Distributed STDMA in Ad Hoc Networks

Jimmi Grönkvist
Swedish Defence Research Agency
SE-581 11 Linköping, Sweden
email: jimgro@foi.se

Abstract

Spatial reuse TDMA is a collision-free access scheme for ad hoc networks. The idea is to let spatially separated radio terminals reuse the same time slot when the resulting interferences are not too severe. In this paper we describe the properties a distributed STDMA algorithm must have to be efficient, and describe the first step towards such an algorithm. Furthermore, we evaluate this first step and show that it can give the same capacity as a a centralized reference algorithm.

1 Introduction

There are many situations where a fixed communication infra-structure cannot be relied upon for wireless communication, and where self-configurable networks must be deployed quickly, e.g., emergency relief or military networks. 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).

One problem in a radio network is the interferences caused by simultaneously transmitting nodes. An important issue is therefore to design the medium access control (MAC) that controls the use of the channel.

An interesting MAC protocol that has great potential for QoS is spatial reuse TDMA (STDMA), 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 [1, 2, 3, 4].

However, in an ad hoc network, 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 [3, 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 [1, 2], 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.

In this paper we describe the properties a distributed STDMA algorithm must have to be efficient and describe a basic algorithm that is the first step towards such an algorithm. Furthermore, we evaluate this algorithm and show that it can give the same capacity as a centralized reference algorithm. We also show that for the limited information case we do get a decrease in capacity, however, part of this capacity loss can probably be regained by using better traffic adaptivity.

We use an interference-based model for STDMA scheduling as suggested in [4].

2 Network Model

Here we describe the interference based model of a radio network. The network is represented by a set of nodes $\mathcal{V}$ and the link gain $G(i, j)$ between any two distinct nodes (v_i, v_j), $i \neq j$.

For simplicity we assume isotropic antennas and that all nodes use equal transmission power.

For any ordered pair of nodes, (v_i, 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 follows

$$\Gamma_{ij} = \frac{P\,G(i,j)}{N_r}, \quad (1)$$

where P denotes the power of the transmitting node v_i, 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 the pair (v_i, v_j) forms a *link*, (i,j), if the SNR Γ_{ij} is not less than a communication threshold, γ_C. That is, the set of links in the network, $\mathcal{L}$, is defined:

$$\mathcal{L} = \{(i,j) : \Gamma_{ij} \geq \gamma_C\} \ . \quad (2)$$

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

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

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

$$I_L(i,j) = \sum_{v_k \in V_\mathrm{T}(L) \setminus v_i} P_k G(k,j). \quad (3)$$

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

$$\Pi_K(i,j) = \frac{P_i G(i,j)}{N_r + I_L(i,j)}. \quad (4)$$

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 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.

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.

STDMA algorithms generally are of two types; transmission rights have been scheduled either to nodes or links. In the first case, only the sender is determined in advance, in the second, both sender and receiver.

In this paper we concentrate on link assignment, often referred to as link activation. This is mainly done because link activation can handle advanced nodes with abilities like power control and adaptive antennas more intuitively (and efficiently).

3 Desired properties

In the following we list some of the desired properties of a distributed algorithm.

1. *No central control, the algorithm is run in parallel in every node in the network.* This is necessary if we want a robust system that can handle the loss of any node and it is the basic meaning of the term distributed.

2. *Only local information is exchanged and needed*, i.e. the information propagation must be limited. The other corner stone of the term distributed. We do not make any specific definition on the term local, except that global information about the network is not needed.

3. *Local adaptation to topological and traffic changes must be possible.* (Ripples are permitted if the probability of updates decrease with distance from the change.) An addition to the previous two assumptions that prevents "unstable" algorithms.

4. *The algorithm should be able to efficiently handle large changes in the number of nodes and density of the network.* (With efficiently we mean that it should not just be able to create a valid schedule but also perform close to the results of a centralized algorithm in a number of very different scenarios).

5. *Adaptivity to traffic, the algorithm should be able to adapt to the different needs of the different links.* There is considerable variation of traffic over the different links of the network, due to the relaying of traffic in multi-hop networks. An STDMA algorithm must adapt to this in order to be efficient [4].

6. *Using an interference-based network model.* The graph-based network model is currently the most used network model for ad hoc-networks. However, this model does not reflect reality sufficiently well in many of our scenarios. In fact, in order to use a graph-based model we need to be more "careful" in our scheduling, resulting in much lower efficiency. Furthermore, a graph-based model has more difficulty in handling properties 7 and 8.

7. *The algorithm should adapt to the level of mobility*. In relatively static network we can get a very good

picture of the situation, e.g. precise path losses and power levels which could be used to make a more efficient schedule. In a high mobility network all information that may be possible to transmit can be the existence of neighbors, and the algorithm should perform well under these circumstances as well.

8. *The algorithm should handle (exploit) heterogeneous nodes in the network.* Some nodes have more advanced abilities than others, e.g. adaptive antennas, able to use variable data rate, lower noise levels and similar abilities.

Although several distributed STDMA algorithms exists, most of these have been designed with the purpose to give an acceptable solution, rather than a solution that utilizes the channel as efficiently as possible under different situations. None of the existing STDMA algorithms fulfill these desired properties.

A systematic approach to the design of STDMA algorithms is lacking. What is an efficient schedule in a specific scenario? How is such a schedule created? Exactly which information is needed and how much in each case?

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, but increasing information also increases the overhead. Therefore, this means that,the amount of information the algorithm passes between the nodes should vary depending on the situation.

In the next section, we will describe a basic algorithm that creates an efficient schedule with given local information about the network. The algorithm does not care how it receives the information, it only acts upon the information it has received.

This basic algorithm is then used to investigate the efficiency of schedules created with different amount of information, ranging from complete information to just basic graph information.

In further work, we will develop methods to convey the appropriate network information and investigate how much this will cost in overhead. This can be used to develop a complete algorithm that includes the control information and which gives as efficient schedules as possible in every situation.

4 The Algorithm

This section gives a description of the algorithm. In short it can be described with the following steps.

- Nodes that have entered the network exchange local information with its neighbors.
- The node/link with highest priority in its local surroundings assigns itself a time slot.
- The local schedule is then updated and a new node/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 has need of many time slots will more often have high priority than a link with low priority.

In the following we 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 it may change over time but this will not change the basic functionality of the scheduling process.

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

- *Active*: In this mode the link has the highest priority in its local neighborhood and it will subsequently assign itself a time slot. For simplicity we assume a random choice if more than one time slot is possible. A link process is in this mode when there exists unused slots or when 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 other links. We later describe under which conditions this may be permitted. Information about which time slot is chosen and the links new priority will be transmitted to its local neighborhood. After this the link process can stay in active 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 on 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.

In this paper we assume that the receiver does the assignment. However, the sender could do this instead with minimum changes.

Eventually, parts of the schedule will not be valid due to mobility. A receiver which detects conflicts will drop the time slot. This might then result in thefts of another slot (or even the same).

Link Priority

Link priority decides in which order the links may attempt to assign themselves a time slot. This figure can depend on many things, but the most important will be the number of time slots h_{ij} the link is assigned and the traffic of the link Λ_{ij}. Since both these values are changing this also results in a constantly changing link priority.

The priority value of a link (i,j) will be $\frac{h_{ij}}{\Lambda_{ij}}$, where the lowest value has the highest priority. The links with the highest priority (lowest value) will then be the links which limits 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 in a local neighborhood will receive at least one time slot before some of the links receive more than one slot. This will for example be the case when a new schedule is initiated.

Theft of time slots

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

The policy for 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 total network throughput must increase.

This means that a link only is permitted to steal a link from another if the priority value of the stealing link is lower than the other links priority value *after* the loss of a time slot, i.e.

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

Limited information

The interference-based model is included mainly by which information is transmitted and its method for determining when links can transmit simultaneously. We use the notation *local neighborhood* of a link (i,j) to mean those links that will be taken into consideration when the 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 are assumed. A remaining issue is then exactly what information the algorithm needs in order to do the scheduling. We need the following:

- *Interference - Received Power* We have an estimate of the interference from all other transmitters I_i. However, if the interference level from a transmitter is below a value δI it is set to zero, i.e. we assume that such node does not affect each others. If δI is set to zero we do in fact have all information about the network. We define the *Interference threshold* γ_I to be $\delta I/N_r$.

- *Local Schedule* We also need information of how much more interference can be handled by the assigned receivers.

- *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.

With this information we have sufficient information for both sender and receiver to determine when to be active and which time slots that can be assigned.

The only further information required is that the sender informs the receiver which slots that are available for the transmitter to use without causing too much interference. The receiver can then assign the time slot. Information about this is then propagated to the local neighborhood.

5 A First Evaluation

In this section we will compare the described algorithm with an existing centralized algorithm [5]. One important property of this algorithm is that it is traffic adaptive. This means that each link receives a number of time slots in direct proportion to the traffic flowing over it.

The comparison will be done for two networks with three different levels of information, i.e. we will use three different γ_I. First we use γ_I equal to zero which is the case when we have the same information as the centralized algorithm. We then use higher value to see how much capacity is lost when the information is limited. We have not implemented the theft of time slots part since this will mostly be useful in a mobile scenario.

We use the schedule length given by the centralized algorithm as input frame length for the distributed algorithms. This frame length is known to be sufficiently long to give efficient traffic compensation.

The two networks considered are of size 10 and 20 nodes, respectively.

	Network 10	Network 20
cent.	0.638	0.636
$\gamma_I = 0$	0.638	0.636
$\gamma_I = -3dB$	0.638	0.468
$\gamma_I = 0dB$	0.638	0.463

Table 1: λ_{max} for different γ_I compared to the centralized algorithm.

As can be seen, for the full information case the distributed algorithm generates schedules with the same capacity as the centralized approach. This is an important property, since this means that we do not lose anything when we use distributed algorithms in terms of capacity. The gain in robustness is significant, though.

For the case with limited information we do necessarily have conflict-free schedules. Due to interference from nodes far away, we end up with a SIR below γ_R for some receivers. In order to handle these nodes we assume that they have the ability to decrease their data rate so that conflict-free transmission is possible even in these cases. For simplicity we assume that the available data rate is linearly dependent on the SIR close to the γ_R threshold. For the small network this does not cause a problem but for the larger network we do get a decrease of the capacity.

This decrease in capacity occurs because some (few) links have a decrease in SIR in most of their assigned time slots. Traffic sensitivity gives each link time slots in proportion to the traffic on the link, with the assumption that all assigned time slots can be used at full rate. This will have the consequence that the decrease of rate on a single link will decrease the throughput of the network. Therefore, a better estimation of link priority, where the data rate is included, would probably make the capacity loss due to limited information much smaller.

6 Conclusions

In this paper we have developed a distributed slot allocation algorithm and evaluated it compared to a centralized reference algorithm. The distributed algorithm have been shown to generate very efficient schedules, with capacity equal to that of the centralized algorithm. For the limited information case we do get a decrease in capacity, however, part of this capacity loss can probably be regained by using better traffic adaptivity.

References

[1] 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.

[2] 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.

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

[4] O. Somarriba, *Multihop Packet Radio Systems in Rough Terrain*, Tech.lic. thesis, Radio Communication Systems, Department of S3, Royal Institute of Technology, SE-100 44 Stockholm, Sweden, Oct. 1995.

[5] J. Grönkvist, *Assignment Strategies for Spatial Reuse TDMA*, Tech. lic. thesis, Radio Communication Systems, Department of S3, Royal Institute of Technology, SE-100 44 Stockholm, Sweden, Oct. 2002.