QoS Routing for Real-time Applications in CDMA Based Ad Hoc Networks

Lin Xiao *Student Member, IEEE* and Eliane Bodanese

***Abstract* — Recently, mobile ad hoc networks have obtained a growing interest because of their advantages in many practical applications. One of the crucial points is the Quality-of-Service (QoS) routing for real-time applications, which requires sufficient and constant bandwidth. Therefore, a routing protocol should also consider the definition of an efficient medium access control (MAC) scheme in a cross layer design. A QoS routing protocol is proposed in this paper to calculate, allocate and reserve resources for CDMA based ad hoc networks with route request. The protocol is called CDMA Bus Lane routing protocol.**

***Keywords* — ad hoc networks, CDMA, routing protocol, QoS, real-time**

I. INTRODUCTION

Together with the growing demand for user mobility, application requirements are becoming more stringent, especially in the area of real-time services, making the support for quality of service (QoS) in ad hoc networks a necessity.

Most routing protocols for mobile ad hoc networks, such as AODV [1], DSR [2], and TORA [3] are designed for best effort routing without considering QoS requirements. Some proposed QoS routing protocols, such as CEDAR [4] and ticket-based probing algorithm [5], are not implemented above a specific MAC layer. In fact, the ability to provide QoS is heavily dependent on how well the resources are managed at the MAC layer. The IEEE 802.11 MAC protocol has been regarded as unsuitable for ad hoc networks [6], and some proposed TDMA schemes [7], [8], [9] only make slot reservations within the neighbourhood of the nodes. The scheduling along a whole communication path is not considered. Therefore, an approach to implement QoS in ad hoc networks is to consider the network layer along with the lower layers. QoS routing protocols proposed in [10] and [11] combine network and TDMA MAC layers together to better support QoS. However, compared with TDMA, CDMA is a widely implemented technology for today’s wireless communications and allows terminals to use the entire channel bandwidth at the same time through different spreading codes. Some researchers focus on using CDMA as MAC layer ([12], [13]) for ad hoc networks. However, they do not consider routing issues.

The CDMA Bus Lane is a novel solution which combines different layers to offer better QoS in mobile ad hoc networks. The network structure and code allocation algorithm for the CDMA Bus Lane have been proposed in [14] by the same authors. This paper mainly presents further work on the Bus Lane QoS routing protocol which combines the resource calculation, allocation and reservation during the on-demand route discovery stage.

Firstly, this paper introduces the model and assumptions of the network. The code allocation algorithm with time scheduling is also presented. Then, the QoS routing protocol for the CDMA Bus Lane is proposed in detail. Finally, the simulations and results are presented and discussed.

II. NETWORK MODEL AND ASSUMPTIONS

Consider a multihop ad hoc network through a common wide-band spectrum spreading channel using CDMA. The traffic is classified into two basic categories: data traffic and real time traffic.

The available bandwidth in a network is divided into two non-overlapping channels: all data traffic shares the same spreading code channel. CDMA Bus Lanes are only built and reserved for real-time traffic. The communication framework is shown in Fig. 1.

On-demand routing	
CSMA/CA (for data, signalling messages)	Code channel allocation + TDMA (for real-time
Contended Single code channel	Scheduled multi-code channel

Fig.1. The CDMA Bus Lane communication framework

Some assumptions need to be made in order to narrow a network scenario for the application of the CDMA Bus Lane scheme:

1. Two sets of transceivers are implemented in each mobile node: one is for data traffic, signalling and message packets; the other is for real-time sessions.
2. A node can transmit and receive at the same time through the use of two orthogonal codes separately at the transmitter and receiver.
3. An ideal PHY layer is assumed. This means the interference from background noise is ignored.
4. A node cannot receive or transmit multiple packets simultaneously for real-time sessions. The multiple real-time sessions across the same node are scheduled in time slots.
5. The link between two nodes is symmetric.
6. The spreading code set $C = \{c_1, c_2, c_3 \ldots c_m\}$ with different lengths are generated by an Orthogonal Variable Spreading Factor (OVSF) tree according to a *n-codes-remain* scheme [14], which means that there are *n* codes left in each level on the tree in order to generate usable codes for the next level.

III. Code Allocation Issues

A. Code and Bandwidth

In the CDMA Bus Lane, the spreading codes are the resources to be allocated to each node along the paths. Spreading codes are sequences of pseudo random bits that are used to expand the bandwidth occupation in a CDMA system. Data symbols are multiplied by orthogonal spreading sequences. The 3GPP specifications restrict the attention to the case of codes with block length:

$$N_c = 2^m = SF \qquad (1)$$

N_c represents the number of chips per data symbol, and also the spreading factor (*SF*). It is important to note that a low spreading factor allows communication at a higher data rate, but at a cost of having less available spreading codes. Contrarily, a higher spreading factor allows communication at a lower data rate, but more spreading codes are available. A chip rate R_c can be represented by the data rate (R_d) and the *SF*.

$$R_c = R_d \times SF \qquad (2)$$

However, in a real-time communication, the data rate of a session (R_s) is constant. Given a *SF*, the required chips per second (*Cps*) for the stream is:

$$Cps = SF \times R_s \qquad (3)$$

Each session across a node takes the necessary number of slots to transmit *Cps*, because the bandwidth in each node is partitioned into a set of time slots $S = \{s_1, s_2 \ldots s_n\}$, which composes a frame. In this way, a node can handle multi-transmission from different sessions.

For a chosen *SF* of a session across a node, the value of *Cps* should be less than or equal to the available transmission chip rate R_a ($Cps <= R_a$). R_a is defined by the remaining available time slots.

Therefore, the channel bandwidth in the node R_c can be represented by the data rate and *SF* of crossing sessions and the available bandwidth R_a.

$$R_c = \sum (R_{session(i)} \times SF_{session(i)}) + R_a \ (R_a >= 0) \qquad (4)$$

B. Path Code Allocation and Bandwidth Calculation

An ad hoc network is represented by a graph *G* with a set of nodes *N* and a set of links *L*, *G = (N, L)*. The code allocation and bandwidth calculation along a given path $P = \{n_0 \rightarrow n_1 \rightarrow \ldots \rightarrow n_i \rightarrow \ldots \rightarrow n_d\}$ is processed hop by hop from source to destination. A code can be reused outside its interference range (two node hops). Therefore, in order to calculate the available transmission code set TC_i^p *(i=0,1,…,d-1)*, where *p* means the path *P* and *i* means node n_i, for link *l(i,i+1)*, the local active transmission code schedule around n_i and n_{i+1} should be considered. According to the code allocation algorithm in [14], RC_i represents the links that n_i receives from its neighbours (NB_i), that we call the reception codes of n_i; TC_i are the links that node n_i transmits (transmission codes of n_i), the same applies to n_{i+1} (RC_{i+1}, TC_{i+1}). RC_x are the reception codes of the nodes in NB_i. TC_y are the transmission codes of the nodes in NB_{i+1}.

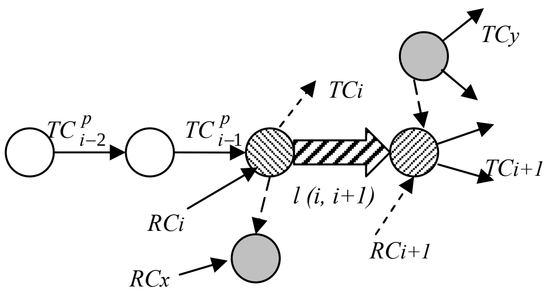


Fig.2. Code allocation for link *l (i, i+1)* on the Path *P*

Fig.2 shows the information that should be considered when allocating a code for *l (i, i+1)* on the path *P*. All the links represented by solid arrows affect the allocated transmission. Obviously, the codes in RC_i, TC_{i+1}, TC_{i-1}^p and TC_{i-2}^p cannot be used for the link, but the codes in TC_i or RC_{i+1} (dashed links) can be reused, because these links are scheduled in time. From the view of node n_i, the link *l (i, i+1)* should also avoid interference with RC_x. However, not all of those codes affect the code allocation. We can ignore the reception codes of RC_x originated from transmissions of n_i, which belong to TC_i, and all the reception codes of the special neighbour n_{i+1}, which has been represented by RC_{i+1}. So, the interference code set from NB_i can be represented by NC_x:

$$NC_x = RC_x - TC_i - RC_{i+1} \qquad (5)$$

In the same way, the interference code set from NB_{i+1} can also be represented by:

$$NC_y = TC_y - TC_i - RC_{i+1} \qquad (6)$$

Therefore, the set of available transmission codes AC_i^p, which can be used for transmission from n_i to n_{i+1} without interference can be described by:

$$AC_i^p = \overline{RC_i \cup TC_{i+1} \cup NC_x \cup NC_y} - TC_{i-1}^p - TC_{i-2}^p \quad (if\ x<0,\ TC_x^p = \emptyset) \qquad (7)$$

Also, the selected codes must have a reasonable *SF* to satisfy the bandwidth requirement of the session.

$$R_s \times SF <= R_c - \sum Cps \qquad (8)$$

IV. QoS Routing And Resource Reservation

The CDMA Bus Lane with QoS implements the code allocation algorithm on an on-demand routing protocol, so that, a route with sufficient bandwidth from the source to the destination can be found and reserved at the route discovery stage. This cross-layer design combines the network layer and the MAC layer to better support the QoS for real-time communications in mobile ad hoc networks.

RREQ_B and *RREP_B* are messages used at the route discovery stage. Aside from the attributes in the classic AODV *RREQ* [1], the *RREQ_B* for the CDMA Bus Lane also contains bandwidth requirements for the session and the local resource information for code and slot allocation. The node who receives the *RREQ_B* will calculate the bandwidth and allocate a valid code and the bandwidth for the link from where the *RREQ_B* was sent. The *RREP_B* is back from the destination to confirm the reservation in each node.

A. Information Maintenance and Exchange

In order to allocate code and slots with a route request, some information should be maintained and exchanged among nodes.

Each node needs to maintain the following tables to record all the necessary information:

Bus Lane routing table*:* It records the routing information for all the QoS sessions across the node. Each session is identified by the source and destination IP address pair.

Transmission (Reception) code table*:* It records the code information of the links across the node, including the code ID, expiry time, address of the next hop (last hop) and the slots used for a transmission (reception).

Transmission (Reception) slot table: It is a simple table that records which slots are used and which code is assigned in each time slot for transmission (or reception). It works at the MAC layer to control the code channel implementation in the time domain, and the packet transmission (or reception) in each time slot.

Seen requests table: It records the *RREQ_B* requests which have been seen and sent out by the node. As the table in the AODV routing protocol [1], it avoids the loop at route discovery stage.

Pre-reserved table: Because the *RREQ_B* is a broadcast, the codes and slots will be allocated on multiple routes with the *RREQ_B*. All the resources allocation in this stage is just pre-reserved. Only the node who received the *RREP_B* can reserve the resources for the incoming session.

Neighbouring information table: It records the information from all the neighbours of the node. A timer is set up to keep the information updated.

Hello Messages can maintain the active links as well as exchange information among neighbours. A node broadcasts a *Hello Message* to its neighbours at every *HELLO_INTERVAL* time, which allows the node to hold the latest information from its neighbours. When a node receives this message, it updates the corresponding entries of its *Neighbouring information table*. If a node never receives it from a neighbour for a certain time, it will no longer consider that node as its neighbour. If the neighbour is on an active route, a breakage will be reported.

B. The Code Assignment during the Route Discovery

The code allocation algorithm proposed in [14] works on pre defined routes and assumes all the useful information is known by the nodes. However, when allocating codes and slots at the route discovery stage, there is no fixed route to follow, and the information is separately maintained among the nodes in the network, which increases the degree of difficulty.

From the view of an *RREQ_B* sender, all the neighbouring nodes could be the next hop on the route. It is impossible for the *RREQ_B* sender to do the calculation and allocation for all these links before the request packet is sent out to its neighbours. However, the node who receives an *RREQ_B* can get sufficient information from one unique last hop node, which makes the calculation simpler. Therefore, the decisions of the code for any hop should be made in the nodes who receive the *RREQ_B*.

According to (5), (6) and (7) the *RREQ_B* needs to present information about RC_i, RC_x, TC_i and the pre-allocated codes for nodes n_{i-1} (TC_{i-1}^{p}) and n_{i-2} (TC_{i-2}^{p}), which affect the code allocation of n_i. However, simply appending them on the *RREQ_B* would make the packet very large. An array called *F_code [codeID]* is used to simply record the availability of the code according to the information received from the sender. That means, part of the code selection task should be performed by the *RREQ_B* sender. The index *codeID* represents the ID of the code in code set *C*. There are three different values which represent the availability of the corresponding code in the array (0, 1 and 2).

F_code[x] = 0 means the code c_x is not in the forbidden code set as far as node n_i knows, which can be allocated as transmission code to next hop from the view of the node n_i.

Value "1" means only one neighbour of n_i uses this code as a reception code, and the transmitting source of this reception is not node n_i.

The element is set to "2" if at least one of the conditions below is true:

- There is more than one neighbour of node n_i using this code as a reception code, and none of these receptions are due to transmissions from node n_i.
- The code is in the set RC_i, which is forbidden for the code allocation.
- The code is either TC_{i-1}^p or TC_{i-2}^p.

For an element with value "2", the corresponding code is not allowed to be allocated.

When a neighbouring node n_{i+1} receives the *RREQ_B*, it checks the array *F_code*. It firstly eliminates all the matching codes with RC_{i+1} (the receptions of the node itself) by subtracting one from the corresponding elements in the array whose value are not zero. In this way, node n_{i+1} takes the share back from the corresponding elements in the array which were counted as reception codes of a neighbour by node n_i. After that, the elements of the array whose values are still not zero are the ones totally forbidden for the code allocation. They can be described as:

$$NC_i \cup RC_i \cup TC_{i-1}^p \cup TC_{i-2}^p \tag{9}$$

Then the forbidden codes from node n_{i+1} and its neighbours ($NC_{i+1} \cup TC_{i+1}$) are easily found with the information from node n_i. A value is set at each corresponding element in the array *F_code.* The forbidden codes for this link can be collected by a code set FC_i^p from the array *F_code*; they are the codes represented by the elements whose values are not zero. The forbidden code set is represented by:

$$FC_i^p = NC_i \cup RC_i \cup NC_{i+1} \cup TC_{i+1} \cup TC_{i-1}^p \cup TC_{i-2}^p \tag{10}$$

The available codes in the code set AC_i^p can be easily collected from the elements of the array *F_code*, whose value are zero. It is represented by:

$$AC_i^p = \overline{FC_i^p} \tag{11}$$

The code with smaller ID is selected to be the reception code of n_{i+1}, because a smaller *SF* uses fewer slots allowing more sessions across the node.

The *RREQ_B* also contains an array with the slots being used for the transmissions in node n_i. It is very easy for node n_{i+1} who receives this request packet to find all the slots available for the communication between the two nodes. If node n_{i+1} finds that there are enough time slots to implement the selected reception code, both the code and the time slots are recorded into the *Pre-reserved table*, otherwise the node refuses to forward *RREQ_B*.

The algorithm realises the code allocation, the bandwidth calculation and the time slots scheduling together with the route request packet *RREQ_B,* hop by hop at the route discovery stage, which makes it possible to find a route with sufficient bandwidth. Once a *RREQ_B* reaches the destination node of the session, the reception codes along the valid route are pre-reserved, and the transmission codes are also determined and pre-reserved with the help of the *RREP_B* packet. The details of the whole routing protocol are described in the following section.

C. QoS CDMA Bus Lane Establishment

The route discovery within the CDMA Bus Lane routing protocol is purely on demand and follows a route request/reply discovery cycle. Each node in an ad hoc network is independent, which can be simulated as an event driven system. Each action of a node is triggered by an internal or external event. The event here could be a packet event (a packet arrives from the upper or lower layer) or a time event (a timer expires).

1) ***Event*:** A data packet is received from the application layer to a given destination with a specific bandwidth requirement.

***Actions*:**

- The node checks its routing table. If there is no active route for that session, a new route is required.
- If a route for such a session is already in the discovery process, the node puts the packet into the corresponding buffer and waits for the reply.
- Otherwise, the node must create an *RREQ_B*, which contains the bandwidth requirement, *F_code[codeID]* and time slot scheduling. After that, the source node broadcasts the packet to its neighbours and then sets up a timer to wait for a reply.

2) ***Event:*** A node receives an *RREQ_B* packet from its neighbour.

Actions:

- The node firstly checks whether the request for this session has been handled before. If so (it has been forwarded or replied), the node silently discards the packet. Otherwise, the node resets its *Pre-reserved table*, allocates the code and calculates the bandwidth.
- If the node cannot find a suitable code or enough time slots, the request packet is discarded. Otherwise, the node writes them into the *Pre-reserved table*. After a successful code allocation, the node records the *RREQ_B* in the *Seen requests table* to avoid another request for the same session to be processed during the same route discovery period.
- If the node (n_i) is not the destination of the session, it needs to update the *RREQ_B* packet and forward it to the next hop. The node firstly reads the RC_{i-1}^{p} (the pre-reserved reception code by the last hop node, which also presents TC_{i-2}^{p}) from the received *RREQ_B*, updates the slot attributes and the *F_code[codeID]* array. It also puts its pre-reserved reception code RC_{i}^{p} into the *RREQ_B* packet for the code allocation on the link *l (i+1, i+2).*
- If the node itself is the destination for the requested session, it puts all the pre-reserved information into the *Bus Lane routing table* and reserves the code and slots by updating the *Reception code table* and the *Reception slots table*. Then it creates a reply packet *RREP_B*. The *RREP_B* contains the reception code reserved for the session and the corresponding reserved time slots.

3) ***Event:*** A node receives a *RREP_B* reply packet.

Actions:

- If an intermediate node receives the *RREP_B*, it sets up the next hop path in the corresponding entry in the *Bus Lane routing table*. The reserved code in the packet is recorded as the transmission code of the node for the session in both the *Bus Lane routing table* and the *Transmission code table*. The slots are also reserved in the *Transmission slots table* which is used to schedule the transmissions in the MAC layer.
- The intermediate node then updates the *RREP_B* packet by changing the attributes of the reserved reception code and slots previously recorded in its *Pre-reserved table*. After increasing the hop count by 1, the node sends the packet to its previous link node which is recorded in its *Pre-reserved table.*
- If the node is the source of the session, it sets up the route entry and reserves the code and slots for the session. Therefore, a CDMA Bus Lane with QoS is set up with constant bandwidth. The node then starts to transmit the data.
- If a node finds the pre-reserved transmission (or reception) slots have been taken by another concurrent session, it gives up the slot reservation, continues the processing of *RREP_B*, and initiates a slot request packet (*SREQ)*, which includes the local slot schedule, to its next (or previous) hop.

4) ***Event:*** The *RREP_WAITING_TIME* has expired before a valid reply packet arrives

Actions:

- If the number of retries is smaller than *RREQ_RETRIES*, which records the maximum number of times that a *RREQ _B* is allowed to be rebroadcast, the node renews the request. Otherwise, it drops the data in the buffer and terminates the discovery process.
- If the node needs to continue a request, it increases the *TTL*, the broadcast ID and the sequence number of itself, as well as resetting the sequence number of the destination. After

creating a new *RREQ _B* with these attributes, the node resets the timer to *RREP_WAITING_TIME* and broadcasts the request packet to its neighbours again.

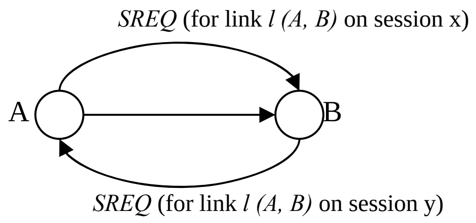
5) ***Event:*** A *SREQ* packet arrives from the next (or previous) hop.
Actions:
- Calculate and find enough slots satisfying the bandwidth requirement according to the current slot schedule of the two nodes.
- If the bandwidth requirement cannot be satisfied, report the breakage by sending a route error (*RERR*) message to both the source and the destination.
- If slots are found, reserve them for the link and send a *slot reply packet* (*SREP*) to the *SREQ* sender.

6) ***Event:*** A *SREP* packet arrives from next (or previous) hop.
Actions:
- Reserve the slots appended on the *SREP* packet.

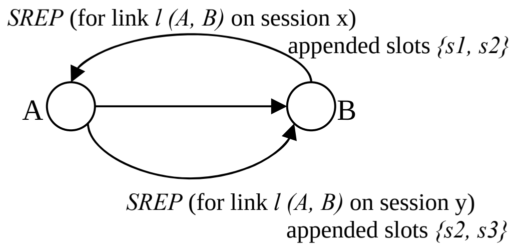
In this way, a CDMA Bus Lane is set up for a real-time session, the *Transmission (Reception) code tables* and *Transmission (Reception) slots tables* are written by the routing protocol. The channel reservation and the scheduling are actually performed by the MAC layer. It partitions the bandwidth into time slots for transmission or reception and assigns a code for each slot according to the tables maintained by the network layer.

Oscillation could happen when two nodes send *SREQ/SREP* packets to each other for slot reallocation on different sessions. When receiving a *SREP*, a node can find out that some slots it has reallocated are the same slots appended on the *SREP* sent by the other node.

Fig.3. gives an example how this oscillation happens. Assume that both sessions *x* and y go through link *l (A, B)*. Nodes A and B receive *RREP_Bs* for different sessions at the same time. Node A finds out that some pre-reserved transmission slots for session *x* have been taken by another concurrent session and node B finds out that some pre-reserved reception slots for session *y* have been taken by another concurrent session as well. These events make nodes A and B to send *SREQs* to each other for slot reallocation on link *l (A, B)* (Fig.3 (a)).Node A and B recalculate the bandwidth when receiving *SREQs* and append the new reserved slots on *SREPs* and send them back to each other. If one or more newly reserved slots are the same for both of the sessions, nodes A and B need to reallocate the slots again. The oscillation can happen again and again (Fig.3 (b)).


SREQ (for link *l (A, B)* on session x)
A
B
SREQ (for link *l (A, B)* on session y)

(a) node A and B send *SREQ* to each other for slot re-allocation for the link *l(A, B)* on different sessions at the same time.


SREP (for link *l (A, B)* on session x)
appended slots *{s1, s2}*
A
B
SREP (for link *l (A, B)* on session y)
appended slots *{s2, s3}*

(b) node A and B send *SREP* back to each other for slot reservation at the same time. The collision happens in slot *s2* again.

Fig.3. Oscillation in slot reallocation stage.

A scheme called *Receive-end first* is used to solve the oscillation problem. The receiving end of the link can reserve the slots appended on the *SREP* packet, but the transmitting end of the link must initiate another *SREQ* for slot reallocation when oscillation happens. In Fig.3 (b), node B reserves the slots appended on the *SREP* packet for session *x*, and node A must send a *SREQ* to node B for session *y* again.

D. Route Maintenance

Once a route has been discovered for a given source/destination pair, it is maintained as long as it is needed. The movement of the active nodes can cause either route breakage or code collision. Both of these events are identified by the periodical *Hello Messages* broadcast.

1) *Route Breakage*

When a node cannot receive a *Hello Message* from a neighbour after a certain time, it deletes the node from its *Neighbouring information table*. The node then checks its *Bus Lane routing table*. If the lost node is a next (or previous) hop node on an active session, it releases its reserved resources for that session and sends a *RERR* message to both the source and the destination in order to report the breakage.

Any node that receives a *RERR* will clean its reservations for the corresponding session and send the *RERR* out again until the message reaches the source or the destination. Therefore, all the nodes including the source node will be ready for a new route discovery.

2) *Code Collision*

The code assignment with the route request is performed according to the code and slot scheduling at that time. However, movement of the nodes in mobile ad hoc networks can bring two nodes, which were far away from each other, into the code interference range and cause code collision.

The identification of a code collision is performed when a *Hello Message* is received. When a node finds that one of its transmission (or reception) codes collides with a reception (or transmission) code of its neighbour, it could start a procedure to change this code. However, it is possible that the neighbour also could detect this collision and start a procedure to change the code. In order to avoid the oscillation caused by a simultaneous code change in both of the nodes, only reception codes are allowed to be changed. Therefore, once it happens, the node sends a *Link Request Packet* (*LREQ*) back to its previous hop node, appending the code and slot information.

The node that receives the *LREQ* will reallocate the code and calculate the bandwidth for the link. If a new code and sufficient bandwidth are found, a *Link Reply Packet* (*LREP*) is sent back to reserve the resources. Otherwise, if the link does not have sufficient bandwidth, a *Route Error* (*RERR)* is sent out to report the breakage and to release the reservation on the route.

V. Simulation And Results

The performance of the CDMA Bus Lane routing protocol in a static ad hoc network is simulated using OPNET Modeler [15]. The MAC layer is implemented to allocate codes and separate the channel in time slots (called *BusLane* in the simulation).The comparison is done between the *BusLane* and a pure CDMA scheme that just sets up code channels without time scheduling (represented by *PureCD*). The *BusLane* is also compared to a *RouteOL* protocol which only implements the routing protocol to find a route, but without code and slot allocation. That means all the transmissions in the *RouteOL* just share a common channel like classic AODV.

A static ad hoc network of 30 nodes is generated in an area of 1000m×1000m. The transmission range is 250*m*. There are up to 30 real time sessions (10 to 30 sessions) issued from the 30 nodes. The destination of the session from each node is selected randomly. All the sessions start during the first 20s of the simulation time and last until the end of the simulation. The simulation time is 120s. User traffic is generated with CBR sources, which generates 20 packets per second. It is a VoIP session source according the Nokia AMR-WB speech coding standard [16], which offers a data rate of 20kbps. Five different traffic patterns are generated and their simulation results are averaged. The number of packets received by the destinations is measured. A session is called “served” if at least 90% packets are received by the destination.

The allocation of the spreading codes was a *4-codes-remain* code set. The *SF* varies in the range of 8 to 128. The transmission rate of the channel is 5.4 Mbps. There are 32 slots in a frame, and each slot is 1.5625 ms, which carries 8.4kbit information. Therefore a packet can occupy one or more time slots. The waiting time for a reply after a *RREQ_B* broadcast (*RREP_WAITING_TIME*) is: 2×*NODE_TRAVERSAL_TIME*×*TTL*. The initial value of *TTL* is two, and it is increased by two, each time there is no reply after the *RREP_WAITING_TIME*. After resetting the *TTL*, the source node rediscovers the route again. *Hello Messages* are broadcast by each node every 1s. The neighbour node information expire time is 2s.

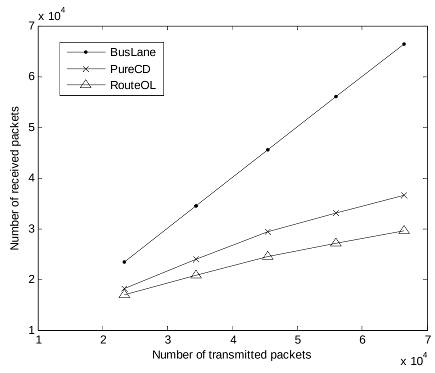


Fig.4. Packet throughput under different traffic loads

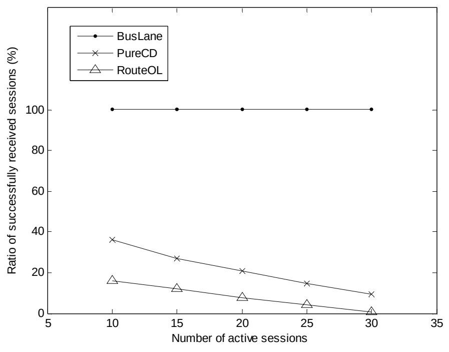


Fig.5. Ratio of successfully received sessions

Fig.4 shows the throughput under different traffic loads, which is defined by the number of concurrent sessions. The five points on each line are collected when the concurrent sessions are 10, 15, 20, 25 and 30 during the simulation time. All the route discovery and channel allocation are performed successfully before the data transmission. The results show that, once the routes are built successfully, *BusLane* can offer a 100% throughput in a static ad hoc network without any interference. There are many collisions under *PureCD* and *RouteOL*, which do not implement the time schedule scheme in their MAC layers.

Fig.5 shows the ratio of successfully received sessions, under different concurrent numbers of real time sessions (10, 15, 20, 25 and 30) in the network. All the sessions can be received under the *BusLane* protocol in a static ad hoc network. However, the other two protocols do not perform as well as the *BusLane*, even under lighter traffic. When increasing the concurrent sessions, the performance of the other protocols degrades.

The results show that the CDMA Bus Lane routing protocol can give each session a protected route from source to destination without any interference. The support from the MAC layer is very important in the route discovery, channel reservation and data transmission stages. The network using the *BusLane* protocol can perform 100% packet throughput, because the simulation was performed in a static network scenario. Once the QoS paths were set up, the packets were transmitted using the code and the scheduled time slots without any interruption. However, changes in the network topology will cause breakages of the links and decrease the packet throughput. Therefore, a good mechanism is required to maintain the QoS routes.

Another simulation is performed to show the performance of the route maintenance in a mobile scenario. The network model and the stream feature are the same as the previous simulations. Only the *Buslane* protocol is implemented with 30 concurrent sessions in the network. The mobility speed changes from 0 to 20m/s. Table 1 shows the throughput and successful session rate with different speeds.

TABLE I
PERFORMANCE OF ROUTE MAINTENANCE

Speed (m/s)	Throughput (%)	Successful sessions (%)	Isolated loss (%)
0	100.00	100.00	0
2	94.16	90.00	0
4	90.40	76.67	0.46
7	86.77	53.33	2.81
10	85.96	46.67	1.92
15	80.45	33.33	1.99
20	79.35	26.67	0.60

A node in the network can move outside of the transmission range of other nodes, becoming an *isolated node*. Sometimes, nodes in the network can be separated into two or more subnets due to node mobility and consequent loss of connection. We refer to these subnets as *isolated subnets*. If a source node or a destination node is an *isolated node*, or they belong to different *isolated subnets*, we have an *isolated problem*. It is not possible to set up a route during the occurrence of an *isolated problem*. The packet loss caused by this problem is called *isolated loss*, which is also presented in Table 1.

Obviously, throughputs and successful sessions decrease with the increase of mobility speed. The results in Table 1 show that the Bus Lane routing protocol works well in walking speed. The *isolated loss* can be represented by the $Session\_Data\_Rate \times Isolated\_Time$. When the speed increases, the chance that an *isolated problem* will happen also increases, but the *isolated time* decreases.

VI. CONCLUSIONS

A QoS routing protocol for the CDMA Bus Lane has been proposed in this paper. It describes how messages with the necessary information are exchanged among nodes, how to implement the code allocation algorithm on the route discovery process, and how to maintain the routes in a mobile scenario. The simulations have been implemented in both static and mobile scenarios. The results show that the CDMA Bus Lane routing protocol can find and reserve a privileged route from source to destination without any interference from other real-time sessions. The support from the MAC layer is very important in the route discovery, channel reservation and data transmission stages. The route breakage and code allocations can be identified by periodical *Hello Messages*, and solved by the implementation of the route maintenance scheme. It works well in walking speed.

Further research focuses on improving the route maintenance in a dynamic network scenario. In order to realise the CDMA Bus Lane for real time services in ad hoc networks, the ability to handle the route breakage and the channel collision, due to mobility is very important. The interference from the PHY layer should also be considered. The purpose is to set up a CDMA Bus Lane, which can provide better support for QoS routing in mobile ad hoc networks.

ACKNOWLEDGMENT

The authors wish to thank Rachel Stoneley for the suggestions to improve the presentation of the paper.

REFERENCES

[1] C. E. Perkins, and E. M. Royer: "Ad Hoc On-Demand Distance Vector Routing", Proceedings of IEEE Workshop on Mobile Computing Systems and Applications 1999, February 1999, pp. 90 –100

[2] D. B. Johnson, and D. A. Maltz: "Dynamic Source Routing in Ad Hoc Wireless Networks", Mobile Computing, Kluwer Academic Publisher, vol. 353, 1996, pp. 153 –181

[3] V.D. Park, and M.S. Corson: "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", Proceedings of IEEE INFOCOM 1997, April 1997, pp.1405 –1413

[4] Raghupathy Sivakumar, Prasum Sinha, and Vaduvur Bharghavan: "CEDAR: A Core-Extraction Distributed Ad Hoc Routing Algorithm", IEEE Journal on Selected Areas in Communications, vol. 17, no. 8, pp. 1454 – 1465, August 1999

[5] Shigang Chen, and Klara Nahrstedt: "Distributed quality-of-service routing in ad hoc networks", Communications, IEEE Journal on Volume 17, Issue 8, Page(s):1488 – 1505, Aug. 1999

[6] Shugong Xu, and Tarek Saadawi: "Does the IEEE 802.11 MAC Protocol work well in Multihop Wireless Ad Hoc Networks?" IEEE Commun. Mag., vol. 39, pp. 130–137, Jun. 2001

[7] C. Zhu, and M. Scott Corson: "A Five-Phase Reservation Protocol (FRRP) for Mobile Ad Hoc Networks", Proc. IEEE INFOCOM, vol.1, pp.322–331, March 1998

[8] Z. Tang, and J. J. Garcia-Luna-Aceves: "Hop Reservation Multiple Access (HRMA) for Ad Hoc Networks", IEEE INFOCOM'99, pp. 194–201, 1999

[9] Z. Tang, and J. J. Garcia-Luna-Aceves: "A Protocol for Topology Dependent Transmission Scheduling in Wireless Networks", in Proc. IEEE WCNC, New Orleans, LA, vol. 3, pp. 1333—1337, Sep. 1999

[10] C.R. Lin and M. Gerla: "Asynchronous Multimedia Multihop Wireless Networks", Proceedings IEEE INFOCOM 1997

[11] Chenxi Zhu and M. Scott Corson: "QoS routing for mobile ad hoc networks", Proc. IEEE INFOCOM, New York, NY, June 2002

[12] Amit Butala and Lang Tong: "Dynamic Channel Allocation and Optimal Detection for MAC in CDMA Ad hoc Networks", 36th Asilomar Conf. Signals, Syst., Comput., Pacific Grove, CA, Nov. 2002

[13] R. Fantacci, A. Ferri and D. Tarchi: "Medium Access control protocol for CDMA ad hoc networks", ELECTRONIC LETTERS vol. 40 No.18, 2nd September 2004

[14] Lin Xiao, Eliane Bodanese, "CDMA Bus Lane: a novel QoS solution for real-time traffic in ad hoc networks", ITC 19, pp. 97-106, 2005

[15] OPNET Modeler software. Available: http://www.opnet.com/ product / modeler/home/html

[16] Dimitrons Miras, "Network QoS Needs of Advanced Internet Application", working document of INTERNET2 QoS WORKING GROUP 2002