WIRELESS MULTIPLE ACCESS CONTROL PROTOCOLS

Hakan Cuzdan

Control Engineering Laboratory – Helsinki University of Technology

PL5400, 02015 TKK - Finland

Email: hakanc@cc.hut.fi

Student No: 55984M

Abstract

In competition based protocols, the channel is shared among many neighbouring users each of which tries to get access to. The user that has got the channel first disables the others from using the channel during its transmission. This report describes the Multiple Access Control (MAC) Layer in a competition based environment only. MAC layer's inherent problems in wireless environments, performance criterions of MAC Layer are explained. Some of the most common MAC protocols are illustrated (i.e. CSMA-CA, MACA, MACA-BI, PAMAS, MARCH, DBTMA). Each protocol's strong and weak points have been underlined.

Keywords: MAC, CSMA, BTMA, DBTMA, MACA, MACA-BI, PAMAS, MARCH

1. Introduction

In ad hoc networks, transmitters use radio signals for communication. Generally, each node can only be a transmitter (TRX) or a receiver (RX) one at a time. Communication among mobile nodes is limited within a certain transmission range. And nodes share the same frequency domain to communicate. So, within such range only one transmission channel is used, covering the entire bandwidth. Unlike wired networks, packet delay is caused not only by the traffic load at the node, but also the traffic load at the neighbouring nodes', what we call as "traffic interference".

Note that source and destination could be far away and each time packets need to be relayed from one node to another in multi hop fashion, medium has to be accessed. Accessing medium properly requires only informing the nodes within the vicinity of transmission. Therefore, MAC protocols deals only with per-link communications, not with the end-to-end communications [4].

Medium access (MAC) protocols control access to the transmission medium. Their aim is to provide an orderly and efficient use of the common spectrum. These protocols are responsible for *per-link connection establishment* (i.e. acquiring the medium) *and per-link connection cancellation* (i.e. releasing the medium free).

MAC protocols can be classified as *synchronous* and *asynchronous*. In synchronous MAC protocols, transmission capacity is divided into slots (i.e. time slots in TDMA, Slotted ALOHA, frequency slots in FDMA, time and frequency slots in FDMA/TDMA in GSM, and spreading codes in UMTS) and nodes are only supposed to use those slots to get access to the transmission medium. These synchronized slots are assigned by a central control unit in the system (i.e. base stations (BS) in GSM, or in UMTS). However, due to the infra-structure less property of ad-hoc networks, synchronization is not possible, leaving out asynchronous MAC protocols being the only option.

To summarize up to this point, in ad-hoc networks;

- In competition based environment, the medium is shared by all neighbouring nodes which are in competition among themselves to seize the channel. And a node can only be a TRX or RX. It can not be both at the same time.

- Transmission delay is caused by the traffic load at the node who has the packet and by the traffic load of neighbouring nodes (i.e. traffic interference)

- Aim of the MAC Protocols is to have orderly and efficient use of the transmission capacity while increasing throughput as much as possible.

- MAC Protocols for ad-hoc protocols can't be synchronous due to their infra-structure less nature. They are asynchronous and access to the medium is contention-based (i.e. competition to get the channel first).

2. Asynchronous MAC Protocols for Ad-Hoc Networks

Before explaining what asynchronous MAC protocols are, the following facts must be underlined;

1. The establishment of routes is the responsibility of network layer. It is done before the node sends a packet belonging to a specific route, so that the node can know to which neighbouring node to relay the packet.

2. Antennas are assumed to be omni-directional. Therefore, each transmission affects the neighbouring nodes within a certain transmission range.

All MAC protocols for ad-hoc networks must address the two important problems. The first problem is called *hidden terminal problem* and it is explained in Figure 1.

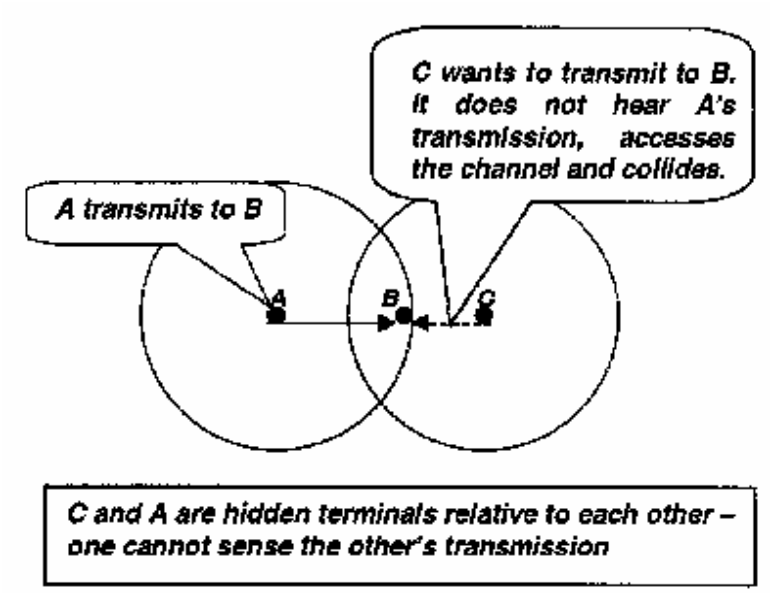

Figure 1 – The hidden terminal problem

The second problem is called *exposed node problem* and it is explained in Figure 2.

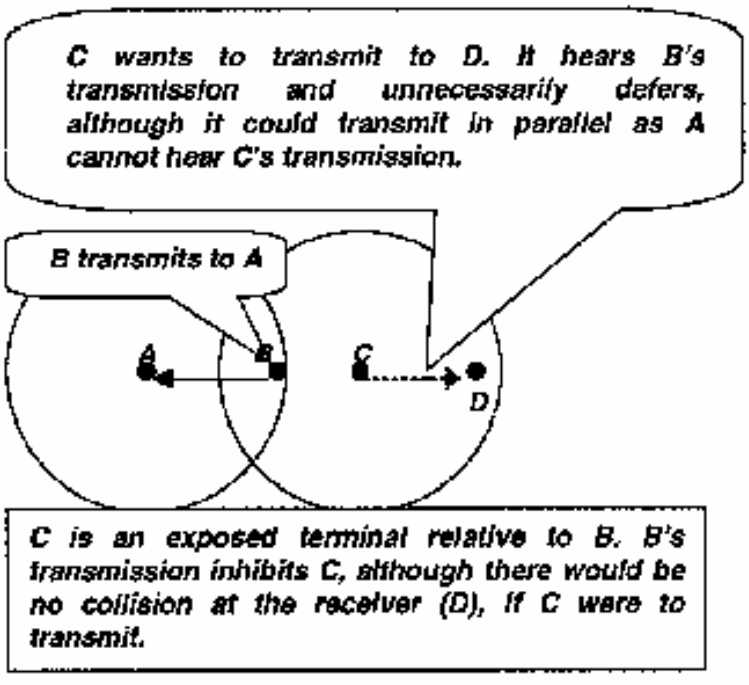

Figure 2 – The exposed terminal problem

Assume the following case where node A wants to send a data packet to B;

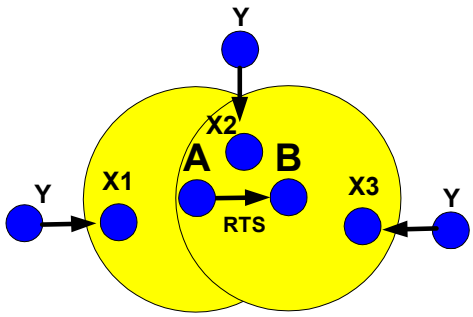

Figure 3 – MAC issues illustration case

Below given the following problems to be solved at MAC layer:

1. How will A acquire the channel while other nodes possibly trying to send data to B as well? Is it going to be by sending a Ready To Send (RTS) packet (i.e. sender initiated MAC) or by the reception of RTR packet (i.e. receiver initiated MAC)

2. How will you minimize the possibility of control packet collisions at the receiver? (or; How will you reduce the control overhead (i.e. the amount of control packets having been sent to get specified throughput)? or; How will you reduce the contention period?)

3. How will you prevent the hidden nodes sending data?

4. How will you enable the exposed nodes sending data?

5. Will you use one channel or several channels for control and data packet transmission?

6. If a node can neither receive nor transmit any packets due to traffic interference, would you consider shutting the power of the node off? If so, which interface (i.e. data or control or both) will you shut off? and how long will you shut them off?

7. How will you guarantee that the channel have been released properly after the data transmission is finished?

The performance parameters of MAC layer are;

- Average end to end delay: The mean time required to transmit a packet through a route.
- Throughput: Successfully received number of data packets without collision.
- Control Overhead
- Power conservation

In the following, the factor's affecting MAC Protocol's design and performance are mentioned.

1. Implementation Technology: (a) Single Channel for Control/Data signalling together (i.e. Medium Access Protocol with Collision Avoidance (MACA), MACA By Invitation (MACA-BI), Floor Acquisition Multiple Access (FAMA)) (b) Control and data channels separated in multiple channels (i.e. Dual Busy Tone Multiple Access (DBTMA), Busy Tone Multiple Access (BTMA))

2. Traffic Characteristics (a) Stationary traffic (b) Non-Stationary Traffic

3. Mobility

Having showed the most common problems of MAC layer and the factor's affecting MAC layer performance; we'll describe the MAC protocols for ad-hoc networks and explain how they address those problems.

2.1 Carrier Sense Multiple Access (CSMA) and Collision Avoidance (CA)

CSMA and CSMA-CD is used for wired LAN networks. CSMA works as follows. A terminal can transmit only when it senses no carrier (i.e. no transmission) within its vicinity. However, transmissions out of range can't be detected. Thus, in spite of carrier sensing a transmission could still collide at the receiver with another transmission from an "out of range" terminal, often referred to as the "hidden terminal". Also, exposed node problem exists in CSMA. Both, the hidden and the exposed terminal problems cause the pure CSMA scheme to be inefficient for ad-hoc networks [1]. To avoid collisions at the receiver node, CSMA-CA is used. In CSMA-CA, a three-way handshake mechanism is used to avoid collisions. An RTS (Request to Send) message can be used by a node to indicate its wish to transmit data. The receiving node can allow this transmission by sending a grant using the CTS (Clear To Send) message. Because of the broadcast nature of the message, all the neighbours of the sender and the receiver will be informed that the medium will be busy, thus preventing them from transmitting and avoiding any collisions. Figure 4 depicts this process.

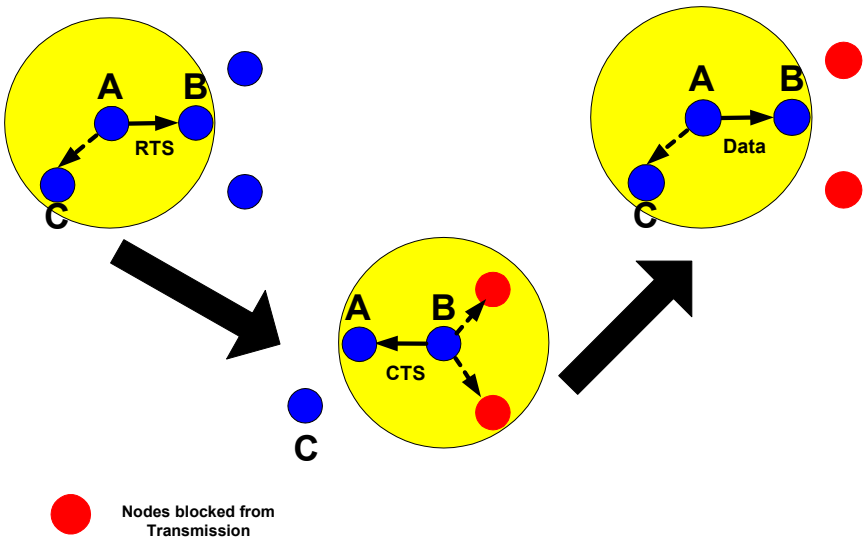

Figure 4 – RTS/CTS/Data Handshake

The RTS-CTS method is not a perfect solution to the hidden terminal problem. There are some cases when collusions occur and the RTS and CTS control messages are sent by different nodes. As shown in Figure 5, node B is granting a CTS to the RTS sent by node A. However, this collides with the RTS sent by node D at node C. Node D is the hidden terminal from node B. Because node D doesn't receive the expected CTS from node C, it re-transmits the RTS. When A node receives the CTS, it is not aware of any collision at node C and hence it proceeds with a data transmission to node B. Unfortunately, in this scenario, it collides with the CTS sent by node C in response to node D's RTS.

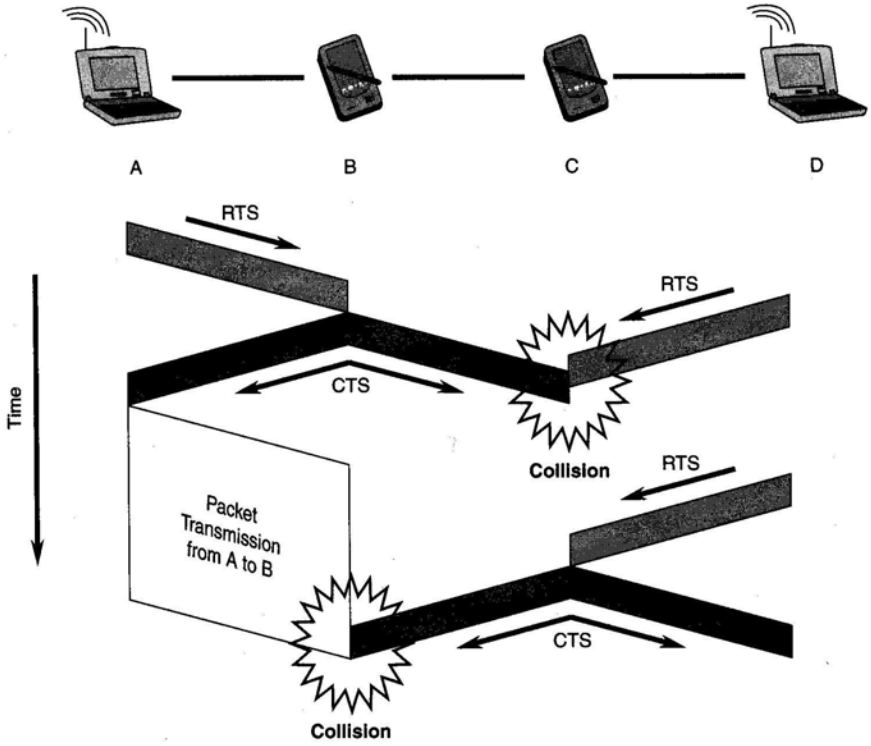

Figure 5 – RTS – CTS drawback 1

Another problematic scenario occurs when multiple CTS messages are granted to different neighbouring nodes, causing collisions. As shown in Figure 6, two nodes are sending RTS messages to different nodes at different points in time. Node A sends an RTS to node B. When node B is returning a CTS message back to node A, node C sends an RTS message to node B. Because node C cannot hear the CTS sent by node B while its transmitting an RTS to node D, node C is unaware of the communication between the nodes A and B. Node D proceeds to grant the CTS message to node C. Since both nodes A and C are granted transmission, a collision will occur when both start sending data.

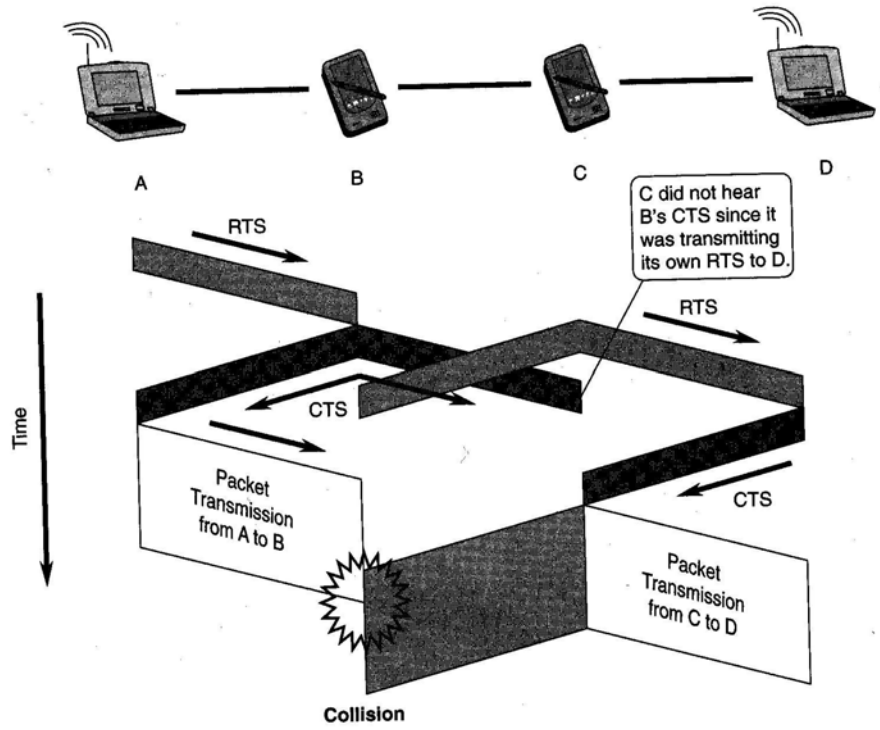

Figure 6 – RTS – CTS drawback 2

The shortcomings of RTS-CTS method have two following reasons:

1. The limitation of the implementation technology, that is, a node cannot be a TRX/RX at the same time. This means, when a node is transmitting, it can't hear any transmission going on.

2. Control packets may collide, causing a node being unaware of the transmission going on in its neighbourhood. Also, control and data packets transmission use the same channel. Those result in a capacity that can be used simultaneously by the two transmissions, which collide into one another.

This section could be summarized as follows:

- In competition based environment, the medium is shared by all neighbouring nodes which are in competition among themselves to seize the channel. And a node can only be a TRX or RX. It can not be both at the same time.

- Transmission delay is caused by the traffic load at the node who has the packet and by the traffic load of neighbouring nodes (i.e. traffic interference)

- Aim of the MAC Protocols is to have orderly and efficient use of the transmission capacity while increasing throughput as much as possible.

- MAC Protocols for ad-hoc protocols can't be synchronous due to their infra-structure less nature. They are asynchronous and access to the medium is contention-based.

- Routing and MAC are two different aspects of ad-hoc networking. Former one deals with finding the best path between s-d pairs in a multi-hop, mobile environment. MAC assumes that the route has been established, so that it knows the node it should forward the packets to. MAC protocol's task is to gain access to the medium where other nodes are seeking for gaining an access too (i.e. traffic interference)

- The most common problems associated with MAC protocols in ad-hoc networks are hidden and exposed terminals problems.

- To cure the hidden terminal problem (i.e. in CSMA), A three way handshake RTS/CTS/Data is used (i.e. CSMA-CA). However, due to the collision of control packets or due to single mode of (TRX or RX) transmission, two parallel transmissions may occur claiming the same capacity. This would cause collisions wasting the capacity.

- CSMA (with hidden and exposed terminal problems) and CSMA-CA (with exposed terminal problem and RTS/CTS flaw) are not suitable for ad-hoc networks' MAC layer.

2.2. Multiple Access with Collision Avoidance (MACA) [1]

The main motivation behind the MACA is that Carrier Sensing (CA) is unnecessary in ad-hoc environment. For example, when hidden terminals exist, lack of carrier doesn't always mean it's OK to transmit. Conversely, when exposed terminals exist, presence of carrier doesn't always mean that it's bad to transmit.

Instead we'll extend the CA part of what we'll call MA/CA (or just MACA). The key to collision avoidance is the effect that RTS and CTS packets have on the other stations on the channel. When a station overhears an RTS addressed to another station, it inhibits its own transmitter long enough for the addressed station to respond with a CTS. When a station overhears a CTS addressed to another station, it inhibits its own transmitter long enough for the other station to send its data. The transmitter is inhibited for the proper time even if nothing is heard in response to an RTS or CTS packet.

This has been illustrated in Figure 4. Node C want to send data to node A, but it overhears an RTS from node A, it will know not to transmit till node A finishes sending its data.

But, how does node C know how long to wait after overhearing RTS from node A? The initiator of the dialogue, node A, includes in RTS packet the amount of data it plans to send, and we have B, the responder, echo that information in its CTS packet (for B's red-coloured neighbours in Figure 4). Now, everyone overhearing B's CTS knows just how long to wait to avoid clobbering a data packet that it might not even hear.

Collisions do occur in MACA, especially during an RTS-CTS phase (i.e. contention period). There is no carrier sensing in MACA. Each mobile host basically adds a random amount of time to the minimum interval required to wait after overhearing an RTS or CTS control message. In MACA, the slot time is the duration of an RTS packet. If two or more stations transmit an RTS concurrently, resulting in a collision, these stations will wait for a randomly chosen interval and try again, doubling the average interval on every attempt (i.e. exponential back off algorithm). The station that wins the competition will receive a CTS from its responder, thereby blocking other stations to allow the data communication session to proceed.

Note also that collisions do occur during the data transmission phase due to the drawback of RTS-CTS mechanism (i.e. refer to Figure 5 and 6). When those collisions occur in MACA, recovery is left up to the transport layer thus greatly reducing throughput. [5]

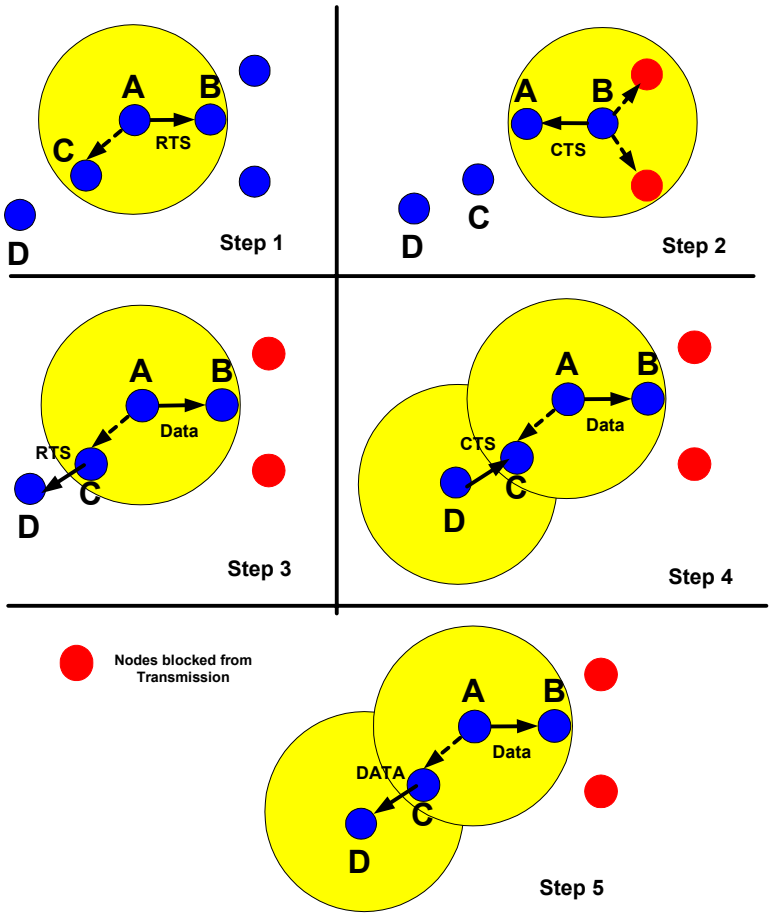

Figure 7 – MACA Protocol overcoming the exposed node problem (node C is the exposed node)

MACA offers solution to *the exposed terminal problem* as follows. If a node hears no response to an overheard RTS, then it may assume that the intended recipient of the RTS is either down or out of range. An example is shown in Figure 7.

Automatic Power Control in MACA on RTS/CTS and Data packets leads to increased throughput in ad-hoc networks. By changing the MACA rule to "inhibit *a transmitter when a CTS packet is overheard*" to "*temporarily limit power output when a CTS packet is overheard*", geographic reuse of the channel can be significantly improved. For example, in Figure 7, if station C has recently sent traffic to station A, it knows how much power is required to reach A. If C overhears station A responding to with a CTS to a third station B, then C need not remain completely silent for the required interval; it need only limit its transmitter power to say, 20 dB below the level needed to reach A. During this time it would be free to transmit to any station that it could reach with that reduce over level, because its signal at A would be overridden by B's signal. (This is analogous to the people at the cocktail party continuing their conversations in whispers instead of stopping completely when Tom tells Bob to go ahead.)

The CTS packets, however, pose a problem. In addition to telling the initiator to send its data, the CTS must inhibit all potential interferers from transmitting. It may therefore need more power than that needed just to reach the initiator to ensure that everyone "gets the message." (A CTS packet might therefore before like Tom shouting "Hey, everyone, shut up! I'm trying to hear Bob speak!" at the cocktail party mentioned earlier).

All this shouting potentially limits the geographic channel reuse ability we've worked so hard to get. But all is not lost. A station responding to an RTS with a CTS can always expect data to follow. If it doesn't arrive within reasonable period, or if a retransmitted RTS arrives instead, then either the CTS was stepped, or the CTS wasn't heard widely enough to prevent the data transmission that follows from being stepped on. It should then respond to the next RTS from the same station (which will likely be a repeated attempt to send the same data) with a CTS at higher power. On the other and, if a responder has had good luck in getting data in response to its CTS packets, it might try lowering the power it uses to transmit them in order to help limit channel loading. Of course, it would never lower its CTS power below the level it knows is necessary to reach the initiator. In sum, MACA with power control automatically determines the exact amount of power required for each RTS and data transmission, and learns by experience (i.e., trial and error) the power required for CTS transmissions.

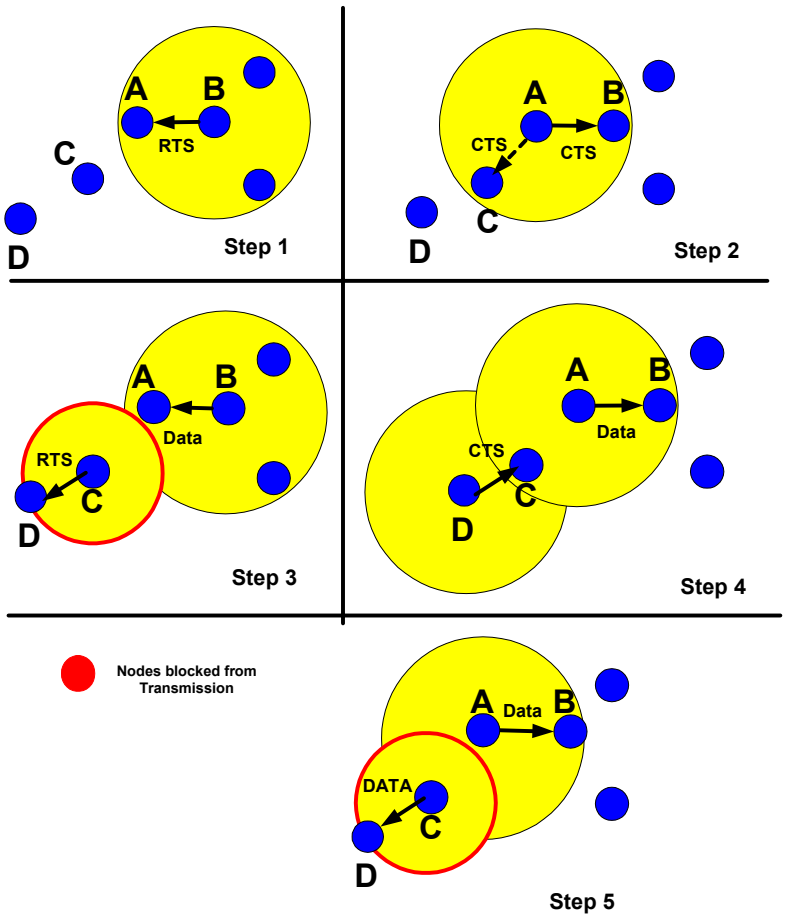

Figure 8 – Automatic Power Control in MACA to increase the throughput

In [4], MACA has the contention only in the RTS-CTS period, after that data transmission occurs over a contention free period. However, since MACA is utilizing RTS-CTS-Data handshake, we know that this can't be true due to the fact that RTS-CTS exchange can lead to two parallel transmissions claiming the same capacity (refer to figures 5 and 6). (Also refer to page 7 of [5])

MACA protocol could be summarized as follows:

- Carrier sensing is not efficient in ad-hoc networks.
- MACA doesn't have any carrier sensing operation [1]
- MACA offers solution to the hidden terminal problem by RTS/CTS message exchange
- MACA RTS/CTS exchange causes collisions during data transmission, recovery is left up to the transport layer thus greatly reducing throughput.[5]
- MACA solves the exposed node problem as follows. If a node doesn't hear a response to an RTS message, it can send packets.

- Since MACA uses power control on RTS/CTS/Data packets, collisions can be eliminated and throughput can be increased

- Collisions in MACA happens in the following cases (i) During RTS/CTS dialog using exponential back off algorithm (ii) Chances of collisions in data packets still exist, if CTS is not heard (i.e. due to mobility and not sensing carrier in MACA, or due to the channel conditions)

- Releasing the resources after transmission ends is done by putting the data amount information in RTS and CTS packages.

2.3 MACA By Invitation (MACA-BI)

There are practical difficulties set by the standards and by the hardware constraints making MACA less attractive to the ad-hoc network applications. These are:

- Time is required for any transceiver to switch from TX to RX state (or vice versa). This is called *TX-RX turn around time*. This turn around time must be less than 25 micro seconds set by the standards.

- Each transmission must wait to give a chance to the previous transmitter to switch to the receive mode.

- RTS-CTS mechanism in MACA is open to collisions. This means that there may pass a considerable amount of time for a node to begin data transmission. This reduces the throughput of the channel. A new way is required to reduce the time passed in the three way handshake.

The principle behind MACA – BI is as follows. MACA – BI replaces RTS-CTS-Data handshake by RTR (Ready To Receive) – Data handshake. This way, a node is allowed only if it is allowed to transmit first. And an RTR message is issued only if there is no data/RTR transmission around. Note that, to accomplish this, MACA-BI doesn't need to utilize carrier sensing. Note also that carrier sensing is different from listening to a specific packet. In this way, the time it takes to establish a connection can considerably be reduced by disabling possible RTS collisions in MACA. Below given an example how MACA- BI works in comparison with MACA.

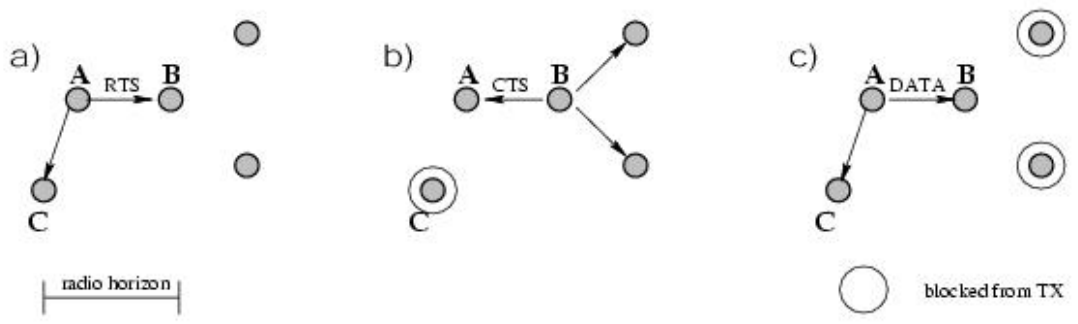

Figure 9 – MACA protocol example

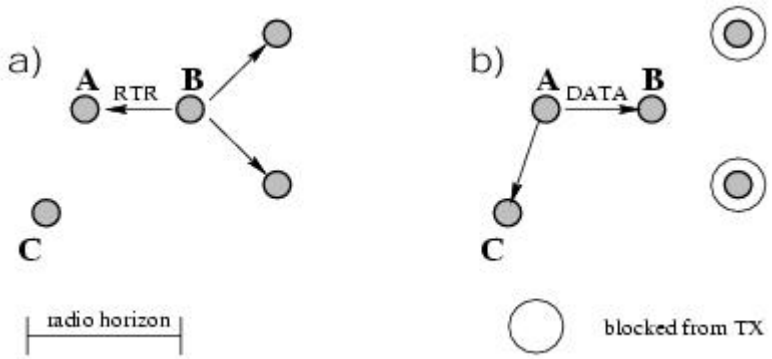

Figure 10 – MACA-BI protocol example

Note that node B in Fig. 10 does not have the exact knowledge of packet queue at node A. Rather; it must estimate queue and average arrival rate. To make this possible, we assume that each data packet carries the information about the backlog in the transmitter (A in this case). From the backlog notification and from previous history, B can decide how many packets to invite.

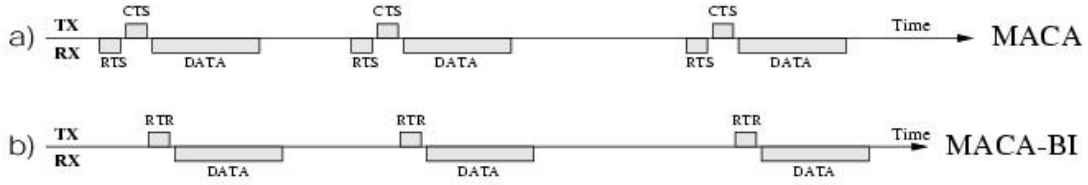

Figure 11 – Two and Three way handshake timing

MH_A replies with the requested packets and the new backlog information. So, the efficiency of the “invitation" scheme rests on the stationarity of the traffic pattern, which permits to predict which neighbours have “how many” packets to send.

To enhance performance in non stationary traffic situations, a node may transmit an explicit RTS if the queue length or delay has exceeded a given threshold before an RTR is received from the intended destination. In the limit, MACA-BI reduces to MACA if traffic burstiness prevents timely invitations.

One of the advantages of MACA-BI over MACA is that MACA-BI is data collision free. This can be explained as follows. Consider the 4-node network in Fig. 12.

Figure 12 – Collision free property of MACA-BI

The channel is assumed to be noise free and symmetric. A data collision occurs if node A transmits a data packet to B and simultaneously, node C transmits a data packet to B or to D. This causes a data collision in B. We will show that such collision cannot occur in MACA-BI. Note that other 4-node topologies with varying degrees of connectivity among nodes can be examined beside Fig. 12.

1. C transmits a data packet to B. This is impossible since node B can invite only one node at a time (either A or C in our case).

2. C transmits a data packet to D. This can happen only if C did not hear the RTR from B to A. Here again, two cases must be considered:

(a) B transmitted RTR to A while C was transmitting (either RTR or data). This is impossible, since transmission from C would have been heard from B, preventing its RTR transmission to A

(b) B transmitted RTR to A while C was receiving an RTR from D. Again, this is impossible because the RTR from D would have conflicted (at node C) with RTR from B, thus preventing the subsequent data transmission from C to D.

Thus, we conclude that collisions among data packets are not possible in MACA-BI. Note that control packets may still collide with each other, either directly (because of carrier sense failure due to non zero propagation delays) or indirectly (because of hidden terminal transmission).

To summarize MACA-BI protocol:

- Two way RTR-Data handshake is utilized to reduce the connection setup time
- A node transmits data only if it receives RTR
- An RTR is issued only if no RTR/Data packets around
- Carrier sensing (CS) is not required for MACA-BI
- RTR is issued for a specific node and for specific number of packets.
- The timeliness of RTR messages affects MACA-BI performance
- MACA-BI can utilize RTS packages in case of access load at sender nodes.

- Receiver node doesn't know how many packets to sender node have to send. Therefore, piggy backing or back logging is used to inform the receiver node about the next incoming packets once the connection has been established.

- MACA-BI is *data collision free* whereas MACA itself is not due to the drawbacks of RTR-CTS-Data handshake (refer to Figures 5 and 6) (Contrary to [2], page 2)

- MACA-BI is less vulnerable to control packet corruption than MACA, since it requires half as many control packets.

- The "receiver driven mechanism" of MACA-BI automatically provides traffic regulation, flow control and congestion control.

- In [2], it has been shown that under static ad-hoc network, and in dual ring, grid and star topologies, MACA-BI performs better than MACA in delay and throughput.

- Releasing the resources after a finished transmission is achieved by putting the invited number of packets information into RTR packets.

- The timeliness of RTR messages affects MACA-BI performance. If RTR message is received earlier when there is no packet to send, or if it is received late where packets are lost due to the overloaded packet queue, then delay and throughput characteristics of MACA-BI would be quite poor.

- Receiver driven mechanism to access the medium arises "when to invite?" question, severely reducing the performance. Periodical invitations or invitations based on a node's own load has been offered [2], however, they don't give satisfactory results.

2.4. Power Aware Multi-Access protocol with Signaling (PAMAS)

The PAMAS protocol is a combination of the original MACA protocol (see [1]) and the idea of using a separate signalling channel. The main motivations behind PAMAS can be summarized as follows:

1. MACA itself uses RTS-CTS – Data handshake which uses a common channel. This causes collisions in data transmissions (i.e. refer to Figure 5 and 6). The recovery is left up to the transport layer [5], thus greatly reducing the throughput.

2. Most of the MAC protocols focus on the issue of maximizing throughput and minimizing the transmission delay. However, if a node cannot transmit/receive data (due to the traffic interference from its neighbours), it may turn off its power. So, power consumption is the 3.rd issue in MAC performance.

PAMAS protocol deals with the above issues very efficiently. Below given the explanation how it works.

The PAMAS protocol is a combination of the original MACA protocol (see [17]) and the idea of using a separate signalling channel. Thus, we assume that the RTS-CTS message exchange takes place over a signalling channel that is separate from the channel used for packet transmissions. This separate signalling channel enables nodes to determine when and for how long they can power themselves off.

The state diagram outlining the behaviour of PAMAS protocol is illustrated in Figure 4. As indicated in the figure, a node may be in any one of six states - **Idle, Await CTS, BEB (Binary Exponential Backoff), Await Packet, Receive Packet**, and **Transmit Packet**.

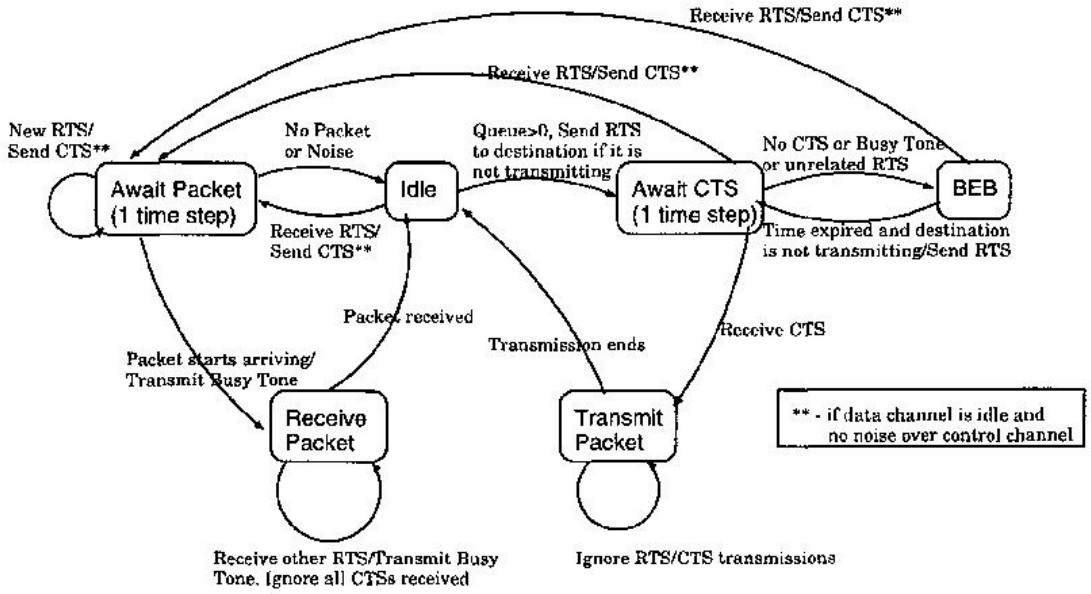

Figure 13 – PAMAS protocol

When a node is not transmitting or receiving a packet, or does not have any packets to transmit, or does have packets to transmit but cannot transmit (because a neighbor is receiving a transmission) it is in the Idle state. When it gets a packet to transmit, it transmits a RTS and enters the Await CTS state. If the awaited CTS do not arrive, the node goes into binary exponential back off (the BEB state in the figure). If a CTS arrives, it begins transmitting the packet and enters the Transmit Packet state. The intended receiver, upon transmitting the CTS, enters the Await Packet state. If the packet does not begin arriving within one roundtrip time (plus processing time), it returns to the Idle state. If the packet does begin arriving, it transmits a busy tone over the signalling channel and enters the Receive Packet state. Let us now look at the functioning of the protocol in some more detail.

When a node in the Idle state receives a RTS, it responds with a CTS if no neighbor is in the Transmit Packet state or in the Await CTS state. It is easy for a node to determine if any neighbor is in the Transmit Packet state (by sensing the data

channel). However, it is not always possible for a node to know if a neighbor is in the Await CTS state (the transmission of the RTS by that neighbor may have collided with another transmission over the control channel). In PAMAS protocol, if the node heard noise over the control channel within T 2 of the arrival of the RTS, it does not respond with a CTS. If, however, it does not hear a packet transmission begin within the next T, it assumes that none of its neighbors is in the Await CTS state anymore.

Now consider a node that is in the Idle state and has a packet to transmit. It transmits an RTS and enters the Await CTS state. If, however, a neighbor is receiving a packet that neighbor responds with a busy tone (twice as long as a RTS/CTS) that will collide with the reception of the CTS. This will force the node to enter the BEB state and not transmit a packet. If no neighbor transmits a busy tone and the CTS arrives correctly, transmission begins and the node enters the Transmit Packet state.

Say a node that transmitted a RTS does not receive a CTS message. It enters the BEB state and waits to retransmit a RTS. If, however, some other neighbor transmits a RTS to this node, it leaves the BEB state, transmits a CTS (if no neighbor is transmitting a packet or is in the AwaitCTS state) and enters the Await Packet state (i.e., it waits for a packet to arrive). When the packet begins arriving, it enters the Receive Packet state. If it does not hear the packet in the expected time (i.e., round trip time to the transmitter plus some small processing delay at the receiver), it goes back to the Idle state.

When a node begins receiving a packet, it enters the Receive Packet state and immediately transmits a busy tone (whose length is greater than twice the length of a CTS). If the node hears a RTS transmission (directed to some other node) or noise over the control channel at any time during the period that it is receiving a packet, it transmits a busy tone. This ensures that the neighbor transmitting the RTS will not receive the expected CTS. Thus, the neighbours' transmission (which would have interfered with the node receiving a packet) is blocked.

Having mentioned the flow diagram of PAMAS, now we can state how PAMAS solves the data collisions in RTS-CTS-Data exchange. Consider in Figure 4 node B's reception of a packet from A will not be affected by the transmission of a CTS by node C (since these transmissions occur over separate channels). In the second example, in Figure 5, when node B begins receiving the packet from A, it transmits a busy tone that is heard by node C. If the busy tone overlaps with the CTS transmission from node D to node C, node C hears only noise and will enter the BEB state and transmit a RTS again, later. This retransmission of the RTS will be met by another busy tone from B if B is still receiving the packet. This continues until either B finishes receiving or D sends a RTS to C (in this case C may begin receiving a packet from D).

The power saving procedure of PAMAS can be explained as follows:

Nodes consume power while transmitting or even while receiving a packet. Unfortunately, in an ad hoc network, it is frequently the case that a packet transmission from one node to another will be overheard by all the neighbors of the transmitter. All of these nodes will thus consume power needlessly.

In order to conserve power and extend the lifetime of mobile nodes, the PAMAS protocol requires nodes to shut themselves off if they are in a situation where they overhear transmissions. There are two conditions under which it is beneficial for a node to turn itself off.

1. If a node cannot receive data transmissions directed (because a neighbour is transmitting a packet) to it.

2. If it cannot transmit a packet (because a neighbour is receiving another transmission).

Every node in our system makes the decision to power off independently. A node knows if a neighbor is transmitting because it can hear the transmission (over the data channel). Likewise, a node (with a non-empty transmit queue) knows if one or more of its neighbors is receiving because the receivers transmit a busy tone when they begin receiving a packet (and in response to RTS transmissions). Thus, a node can easily decide when to power off. There are, however, two additional questions to be answered:

1. For how long time is a node powered off?

If the node only powers off its data interface but always leaves the signalling interface powered on, then this will enable the node to always know the length of new transmissions and keep the data interface powered off appropriately.

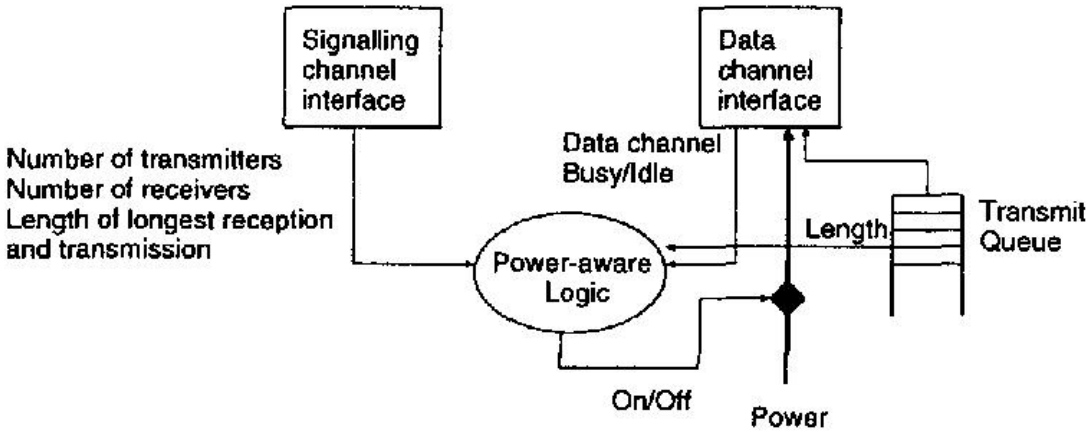

Figure 14 – Separate interfaces for signalling and data

Figure 14 illustrates the block diagram needed for this type of a communications device. Here, the signalling interface listens to all RTS/CTS/Busy Tone transmissions and records the length of each transmission and reception. This information (along with the length of the transmit queue) is fed to the power aware logic which determines whether to turn the data interface off or on.

2. What happens if a neighbor wishes to transmit a packet to a node that has powered itself off? (Consider it from the delay and throughput point of view)

Consider an example. Say we have a line network with four nodes (A-B-C-D) and node B is transmitting to node A. The transmission is overheard by node C (who powers itself off). Say node D has a packet to transmit to node C. Since C is powered off, D's RTSs go unanswered causing D to go into BEB.

What happens if C was not powered off? In this case, since C's neighbor B is transmitting a packet, C will not respond to D's RTSs. Thus, C's behavior, from the viewpoint of D, is the same irrespective of whether C is powered off or not! As a corollary, we can see that packet delays do not increase as a result of powering off nodes. This is because the period of time when a node is powered off is one where it can neither receive packets nor can it transmit packets.

Below the simulation results regarding power conservation in PAMAS are given. Referring to [5], the assumed system model is

1. Random Network Topology, 10 and 20 node network
2. Fixed size data packets (512 bytes), RTS and CTS packets are 32 bytes
3. The busy tone is 64 bytes
4. 1 unit of energy is consumed for 32 bytes of transmission
5. 0.5 units of energy is consumed for 32 bytes of reception
6. Traffic arrivals are Poisson with arrival rate λ
7. The length of the buffer is fixed and packets dropped if the buffer is full.

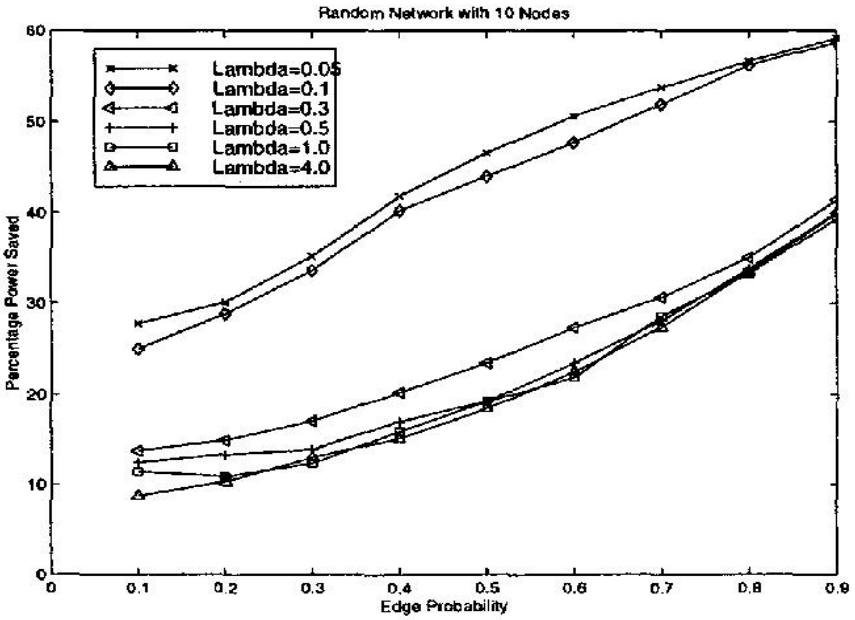

Figure 15 – Power saved in random networks with 10 nodes

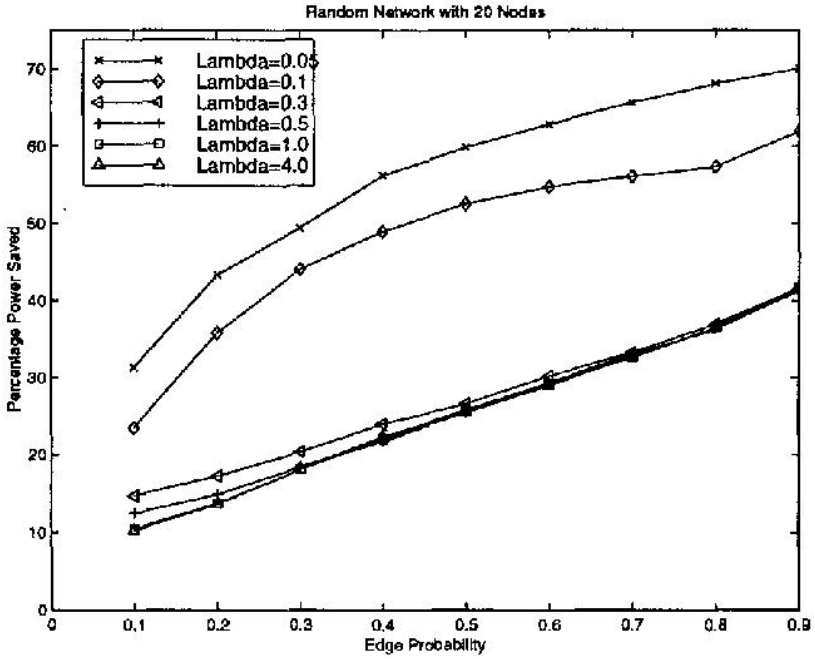

Figure 16 – Power saved in random networks with 20 nodes

Figure 15 and 16 shows that in PAMAS;

- The higher the arrival rate λ, the length of the contention period increases. Therefore the percentage of the power saved decreases.
- Power consumption is reduced if PAMAS is used due to the neighbouring nodes not hearing unnecessary power consuming packet traffic not oriented to themselves

Below given a summary of PAMAS algorithm:

- PAMAS is based on MACA with a separate signalling channel and power conservation by turning mobile nodes data channel off when unnecessary.
- PAMAS solves MACA's problem of collisions during data transmissions (i.e. RTS-CTS-Data exchange, Figure 5 and 6) by separating data and control channels.
- Nodes consume power even while receiving a packet. If a packet is not destined for the node received, power is wasted.
- PAMAS conserves battery power by turning off the data interface when a node can not receive and transmit data. Also, nodes power their data interfaces down if the packet coming is not meant for them. This doesn't affect delay since a node can still hear the control signalling around and can determine exactly when it will turn itself on.
- PAMAS requires Carrier Sensing and a relatively complex circuitry compared to MACA.
- Releasing the capacity in PAMAS is done by turning off the busy tone in control channel by the receiver node and finishing the transmission at the transmitter node.

2. 5. Dual Busy Tone Multiple Access (DBTMA)

In [7], it was shown that "packet sensing" schemes (i.e. RTS-CTS mechanisms), such as FAMA, MACA, MACAW, couldn't solve the problems of hidden terminal problem completely (i.e. refer to figure 5 and 6). DBTMA addresses the hidden terminal problem in RTS-CTS-Data scheme by separating data and control channels.

In the DBTMA protocol, two narrow-bandwidth tones are implemented with enough spectral separation on the single shared channel. Transmit busy tone (BT_t) and receive busy tone (BT_R), indicate whether the node is transmitting RTS packets or receiving data packets, respectively. The transmit busy tone BT_t provides protection for the RTS packets to increase the probability of successful RTS reception at the intended receiver.

We use the receive busy tone to acknowledge the RTS packet and provide continuous protection for the transmitted data packets. All nodes sensing any busy tone are not allowed to send RTS requests. When the start of the signal is sensed, a node sending the RTS packet is required to abort such transmission immediately. Indeed, the RTS packets and the receive busy tone solve the hidden and the exposed-terminal problems.

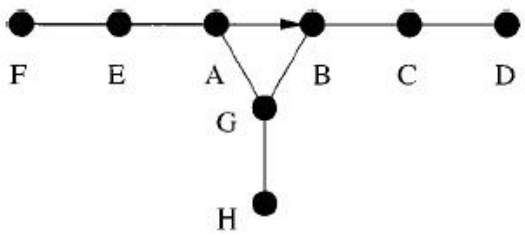

Figure 17 – An example network to demonstrate the hidden and the exposed node problems

The operation of the DBTMA protocol will be explained by the way of a network example, shown in Fig. 17. In this figure, a solid line between any two nodes indicates that the nodes can hear each other. Hence, node C is a hidden terminal to the transmission from node A to node B, and node E is an exposed terminal, if it wants, for example, to communicate with node F (but not with node A).

A node implementing the DBTMA protocol can be in one of the following seven states: **IDLE, CONTEND, S_RTS, S_DATA, WF_BTR, WF_DATA**, and **WAIT**. Fig. 18 depicts the finite state machine (FSM) of the DBTMA scheme.

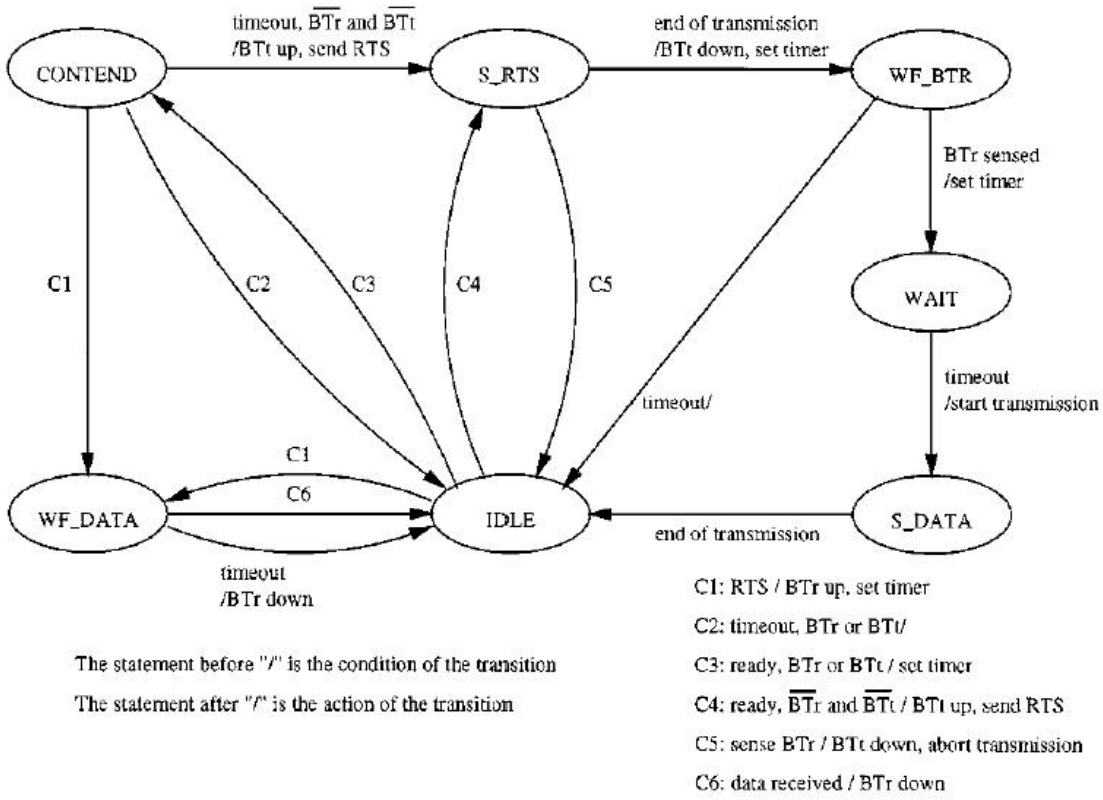

Figure 18 – The state diagram of DBTMA

A node with no packets to send stays in the IDLE state. When a node has a packet to send, but it is not allowed to send the RTS packet, it stays in the CONTEND state. Nodes sending RTS or DATA packets are in the S_RTS or S_DATA states, respectively. The RTS packet sender waits for the acknowledgment from its intended receiver in the WF_BTR state. The receiver waits for the data packet in the WF_DATA state.

When node A has a data packet to send while it is in the IDLE state, it tries to sense the BT_r and the BT_t busy tone signals. If none of the busy signals is present (which means that no one in node A's transmission area is receiving data packet or sending RTS packets), it turns on its BT_t signal, sends an RTS packet to node B, and goes into the S_RTS state. Otherwise, it sets a random timer and goes into the CONTEND state. By the end of the RTS transmission, node A turns off its BT_t signal, sets a timer, and goes into the WF_BTR state. When node B receives the RTS packet, it turns on its BT_r signal, replying to node A and announcing that it is waiting for the incoming data packet. Then it sets up a timer and goes into the WF_DATA state.

Node A continuously monitors the BT_r signal when it is in the WF_BTR state. When a signal is sensed, it knows that its channel request has been successful. Before node A sends the data packet, it waits a mandatory waiting time, t_{MW}, in the WAIT state. This mandatory waiting time is meant to allow all possible RTS transmissions in the range of the receiver to be aborted. Upon timeout in the WAIT state, node A goes into the S_DATA state and sends the data packet. By the end of its transmission, node A goes into the IDLE state. Upon successful reception of the data packet, node B turns off the BT_r signal and goes into the IDLE state, ending the communication. If, for any

reason, node B does not receive the data packet before the timer expires, it turns off the BT_r signal and goes into the IDLE state.

Upon timeout in the CONTEND state, node A turns on its BT_t signal and sends its RTS packet if no busy tone signal is sensed. Otherwise, it goes back into the IDLE state. From the perspective of the other nodes in the neighborhood, their operations can be described as following: When the BT_r and/or the BT_t signal is sensed, a node (e.g., node E, G, or C) is not allowed to send any RTS request. When the start of a BT_r signal is sensed while a node (e.g., node G or C) is in the S_RTS state, it aborts its RTS transmission, turns off its signal, and goes back to the IDLE state. We show the time diagram with the operation of node A and node B in Fig. 19.

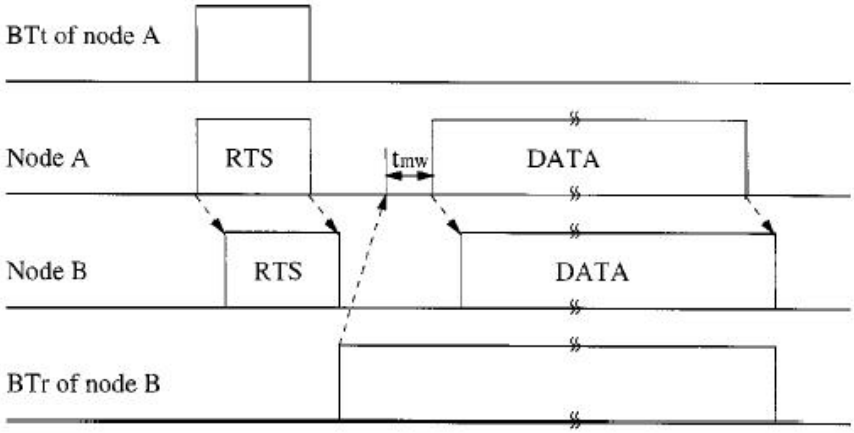

Figure 19 – Time diagram of DBTMA

Additional details of the DBTMA operation rules and a summary of DBTMA are presented in the following, the proof of those can be found in [6].
Summary of DBTMA;

- It has been proved that "packet sensing" RTS-CTS-Data handshake on a single channel for Data and Control signals doesn't prevent hidden terminal problem (refer to Figures 5 and 6) [7]

- DBTMA solves the RTS-CTS-Data handicap by separating data and control signalling into three separate channels. One channel for Data and RTS exchange, one channel for transmitting tone and one channel for receiving tone.

- BT_r and BT_t solves the problem of hidden nodes in DBTMA
- Not hearing BT_r solves the problem of exposed nodes in DBTMA
- BT_t and RTS message reduces the contention probability at the receiver (i.e. the probability of control packet collisions)

2.6. Multiple Access with Reduced Handshake (MARCH) [8]

The main driving forces behind MARCH can be explained as follows:

- In RTS-CTS schemes, the contention time interval to claim the channel can be quite high due to multiple RTS packet collisions especially at high loads.
- RTS-CTS handshake inherently has unnecessary packet transmissions. This can be illustrated as follows;

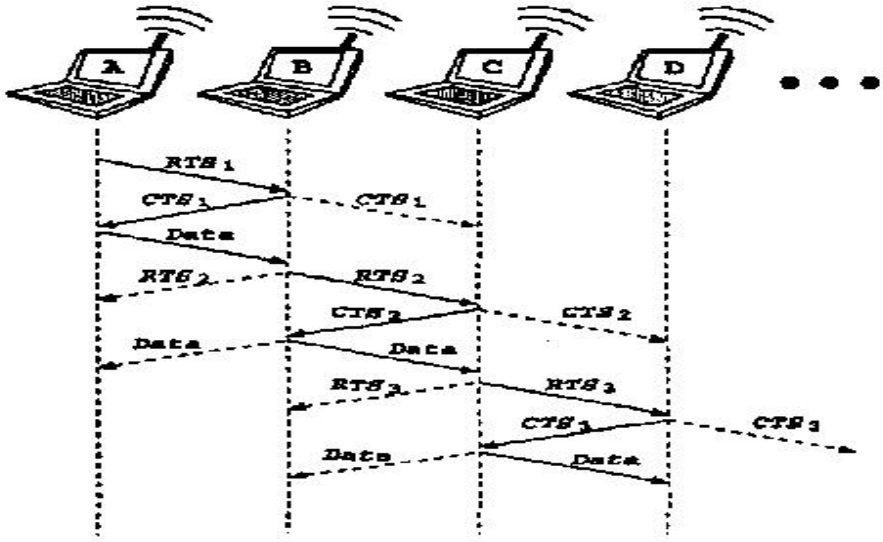

Figure 20 – The RTS – CTS handshake

From figure 20, we observe that to forward a data packet, B needs to transmit two control packets: a CTS_1 packet destined to A and an RTS_2 packet to C. However, due to the broadcast nature of omni-directional antenna, C will receive both the CTS_1 packet and RTS_2 packets. This characteristic implies that overheard CTS_1 packet can also be used to convey information of a data packet arrival at B to C. Then, after the data packet has been received by B, C can invite B to forward that data via the CTS_2 packet and therefore the RTS_2 packet can be suppressed and so on...

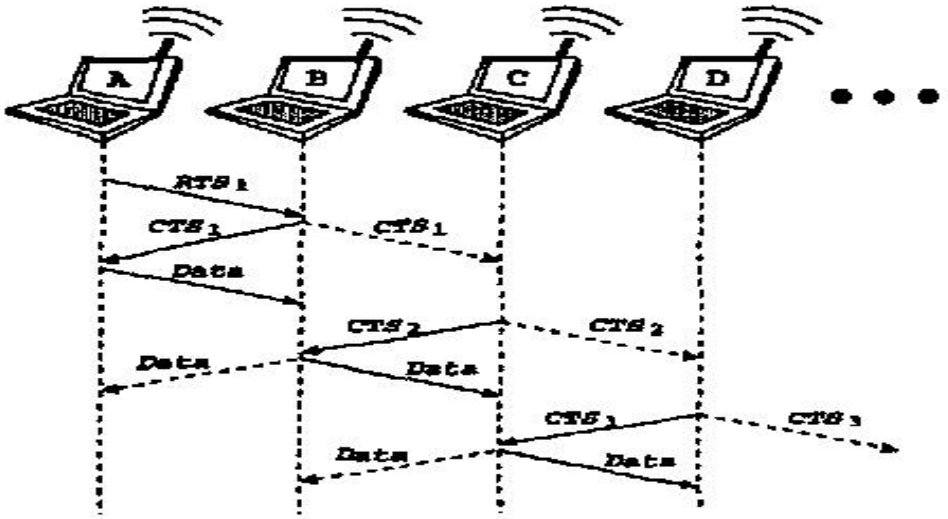

Figure 21 – MARCH Protocol's handshake mechanism

Figure 21 shows the new handshake process to forward a packet through the route. As can be seen, the RTS-CTS mechanism is reduced to a single CTS (CTS-only) handshake after the first hop, and the reduction in the control overhead is a function of the route length.

To support the CTS overhearing mechanism in MARCH, the following information in an CTS/RTS packet is included;

- The MAC addresses of the sender and the receiver
- The route identification number (RT_{ID})

The route identification number is used to prevent unnecessary CTS handshake from the neighbouring nodes who heard the pervious CTS exchange. A neighbouring node sends CTS if and only if it is on the route that has been specified for the data packet.

In MARCH, the MAC layer has access to tables that maintain information on the routes the node participates into, as well as its upstream and downstream neighbours in those routes. This doesn't mean that MARCH performs any network layer routing; it just consults those tables to understand it should respond to a control message (RTS/CTS) particular to a certain route.

The operation of MARCH can be illustrated with the following example.

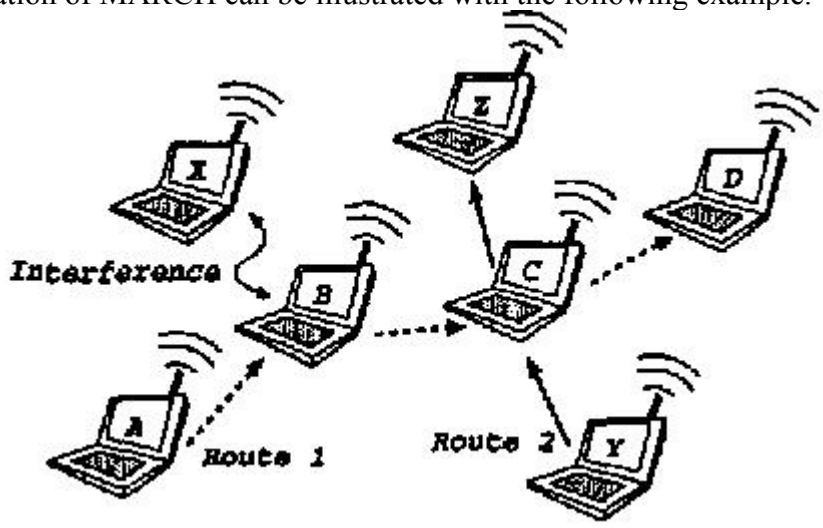

Figure 22- MARCH example

In figure 22, two routes intersect at C. Route one consists of A→B→C→D and route 2 includes Y→C→Z. To begin data transmission in route 1, RTS_1 packet is first sent from A to B. If this packet is successfully received by B, then it will reply with a CTS_1 packet to grant the data transmission. Meanwhile, CTS_1 is also heard by C. According to the MAC address and RT_{ID}, C knows that the packet is sent by its upstream neighbour B in route 1. A timer T_w is then invoked at C. T_w is set to a value long enough for B to receive and process the data packet. Upon timeout, if the channel is free, C sends a CTS_2 packet to B to acquire the data packet. Similarly, D

will overhear CTS_2 sent by C and will subsequently invite C to relay the data packet once its T_w timer expires.

Below given a summary of MARCH;

- In RTS-CTS schemes, the contention time interval to claim the channel can be quite high due to multiple RTS packet collisions especially at high loads.

- RTS-CTS handshake inherently has unnecessary packet transmissions.

- MARCH reduces the amount of control packet exchange. For an ad hoc of L hops, the number of handshakes needed to send a data packet from the source to destination is 2L in MACA, L in MACA-BI, and (L+1) in MARCH. Hence, as L is large, MARCH will have very similar number of handshakes as in MACA-BI.

- Releasing of the channel is performed using CTS. Any neighbouring node will wait for single packet reception time and then try to transmit its own data packet.

- There is a deadlock in MARCH. The success of packet transmission through a route depends on the successful reception of overheard CTS packets. Referring to figure 22, if C receives a RTS packet from Y while B is transmitting RTS_1 to A, then C will not know when the packet has successfully arrived to B. No invitation will take place for the packet in route 1. In this case, B must initiate another RTS-CTS handshake after some time interval and if the channel is free. This point has not been mentioned in [8].

Conclusions

In this report, wireless MAC layer protocols are presented for ad-hoc networks in a competition based environment. In competition based environment, the medium is shared by all neighbouring nodes which are in competition among themselves to seize the channel. Once a node seizes the channel, it prevents its neighbouring nodes to receive or to transmit based on where the neighbouring nodes are. In this type of environment, MAC Layer only deals with per-link connections establishment and cancellation. The aim of MAC layer is to provide orderly and efficient use of the capacity among the nodes while increasing throughput and reducing the delay. Also, due to the infra-structure less property of ad-hoc networks, MAC layer protocols have to be asynchronous. MAC layer protocols have delay, throughput, control overhead and power consumption performance criterions.

The hidden and exposed node problems are the most common problems at the MAC layer, reducing the throughput and increasing the delay. To solve the hidden and exposed terminal problems, RTS-CTS-Data exchange is proposed and the flaws of this method have been described.

Contention causes packet collisions, delays and decrease in throughput. Then, a simple example is given in Figure 3 and the problems that may be encountered during the channel acquisition and data transfer are mentioned.

Different MAC protocols have been suggested in the literature to overcome the limitations of Ad hoc wireless MAC layer. Each protocol has its own advantages and drawbacks.

In the following parts of the paper, several MAC protocols have been described. Most of them try to solve the hidden and exposed node problem using RTS-CTS-Data exchange in an insufficient manner (i.e. MACA [1], FAMA [9]). The separation of control and data channels is used (i.e. DBTMA [6], BTMA) to address the hidden and exposed node problems.

One of the ways to reduce the contention time interval is the invitation scheme (i.e. in MACA-BI). This method only utilizes RTR packets sent by receivers (i.e. receiver initiated MAC protocol). However, one drawback is that the receiver must know that its neighbour has a packet for itself. This requires a very good RTR timing estimation; otherwise the delay and throughput characteristics would be severely affected.

MARCH abolishes the RTR timing estimation requirement of MACA-BI in a receiver initiated manner. It utilizes higher network layer and the broadcast nature of the omni directional antenna. Each RTR packet has the route information and this is used by the next hop on a route to suck the packet up if the channel conditions are appropriate.

Another way to reduce the contention time interval is to use busy tones as in DBTMA [6]. This method separates control and data channels. When a control packet is sent, BT_T is ON reducing the control packet collisions at the receiver. Similarly, to reduce the data packet contentions BT_R is used at the receiver side.

Another issue in MAC layer is to conserve battery. Battery power is wasted by receiving packets not destined for a node. To address this problem and as well as hidden and exposed nodes problems, PAMAS is offered [5]. It is a combination of the original MACA protocol using a separate signalling channel as well as power control.

References:

[1] Phil Karn, "MACA – A New Channel Access Method for Packet Radio," 9.th Computer Networking Conference, pp. 134 – 140, ARRL/CRRL Amateur Radio, 1990.

[2] Fabrizio Talucci and Mario Gerla, "MACA-BI (MACA By Invitation): A wireless MAC protocol for High Speed Ad-Hoc Networking", Proc. IEEE ICUPC '97, 1997.

[3] L. Kleinrock and F. Tobagi. Packet switching in radio channels: Part I – carrier sense multiple access modes and their throughput-delay characteristic. IEEE Trans. Comm., COM-23 n 12:1400-1416, 1975.

[4] C-K Toh, "Ad Hoc Mobile Wireless Networks, Protocols and Systems", Prentice Hall, ISBN 0-13-007817-4

[5] S. Singh and C.S. Raghavendra. "PAMAS - Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks", ACM SIGCOMM, 1999

[6] Zygmunt J. Haas, Jing Deng. "Dual Busy Tone Multiple Access (DBTMA) – A Multiple Access Control Scheme for Ad Hoc Networks", IEEE Transactions on Communications, Vol. 50, No.6, June 2002

[7] Chane L. Fullmer and J.J. Garcia-Luna-Aceves, "Solutions to hidden terminal problems in wireless networks," pp. 39–49, in Proc. ACM SIGCOMM '97, 1997.

[8] C-K. Toh, Vasos Vassiliou, Guillermo Guichal, and C-H Shih, "MARCH: A Medium Access Control Protocol for Multihop Wireless Ad Hoc Networks", p.512-516, IEEE Transactions on Communications 2000.

[9] Chane L. Fullmer, J.J. Garcia-Luna-Acaves, "Floor Acquisition Multiple Access (FAMA) for packet radio networks", ACM Press, p. 262 – 273, 1995, ISBN:0-89791-711-1