FOI-R- -1338- -SE
September 2004
ISSN 1650-1942
Technical Report

Jimmi Grönkvist

Overhead Traffic for Distributed STDMA Algorithms

Command and Control Systems
P.O. Box 1165
SE-581 11 LINKÖPING
SWEDEN

FOI- SWEDISH DEFENCE RESEARCH AGENCY
Command and Control Systems
P.O. Box 1165
SE-581 11 LINKÖPING
SWEDEN

FOI-R- -1338- -SE
September 2004
ISSN 1650-1942
Technical Report

Jimmi Grönkvist

Overhead Traffic for Distributed STDMA Algorithms

Issuing organization FOI- Swedish Defence Research Agency Command and Control Systems P.O. Box 1165 SE-581 11 LINKÖPING SWEDEN	**Report number, ISRN** FOI-R- -1338- -SE	**Report type** Technical Report
	Research area code 4. C^4ISTAR	
	Month year September 2004	**Project No.** E7035
	Customers code 5. Commissioned Research	
	Sub area code 41. C^4I	
Author/s Jimmi Grönkvist	**Project manager** Mattias Sköld	
	Approved by Sören Eriksson	
	Sponsoring Agency Swedish Armed Forces	
	Scientifically and technically responsible Jan Nilsson	

Report title

Overhead Traffic for Distributed STDMA Algorithms

Abstract

Spatial reuse TDMA (STDMA) has been proposed as an access scheme for multi-hop radio networks where real-time service guarantees are important. The idea is to achieve high capacity by letting several radio terminals use the same time slot 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. Due to the mobility of the users, this schedule must be constantly updated. 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 have expanded our earlier work on distributed STDMA.

We have now added methods for conveying the appropriate amount of information, described what information each node should send to its neighborhood, as well as discussed how the nodes will be able to control their local neighborhoods.

We have also described which parameters that can be varied in order to give good appropriate balance between schedule capacity and overhead traffic.

Keywords

STDMA, ad hoc networks, multi-hop networks, MAC, distributed algorithms

Further bibliographic information	**Language** English
ISSN 1650-1942	**Pages** 40 p.
	Price acc. to pricelist

Utgivare Totalförsvarets Forskningsinstitut-FOI Ledningssystem Box 1165 581 11 LINKÖPING	**Rapportnummer, ISRN** FOI-R- -1338- -SE	**Klassificering** Teknisk Rapport
	Forskningsområde 4. Ledning, informationsteknik och sensorer	
	Månad, år September 2004	**Projektnummer** E7035
	Verksamhetsgren 5. Uppdragsfinansierad verksamhet	
	Delområde 41. Ledning med samband och telekom och IT-system	
Författare Jimmi Grönkvist	**Projektledare** Mattias Sköld	
	Godkänd av Sören Eriksson	
	Uppdragsgivare/kundbeteckning Försvarsmakten	
	Teknisk och/eller vetenskapligt ansvarig Jan Nilsson	
Rapportens titel Overhead Trafik för distribuerande STDMA-algoritmer		

Sammanfattning

Spatiell TDMA (STDMA) är en accessmetod för radionät med flerhoppsfunktion där realtidstjänster är viktiga. Grundidén är att låta flera radioterminaler använda samma tidlucka när de är så långt från varandra geografiskt att interferensen mellan dem är tillräckligt låg. Sändningsrättigheterna för terminalerna beskrivs med ett schema. Mobiliteten gör att detta schema kontinuerligt måste uppdateras. För att göra detta praktiskt måste dessa uppdateringar göras parallellt i olika delar av nätet med bara lokal information, dvs distribuerade STDMA-algoritmer är nödvändiga.

I den här rapporten har vi utvidgat vårt tidigare arbete på distribuerad STDMA. Vi har i första hand lagt till metoder och funktioner för att överföra lämpliga mängder information mellan noderna och beskrivit exakt vilken information och till vem varje nod ska sända informationen i olika situationer.

Vi beskriver också vilka parametrar som kan varieras för att skapa en bra balans mellan skapade scheman och overhead trafik.

Nyckelord

STDMA, ad hoc-nät, multihopnät, MAC, distribuerade algoritmer

Övriga bibliografiska uppgifter	**Språk** Engelska
ISSN 1650-1942	**Antal sidor:** 40 s.
Distribution enligt missiv	**Pris: Enligt prislista**

Contents

Chapter 1

Introduction

1.1 Background

In many military scenarios it is not possible to rely on a fixed communication infra-structure for wireless communication, instead self-configurable networks must be deployed quickly.

Common features of these networks are that they are not pre-planned, and area coverage is achieved by letting the radio units relay the messages, i.e. multihop networks. These kind of networks are often referred to as ad hoc networks. One of the most challenging problems in ad hoc networks is to guarantee Quality of Service (QoS), especially delay guarantees.

An interesting medium access control protocol that has great potential for QoS is spatial reuse TDMA (STDMA) [1], which is an extension of TDMA. In STDMA the capacity is increased by spatial reuse of the time slots. An STDMA schedule describes the transmission rights for each time slot. Different algorithms for generating STDMA schedules have been proposed, see e.g. [2–4].

However, a problem in ad hoc networks is that the nodes will be moving, and a schedule that is conflict-free at one moment will probably not be that later.

Therefore, the STDMA schedule must be updated whenever something changes in the network. This can be done in a centralized manner, i.e. all information is collected into a central node which calculates a new schedule [4, 5]. Unfortunately, for fast moving or large networks this is usually not possible — by the

time the new schedule has been propagated to all nodes it is already obsolete due to node movements.

Another way to create STDMA schedules is to do it in a distributed manner [2,3,6], i.e. when something changes in the network, only the nodes in the local neighborhood of the change act upon it and update their schedules without the need to collect information into a central unit.

1.2 Previous Work

A lot of work has been done on distributed STDMA, but all existing algorithms have been designed with the purpose of giving an acceptable solution, rather than a solution that handles the channel as efficiently as possible under different situations.

A more systematic approach to the design of STDMA algorithms is lacking. We know that the more information about the network we have the better schedules we can create, thereby increasing the total capacity of the network. However, increasing information also increases the overhead. This means that the amount of information the algorithm has about the network should vary depending on the situation.

The use of interference-based scheduling can give us the means to vary the amount of information. In report [7] we described the first distributed algorithm that uses interference information.

However, that report focused on which information a node needed in order to create efficient schedules and how much capacity such schedules could give us rather than to study how such information can be conveyed and what overhead traffic this would result in.

This of course gave some obvious results, e.g. the more knowledge we have about the network the better schedules we create (that is higher capacity). In this report we will study another side of the problem, i.e. not what we can do with a certain amount of information but how to convey it and how much overhead this will cost. We will describe when information should be transmitted and to whom.

1.3 Delimitation

Also in this report we need to limit the study somewhat. Interference-based scheduling is much more complicated than the usually used graph-based scheduling [8]. A complete algorithm that can directly be implemented, that can handle multiple concurrent events, error prone channels and other complicated network behavior will be left for future work.

Therefore we will make some simplifications that in a fully realistic scenario may cause problems or even make the network end up in deadlocks if further additions to the algorithm is not added.

- Exact queuing behavior of overhead messages will not be studied. We assume that there is no delay of the control messages, we will thereby only study a capacity limited system which have sufficient with capacity to update a schedule before the next event occurs.

- Messages are always delivered in the order they were sent and no are lost. This could be implemented by adding sequence numbers and retransmitting lost packets but in this report such methods will only be used for other reasons by overhead messages.

A problem with interference-based scheduling that does not exist for graph-based scheduling is that some of the parameters used are not discrete variables, an example of such a parameter is path-loss. Such variables may take any value and can change continuously. At least one such variable (maximum interference) must be sent to a node's neighbors. An exact representation of such a variable cannot be sent, therefore it needs to be sampled, both in time and in value. (If we use measurements the variable may be discrete in time, but probably with to great rate of change for us to use directly.)

A problem is therefore which sample rate and number of possible levels of description we need. Time is the most critical parameter. This is because we only need to increase the level value with one bit to double the accuracy, while the sample rate must be doubled to do the same in the time plane, thus doubling the overhead.

Appropriate choices for these values must be decided for the algorithm to work well, but this will be left until a further evaluation of the algorithm by simulations.

Chapter 2

Network Model

This chapter introduces the network model we use and the assumptions required. We start by describing the interference-based model of a radio network. This model gives a realistic description of when users can transmit on the channel without conflicts. Traditionally for ad hoc networks, the most frequently used model is graph-based. Such a model is much simpler to use, but its ability to describe the communication channel is not very good.

2.1 Interference-Based Scheduling

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), Γ_{ij}, as

$$\Gamma_{ij} = \frac{P_i G(i,j)}{N_{\mathrm{r}}}, \tag{2.1}$$

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_{r} 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*, γ_{C}. That is, the set of links in the network, $\mathcal{L}$, is defined:

$$\mathcal{L} = \{(i,j) : \Gamma_{ij} \geq \gamma_{\mathrm{C}}\} \ . \tag{2.2}$$

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

$$V_{\mathrm{T}}(L) = \{v_i : (i,j) \in L\} \ .$$

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

$$I_L(i,j) = \sum_{v_k \in V_{\mathrm{T}}(L) \setminus v_i} P_k G(k,j). \tag{2.3}$$

Furthermore, we define the *signal-to-interference ratio* (SIR):

$$\Pi_L(i,j) = \frac{P_i G(i,j)}{(N_{\mathrm{r}} + I_L(i,j))}. \tag{2.4}$$

We assume that any two radio units can communicate a packet without error if the SIR is not less than a *reliable communication threshold*, γ_{R}. A schedule S is defined as the sets X_t, for $t = 1, 2, \ldots, T$, where T is the period of the schedule. The sets X_t contain the nodes or links assigned time slot t. A schedule is called conflict free if the SIR is not less than the threshold γ_R for all receiving nodes in all sets X_t.

However, due to mobility and limited information conflict-free schedules are very difficult to create and uphold. In order to make comparisons for distributed scheduling we assume that links that have a lower SIR than γ_{R} in a time slot can decrease its data rate as compared to the nominal data rate R_N used by links with SIR above γ_{R}, i.e.

$$\frac{SIR}{\gamma_{\mathrm{R}}} = \frac{\text{Used Rate}}{R_{\mathrm{N}}}, \tag{2.5}$$

Furthermore, 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. For simplicity we assume a fixed transmission power and omni-directional antennas. However, the use of power control and directional antennas is quite easy to include in interference-based scheduling compared with graph-based scheduling. This is important, since it has been shown that adaptive antennas can have a vast improvement on network capacity [9].

2.2 Node and Link Assignment

Traditionally, there are two different assignment methods for STDMA.

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. In a *link-assigned* schedule, the directed link is assigned a slot. A node can then 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 network throughput [10] (at least for unicast traffic).

In this report we will concentrate on link assignment, often referred to as link activation. This is mainly done because it more intuitively (and efficiently) can handle advanced nodes with abilities like power control and adaptive antennas. Furthermore, using link assignment we can extend the transmission rights (LET – Link assignment with Extended Transmission rights), which gives huge improvements [11].

In the following, we describe the criteria needed 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 link $(i, j) \in L$ iff $\{i, j\} \cap \{k, l\} \neq \emptyset$. 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.

The assumptions 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, can also be described as that a set of links L and the set of its adjacent links $\Psi(L)$ must be disjoint:

$$L \cap \Psi(L) = \emptyset. \tag{2.6}$$

The signal-to-interference criteria (2.4) gives the following condition

$$\Pi_L(i, j) \geq \gamma_{\mathrm{R}} \ \forall \ (i, j) \in L. \tag{2.7}$$

If the above two conditions, (2.6) and (2.7), hold for a set of links $L \in \mathcal{L}$, we say that the links in L can *transmit simultaneously* at full data rate .

We will say that the schedule generated by a distributed STDMA algorithm is conflict-free if equations (2.6) and (2.7) are valid for all links in any time slot.

2.3 Traffic in Multi-hop Networks

The relaying of traffic in multi-hop networks causes a considerable variation of the traffic load on the links. To achieve large capacities, efficient traffic-controlled schedules have to compensate for this problem.

In a traffic-controlled schedule, links or nodes can use several slots, see [12], according to the traffic load. We define h_{ij} as the number of slots allocated to link (i, j) within a frame in a schedule.

In our traffic model we assume point-to-point traffic, i.e. a packet entering the network has only one destination. However, it is easy to expand the traffic model to other forms of traffic, e.g. multicast traffic.

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 λ 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 cannot be calculated until the traffic has been routed.

Now, let R denote the routing table where the list entry $R(v, w)$ at v, w is a path from entry node v to exit node w. Let the number of paths in R containing the *directed* link (i, j) be equal to Λ_{ij}.

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

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

The maximum traffic load giving bounded packet delay is commonly referred to as the throughput of the network. We define the throughput as the

number λ^* for which the following expressions hold for all traffic loads λ

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

The maximum throughput for a link assigned schedule can be written as [11]

$$\lambda_{\mathrm{L}}^* = \min_{(i,j)} \frac{N(N-1)h_{ij}}{T_{\mathrm{L}}\Lambda_{ij}}, \tag{2.8}$$

where T_{L} is the length of the link-assigned schedule, i.e. the number of time slots in the schedule.

2.4 Routing assumptions

We assume that the routing is perfect and we ignore any necessary overhead caused by routing traffic. This also means that the routing protocol is assumed to be useful for the multicast transmissions that the STDMA algorithm needs. However, we also assume that the routing protocol reacts to link changes slower than the MAC layer is capable of handling, which means that new multicast routes is available only when the MAC protocol has figured out its new local neighborhood after a change.

This means that between link changes the STDMA algorithm may use the routing protocol for its updates regarding assignment of slots. But when a change takes place, the STDMA algorithm must first figure out its new local neighborhood with the old routing information before the routing protocol adapts to the change.

The routing protocol in each node will have a better picture on how the whole network looks like than the local picture from the STDMA algorithm in the node, but the routing protocol will be locally updated through the MAC layer.

2.5 Interactions with the Link Layer

The network model we have described in this chapter is somewhat simplistic (although more complex than a graph model). In reality, all packets will not be

perfectly received if the SIR is above γ_C, and all packets will not be lost just because SIR is below the threshold. If the sent packets had infinite size, such assumptions would be more accurate.

In addition, exact path gains will not be available and fast and slow fading will complicate the situation even further. An exact representation on the details of the link is difficult to obtain and even if we could get such information from the links in the network, an STDMA algorithm could not handle such fast changes.

Instead, we have to hide the volatility of the link from the MAC protocol. The link gain given from lower layer will be an expected average. Variable data rates for the link (as seen from STDMA point of view) must be average data rates, not what actually is transmitted on the link. An adaptive radio node may actually change data rate on the link from one time slot to the next (in extreme cases we could even change it within the time slot).

The purpose of the STDMA algorithm is to create sets of simultaneously transmitting links that can be efficiently handled locally by the links (without any further interaction between them in terms of transmitted information).

Nevertheless, the actual functionality of the link will be ignored in this report and we will concentrate on the functionality of STDMA when the details of the link layer already have been hidden.

2.6 Traffic Estimations

A similar issue is traffic from upper layers. In order for STDMA to work well we will need to have accurate information on traffic loads in order to compensate for these different traffic loads. However, such information is not always available. Sometimes the applications may give some information and sometimes we have to do estimations from the arriving traffic and existing queue lengths.

Such estimations may vary quite a lot over time and also this information may have to be hidden from the STDMA algorithm. How to do traffic estimations efficiently is a research area in itself and not so much is yet done for ad hoc networks. But since this is also out of the scope of this report we will assume that the actual expected traffic loads are known, as was calculated in section 2.3.

Chapter 3

A distributed STDMA algorithm

In this chapter we will give a description of the algorithm presented in [7]. In this form the algorithm does not care about how information is obtained but it will only act on what is available. In the next chapter we will discuss how the nodes obtain such information.

3.1 How to create a schedule

In short, the distributed STDMA algorithm can be described by the following steps:

- Nodes that have entered the network exchange local information with their neighbors.
- The link with highest priority in its local surroundings assigns itself a time slot (this is done in the receiver).
- The local schedule is then updated, and a new link has highest priority. This process is then continued until all slots are occupied.

We will include traffic sensitivity through the link priorities, i.e. a link that needs many time slots will have high priority more often than a link with low priority.

The algorithm is interference-based since it uses interference information, i.e. interference information is transmitted and used when the algorithm decide

when links can transmit simultaneously. We will use the term *local neighborhood* of a link (i, j) to mean those links that will be taken into consideration when a link determines whether it can transmit simultaneously with all other assigned links. Links outside the local neighborhood will not be considered and therefore no information about these links is assumed. Exactly how large this neighborhood is will be discussed later.

In the following we will assume that each link has a given schedule length T. This length is not necessarily the same length in all parts of the network and may change over time. But this will not change the basic scheduling process.

The STDMA algorithm is run in parallel for each link, i.e. each link can be seen as a separate process which will be run at the receiving node of the link, which means that each node will run a process per incoming link. These processes can be in three modes: active, waiting, or asleep.

- *Active*: In this mode, the link has the highest priority in its local neighborhood and will subsequently assign itself a time slot. Which time slot is chosen if more than one is available will be discussed later. A link process can be in this mode because of the existence of unused slots or because the link's share of the time slots in its local neighborhood is too low. In the latter case, it can steal time slots from another link. We will describe later under which situations this may be permitted. Information about which time slot is chosen and the link's new priority will be transmitted to all nodes in the local neighborhood of the link. After this, the link process can stay in this mode or change into one of the others.
- *Waiting*: In this mode, a link wants to assign itself a time slot, but another link has higher priority. The link will wait its turn. However, since time slots are taken by active users, the link may change into asleep mode instead if all time slots are taken and the link does not have the right to steal slots.
- *Asleep*: In this mode, there are no available slots for the link and it simply waits for a change of the network, either in topology or in traffic levels.

3.1.1 Link Priority

Link priority decides in which order the links may attempt to assign themselves a time slot. The link priority can depend on many things, but the most important

will be the number of time slots the link is assigned, h_{ij}, and the traffic of the link, Λ_{ij}. Since both these values are changing, the link priority is constantly changing.

The priority value of a link (i, j) will be $\frac{h_{ij}}{\Lambda_{ij}}$, where the lowest value has the highest priority. The motivation for this comes from the maximum throughput formula (2.8). The links with the highest priority (lowest value) will be the links which limit the maximum throughput of the network. The network throughput will not rise above zero until all links with traffic levels above zero have received at least one time slot. An obvious consequence of this is that all links with traffic in a local neighborhood will receive at least one time slot before any of the links receive more than one slot. This will be the case, for example, when a new schedule is initiated.

3.1.2 Theft of Time slots

Sometimes, the relative traffic levels will change in a local area (or other changes may take place). This will result in a situation where a link has a smaller proportion of the time slots than its priority value merits. If there are free slots, the link may assign itself slots until it is on a similar priority level as its surrounding links. However, if no time slots are free, the link can sometimes steal time slots from other nodes.

The policy for assigning time slots in the case of free time slots is always that the link which limits the throughput will be the one that receives an extra time slot. This is also the case when a link is permitted to steal a time slot. When stealing a time slot, the local network throughput must increase.

This means that a link (i, j) is only permitted to steal a time slot from another link (k, l) if the priority value of the stealing link is lower than the other link's priority value *after* the loss of a time slot, i.e.

$$\frac{h_{ij}}{\Lambda_{ij}} < \frac{h_{kl} - 1}{\Lambda_{kl}}.$$

The theft of time slots means that it is the link with too few time slots that reacts to unfairness situations. This is a better solution than if links that have too many time slots would give them up since such a link can never know if the time slot can be used by the link with too few time slots. The time slot can always be blocked by another node further away.

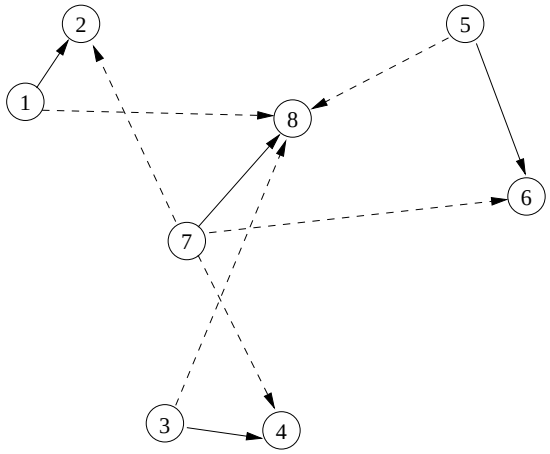

Figure 3.1: A small part of a network.

3.2 What information does a node need?

An important issue is to decide exactly what information the algorithm needs in order to do the scheduling.

Two things must be fulfilled if a link can be allowed to transmit in a time slot.

First, the receiver must have sufficiently low level of interference from the assigned transmitters. This is calculated in the receiver.

Second, the interference from the transmitter is not allowed to cause interference problems in any of the other receivers already assigned time slots. This is calculated in the sender and then the information is sent to the receiver (since it does the actual scheduling) as a variable called $AVTS$ (available time slots to send).

The problem can be illustrated by the example shown in Figure 3.1. In this case links $(1, 2), (3, 4)$, and $(5, 6)$ are assigned to a particular time slot. Now, link $(7, 8)$ wants to be assigned as well. This means that the received power in node v_8, must be sufficiently large compared to noise and the combined interference from nodes v_1, v_3, and v_5. Furthermore, the interference caused by node v_7 in the receiving nodes v_2, v_4, and v_6 must not be so large that any of

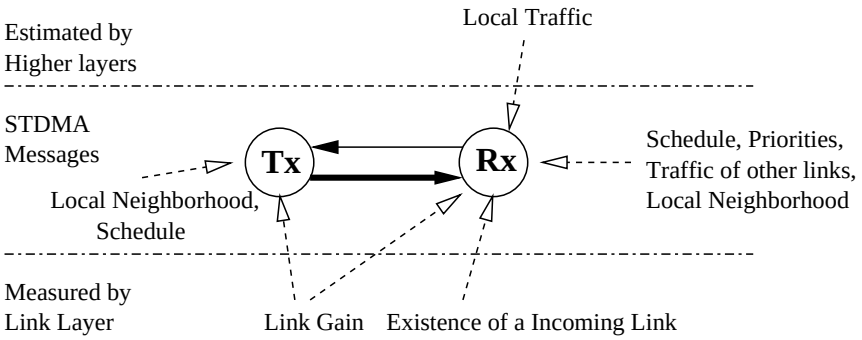

Figure 3.2: The needed information for a node.

these fall below the SIR threshold.

In order to achieve conflict-free scheduling on link (i, j) the receiver v_j need the following information (which also can be seen in picture 3.2):

Interference - Received Power

We need an estimate of the received power from each of the other transmitters, i.e. $P_k G(k, j)$. We assume that the channel is equal in both directions, that is

$$G(k, j) = G(j, k).$$

This assumption can then be used by the transmitter to determine whether its transmission will cause problems for somebody else.

If the received power level from a transmitter is below a value δI, it is assumed to be zero by the algorithm, i.e. the algorithm assume that such nodes do not affect one another. The size of this threshold is given by the interference threshold, i.e. $\gamma_{\mathrm{I}} = \delta I / N_r$.

We assume that this information can be obtained by measurements on the channel and no specific information needs to be transmitted to handle this.

To simplify notation, we say that a pair of nodes v_i and v_j form a *interference link* if the signal-to-noise ratio (SNR) is less than γ_{C} but higher than γ_{I}.

Local Schedule

A node needs to know the local schedule and how much more interference can be handled by the assigned receivers $I_{\max}(l,\tau)$ in each time slot τ, i.e. the largest value of $I_{\max}$ such that the following inequality is still valid

$$\frac{P_k G(k,l)}{(N_{\mathrm{r}} + I_L(k,l) + I_{\max}(l,\tau))} \geq \gamma_{\mathrm{R}}.$$

This information ($I_{\max}(l,\tau)$) is required for node v_i to be able to determine whether its transmission can be handled by the other, already assigned receivers. A link can be assigned the time slot if:

$$P_i G(i,l) < I_{\max}(l,\tau)$$

for all assigned receivers v_l in time slot τ.

The local schedule is also used to determine whether the receiver v_j can handle all existing interference. Measurements of the channel in the specific time slot can be of help when doing this, but is not sufficient since node v_j cannot know whether all assigned transmitters v_k actually are using their time slot. Instead the actual SIR is calculated using the local schedule and received power levels. If the interference levels that are measured are higher than this estimate, we conclude that the we have extra interference from outside the local neighborhood and the interference measured should be used while this lasts.

The local schedule S and all values of $I_{\max}(l,\tau)$ needs to be sent over the channel. However, only information about nodes that v_j can cause interference to or v_j can be interfered by is needed.

Priorities

A node needs to know when it should be active. It also needs to know if a node in the neighborhood is asleep, since such nodes are not considered in terms of priority. The exception to this is the case with theft of time slots, in which case sleeping nodes are also considered. The node needs such information from the entire local neighborhood since we cannot allow two nodes to be active simultaneously in the local neighborhood.

Available Time Slots to Send (AVTS)

The receiver also needs information from the transmitter about which time slots it can assign. This can be represented by a vector of the same length as the frame length or specific information about changes of a single time slot. Only information from the transmitter is needed by the receiver.

With this information the receiver has sufficient information to determine when link (i, j) should be active and which time slots it can assign.

Local neighborhood

Due to mobility the local neighborhood will change constantly and we also need to have accurate information about what the local neighborhood is at the present moment.

Chapter 4

Overhead Cost

In this chapter we will study what information the nodes need to convey to the other nodes in their local neighborhood, how often such messages need to be sent, and finally discuss how much capacity will be lost to this overhead traffic.

4.1 What information must each node send and to which receivers?

In the end of the previous chapter we stated what information a node needs to make a correct decision on whether it can assign a time slot or not. In this section we will look at the problem from the other side and study what information a node needs to send to its local neighborhood so that the rest of the nodes can make correct decisions. Returning to the previous points we can divide them into three regions depending on how far away the information is needed.

Available time slots to send ($AVTS$):

The sender v_i to a link (i, j) must send this information to the receiver v_j, no one else needs the information. This can be seen as region 1 in figure 4.1. This information is sent every time the availability changes for any time slot due to events in the network. However, with omni-directional antennas, whether or not a node can send is independent on which link it attempts to use. This means that $AVTS$ should be equal for all outgoing links and it will normally be sent

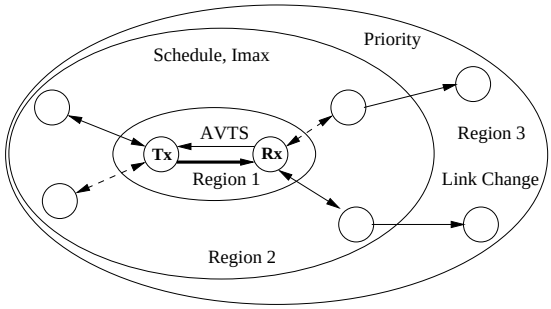

Figure 4.1: The different distances information needs to be transferred.

as a broadcast message to all neighbors. An exception is the concept of theft at which case $AVTS$ can differ for the different links, the message will then be sent to a subset of the neighbors. The distance to send such a message can be seen as region 1 in figure 4.1, or a union of such regions when it is combined to all neighbors. It is still only one transmission though.

Local Schedule and I_{max}

This information is usually updated after an assignment or deassignment of a time slot, although not necessarily by the actual link. We can also have updates due to changes on path gain caused by mobility. This information should be sent to nodes which transmissions can interfere with the receiver of the link and nodes which reception may be affected by the sender.

The local schedule will be sent to the region 2 in figure 4.1, this region consists of all nodes with a link or interference link to the receiver or transmitter. The $I_{\max}$ value can actually be sent to a smaller region than this since only nodes with links or interference links to the receiver needs this information, but for now we ignore this since we assume that they are sent in the same message when an assignment has taken place.

Priority and state

Knowledge about the priority and state of links is needed even further away. The reason for this is that the receiver is in control of the link, which also gives it

control of when the sender to the link transmits. This means that the node needs information also about links that the transmitter can create interference for, in addition to knowledge about receivers to links which transmissions can cause problems for the reception of the link. The sender will only test already assigned slots, it cannot prevent its neighbors from attempting to do assignments. Only if the priority and state of such links are known by the receiver can we prevent that. The range of such updates is shown as region 3 in figure 4.1.

Link updates

Every time there is a link change, i.e. link is created or destroyed or if the received power from another node passes the interference threshold, it creates a change in the local neighborhood of not only the transmitter and receiver of that link but also to nodes further away. If a link has changed to a lower state, e.g a link changes into an interference link, this handling is rather simple since it means that the local neighborhood decreases for the nodes and they only need information about the changing link itself. The new local neighborhood can decided by all nodes on their own with information about the changing link.

To actually send messages after this may be a little bit more complex since the routing will change but we assume that this problem belongs to the routing layer, which we have assumed can handle this efficiently as long as the MAC layer can keep track on their local neighborhoods. (see assumptions in chapter 2)

The second case occurs if a link has changed to a higher state, e.g an interference link into a link. This will result in a situation where nodes on both side of the link may need to increase their local neighborhood with new links and maybe also new nodes. In such a case we assume that the sender and receiver of the changing link is responsible for updating their local neighborhood.

Such an update will take place in a number of steps:

1. The sender and receiver exchange their local neighborhood (or an appropriate subset, those links that could be of interest).

2. Both these nodes receive this information and retransmit new information in their local neighborhood. Information is only sent if it is needed, i.e. information about a link is sent to those nodes that should have the link in their region 3 neighborhood. This means that nodes at two hops distance

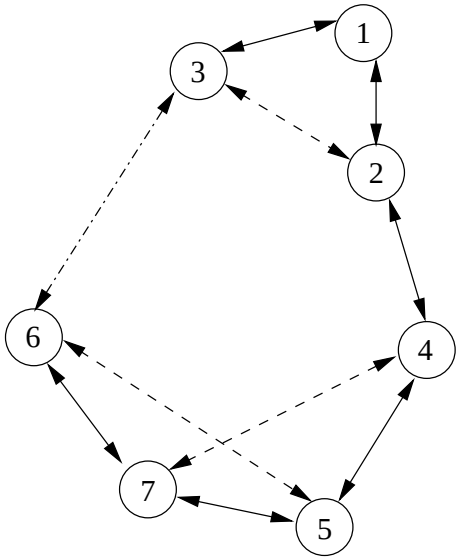

Figure 4.2: A 7-node network.

does not need information about anything except perhaps the new link, see region 3 as an example.

We may also need to send information about links that is already known. A new link may make a node, previously at two hop distance, be a direct neighbor which should be known by other nodes in our neighborhood.

3. Nodes receiving this information upgrade their local neighborhood accordingly.

Notice that the only nodes actively involved, i.e. sending messages, in this exchange are the sender and receiver to the changing link.

To clarify this we will give an example. In figure 4.2 we see a 7-node network. At the start of the example we will assume that there is no link between node v_3 and v_6, then an interference link between them is created and finally this interference link is upgraded to a full link.

At the start of the example node v_3 have knowledge about the nodes v_1, v_2, and v_4 and all links between these nodes. The only knowledge about node v_4 it

has is the link $(2, 4)$. Node v_6 have knowledge about the nodes v_4, v_7, and v_5 and all links between these nodes. It also have knowledge about link $(2, 4)$, but node v_2 is not included in the local neighborhood.

At some moment nodes v_3 and v_6 detects that they are sufficiently close so that they no more can ignore each other. We will discuss how this is done later in this chapter when we describe how the information is sent on the channel.

Both of these nodes now send an appropriate part of there local neighborhood to each other. How this is done is left for the routing layer, but normally we do not need six hops to find the other side as in this example. In more practical implementations one could probably set a maximum number of allowed hops, and let the algorithm handle interferences as unidentified in such cases as described in report [7], but here we will assume that information is always sent if possible.

Node v_3 sends information about itself v_3, v_1 and links $(1, 3)$ and $(3, 1)$ since more is not needed on the other side of an interference link (see figure 4.1).

Node v_6 sends information about itself v_6, v_7 and links $(7, 6)$ and $(6, 7)$.

This information is then to be spread in the "old" local neighborhoods of v_3 and v_6. Once again though, we limit the necessary information. Node v_6 only needs to inform node v_7 about node v_3, it's incoming link $(1, 3)$ and the existence of the new interference link. Node v_3 informs node v_1 about node v_6, it's incoming link $(7, 6)$ and the new interference link.

Once these updates are done the new local neighborhoods are updated.

Now, let us assume that nodes v_3 and v_6 move closer and the interference link will eventually be useful as a communication link. The reaction by v_3 and v_6 are similar as before but the appropriate information is different to before. Node v_3 sends information about v_3, v_1 and v_2 and links $(1, 3)$, $(3, 1)$, and $(4, 2)$. In addition to this, information about the existence of the interference link between v_3 and v_2 is also given.

Similarly, node v_6 sends information about v_6, v_7, v_5, and links $(7, 6)$, $(6, 7)$, $(4, 5)$, and also information about the interference link between v_6 and v_5.

Some of this information could of course be omitted, since many of these links have already been known. But to be on the safe side we assume that it is transmitted anyway, as a safety precaution if the link gain changes so fast that the interference links do not have time to be generated. This could happen if we have obstacles between the nodes. This information does not need to be relayed

any more since a direct link is available, however, care should be taken since the routing algorithm possibly does not know anything of this link yet.

Nodes v_3 and v_6 should now inform their local neighborhoods about the change.

In addition to this we have also added sequence numbers to these messages, since simultaneous events may have the effect that old information may arrive later than the new information if several links are changing at the same time. When nodes in such cases send their local neighborhoods there is no guarantee that no old information is received after the new. By adding a sequence number that only the owner (receiver) of a link may change we make certain that old information is not used. For now, these are only necessary for links updates since these messages are sent by nodes other than the owner of the link.

In addition to this information, the local schedule must also be updated. However, this information is only needed in region 2, which means that the sender and receiver only needs to send the time slots they have outgoing or incoming links assigned. In addition, this information does not need to be further conveyed.

We conclude this section by giving a overview of the existing messages shown in table 4.1.

Message	Purpose	Destinations	Region
AVTS	When the sender can transmit	Only the receiver	1
Schedule	Assigned Time slots and their I_{max}	Those nodes that can be affected by the assignment, i.e at least a direct interference link	2
Priority	Controls Access in the local neighborhood	Sufficient so that two simultaneous assignments do not cause problems	3
Link Changes	Updates the local neighborhood	Same as above	3

Table 4.1: An overview of the different types of messages used in the scheduling.

4.2 How often are these messages sent?

We also need to know how often each type of message are going to be sent.

There is two different causes behind virtually all changes in a ad hoc network: link gain changes and traffic changes.

The first of these is usually dependent on mobility, but nodes that leave or enter the network can be seen as more abrupt changes in link gain. Traffic changes will, in our traffic model, mainly depend on rerouting due to changes in path gain but in general it will depend on the input traffic to the network.

For small changes in link gain, $I_{\max}$ may change value and it is possible that time slots can now be assigned or must be deassigned because of the SIR criteria. This can have consequences for nodes further away, e.g. new slots that can be assigned or stolen.

For larger changes in link gain, links may be broken or created. This can create substantial changes in the scheduling process and perhaps also rerouting which can create changes also far away in the network

These are very complex processes and an analytic calculation is difficult, we will therefore leave this for future work when this can be simulated.

For most of these variables, it is rather obvious when they will be sent and by using simulations it would be rather simple to measure how many of each type of message that are sent. However, an exception to this is updates of $I_{\max}$ and Priority values. The problem with updates of these variables are that they change continuously as the nodes move. We will therefore have to limit the transmissions of these messages. This must be done so messages only will be sent when the value has changed more than a chosen threshold since the last update. We can also choose such threshold so that we quicker react to decreasing values than increasing.

4.3 Exactly how are each these messages sent?

The basic idea is to divide time into two parts as shown in figure 4.3, in the same way USAP [13] uses its bootstrap slots.

- One TDMA frame with mini slots used for administrative information (STDMA overhead) where one is given to each node.

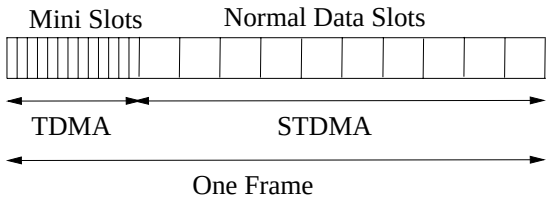

Figure 4.3: The used frame.

- One STDMA frame which are used for user traffic.

In practice the different frames do not need to be divided as the figure shows and for large networks a pure TDMA frame is probably not necessary, but we want to use two advantages of TDMA here.

First, TDMA is independent on mobility, as long as we have no new nodes the schedule do not need to be updated. This means not just that such updates can be ignored but also that the slots which each node uses can be known in the entire network.

Second, since only one node is sending in these slots and the rest of the nodes know which one this is, it can be used for determining both link gain and when interference links exists.

In a large network a pure TDMA frame is not very efficient, we may also have problems with traffic adaptivity, if the administrative traffic is different on the different nodes.

As previously stated we will assume that we have no delay for these TDMA messages and that two independent events cannot take place simultaneously. (They may if they take place at the exact same moment in time, though.)

The overhead cost will now be how large part of the time that needs to be mini slots slots.

4.4 What affects the overhead?

We will leave a more detailed investigation of the exact overhead cost for future work when it can be simulated, but we can already see a number of different network properties that will affect the overhead:

- *Network size*

  The network size has a direct effect on the TDMA frame; a doubling of the network size doubles the TDMA frame. It may also increase the overhead per node if the local neighborhoods increase in size. The network size is not a parameter we can control but it will affect the performance of the algorithm.

- *Network Connectivity*

  Network connectivity is a measure on how large part of the network that a node can reach with one hop. Connectivity will have a direct effect on the local neighborhoods, thereby increasing the number of nodes information needs to be spread to. Similar as above, network connectivity is difficult to control, but since it is possible to do this to some extent by power control, it would be interesting to study this for low connectivity.

- *Choice of* γ_I

  Connectivity is a measure on normal links. The interference threshold will affect the local neighborhood through interference links. The smaller this parameter is chosen, the larger part of the network will be included in the local neighborhood. This is one of the most important parameters under our control and a good choice of this parameter is necessary to make the algorithm work well.

- *Mobility rate*

  Mobility rate is the rate of changes in the network. A faster network will require updates more often. However, as long as the STDMA algorithm manage to keep the schedule updated, mobility rate should have approximately linear effect since all updates only react to changes.

- *Frame length*

  The more time slots we have to assign, the more overhead it requires. Notice though that the overhead may grow faster that linear. This is because that theft will be much more common in long frame lengths. For a short frame length where each link will receive at most one time slot theft is close to non existing, while for long frame lengths theft will be much

more common in order to even out the average load between the links. This will cause an addition of the overhead traffic.

Chapter 5

Conclusions

In this report we have expanded the distributed STDMA algorithm first described in [7].

We have now added methods for conveying the appropriate amount of information, described what information each node should send to its neighborhood as well as discussed how the nodes will be able to control their local neighborhoods.

We have also described which parameters that can be varied in order to give good appropriate balance between schedule capacity and overhead traffic.

5.1 Future Work

In future work we will study the following problems:

- The values of the important parameters needs to be decided for different network. This must in practice be done with the help of simulations in each case.
- From this also decide when STDMA works well, i.e. for which mobility rates, network size and mobility will the schedules be acceptable without unreasonably high overhead traffic.
- Expand the algorithm further so we can handle also error prone channels, lost packets, variable delays of packets and other complex situations.

- Further optimization, for example, faster initialization of the network can be done by letting nodes assign several timeslots at once.

Bibliography

[1] R. Nelson and L. Kleinrock, "Spatial-TDMA: A collision-free multihop channel access protocol," *IEEE Trans. Commun.*, vol. 33, no. 9, pp. 934–944, Sept. 1985.

[2] I. Chlamtac and S. Kutten, "A spatial reuse TDMA/FDMA for mobile multi-hop radio networks," in *Proceedings of IEEE INFOCOM '85*, 1985, vol. 1, pp. 389–394.

[3] I. Chlamtac and A. Lerner, "A link allocation protocol for mobile multi-hop radio networks," in *GLOBECOM '85, IEEE Global Telecommunications Conference, Conference Record*, 1985, vol. 1, pp. 238–242.

[4] B. Hajek and G. Sasaki, "Link scheduling in polynomial time," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 910–917, Sept. 1988.

[5] O. Somarriba, *Multihop Packet Radio Systems in Rough Terrain*, Tech.lic. thesis, Radio Communication Systems, Department of Signals, Sensors and Systems, Royal Institute of Technology, SE-100 44 Stockholm, Sweden, Oct. 1995.

[6] L. Pond and V. Li, "A distributed time-slot assignment protocol for multi-hop broadcast packet radio networks," in *MILCOM*, 1989.

[7] J. Grönkvist, "Distributed STDMA algorithms for ad hoc networks," Scientific Report FOA-R–1059–SE, Swedish Defence Research Agency., Div. of Command and Control. Linköping, Sweden, 2003.

[8] J. Grönkvist and A. Hansson, "Comparison between graph-based and interference-based STDMA scheduling," in *Proc. MobiHOC*, 2001.

[9] F. Eklöf et al, "On the performance of antenna arrays in spatial reuse TDMA ad hoc networks," in *Proc. of MILCOM'2002*, 2002.

[10] J. Grönkvist, "Assignment methods for spatial reuse TDMA," in *Proc. of MobiHOC 2000*, 2000.

[11] J. Grönkvist, *Assignment Strategies for Spatial Reuse TDMA*, Tech. lic. thesis, Radio Communication Systems, Department of Signals, Sensors and Systems, Royal Institute of Technology, SE-100 44 Stockholm, Sweden, Mar. 2002.

[12] J. Shor and T. Robertazzi, "Traffic sensitive algorithms and performance measures for the generation of self-organizing radio network schedules," *IEEE Trans. Commun.*, vol. 41, no. 1, pp. 16–21, Jan. 1993.

[13] D. Young, "USAP: A unifying dynamic multichannel TDMA slot assignment protocol," in *IEEE MILCOM*, 1996.