffic load of mainly broadcast traffic.

Figure 5.9: The figure shows the ratio between delay for the modified LET and delay for node assignment with broadcast traffic. This is plotted for networks of different size 10 and 20 nodes.
Chapter 6

Model Comparison

In this section we investigate the loss of efficiency (in terms of throughput and delay) when the STDMA algorithm only has knowledge about the two-level graph model of the network compared to having full knowledge of the attenuation between all pairs of nodes. This is done for conflict-free scheduling with link assignment since this is the base for LET.

6.1 Graph-based Network Model

The traditional approach in designing reuse schedules is to use a graph model of the network. Given a graph, a reuse schedule can be obtained by studying the set of edges. One problem with this approach is that, depending on how the graph is chosen, it may result in schedules with serious interference in terms of SIR.

In the graph-based method, a graph representation $G_\gamma$ is chosen. In order to represent the radio network as a directed graph, we denote by $G$, the directed graph that is obtained by defining the set of nodes $V$ as vertices and the set of edges $E$ as follows

$$(i, j) \in E \text{ if and only if } \Gamma_{ij} \geq \gamma,$$

i.e. the set of edges is the set of node pairs with SNR not smaller than $\gamma$.

The schedule is then designed from the graph $G_\gamma$. Interferences from other nodes are not taken into account. The traditional method for node
assignment given a set of edges is to say that two nodes \( v_i \) and \( v_j \) can use the same time slot if and only if:

- edge \((i, j) \notin E\) and edge \((j, i) \notin E\), and
- there is no \( v_k \) such that \((j, k) \in E\) and \((i, k) \in E\)

The first criterion is based on a node not being able to receive and transmit simultaneously in the same slot. The second criterion is that a node cannot receive a packet from more than one node in the same slot. These criteria translate to the condition that nodes must be at least two hops away from each other in order to be scheduled to the same slot. For a more precise description of this problem see \([16]\).

Observe that the above criterions are not sufficient to guarantee that the assignment is conflict-free in terms of SIR, as defined in the previous section. The assignments that fulfill the above two criteria do not necessarily fulfill the SIR criterion given in condition (3.2). They may therefore not be able to transmit simultaneously according to our definitions. We illustrate this with a small example.

In figure 6.1 we see the edges obtained for a sample network by choosing the threshold \( \gamma_C \) to be 13 dB.

Now, assume that links (2,4), (7,5) and (8,9) have been assigned the same time slot. This is possible according to the graph model of the network. If all of these nodes transmit at the same time, then the SIR calculated at node 5 will be only 1.6 dB. This is because the SNR between nodes 5 and 8 is just below what is needed for communication and the SNR between 5 and 7 is just above.

From the example we see that the graph approach if applied as above will result in serious interferences.

However, graph-based algorithms can still be useful. One method of avoiding the serious interference levels shown in the example above is to base the schedule on a graph where also node pairs with SNR less than \( \gamma_C \) are included as interference edges \([31]\). The edges with SNR lower than \( \gamma_C \) will not be assigned time slots, but only be used in the test criterion. By considering a graph \( G_\gamma \) and letting \( \gamma \) take a value \( \gamma_l \) smaller than \( \gamma_C \), the set of edges will contain not only the links but also interference edges, which represents the case when the signal from one
user is too weak to be used for communication but is still strong enough to interfere. We will call $\gamma_I$ the *interference threshold*.

However, all transmissions in the time slot will add to the interference. The choice of $\gamma_I$ thereby determines the remaining interference. By choosing the threshold for a communication link, $\gamma_C$, slightly greater than what is needed for reliable communication, $\gamma_R$, we assure that the communicating link can handle these remaining interferences. The following example illustrates this procedure.

Consider our previous example. Figure 6.2 shows the $G_\gamma$ with $\gamma$ chosen to be 7 dB. The interference edges obtained are illustrated with dashed lines. With this graph description we can see that the links (2, 4), (7, 5) and (8, 9) cannot share the same slot. This will not be allowed since there are edges (2, 5) and (5, 8), which means that interference violation in node 5 will be avoided.

In some cases the interference edges will also prevent assignment of nodes that can be assigned to the same slot without violating the SIR
Figure 6.2: The graph $G_\gamma$ obtained for $\gamma = 7$ dB. The interference edges are indicated with dashed lines.

criterion. One such example is links $(10, 5), (2, 1)$ and $(8, 6)$. These three links will not be allowed to share a slot since there is an edge $(2, 5)$ and an edge $(8, 5)$. However, the SIR values on all these three links are above 10 dB. This is because the signal levels on these links are so strong that quite strong interferences can be accepted.

The graphs $G_\gamma C$ and $G_\gamma I$ with a properly chosen $\gamma C$ and $\gamma I$ can now be used to generate a reuse schedule with any assignment algorithm taking a graph as a network model.

### 6.2 Evaluation and Results

Since we study conflict-free schedules, we have to generate these based both on the SIR model and the two-level graph model with appropriate SIR. Remember from section 2.4 that a schedule is considered to be conflict-free if the SIR is above a threshold $\gamma R$. However, it is difficult to design a graph-based schedule to a certain minimum value of $\gamma R$, since
given a $\gamma_R$ we have to find appropriate values of $\gamma_C$ and $\gamma_I$ such that the resulting SIR is as close to $\gamma_R$ as possible.

If we have one strong interferer with an interference that lies just below the interference threshold, the resulting SIR will at least have the value

$$\frac{\gamma_C}{1 + \gamma_I}.$$ 

A simple choice here is to set $\gamma_I$ to 0 dB. Then all remaining interferences will be weaker than the receiver noise. If, for example, we choose $\gamma_C$ to be 10 dB, our single interferer will give us a resulting SIR of 7 dB. For more than one strong interferer, we can get a lower resulting SIR. Of course, this is under the assumption that the communication link has a SNR close to the threshold $\gamma_C$.

Now we can see that a very low value of $\gamma_I$ (below 0dB) will, even in a worst case scenario, result in a SIR close to to the communication threshold. But this will also result in very many interference edges in the graph and thereby a very low spatial reuse. On the other hand, a high value of $\gamma_I$ (close to $\gamma_C$) can result in very low SIR, even below 0dB independent on the choice of $\gamma_C$, although it allows for high spatial reuse.

Without any further investigations we will use 10dB as the communication link threshold and 0dB as the interference threshold.

Since we have no perfect way of designing a graph-based schedule so that the minimum SIR becomes $\gamma_R$, the choices of $\gamma_C$ and $\gamma_I$ have to be made such that we are certain that the resulting SIR values are equal to or higher than $\gamma_R$, thereby most often achieving resulting SIR values that are much higher than necessary (with of course, the corresponding loss in spatial reuse).

SIR-based scheduling, on the other hand, can set its target SIR to $\gamma_R$ and achieve a resulting SIR that is very close to its target. However, since we have not investigated the choices of $\gamma_C$ and $\gamma_I$, we ignore this fact and assume that appropriate values of $\gamma_C$ and $\gamma_I$ can be found to achieve the desired $\gamma_R$ in the comparison.

In order to do this, we will use $\gamma_C$ and $\gamma_I$ to be 10dB and 0dB, respectively, in the generation of the graph-based schedule and use the resulting
minimum value of SIR over all time slots for $\gamma_R$ when we are generating the SIR-based schedule.

In figure 6.3, the ratio between the network delay $D$ for SIR-based schedules and the network delay $D_g$ for graph-based schedules is plotted. In figure 6.4 the corresponding ratio $\lambda/\lambda_g$ for the maximum throughput is plotted. The graph-based schedules perform worse than the SIR-based schedules. For example, almost a third of the maximum throughput is lost in some cases.

To motivate why this is the case we plot the distribution of the minimum SIR of the time slots for both schedules for one of the networks. As can be seen, the algorithm using the SIR-model manages to schedule its slots to a resulting SIR much closer to $\gamma_R$ than an algorithm using the graph model. Since the graph-model has much less information about the network, it has to behave more carefully in its assignment of the time slot.
6.3. Conclusions and Comments

Our simulation results indicate that the network performance of the graph-based scheduling may suffer compared to the interference-based scheduling, depending on the SIR thresholds.

In order to achieve collision-free schedules, up to one third of the network capacity can be lost with the more careful graph-based scheduling due to limited information about the network. No clear dependence of the network connectivity has been observed.

Since the interference-based scheduling is very complex, it is difficult to use in distributed scheduling algorithms, which means that some form of model with limited information must be used. On the other hand, for the graph-based scheduling, more investigation is necessary about how to determine the thresholds used in the scheduling, as we might otherwise lose too much of the network capacity. In the stationary or temporary stationary case, where the knowledge of the full interference environment can be assumed, the SIR model offers sufficient improvement to be worth its increased complexity.
Figure 6.5: The figure shows the distribution of the minimum SIR of the time slots both for a graph-based schedule (dashed) and for a SIR-based schedule.
Chapter 7

Conclusions

In this thesis we have studied the different assignment strategies used for STDMA. The first comparison was between the two most commonly used assignment methods, i.e. node and link assignment. We conclude that, in the unicast case, link assignment behaves better for high traffic loads, achieving a higher throughput. However, this comes at a cost of higher delay than node assignment for low traffic loads. This shows that for unicast traffic there is a certain traffic arrival rate for which node and link assignment are equal good in terms of delay. Our simulation results show that only the size and the connectivity of the network are necessary to determine this traffic arrival rate, thereby making it possible to generally determine which assignment method one should use for a specific network.

For broadcast traffic, node assignment is always preferable. Basically, for heavily loaded networks with mainly unicast traffic, link assignment is to be preferred. If we have a considerable part which is broadcast traffic, we should use node assignment.

In order to improve the behavior of the STDMA network we have proposed a novel assignment strategy with the advantages of both assignment strategies. The strategy proposed is based on a link schedule, but where transmissions rights are extended. This results in an assignment method that behaves as node assignment for low traffic loads and link assignment for high traffic loads.
We have also shown that there can be significant benefits from designing the algorithms that generate the link schedules so that they take better advantage of the LET strategy.

From this investigation, we can conclude that LET is the preferable assignment method for all networks except for networks with a high traffic load of mainly broadcast traffic.

Finally, we have studied the loss of efficiency in terms of throughput and delay when the STDMA algorithm does not have the full knowledge of the attenuation between all pairs of nodes, but instead only has knowledge of the two-level graph model of the network. This is important since STDMA will have its biggest use in a mobile scenario, in which case knowledge about the full interference environment cannot be assumed.

The results here indicate that the network performance of the graph-based scheduling may suffer compared to the interference-based scheduling, depending on the SIR thresholds. Up to one third of the network capacity can be lost with the more careful graph-based scheduling in order to achieve collision-free schedules, due to limited information about the network.

Further work in this area will be necessary if we want to design efficient STDMA schedules, not to mention a distributed version of the STDMA algorithm.
Appendix A

Simulation Model

A.1 Simulation Setup

Due to the relaying of packets the statistical properties of the behavior of the network is complicated, and an exact analytical analysis of the expected delay is difficult. Therefore computer simulations will be used in order to determine the network delay [7]. Here we give a more detailed description of the simulation setup used.

We will first describe how the networks are generated and then how the basic path loss between all nodes are calculated. We will also discuss the specific routing and STDMA algorithm that we will use.

A.1.1 Generation of networks

All networks used in the comparison are generated in the following way; \( N \) nodes are spread out uniformly in a sample terrain consisting of mixed meadows and forest. The center frequency is chosen as 300 MHz, and antenna heights 3 m. Both communication threshold and reliable communication threshold are set at 10dB.

In the comparisons 500 networks of size 10, 20, and 40 nodes have been generated with different connectivity. The connectivity is varied by changing the transmission power for a network. The transmission power is always set sufficiently high so that it gives a connected network.
A.1.2 Link Gain

An essential part of modeling an on-ground or near-ground radio network is to model the electromagnetic propagation characteristics due to the terrain variation. A common approach is to use the basic path-loss $L_b$ between two nodes. The most simple assumption concerning the wave propagation is that no obstacles appear between the transmitting and receiving antennas, and no reflection or diffraction exist in the neighborhood of the path between the receiving and transmitting antennas. This model is the so-called free-space assumption.

Refined assumptions of wave propagation conditions near ground include terrain-height information and terrain-type information to estimate the basic path-loss. See Parsons’ [32, Sec. 2.3] for an introductory description of the problem. In the digital terrain database we have used, the height is represented as terrain-height samples at equidistant square lattice intersection points, and the terrain type is one of maximum 256 terrain types such as fresh water, salt water, forest, wet ground, etc. We assign electromagnetic ground constants such as relative dielectric constant $\varepsilon_r$ and conductivity $\sigma$ for each terrain type, and we also assign surface roughness for terrain types. Our database has a 50-meter grid for the terrain-height samples and a 25 meter by 25 meter terrain-type area.

All our calculations are carried out using the wave propagation computations library DetVag-90®, [33]. Here, we use the multiple knife-edge model of Vogler [34] with five knife edges. Currently, DetVag-90® works with narrow-band signals, i.e., the bandwidth of the transmitted signal should be of percentage order of the carrier frequency.

With the assumption of omnidirectional antennas, the link gain $G$ will be equal to $1/L_b$.

A.1.3 Acknowledgment scheme

Although STDMA scheduling can accomplish the target SIR, this does not necessary result in all packets being perfectly received, especially if the target SIR is set low. The noisy channel can cause more errors than can be handled by Forward-Error Correction (FEC). By using part of the error-correction capability one can instead use the code to detect more
errors, and use an acknowledgment scheme to retransmit the packet. This can be done both on an end-to-end level and on the link level. Which one is used is usually dependent on the packets-loss ratio.

In our simulations we will assume that for collision-free scheduling, i.e. the target SIR is achieved, we assume that the Forward-Error-Correction (FEC) can handle all errors on the channel and that we will have no undetected errors. In these cases no retransmissions will be considered.

For the strategies that are not collision-free a perfect feedback channel with no delay is assumed. The only strategy of this type is LET, described in chapter 4. All the rest are collision-free strategies.

This type of feedback channel is not very realistic, but as will be seen, LET will be collision-free for high traffic loads, and the effect of this acknowledgment scheme is small for very low traffic levels when no collisions occur. Since high traffic load and low traffic load will be the interesting areas in this investigation, the effect of this assumption is only a simplification of the simulations. In a realistic scenario, more complex acknowledgment schemes should be used since STDMA assumes safe transfer of packets over the links.

For example could an acknowledge scheme similar to the one used in TCP be used [35].

A.1.4 Traffic model - end to end

In our simulations, we assume that packets are generated by a pseudo-Poisson process with intensity \( \lambda \) and with a uniform traffic distribution.

We choose Poisson traffic for simplicity and because we want to compare the simulation results with the approximations which assume this traffic model.
Bibliography


85


