A Self-Balanced Receiver-Oriented MAC Protocol for Multiple channels Multihop Ad-Hoc Networks

Hicham Anouar, Christian Bonnet
Institut Eurecom
2229 route des Crêtes, 06904 Sophia Antipolis, France
anouar,bonnet@eurecom.fr

***Abstract*— A novel medium access control protocol for multiple channels ad hoc networks is presented. SEBROMA (self-balanced receiver-oriented MAC) is fully distributed, code assignment free and does not need global network synchronization. The proposed scheme is analyzed through Markov chain modeling in multihop configuration and in saturation condition.**

I. Introduction

The main concern of Medium Access Control (MAC) protocols for wireless network is to share efficiently and fairly communication medium among many contending users. These protocols and their performances differ according to the environment in question and the sysytem requirement to be satisfied. In ad hoc network, multiple stations communicate without presence of any fixed infrastructure. In this case, multiple access is basicaly distributed and random. It has been shown that CSMA based protocols [1] with RTS/CTS [2] handshake mechanism are well suited for single channel systems in single-hop or multi-hop configuration. In multiple channels networks carrier sensing is not always feasible and MAC protocols have to resolve in addition the channel assignement/reservation problem. Several protocols have been proposed for taking advantage of spreading codes for multiple access. Sousa and Silvester [3] analyzed the throughput of some code assignment schemes such as transmitter-based, receiver-based, or transmitter-receiver-based. The channel assignment problem is trivial when the network size is small; it becomes inefficient to assign a unique code to each transmitter or receiver when the network size grows or the topology changes. Another important design aspect of MAC protocols is the choice of the communication initiator. Transmitter based schemes perform better at low loads as collision probability is small while receiver based ones are better suited for high loads. A performance limitation of all random access protocols is that they cannot provide delay guarantees. This occurs at high load when nodes spend most of time trying to resolve contentions, by consequence this leads to a quasi deadlock situation.

In this work we present SEBROMA, a new MAC protocol for multiple channels ad-hoc network, and we derive its performances through a Markov chain modeling. The proposed protocol is receiver-oriented, fully distributed, code assignment free, and does not need global network synchronization. Section II deals with the MAC protocol description and the fundamental design choices behind it. In section III we derive the equivalent Markov chain model for the proposed system in saturation condition and in multihop environment, and use it to obtain the achievable throughput and system delay. Finally in section IV we present our conclusions.

II. Protocol Description

SEBROMA is a realistic and fully distributed multiple access protocol for multiple channels ad hoc networks. The basic philosophy of the developed scheme is to reduce as much as possible signalization overhead and avoid global network synchronization due to the difficulties related to its practical realization. All nodes are given the same responsibility (i.e flat architecture), thus, single points of failure are avoided and the protocol is topology transparent. To minimize collisions at high loads, each communication is initiated by receivers. Hence, only local synchronization is performed between each receiver and its intended transmitters and eventually maintained for data transfer between the receiver and the contention's winner. Multiple channels system is considered, where a common channel is used for signalization and other channels are used randomly by all nodes for communications setup and data transfer. This simplifies the code assignment functionality since no inter-node collaboration is needed.

Each node transmits, after a random time period of mean T without activity (i.e. without getting a packet to transmit from the uplayer), on the common channel, an invitation message RTR (ready to receive) containing a synchronization sequence, its ID and a code (channel) ID randomly chosen at each RTR message transmission (Fig. 1). The synchronization sequence allows the listening nodes to detect the transmission of the RTR message and get synchronized with its sender in order to be able to correctly receive its message. The code in the RTR message is used later for communication setup. Randomly choosing this code, at each RTR message sending, avoids the need of a centralized code assignment. Collision on the common channel is reduced by the use of random period T and it's resolved by making multiples attempts.

Conversely, if the node get a packet to transmit durring T, it starts a random time period of mean T_{out} waiting for RTR messages.

The RTR message is followed by the communication setup phase realized on the data channel chosen by the receiver and

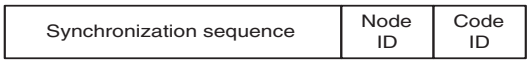

Fig. 1. RTR message form

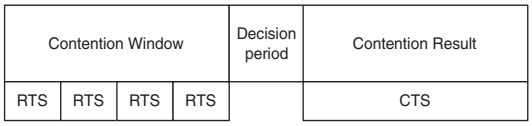

Fig. 2. Communication setup on the receiver code

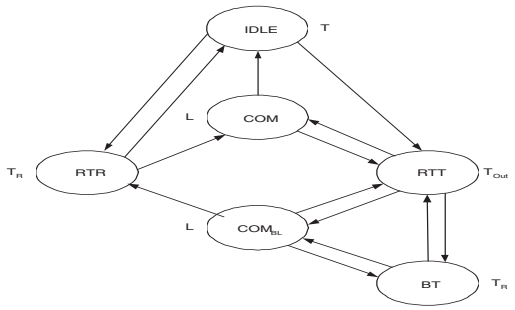

Fig. 3. Modes diagram

denoted in the RTR message. It consists on a contention phase and eventually data transfer phase. A contention window is dedicated to the reception of the Request to Send (RTS) messages and it is divided into R several contention sub-windows in order to reduce the collision probability.[1] Among the successfully decoded RTS messages, the receiver answers the accepted request [2] by sending a Clear to Send (CTS) message (Contention resolution phase). Then data transfer can start (Fig. 2). Once again, Contention among transmitters is reduced by the use of random waiting periods T_{out} and multiple contention sub-windows and resolved by making multiples attempts.

In classical approaches, every node is given some credit (time, attempts...) to send its packets. In case of failure, the packet is assumed lost and the node restarts later with a new packet. Our aproach to resolve this problem is quite intuitive; If a transmitter fails to reach its destination in a given random time period T_{out} (because of collisions or a not available receiver...), it sends directly an RTR message and tries to receive others nodes traffic. Right after the end of its RTR message, if no communication is successfully setup, or after the end of the data traffic transfer in the contrary case, the transmitter restarts a new random waiting period T_{out} for invitation messages RTR. The transition between transmission and reception phases is repeated until succesfull transmission of the packet, it allows to unlock situations where numerous transmitters are trying simultaneously to initiate a communication, which keep all of them blocked infinitely. So, in SEBROMA, each node carries fairly other nodes's traffic as well as its own traffic, this ensures the permanent presence of receivers in the network and hence maintains the network communications capability. This is what we call **self-balancing** aspect of the protocol (in sense of number of receivers and transmitters simultaneously present in the network)

A. state diagram

We define the system states as follow (Fig. 3):

1) *Idle state:* a node is given a random time periode of mean T for getting packets from its uplayer. Durring this time, it's said to be in the *Idle state*. If no new packet is received during T, the node becomes Ready Receiver (RR) and passes to the *RTR state*, otherwise it becomes Ready Transmitter (RT) and passes to the *RTT state*.
2) *Ready To Receive state:* A node is said to be in the *RTR state* when it is sending RTR message, if it succeeds to initiate a communication it passes to the *COM state*, otherwise it goes back to the *Idle state*.
3) *Ready To Transmit state:* A RT is given a random time periode of mean T_{out} to reach its destination. If it does not succeeds, it becomes directly RR and passes to the *BT state* (ready receiver but with a packet to transmit: blocked transmitter (BT)), otherwise it passes to the *com state*.
4) *COM state:* A node (RR or RT) is said to be in the *COM state* when it is communicating. In case of collision on the data channel, the RR goes back to *Idle state* and the RT goes back to the *RTT state*. Otherwise, at the end of the communication, the RR and RT passe to the *Idle state*.
5) *BT state:* A node is said to be in the *BT state* when it is sending RTR message after expiration of its time period T_{out}. If it succeeds to initiate a communication (as receiver), it passes to the COM_{BL} *state*, otherwise, it goes back to the *RTT state*.
6) COM_{BL} *state:* A node (RT or BT) is said to be in the COM_{BL} *state* when it is communicating . In case of collision on the data channel, the BT goes back to *RTT state* and the RT passes to the *BT state*. Otherwise, at the end of the communication, the BT passes passes to the *RTT state* and RT passes to the *RTC state*.

B. Example

Figure (4) illustrates message exchanges in the main network scenarios.
In the first case, node A has no packet to send after time period T on the *Idle state*, so node A becomes ready receiver, sends RTR message on the common channel and waits for reply from any correspondent. Node B is ready transmitter and has a packet for node A, so it sends RTS message on the data channel choosen by A. node A replies then with a CTS,

[1] We may reserve one or several higher priority sub-windows for multicasting, handover traffic...

[2] Based on requests priority, capability criterion...

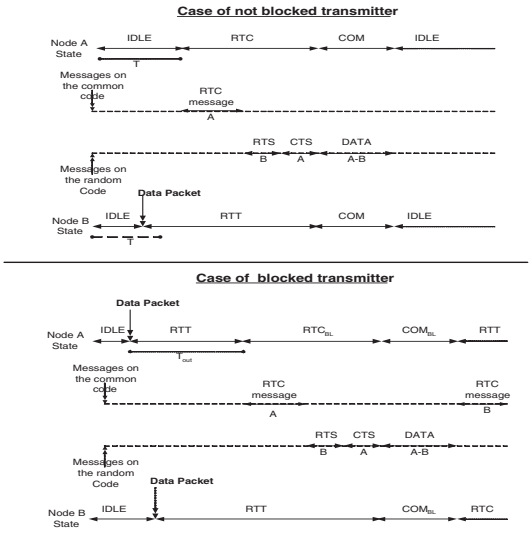

Fig. 4. Example of control messages exchanges

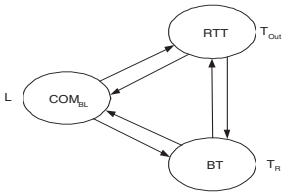

Fig. 5. States diagram in heavy load situation

and the data transfer starts. At the end of the communication, the two nodes return to the Idle state.

Now, in the second case, node A has received a packet to send from the up-layer durring T, it enters then the *RTT state* and waits invitation message RTR from its destination. Node B receives a packet to send to node A, he enters also to the *RTT state* and wait invitation message from its destination. After a random time period T_{out} in the *RTT state*, node A did not success to reach its destination, it enters to the *BT state* (since he can not transmit, it tries to receive) and sends an RTR message (as blocked transmitter), node B is still in the RTT state and has a packet for it, so they start a communication: A and B are now in the *COM_{BL} state*. At the end of the communication, node A returns to the *RTT state* while node B, as it was served by a BT, goes to the *RTR state* and tries also to serve other nodes.

III. Saturation Throughput Analysis

In heavy load situation, nodes in Idle state get instantaneously packets to transmit. The Corresponding state machine in this case is reduced to the one given in figure (5). We analyse the system in this situation considering the following assumptions:
nodes are bi-dimesional poisson distributed on an infinite area (no edge effect) with average number of terminals per unit area λ. We assume that a new sample of the spatial distribution is given for every RTR message transmission. All nodes have the same circular transmitting and receiving range of radius r. Nodes traffic is uniform over a circular range of radius R (routing area). Each terminal is assumed to know the next destination of a packet in its range, this destination is assumed to be the best route for the packet. The physical layer offers (D+1) orthogonal and identical channels (D for data and one for signalization); only one transmission at time is allowed on every channel. There are L contention sub-windows for the reception of RTS messages. The elementary time unit is taken equal to the duration of the RTR message T_R. Random waiting periods for transmitters are exponentially distributed with mean T_{out}. Packets length is assumed to be exponentially distributed with mean l.
We take $N+1$ as the average number of nodes in a circular range of radius r, $N+1 = \lambda\pi r^2$. So every node has in average N neighboors (N is the nodes's degree). Since all nodes have the same capabilities, the same requests and evolve in the same environment, their performances are then similar. Observe that we have made no hypotheses about existance of any exception in the model.
We use a three-dimensional continuous-time Markov chain to model the system's behavior in the range of some given node. The resulting system can be viewed as a closed network of queues with network's state dependent routing probabilities. Towsley [4] showed that this type of systems still have product-form solution. In fact, there is no queueing in each state and the system is simply a delay model. Applying then Mean Value analysis (MVA)[5] gives us the relation:

$$Q_N(i) = N.Q_1(i) \tag{1}$$

Where $Q_N(i)$ denotes the mean number of customers in queue i where there is N customers in the system. We use a modified Equilibrium Point Analysis (EPA) to derive our system's performances. We suppose that at equilibrium point, the N neighboors of some given node are in the state $N_e = (N_t, N_r, N_c)$ where N_T is the number of ready transmitters, N_R the number of ready receivers and N_C the number of communicating stations. We analyse then the system for the node in the center of this range and we deduce N_e by equation (1). Hence, equilbrium point provide us with the mean routing probabilities.

A node in the *RTT state* transites to the *Com state* if and only if there is only one RTR message sent in it's range, the RR is the destination of its packet, there are no collisions on the data channel choosen for communication setup and no collisions on the choosen RTS sub-window.

$$\begin{aligned} P(\text{only one RTR in the range of the RT}) &= N_r e^{(-N_r)} \\ P(\text{RR is the destination of RT}) &= \frac{1}{N} \\ P(\text{No collision on the choosen data channel}) &= 1 - \frac{N_c}{2D} \\ P_{CR} = P(\text{collision on RTS sub-windows}) &= \left[\sum_{i=2}^{N_t} C_{N_t}^i \left(\frac{1}{LN}\right)^i\right] \end{aligned}$$

We get:

$$P_{CR} = \left[\left(1+\frac{1}{LN}\right)^{N_t} - \frac{N_t}{LN} - 1\right] \leq \left[\left(1+\frac{1}{LN}\right)^{N} - \frac{1}{L} - 1\right]$$

A sufficient condition to get P_{CR} lower than some given threshold y is to have $L \geq \sqrt{\frac{1}{2y}}$. For example, to get $P_{CR} \leq 0.01$ we need to put $L \geq 7.08$. In the following, we suppose that this is the case and we neglect then collisions probabilities on RTS sub-windows. So

$$P(RTT \rightarrow COM) = \frac{N_r e^{(-N_r)}}{N}\left[1 - \frac{N_c}{2D}\right] \quad (2)$$

Observe that the receiver node doesn't care if there is another RTR message sent in it's range since communication setup is not performed on the common channel. This improve greatly the performances of the protocol in multihop environment.

A node in the *RTR state* transites to the *Com state* if and only if there is one RT in its range who succeed to initiate communication with it.

$$P(RTR \rightarrow COM) = \frac{N_t e^{(-N_r)}}{N}\left[1 - \frac{N_c}{2D}\right] \quad (3)$$

A node in the *Com state* transites to the *RTT state* or the *RTR state* if and only if a collision occurs on the used data channel or in the contrary case the communication finishes successfully. In the two cases, we suppose that nodes stay in *Com state* for the duration of the packet.

$$P(COM \rightarrow RTR) = P(COM \rightarrow RTT) = \frac{1}{2} \quad (4)$$

Taking into account equation(1), the global balance equations can be written as:

$$\frac{N_t}{T_{out}} = \frac{N_r\left[1 - \frac{N_t e^{-(N_r)}}{N}\left(1 - \frac{N_c}{2D}\right)\right]}{T_r} + \frac{N_c}{2l} \quad (5)$$

$$\frac{N_r}{T_r} = \frac{N_t\left[1 - \frac{N_r e^{-(N_r)}}{N}\left(1 - \frac{N_c}{2D}\right)\right]}{T_{out}} + \frac{N_c}{2l} \quad (6)$$

$$\frac{N_c}{l} = \left[\frac{N_r N_t e^{-(N_r)}}{N}\left(1 - \frac{N_c}{2D}\right)\right]\left[\frac{1}{T_r} + \frac{1}{T_{out}}\right] \quad (7)$$

Where the normalizing equation is

$$N_c + N_r + N_t = N \quad (8)$$

To make the protocol fair, a necessary condition is to have flows from *RTT* and *BT* states to COM state equals, otherwise users will be penalized when alternating between transmission and reception pahses. A sufficient condition is then to put $T_{out} = T_r$.
In this case we have

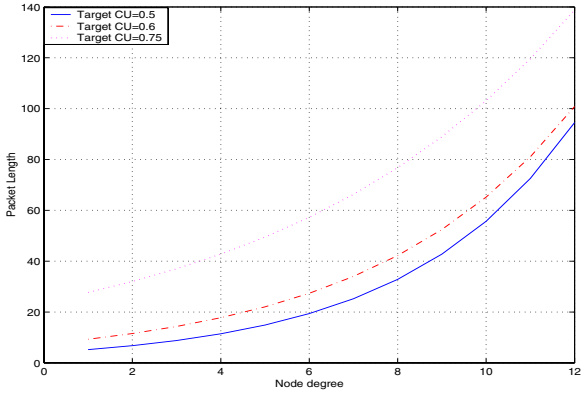

Fig. 6. Packet Length Vs Node degree

$$N_c = \frac{2Dx^2 e^{-(x)}}{x^2 e^{-(x)} + \frac{NDT_r}{l}} \quad (9)$$

Where $x = N_t = N_r$.
In single channel network, carrier sensing improve performance of MAC protocols by providing channel state information to the contending users. In multiple channels network where carrier sensing is not feasible and where the access is random, one possible way to reduce collision is to use long data packet. In this case, nodes involved in data transfer will stay long time in data channels and reduce then contention on the common channel amomg remaining nodes. In the following, we calculate the necessary packet lenght l needed for achieving some target channel utilization (single hop CU) $CU = Nc = N - 2x$, we find that the corresponding l is

$$l = \frac{ND\left[N - 2\epsilon\right]T_r}{x^2 e^{(-x)}\left[2D - (N - 2x)\right]} \quad (10)$$

This gives us a lower bound on x: $x \geq \frac{N}{2} - D$.

Figure (6) gives examples of needed packet lenght to achieve different normalized channel utilization for different values of N where using 20 data channels.

In fact, channel utilization is reduced by collisions on data channels. A collision on a currently used data channel occures if a communication is successfully setup on this data channel in the range of the transmitter or the receiver, the corresponding probability is given by:

$$P_{cd} = \frac{2x^2 e^{-x}}{ND} \quad (11)$$

Taking into account this probability, the channel utilization becomes

$$CU = \frac{2Dx^2 e^{-(x)}}{x^2 e^{-(x)} + \frac{NDT_r}{l}}\left(1 - \frac{(N-2x)x^2 e^{(-x)}}{ND}\right)$$

$$CU \geq \frac{2Dx^2 e^{-(x)}}{x^2 e^{-(x)} + \frac{NDT_r}{l}}\left(1 - \frac{1}{eD}\right) \quad (12)$$

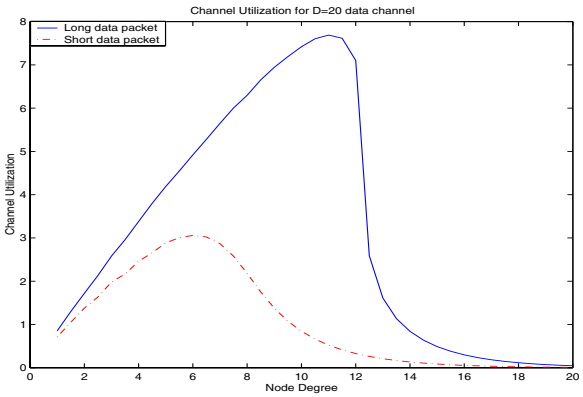

Fig. 7. Channel Utilization Vs Node degree

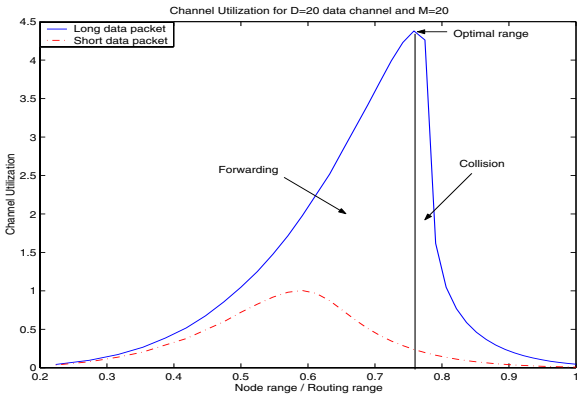

Fig. 8. Channel Utilization Vs Mac/Routing ranges ratio

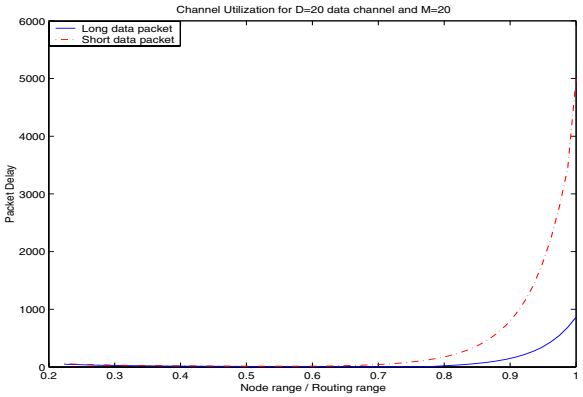

Fig. 9. Packet delay Vs Mac/Routing ranges ratio

Figure (7) shows the obtained channel utilization versus node's degree for long and short data packet ($l = 100, l = 10$) when using 20 data channels. We observe that CU increase linearly as node degree is lower than 11 (long data packet) and then decrease exponentialy. We conclude that, for some given data packet lenght and some number of data channels, there exist a threshold on node's degree above wich performances degrade severely due to a high collisions probability.

The traffic is supposed uniform over a circular routing area of radius R, the average number of forwarding operations is then
$f = \frac{R^2}{r^2}$, and the average end-to-end channel utilization in this case is:

$$CU_e \geq \frac{2Dx^2e^{-(x)}}{x^2e^{-(x)} + \frac{NDT_r}{l}} \left(1 - \frac{1}{eD}\right) \frac{1}{f} \tag{13}$$

The corresponding delay is:

$$Delay \leq \frac{2N}{CU_e} \tag{14}$$

We take (M+1) as the mean number of node in the routing range and we define the Mac/Routing ranges ratio as r/R . Figure (8) shows the end-to-end channel utilization (CU_e) Versus the Mac/Routing ranges ratio for M=20 and D=20. Figure (9) shows the corresponding end to end delay. We observe that there exist an optimal mac range for which CU_e is maximal. This can be explained as follow, bellow the optimal range, the single hope channel utilization is high as well as the number of forwarding operations needed to deliver the packet to its end destination. While above optimal range, single hop CU is low as well as forwarding operations.

IV. Conclusion

We have presented SEBROMA, A MAC protocol for multiple channels ad hoc networks. SEBROMA is receiver-oriented, fully distributed, asynchronous and code assignment free. Fair transitions between transmission and reception phases permit the permanent presence of receivers even at high load. Obtained performances of the protocol in saturation condition in multihop configuration are shown to be superior than those of existing protocols. Degradation in performances due to the use of random access to data channels is compensated by the establishment of RTS/CTS handshake on data channels and not on the common channel which make the protocol much robust in multihop environment. In future work, we will focus on analysing the system for general load.

References

[1] Kleinrock L., Tobagi F.A., "Packet switching in radio channels: PartI - Carrier sense multiple-access modes and their throughput-delay characteristics " IEEE Transactions on Communications, Vol. 23 Issue 12 , December 1975

[2] Garcia-Luna-Aceves, J.J. , Fullmer, C.L. , "Performance of floor acquisition multiple access in ad-hoc networks " Computers and Communications, 1998. ISCC '98. Proceedings. Third IEEE Symposium on , 1998 PP. 63-68

[3] Sousa, E.S., Silvester, J.A. , "Spreading code protocols for distributed spread-spectrum packet radio networks" IEEE Transactions on Communications, Vol. 36 Issue 3 , March 1988 PP. 272-281

[4] Towsley D., "Queuing Network Models with State-Dependent Routing" Journal of the ACM, Vol. 27, Issue 2, April 1980, PP. 323-337

[5] Reiser, M., Lavenberg S.S., "Mean-value analysis of closed multichain queueing networks" Journal of the ACM, Vol. 27, Issue 2 , April 1980, PP. 313-322 Pages: 272 -281

[6] Kleinrock L., "Queueing systems Volume 1: Theory" John Wiley & Sons, 1st ed. , 01/1975

[7] Trivedi, K. S., "Probability and statistics with reliability, queuing, and computer science applications" John Wiley & Sons, 2nd ed. , 10/2001