Channel Access Using Opportunistic Reservations in Ad Hoc Networks

Xiaoqiao Meng*, J.J. Garcia-Luna-Aceves†‡
*UCLA Computer Science, Los Angeles, CA 90095
†Palo Alto Research Center (PARC), Palo Alto, CA 94304
‡Computer Engineering Department, University of California, Santa Cruz, CA 95064
E-mail:*xqmeng@cs.ucla.edu, †jjgla@parc.com

***Abstract*— We introduce a medium access control protocol for ad hoc networks. The new protocol, which we call ORMA (opportunistic reservation multiple access) is aimed at providing both high throughput and bounded channel access delays, which are critical for supporting integrated voice and data services over ad hoc networks. In ORMA, the channel is divided into a random access section and a scheduled access section. The first is used to exchange neighborhood information, the latter is used for data transmissions over time slots organized in frames, with each time slot being accessed through reservations or probabilistic elections. To attain high throughput, nodes access data slots based on a fair election in which they win with a certain probability. To attain bounded channel access delays, nodes reserve time slots by using a novel opportunistic reservation. The performance of ORMA is studied by both analysis and simulations. It is also compared against the performance of schemes based entirely on probabilistic or fixed conflict-free slot assignment.**

I. Introduction

Recent advances in ad hoc networks have stimulated lots of interests in supporting many voice-related applications such as voice over wireless IP and mobile games. It is envisioned that these applications may be running on stations concurrently with legacy data-centric applications. To better support such integrated voice/data traffic in an ad hoc network, the underlying channel access protocol needs to satisfy two requirements: (a) high channel utilization, and (b) bounded channel access delay. While the former is important for serving those data-centric applications, the latter is critical for providing uninterrupted data delivery to those voice-related applications.

Although a large number of channel access schemes have been proposed in the past for ad hoc networks, they do not fully meet both the two requirements. For example, contention-based schemes such as as CSMA [11] and CSMA/CA [2], are arguably most widely used today. They cannot sustain high utilization in heavily loaded situations. They cannot provide bounded channel access delay, thus cannot avoid occasional starvations. On the other hand, many contention-free schemes (e.g., [16], [10], [1]) allow nodes to periodically access channel without collision. However, these schemes rely on global or local topological information which involves high control overhead. Recently, a probabilistic channel access approach is proposed to partially address the aforementioned limitation. The schemes (e.g., [17], [8], [9]) in this approach schedule channel access probabilistically in the sense that each node (link) has certain probability to access the channel during every time slot. Although these schemes incur less control overhead, their probabilistic nature can lead to infinite channel access delay in the worst case.

In this paper, we introduce the Opportunistic Reservation Multiple Access (ORMA) protocol to attain both high channel utilization and bounded channel access delay. ORMA is a node activation protocol based on time-division multiple access. ORMA requires each node to know the set of its contenders which are essentially the nodes within its two-hop neighborhood. By using the two-hop neighborhood information, nodes participate in a fair election in each time slot and have a certain probability to win. Nodes winning the election can access the channel without collision. Further, a node winning the election can opportunistically reserve the probabilistically allocated slots. With such an opportunistic reservation in place, nodes can gradually firm up the probabilistic transmission schedules to form deterministic ones.

The novelty of ORMA lies in its *opportunistic reservation* scheme, which allows ORMA to support delay-bounded channel access without sacrificing high throughput. The nodal throughput achieved by ORMA is $\frac{1}{D}$ where D is the local two-hop node degree. This

1-4244-0507-6/06/$20.00 ©2006 IEEE

is comparable to traditional probabilistic schemes (e.g., [17], [8], [9]) which were considered to provide high channel utilization. The channel access delay achieved by ORMA is D_{max} where D_{max} is the network-wide maximum two-hop node degree. This is comparable to those deterministic schemes ([1], [3], [4], [7]) which are considered to provide fairly good delay bound. We also compute the convergence time of ORMA, which measures how long it takes ORMA to firm up probabilistic schedules into deterministic ones. The convergence time is shown to be less than a few times of D_{max} (time slots). In overall, ORMA strikes a balance between high channel utilization and low channel access delay. It is suitable for supporting voice/data integrated traffic.

The rest of this paper is organized as follows: Section II we discuss related work. We present the design of ORMA in Section III. In Section IV we analyze the properties of ORMA. In Section V, we evaluate the performance of ORMA and compare it with existing schemes. We discuss several design issues in Section VI and conclude the paper in Section VII.

II. Related work

A large body of channel access schemes have been proposed in the literature. In general they fall into two classes: contention-based and contention-free. In the following we briefly describe these two schemes, and discuss their pros and cons.

Contention-based schemes include random access schemes (ALOHA [12] and Carrier Sense Multiple Access (CSMA) [11]) and reservation/collision-resolution schemes (MACA [13], MACAW [18], FAMA [6], and IEEE 802.11 DCF [2]). Contention-based schemes are suitable for light traffic load since their channel utilization ratio is low. Moreover, they cannot provide bounded nodal channel access delay and are prone to unfairness problems.

In contention-free schemes, channel is divided into time slots. Based on how time slots are assigned to nodes, these schemes can be further classified into two categories: deterministic and probabilistic.

In the deterministic category, time is organized as frames and each frame contains equal number of time slots. In each frame, a few time slots are assigned to nodes. Such an assignment is fixed for every frame. Based on how the topology information is required, the schemes in this approach are classified into three categories: centralized schemes, distributed schemes, and topology-transparent schemes. The Spatial TDMA [15] and UxDMA framework [16] are centralized schemes in which each node acquires global topology information. They formulate the channel access issue into a graph coloring problem, then solve the problem to obtain collision-free channel access schedule. A few other MAC schemes [3][10][1] are fully distributed. By using these schemes, nodes use a dedicated control channel (or segment) to dynamically reserve deterministic time slots. Although [3][10][1] provide bounded worst-case channel access delay, the bound is in proportion to the network size, which can be very large. The third class of deterministic schemes are topology-transparent. The representative work include the topology-independent schemes [4][5][7]. These schemes divide time into frames and ensure that a node (or link) can have collision-free transmission in at least one time slot for every frame. These schemes, however, are shown [14] to provide throughput at best equal to that of slotted ALOHA. Another type of randomized contention-free schemes [8][9][19][17] require local topology information. In each time slot, nodes have a certain probability to access the channel without collision. The throughput for these schemes are high but they have no channel access delay.

III. Opportunistic Reservation Multiple Access

This section presents the design of ORMA. We first give an overview of ORMA by showing its diagram in Figure 1. ORMA schedules transmissions in a time-division multiple access manner. Prior to each time slot, a node needs to decide whether the slot is reserved. If the slot is reserved by itself or by any other node, it remains in reception mode. Otherwise, the node uses collected two-hop neighborhood information to participate in a priority comparison (a.k.a. election) with its channel contenders, which are essentially all its two-hop neighbors. If the node has the highest priority, it wins the election, so it can transmit data in that time slot. Moreover, the node can take advantage of this opportunity to request to reserve the slot. Once the reservation succeeds, the node can always access channel in the same time slot within every time frame.

The ORMA operations described above, seemingly simple, yet raise several issues to address. For example, how two-hop neighborhood information is collected? How are opportunistic reservations realized? Given that each individual node can request reservations, will these reservations be in conflict? If a conflict do occur, how should it be handled? The rest of this section is to answer these questions. We first describe how the channel is organized by ORMA in Section III-A. We then de-

scribe opportunistic reservations, the kernel component of ORMA, in Section III-B and III-C. In Section III-D, we describe how to handle reservation conflicts.

A. Channel organization

We assume time is synchronized and the channel is divided into random access and scheduled access periods, as shown in Figure 2. In the random access period, time is further slotted into mini-slots. The network is initially in the random access period. Each node randomly picks up a mini-slot to broadcast its identity as well as any one-hop neighbor's identity it has acquired. This way, nodes incrementally build up their two-hop neighborhood information. Since packets could be lost due to collision, nodes may need to retransmit for multiple times.

The random access period is followed by a scheduled access period, which is organized frame by frame. Each frame is further divided into time slots. A time slot begins with a Request-to-Reserve (RTR) mini-slot and K consecutive Report-Reservation-Conflict (RRC) mini-slots. K is a network-wide parameter no less than the maximum number of any node's one-hop neighbors. The K RRC mini-slots are followed by a data mini-slot used for transmission payload. The $K+1$ mini-slots used for RTR and RRC are used for transmitting short control messages. Their length should be much shorter than the data mini-slot. We will explain the role of RTR and RRC in the following ORMA design.

B. Opportunistic reservation

In time slot t, a node i applies the Neighborhood-aware Contention Resolution (NCR) algorithm [8] to generate a probabilistic schedule. More specifically, node i assigns itself a priority given by

$$\text{Pri}_i(t) = \text{hash}(i \oplus t)$$

where $\oplus$ represents a concatenation operation, and $hash(.)$ is a hash function that maps an input string to a pseudo-random real values between 0 and 1. Suppose N_{12} refers to all the nodes within 2-hop distance to node i (not including node i itself). Node i should already know N_{12} in previous random access periods. Thus node i can infer the priority for every node in N_{12}. If node i's priority is higher than any node in N_{12}, node i is said to win the election [1]. Clearly, the probability for node i to win the election is $\frac{1}{|N_{12}|+1}$, where $|N_{12}|$ is the cardinality of N_{12}.

[1]If n ($n > 1$) nodes have the highest priority and node i is one of them, node i wins the election with a probability $\frac{1}{n}$.

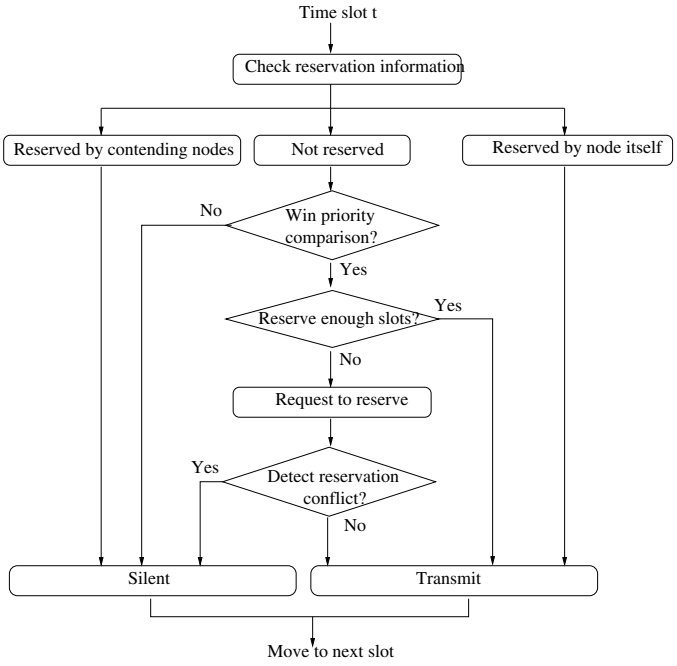

Fig. 1. Diagram of ORMA

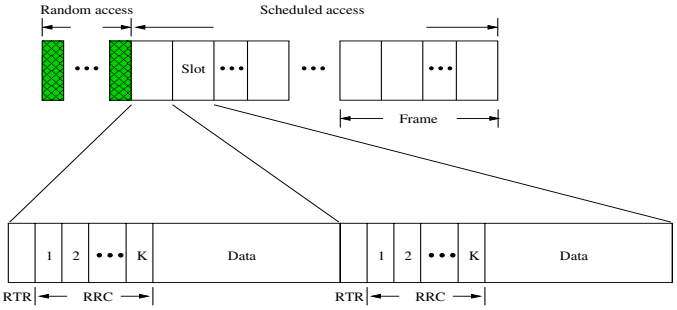

Fig. 2. Time division organization (RTR: Request-to-Reserve. RRC: Report-Reservation-Conflict)

Once node i wins the election in time slot t, node i is eligible for transmitting data. Moreover, node i requests to reserve slot t so that node i can transmit in the same slot for every following frame. To this end, first node i broadcast its identity in the RTR mini-slot (we refer to the time organization in Figure 2). Node i then switches to reception mode in the next K RRC mini-slots. if node i does not receive a RRC in *any* of the K mini-slots, it considers the reservation successful. Otherwise, a reservation conflict must occur. Then node i should give up its reservation request, and remain silent in the rest of the time slot.

Next, we describe actions taken by a node when it receives a RTR. Suppose node j receives a RTR from node i, node j should remain silent and expect data

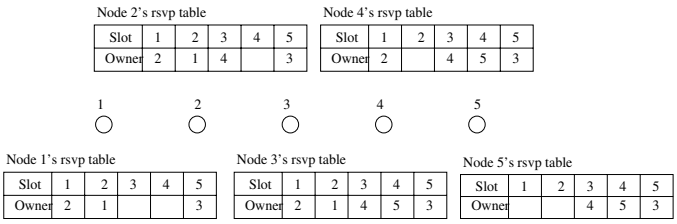

Fig. 3. An example scenario for illustrating reservation tables

transmission from node i. Only until node j correctly receives data packets from node i, node j ascertain the success of node i's request to reserve the current time slot. If such information is new to node j, node j should update its 1-hop neighbors whenever it has a chance to transmit in a future time slot. In this way, all the two-hop neighbors of node i get to know the reservation made by node i.

Each node maintains a reservation table to record how each slot in the frame is reserved. If a node finds that a time slot is reserved, the node should be silent during that slot. Otherwise, it can participate in the election of that time slot. Each node keeps reserving more slots until it has reserved $\frac{T}{|N_{12}|+1}$ out of the T slots in a frame. As $\frac{T}{|N_{12}|+1}$ might be a float-point value, it can be replaced by $\lfloor \frac{T}{|N_{12}|+1} \rfloor$ where $\lfloor . \rfloor$ is the floor function that round a float-point value to its closest integer lower bound.

In the example scenario of Figure 3, we show the reservation tables maintained by each node in a linear topology. As $T = 5$, each node only needs to reserve one slot. Some nodes may find certain slot(s) unreserved. For example, from node 1's perspective, slot 3 and 4 are not reserved. Node 1 can still transmit in these two slots if it wins the election, thus the actual time slots used by node 1 come from both reservations and probabilistic scheduling.

C. Controlling distribution of reserved slots

While ORMA ensures that every node eventually reserve a number of time slots, how these time slots are distributed within a time frame is not specified. Clearly, such a distribution will affect the channel access delay. In particular, the worst-case channel access delay is equivalent to the longest time interval between any two consecutively scheduled time slots. Regarding the distribution, ORMA has the following two strategies:

1) **Reserve-ASAP** A node makes an immediate reservation request upon winning an election. The node continues to do this until it has reserved enough slots.
2) **Reserve-at-regular-interval** A node divides the frame into $\lfloor \frac{T}{|N_{12}|+1} \rfloor$ sections with equal length (the last section could be longer when $\frac{T}{\lfloor \frac{T}{|N_{12}|+1} \rfloor}$ is not an integer). The node reserves one and only one slot in each section.

Clearly, both strategies lead to equal number of reserved slots in the long run. Nevertheless, the *convergence time*, which is the time before every node reserves enough slots, is different under the two strategies. By using the first strategy, nodes have a shorter convergence time, yet the resulting reserved slots could be unevenly distributed. In the worst case, all reserved slots are grouped together within the frame so that no channel access opportunity exists for the rest of the frame. On the other hand, the second strategy results in an even distribution of reserved slots, but the nodes may need a longer time to firm up the desired number of slots. Obviously there is a trade-off between convergence time and worst-case channel access delay. We will analyze and compare the two strategies in Section IV-A.

D. Handling reservation conflicts

A reservation conflict means that two nodes within 2-hop distance reserve the same time slot. Reservation conflicts happen when reservation information is not timely propagated. In this section, we first exemplify how such conflicts could occur, then present mechanisms to address this issue.

In the example scenario of Figure 3, suppose node 3 succeeds in reserving a slot. Node 3 then update its direct neighbors in the data mini-slot. Node 4 receives the update. Unfortunately, node 4 happens to have no chance to transmit until the same slot in the next frame, which makes node 5 unaware of the update. When the time moves to the the same slot in the next frame, if node 5 happens to win the election, a collision will occur at node 4 because both node 3 and 5 transmit[2]. Obviously, the collision happens because node 4 did not forward reservation updates on time.

ORMA handles reservation conflicts by using thw following two mechanisms.

The first mechanism is **Reiterative RTR**. When a node requests to reserve a slot, the node should send RTR. As long as the reservation succeeds, ORMA requires the node to always send RTR in the same slot for every following frame. By listening to such a RTR,

[2]We do not consider *capture effect*. When two packets arrive in the same node at the same time, we assume neither of them are correctly received.

nodes can periodically validate the freshness of their reservation tables.

The second mechanism is **Prioritized RRC**. Suppose node i and j are 1-hop neighbors. Once node i knows that another node j has reserved a certain slot, node i should expect a RTR in that slot for every frame given that reiterative RTR is employed. If node i does not receive a RTR from node j, a reservation conflict must occur. Such a conflict is only possible because of the following situation: node i has a one-hop neighbor k. Node k does not know the slot has been reserved by node j, so node k participates in the election and happens to be the winner. Node k sends a RTR. Both this RTR and the RTR sent by node j will arrive at node i and cause collision.

Clearly, the conflict described above is caused by the outdated reservation table in node k. Since node i is the one who detects the conflict, it is node i's responsibility to notify node k about the conflict and update the reservation table in node k. Accordingly, node i needs to take two actions: first, node i should know the identify of node k. Identifying node k is simple because node i can exploit the fact that node k has the highest priority among $N_1(i)$, all the 1-hop neighbors of node i. Second, node i should further send a RRC to tell node k that the current time slot is reserved by node j. Considering that there may exist other nodes in $N_1(k)$ who are in the same situation as node i, node i must ensure that its RRC does not collide with the RRC possibly sent by any other node in $N_1(k)$. To this end, node i compares its own priority Pri_i with the priorities of other nodes in $N_1(k)$. If Pri_i is ranked the n-th highest one, node i will send RRC in the n-th RRC mini-slot. Since the total number of RRC mini-slots is no less than the maximum one-hop node degree, it is no less than n as well. Therefore, the n-th RRC mini-slot always exists and it is guaranteed collision-free. Once node k receives the RRC from node i, node k updates its reservation table and remains silent in the rest of the slot. In the meantime, since nodes in $N_1(k)$ do not receive data packets from node k, they will realize that node k has canceled its reservation request, thus they will not maintain or further spread such information.

For clarity purposes, we present the pseudo-code of ORMA in Algorithm 1. The pseudo-code describes opportunistic reservation, Reserve-ASAP, and handling reservation conflicts.

IV. Behavior of ORMA

A. Throughput and worst-case channel access delay

To calculate the long-term throughput provided by ORMA, we first need to decide how many slots in each frame are eventually received by any node i. The received time slots contain two parts: the reserved $\lfloor \frac{T}{|N_{12}(i)|+1} \rfloor$ slots, and those slots reserved by nobody but assigned to node i because node i wins the election. Let us consider a homogeneous scenario in which each node in $N_{12}(i)$ reserves $\lfloor \frac{T}{|N_{12}(i)+1|} \rfloor$ slots as well. Suppose T_u is the number of slots reserved by nobody, T_u must satisfy

$$T_u \geq T - (|N_{12}(i)| + 1)\lfloor \frac{T}{|N_{12}(i)| + 1} \rfloor$$

The inequality above is due to the fact that some nodes in $N_{12}(i)$ are more than two-hop away from each other, so that some of their reserved slots may be the same. In each of the T_u unreserved slots, the probability that node i can win an election is $\frac{1}{|N_{12}(i)|+1}$. Therefore, the total number of slots assigned to node i is

$$\begin{aligned} T_{total} &= \lfloor \frac{T}{|N_{12}(i)| + 1} \rfloor + T_u \frac{1}{|N_{12}(i)| + 1} \\ &\geq \frac{T}{|N_{12}(i)| + 1} \end{aligned}$$

This shows that the long-term average throughput achieved by node i is no less than $\frac{1}{|N_{12}(i)|+1}$.

Next, we calculate the worst-case channel access delay in ORMA. Clearly, the delay differs between the two strategies: reserve-ASAP and reserve-at-regular-interval. When former one is used, the worst case occurs when the $\lfloor \frac{T}{|N_{12}(i)|+1} \rfloor$ slots reserved by node i are consecutively together and node i does not win elections in any other slot. Accordingly, the worst-case channel access delay is $T - \lfloor \frac{T}{|N_{12}(i)|+1} \rfloor$. Given T is usually chosen to be $D_{max}+1$ (D_{max} is the maximum two-hop node degree), such a worst-case channel access delay is approximately equal to D_{max}.

When the second strategy, reserve-at-regular-interval, is used, each time frame is divided into $\lfloor \frac{T}{|N_{12}(i)|+1} \rfloor$ portions, and each portion has exactly one reserved slot. If $\frac{T}{\lfloor \frac{T}{|N_{12}(i)|+1} \rfloor}$ is not an integer, the last portion could be longer than the other portions. In an extreme case, the last portion has $2\lfloor \frac{T}{|N_{12}(i)|+1} \rfloor - 1$ slots and its first slot is reserved by node i; Moreover, the last slot of the first portion is reserved by node i and node i does not win elections in both portions. In such a case, the worst-case channel access delay is $3\lfloor \frac{T}{|N_{12}(i)|+1} \rfloor - 3$.

Algorithm 1 Pseudo-code of ORMA

Procedure: ORMA(i, t)
if (t reserved by j, $j \in N_{12}(i), j \neq i$) **then**
  if (i recv RTR from j) **then**
    i waits for data from j. Update reservation table if the data contains new reservation information
  else {/*No RTR recvd. Reservation conflict detected*/}
    $k \leftarrow \arg \max_j Pri_j(\forall j \in N_1(i))$/*$k$ causes conflict */
    $r \leftarrow$ Rank Pri_i among $Pri_{N_1(k)}$
    i transmit RRC in the r-th RRC mini-slot/*Report reservation conflict */
  end if
else if (t reserved by i) **then**
  i send RTR
  i **SendDataAndUpdate(**i,t**)**
else {/*t not reserved*/}
  if ($Pri_i > Pri_j$, $\forall j \in N_{12}(i), j \neq i$) **then** {/*$i$ wins priority comparison*/}
    if (i.ReservedSlot $\geq \lfloor \frac{T}{|N_{12}(i)|} \rfloor$) **then** {/*$i$ has enough reservations*/}
      SendDataAndUpdate(i,t)
    else
      RequestToReserve(i,t**)**
    end if
  else
    if (i recv RTR from j, $j \in N_1(i)$) **then**
      j reserves t. i updates its own reservation table
    end if
  end if
end if

Procedure: RequestToReserve(i,t**)**
i send RTR
if (i recv RRC from j, $j \in N_1(i)$) **then** {/*i's request causes conflict*/}
  Give up request and keep silent in rest of t
else {/*No RRC recvd. Reservation succeeds*/}
  i reserves t, updates its reservation table
  SendDataAndUpdate(i,t)
end if

Procedure: SendDataAndUpdate(i,t**)**{/*Transmit data and updated reservation table in the data mini-slot */}
if (i's reservation table updated since last transmission) **then**
  i broadcast updated reservation table
end if
i transmit data

Scheme	Nodal throughput	Worst-case nodal channel access delay (unit: slot)	Required topology information
ORMA	$\frac{1}{D}$	D_{max}	2-hop topology
NAMA [8]	$\frac{1}{D}$	Infinity	2-hop topology
Skeleton TDMA [1]	$\frac{1}{N}$ + local channel reuse	N	Local topology
Distributed channel assignment [3]	Local topology dependent	At least N (when long-term assignment used)	Local topology
Topology-immune [4]	$O(\frac{\log^2 N D_{max}}{D_{max}^2 \log^2 N})$	$O(\frac{D_{max}^2 \log^2 N}{\log^2 D_{max}})$	Not required
Optimal topology-transparent [7]	$\frac{1}{4kD_{max}}$ (k is usually a small integer)	$4kD_{max}$	Not required

TABLE I

COMPARISON AMONG EXISTING TDMA-BASED CHANNEL ACCESS SCHEMES (D: LOCAL TWO-HOP NODE DEGREE, D_{max}: NETWORK-WIDE MAXIMUM TWO-HOP NODE DEGREE, N: NETWORK SIZE)

Having determined the throughput and delay for ORMA, we now compare ORMA with those most popular TDMA-based channel access schemes, including NAMA [8], skeleton TDMA [1], distributed chan-

nel assignment [3], topology-immune [4], and optimal topology-transparent [7]. The comparison result is given by Table I. As we can see, in typical scenarios where $N \gg D_{max}$ (N is network size), ORMA outperforms other schemes in terms of both throughput and worst-case channel access delay. The price we pay is that ORMA requires 2-hop local topology information, which is not required in the topology-immune [4] and optimal topology-transparent [7] schemes.

B. Convergence time

We now evaluate ORMA by estimating the convergence time, which measures how long it takes a node to reserve the desired number of slots. Ideally, the convergence time should be short so that ORMA can quickly provide bounded channel access delay in case the network topology is changed.

For simplicity, we still study a homogeneous scenario where each node has D_{max} channel contenders and frame length $T = D_{max}$. Accordingly, each node intends to reserve one slot in each frame. As long as a node wins an election in any slot of the initial frame, the node reserves that slot and immediately broadcast such information to its 1-hop neighbors, namely, all the nodes in N_1. Each node in N_1 then further propagates such information to N_2, which are the nodes 2-hop away. To make analysis tractable, we assume new reservations made in a frame are known to all the nodes in N_{12} only at the end of that frame. In other words, if a node has n unsatisfied[3] channel contenders in the first slot of a frame, the node will always compete with these n nodes in the rest of that frame. Only at the beginning of the next frame can reservation tables be updated.

We first calculate how the number of unsatisfied nodes decreases over time. For simplicity, we assume that at the beginning of a frame, a node i has m unsatisfied contenders, each of which further has m unsatisfied contenders. We first calculate $P_m(n)$, the probability for n out of the m unsatisfied contenders succeed in making a reservation within the frame. A direct calculation of $P_m(n)$ is difficult. Instead, we turn to estimate the upper and lower bounds of $P_m(n)$ by considering the following two extreme cases:

- **Case 1** Each unsatisfied contender can access channel with an independently probability $\frac{1}{m+1}$, i.e., these m contenders are not mutually exclusive. This case corresponds to the situation where the m contenders have sparse geographic distribution so that any pair of them are more than two-hop away from each other. Considering that the frame has T slots, we have

$$P_m(n) = \binom{m+1}{n}[1-(1-\frac{1}{m+1})^T]^n [(1-\frac{1}{m+1})^T]^{m+1-n} \quad (1)$$

- **Case 2** The m unsatisfied contenders are mutually exclusive. In every slot only the node with the highest priority can access the channel. Such a case corresponds to the situation that all of the m contenders are within one-hop distance. Accordingly we have

$$P_m(n) = \sum_{\substack{i_1, i_2, \ldots, i_n \geq 1, \\ i_1 + \ldots + i_n = T}} \frac{T!}{i_1! i_2! \cdots 1_n!} (\frac{1}{m+1})^T \quad (2)$$

In an actual case, some of the m contenders are mutually exclusive while the others are independent. Thus, the actual $P_m(n)$ should stand between (1) and (2). Obviously, Case 1 is an over-estimation, so (1) is the upper bound for $P_m(n)$. Case 2 is an under-estimation, so (2) is the lower bound.

Now we compute the probability for a node i to successfully reserve a slot. We start from frame 0 at the beginning of which all the $T-1$ two-hop neighbors are unsatisfied contenders. Let T_c denote the convergence time for node i. The probability for T_c ending in frame 0 is

$$P_{T_c}(t) = (1-\frac{1}{T})^t \frac{1}{T}, \ 0 \leq t \leq T-1$$

To compute $P_{T_c}(t)$, the probability for T_c ending in frame $k (k > 0)$, we need to decide m_k, the expected number of unsatisfied contenders in frame k. Given that frame k has m_{k-1} unsatisfied contenders, the number of unsatisfied contenders for frame $k+1$ is

$$m_k = m_{k-1} - \sum_{i=0}^{m_{k-1}} P_{m_{k-1}}(i) i, \ k > 0 \quad (3)$$

The lower and upper bounds for $P_{m_{k-1}}(i)$ are derived from (1) and (2). Since all the T nodes are unsatisfied at the beginning of frame 0 , we have $m_0 = T$.

Now we can iteratively compute $P_{T_c}(t)$ (the probability for T_c taking value t) as following

$$P_{T_c}(t) = [1-\sum_{i=0}^{t-1} P_{T_c}(i)](1-\frac{1}{m_k})^{t-kT}\frac{1}{m_k}, \quad (4)$$
$$kT \leq t \leq (k+1)T-1$$

[3]An *unsatisfied* node is a node that hasn't reserved its desired number of slots.

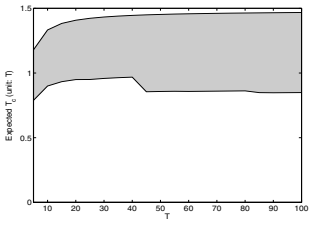

Fig. 4. Average T_c

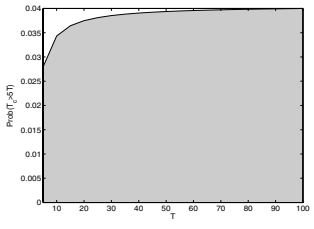

Fig. 5. Prob($T_c > 5T$)

where $1 - \sum_{i=0}^{t-1} P_{T_c}(i)$ is the probability for $T_c < t$.

To estimate T_c, we first need to decide $P_m(n)$. Since $P_m(n)$ is bounded by (1) and (2). Our strategy is to replace $P_m(n)$ by the two bounds and use the two bounds in the following computation separately. We then use $P_m(n)$ and (3) to estimate m_k, which is further used in (4) to estimate $P_{T_c}(t)$. Once $P_{T_c}(t)$ is known, we define the following two metrics to evaluate T_c

$$\bar{T_c} = \sum_{t=0}^{\infty} P_{T_c}(t)t$$

$$P(T_c > 5T) = \sum_{t=5T}^{\infty} P_{T_c}(t)$$

$\bar{T_c}$ is the average convergence time. $P(T_c > 5T)$ is the probability for very long convergence time (A long convergence time is defined as a T_c four times longer than frame length).

We numerically compute $\bar{T_c}$ and $P(T_c > 5T)$ at T. In Figure 4, the two curves are $\bar{T_c}$ obtained from using (1) and (2) separately. The shadow area between these two curves represents the range of the actual $\bar{T_c}$. Figure 4 shows that the average T_c is less than $1.5T$. Next we plot $P(T_c > 5T)$ in Figure 5, in which one of the curve derived from the second extreme case is very close to the X-axis and thus not visible. Again, the actual $P(T_c > 5T)$ should fall into the shadow area. Figure 5 shows that the probability for very long convergence time is less than 0.04.

V. Performance evaluation

In this section, we evaluate the performance of ORMA through simulations. We first describe our simulation methodology, metrics in Section V-A. Then we present the simulation results in Section V-B. The results confirm nthe high-throughput and low channel access delay of ORMA.

A. Methodology and metrics

We implemented ORMA in a discrete event driven simulator. For comparison purposes, we also implement NAMA [8] and the skeleton TDMA scheme [1]. There are two reasons for choosing these two protocols: first, ORMA, NAMA and skeleton TDMA scheme are all designed for collision-free broadcast scheduling. Secondly, NAMA is a representative probabilistic scheme while the skeleton TDMA is a representative reservation-based scheme. We do not compare with CSMA and IEEE 802.11 because they are contention-based and have very different parameter settings and design goals.

The simulation settings are summarized as following:

- *Topology* The simulations are conducted for random topology. A certain number of nodes are randomly placed in an area of $700*700$ square meters. To simplify performance evaluation, we expect that each node has identical two-hop neighborhood topology. To this end, we let the opposite sides of the square to be seamed together. This turns the square area into a torus. To reduce the impact of uneven node distributions, 10 random topology are generated while keeping all the other settings. The final simulation results are averaged over these 10 topology. The communication range for each node is fixed to be 100 meters. The total number of nodes are set to be 30, 100, respectively, and the network-wide parameter K is 3, d11 respectively. During the simulation, nodes are static, thus there are no topological changes.
- *Channel model* Signal strength attenuates by following the free-space model. Communication range is identical for every node. Interference caused by radio outside the communication range is negligible. Capture effect is negligible as well. The channel transmission rate is constant and set to be 15 packets per slot.
- *Traffic and queuing model* New packets arrive at every node by following a Poisson process. Packets are destined for all one-hop neighbors. Whenever a node is scheduled a time slot, the node sends out packets from its buffer at the channel transmission rate. The queue limit is 200 packets. When the queue is full, all the arriving packets are silently dropped.

We use three metrics to evaluate the performance of channel access protocols. **Nodal throughput** is defined as the total number of packets transmitted by a node divided by the duration of the simulation. **Packet delay**

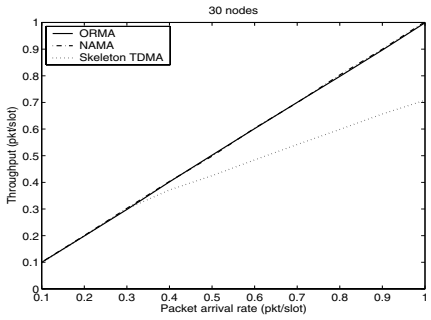

Fig. 6. Average nodal throughput

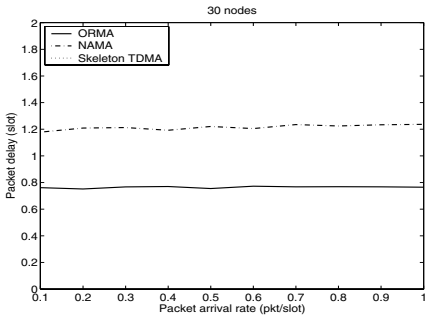

Fig. 7. Average packet delay

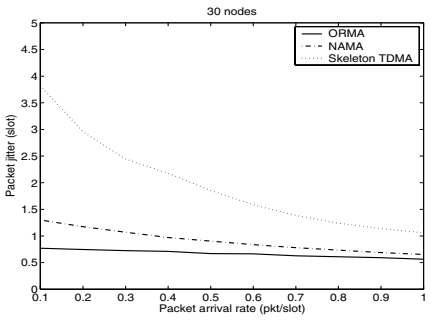

Fig. 8. Average packet jitter

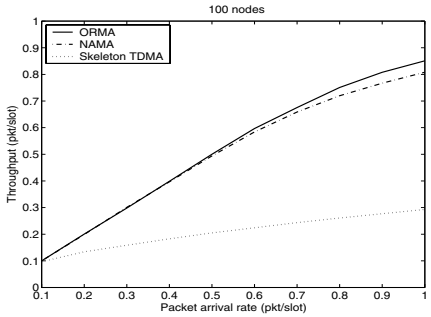

Fig. 9. Average nodal throughput

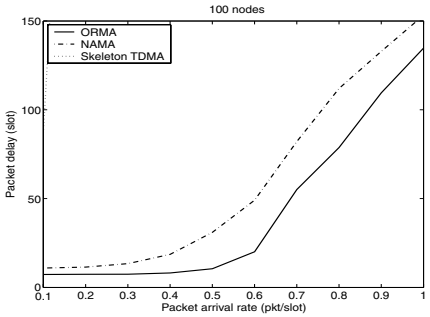

Fig. 10. Average packet delay

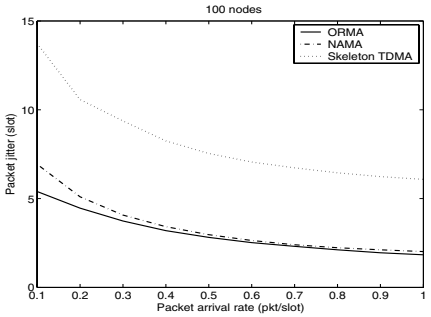

Fig. 11. Average packet jitter

is defined as the average delay incurred by all successfully transmitted packets. **Packet jitter** is the difference between the delay of two packets consecutively transmitted by the same node. The first metric is important for data-centric applications while the latter two are critical for voice services. All three metrics are averaged among either nodes or packets. In addition, the duration for each run is 10000 time slots. Note that we do not simulate mobility and wireless channel fading. We leave this as a part of future work.

B. Results

Figure 6-8 show the three metrics when the total node number is 30. The average nodal throughput (see Figure 6) achieved by ORMA and NAMA are almost identical, but much higher than that of the skeleton TDMA scheme. The fact that ORMA and NAMA give almost equal throughput is because ORMA is converting the random schedule in NAMA to a fixed form without changing the actual channel assignment. Nevertheless, the average packet delay by ORMA is much lower than that of NAMA. When the packet arrival rate is low, e.g., 0.1 pkt/second, the packet delay of ORMA is 35% lower than that of NAMA. Such a gap becomes even more significant when the packet arrival rate becomes larger. The average packet delay by the skeleton TDMA scheme is significantly worse than that of ORMA and NAMA. The main reason is that the skeleton TDMA scheme suffers from fairness issue. More specifically, in the skeleton TDMA scheme, nodes use their identities as static priorities to compete for local channel reuse, thus, some nodes take too much local channel reuse while some others can not. This will cause long packet delay. About the average packet jitter as shown in Figure 8, ORMA can reduce it by almost half compared to NAMA. However, when the packet arrival rate increases, individual packet delays are so large that the difference between the delays of consecutive transmitted packet are insignificant.

Figure 9-11 are the results for a higher node density. One can notice that, for the same packet arrival rate, when the node density increases, the nodal throughput becomes smaller and packet delay and jitter become much higher. This is because when the density is higher, the average two-hop neighborhood degree becomes larger, which reduces the throughput allocated to each node.

Overall, ORMA achieves similar throughput as NAMA, but significantly reduces packet delay and jitter. Compared with the well-known skeleton TDMA scheme, ORMA performs much better in terms of nodal throughput, packet delay and jitter.

VI. Discussions

In this section, we comment on several design issues and discuss future work.

Node activation ORMA is a node activation protocol which is aimed at ensuring collision-free broadcast traffic. Although ORMA supports unicast or multicast traffic, its idea of opportunistic reservations can be readily extended to better support unicast or multicast traffic.

Mobile nodes Mobile nodes can cause topological changes, which further bring two challenges to the operation of ORMA: outdated two-hop neighborhood information and reservation conflicts. To cope with these challenges, by using ORMA, each node updates neighborhood information in the random access section and then restarts the opportunistic reservation process.

k: number of RRC mini-slots So far we assume k is no less than the maximum two-hop node degree. However, when the network topology changes, this may not always hold. In this case, the transmission of RRC is no longer collision-free and ORMA may not handle reservation conflicts as expected. The reiterative RTR help address this problem by detecting and handling such unresolved conflicts at least once every frame, so the chance for RRC collision in every frame is usually low after several frames. Although a larger k help more efficiently handle reservation conflicts, more bandwidth could be wasted. As a part of future work, we will study how to choose k to balance between efficiently handling reservation conflicts and improving channel utilization.

VII. Conclusion

This paper introduced the opportunistic reservation multiple access (ORMA) protocol for ad hoc networks. ORMA supports broadcast communications through time-division multiple access. In ORMA, nodes participate in a priority comparison to determine which node can transmit during a time slot that has not been reserved. Once a node wins, it opportunistically requests to reserve the time slot. In this way, ORMA allows nodes to incrementally reserve channel-access time, until nodes reserve the desired number of slots in each time frame. The advantage of ORMA is that it inherits both the high throughput merit of probabilistic channel access schemes and the bounded access delay merit of reservation-based schemes. ORMA will be suitable for ad hoc networks when both voice and data services are provided.

References

[1] A.Ephremides and T.V.Truong. Scheduling broadcasts in multihop radio networks. *IEEE Transactions on Communications*, 38(4), April 1990.

[2] I. . W. Group. Draft standard for wireless LAN: Medium access control (MAC) and physical layer (PHY) specifications. *IEEE*, July 1996.

[3] I.Cidon and M.Sidi. Distributed assignment algorithms for multihop packet radio networks. *IEEE/ACM Transactions on Computers*, 38(10), October 1998.

[4] J.Chlamtac and A.Farago. Making transmission schedules immune to topology changes in multihop packet radio networks. *IEEE/ACM Transactions on Networking*, 2(1), February 1994.

[5] J.Chlamtac and A.Farago. Time-spread multiple-access (TSMA) protocols for multihop mobile radio networks. *IEEE/ACM Transactions on Networking*, 5(6), December 1997.

[6] J.J.Garcia-Luna-Aceves and C.L.Fullmer. Floor acquisition multiple access (FAMA) in single-channel wireless networks. *Mobile Networks and Applications*, 4(3), October 1999.

[7] J. Ju and V. Li. An optimal topology-transparent scheduling method in multihop packet radio networks. *IEEE/ACM Transactions on Networking*, 6(3):298–306, June 1998.

[8] L.Bao and J.J.Garcia-Luna-Aceves. A new approach to channel access scheduling for ad hoc networks. In *ACM MOBICOM*, Rome, Italy, July 2001.

[9] L.Bao and J.J.Garcia-Luna-Aceves. Hybrid channel access scheduling in ad hoc networks. In *IEEE ICNP*, Paris, France, November 2002.

[10] L.C.Pond and V.O.K.Li. A distributed time-slot assignment protocol for mobile multihop broadcast packet radio networks networks. In *IEEE MILCOM*, volume 1, Boston, MA, July 1989.

[11] L.Kleinrock and F. Tobagi. Packet switching in radio channels: Part i-carrier sense multiple access modes and their throughput-delay charcteristics. *IEEE Transactions on Communications*, 23:1400–1416, 1975.

[12] N.Abramson. The ALOHA system-another alternative for computer communications. *Fall Joint Computer Conference*, pages 281–285, 1970.

[13] P.Karn. MACA- a new channel access method for packet radio. *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pages 281–285, September 1990.

[14] C. Rentel and T. Kunz. On the average-throughput performance of code-based scheduling protocols for wireless ad hoc networks. In *Poster presentation, ACM MOBIHOC*, Urbana-Champaign, IL, May 2005.

[15] R.Nelson and L.Kleinrock. Spatial TDMA: A collision-free multihop channel access protocol. *IEEE Transaction on Communications*, COM-33:934–944, 1985.

[16] R.Ramanathan. A unified framework and algorithm for channel assignment in wireless networks. *Wireless Networks*, 5(2):81–94, 1999.

[17] R.Rozovsky and P.R.Kumar. SEEDEX: A MAC protocol for ad hoc networks. In *ACM MOBIHOC*, Long Beach, October 2001.

[18] V.Bhargavan, A.Demers, S.Shenker, and L.Zhang. MACAW-a media access protocol for wireless LANs. In *ACM SIGCOMM*, pages 212–225, 1994.

[19] V.Rajendran, K.Lbraczka, and J.J.Garcia-Luna-Aceves. Energy-efficient, collision-free medium access control for wireless sensor networks. In *ACM SENSYS*, Los Angeles, 2003.