Title

CSIR: Cellular scheduling with Interest-driven Routing

Authors

Masilamani, A.
Dabirmoghaddam, A.
Garcia-Luna-Aceves, J.J.

Publication Date

2013

Peer reviewed

CSIR: Cellular Scheduling With Interest-Driven Routing

Ashok N Masilamani, Ali Dabirmoghaddam, J.J. Garcia-Luna-Aceves
Department of Computer Engineering
University of California, Santa Cruz
Santa Cruz, CA 95064
Email: {ashok, alid, jj}@soe.ucsc.edu

***Abstract*—CSIR (Cellular Scheduling with Interest-driven Routing) is proposed for the effective dissemination of real-time and elastic traffic in wireless ad-hoc networks. CSIR is a novel cross-layer framework consisting of five components: flow-based priority queuing of packets, distributed interest-based routing, distributed transmission scheduling, a neighbor protocol, and bandwidth reservations. Nodes are scheduled for transmission in coordination with the routes established and the end-to-end delay and bandwidth requirements of the flows. Most of the signaling overhead for active flows is confined to the nodes that are required to maintain them. Results from detailed simulations indicate that, compared to a protocol stack consisting of IEEE 802.11e for channel access, AODV and OLSR for unicast routing, and ODMRP for multicast routing, CSIR attains much better performance in terms of packet delivery and end-to-end delay for elastic and real-time unicast and multicast traffic.**

I. Introduction

Recent advancements in processor design, radio technology and storage elements have made Mobile Ad hoc Networks (MANET) a reality. Being designed to operate dynamically in absence of any fixed infrastructure, in theory, MANETs provide an ideal framework for many distributed applications such as disaster relief, search and rescue and battlefield operations. Nevertheless, in practice, such applications cannot be supported effectively nowadays, because the protocol architectures used in MANETs are derivatives of the protocol-stack originally tailored for wired networks.

In wired networks, topologies are static, bandwidth is plentiful and links fail independently. Such characteristics enable decoupling protocols from the physical medium and making the operations of the protocols independent of each other. However, such a strategy is not well-suited for wireless networks, given that the wireless medium is shared, links suffer from multiple access interference (MAI) and bandwidth is relatively scarce. In addition, the topologies in wireless networks are usually dynamic, due to the mobility and environmental effects (*e.g.*, fading and noise). Therefore, since wireless links are not isolated, transmissions by a particular node may impact the performance of all routes with nodes in the interference range of the transmitter.

The fundamental differences between wireless and wired links make cross-layering an attractive approach for managing network resources and information dissemination in MANETs. Section II summarizes the prior work focusing on cross-layering approaches to routing and transmission scheduling. This summary reveals that the solutions proposed in the past are either based on centralized algorithms requiring global knowledge of the network at every node, or do not integrate routing and scheduling with the establishment of bandwidth reservations and traffic management. Most prior solutions do not address the integration of multicast routing with transmission scheduling. Furthermore, to the best of our knowledge, no prior solution exists that offer a cross-layer framework for networks in which nodes are equipped with GPS system.

This paper introduces a new cross-layer framework for the dissemination of unicast and multicast flows that may be real-time (*e.g.*, VOIP) or elastic (*e.g.*, HTTP). Section III presents the Cellular Scheduling with Interest-driven Routing (CSIR) approach. It consists of five components: (a) A flow-ordered priority-based queuing system to differentiate and handle signaling traffic, elastic traffic and real-time traffic; (b) an interest-driven routing algorithm that constructs and maintains unicast and multicast routing structures and confines signaling traffic to those nodes associated with the flows; (c) a distributed transmission scheduling algorithm based on GPS coordinates of nodes with minimal signaling overhead; (d) a channel allocation algorithm that integrates scheduling and routing decisions to guarantee end-to-end delays for real-time traffic; and (e) a neighbor protocol used to exchange the information necessary for scheduling and channel allocation.

Section IV describes the channel access approach used in CSIR. The geo-spatial coordinates of nodes are used together with their transmission and interference ranges to define collision-free transmission schedules using deterministic distributed algorithms. Each node needs to know only the geo-spatial coordinates of its immediate neighbors to derive collision-free transmission schedules in the presence of hidden terminals. Furthermore, the length of transmission frames used to organize the wireless channel consists of the minimum number of time slots needed to avoid multiple access interference. The transmission-frame length is determined by the transmission and interference ranges of the nodes, and is independent of the number of nodes or density of nodes.

Section V describes the priorities used to transmit signaling traffic, real-time data or elastic data; and Section VI presents the interest-driven approach used in CSIR for unicast and multicast routing. CSIR establishes routing meshes using the interest-driven routing scheme proposed in [7], and orders the time slots over the routing meshes created for active flows to guarantee end-to-end delays for unicast and multicast traffic without sending multiple reservation packets. The signaling traffic used for the maintenance of routes and allocation of time slots to flows is limited to those nodes that are essential to the establishment and maintenance of the flow.

Section VII describes the results of detailed simulation experiments used to study the performance of CSIR and compares it with the performance of a traditional MANET

protocol stack consisting of the IEEE 802.11e DCF [1] for channel access which works independently of the routing protocols used for unicast (AODV [15], OLSR [8]) and multicast (ODMRP [9]) traffic. The results for combined unicast and multicast traffic demonstrate that CSIR outperforms the traditional approach for elastic traffic in terms of packet delivery ratio, and that it attains end-to-end delays well within the 200 ms required to support real-time voice applications, even in the presence of high traffic loads in the network.

II. Related Work

Many cross-layer frameworks have been proposed in the past that attempt to make routing and channel access more efficient in ad hoc networks. Due to space limitations, we review only a small representative sample of prior approaches.

Chen and Heinzelman [6] present a comprehensive survey on cross-layer approaches in which routing protocols provide some sort of support for quality of service in MANETs, and Melodia *et al*. [12] provide a survey of cross-layer scenes for wireless sensor networks. MACA/PR (multiple access collision avoidance with piggyback reservations) [11] was one of the first approaches attempting to integrate channel access, routing and traffic management. MACA/PR is designed for unicast traffic. It extends IEEE 802.11 DCF for channel access to incorporate a bandwidth reservation mechanism and a modified version of DSDV [14] that keeps track of the bandwidth of the shortest paths to each destination and the maximum bandwidth available over all possible paths. Bandwidth reservations are made by the first data packet of a real-tie flow. One-hop scheduling information is piggybacked in data packets and reservations are made taking into account two-hop neighborhood information. However, reservations are done without any coordination with the routing protocol in that scheme.

DARE [5] is a channel access protocol for MANETs that provides end-to-end reservations at the MAC layer using traditional on-demand routing protocol at the routing layer. It propagates request-to-reserve messages that travel from sources to destinations through routes established by the routing protocol. Destinations reply with clear-to-reserve messages that travel along reverse paths establishing the actual reservations. The main limitation of DARE is that routing decisions do not take into account any information regarding the reservations being made or any data collected for channel access.

Cai *et al*. [4] propose an optimization algorithm for end-to-end bandwidth allocation. It focuses on maximizing the number of flows with bandwidth restrictions that a MANET can accommodate. The disadvantage of that algorithm is that it requires global resource information along the route and the route itself is not considered in the optimization algorithm. Setton *et al*. [16] propose a cross-layer framework that incorporates adaptations across all layers of the protocol stack. The main limitation of the proposed framework is that it is mostly based on centralized algorithms and a link-state approach. This makes it unsuitable for the highly dynamic MANETs or very large ad hoc networks.

STORM [13] is a cross-layer framework that integrates unicast and multicast routing with a MAC scheme based on elections of time slots [2]. It provides end-to-end bandwidth and delay guarantees for real-time flows and limits signaling only within the regions of interest. The limitation of STORM is that multiple reservation packets are required for nodes to allocate the required bandwidth along the link, which may result in long convergence times and sensitivity to mobility. The scheduling in STORM is done by using a slot identifier at the core node and allocating cascaded slot assignments from the core node to the source along the path. Therefore, the scheduling decisions are made in coordination with all nodes along the path with multiple reservation packets forwarded by one-hop neighbor nodes.

Other proposals [17], [18] addressing multicast communication have focused on static networks. In those works, the joint multicast routing and the problems of power control [18] or network planning [17] are formulated as a cross-layer optimization problem.

Many routing protocols have been reported that take advantage of location information [10]. However, the location-based routing protocols work independently of the MAC protocol and do not offer any end-to-end guarantees. Limited work exists on using geographical location to help define how nodes should share a common channel [19], [20]. Though those protocols use GPS coordinates for scheduling, the schedules are independent of the application and routing requirements for the flows.

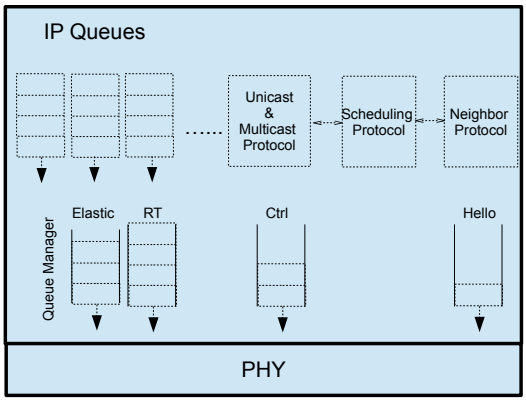

Fig. 1: CSIR Architecture

III. CSIR: Cellular Scheduling with Interest Driven Routing

In the context of this work, it is assumed that: (a) the radios used by the nodes are half-duplex and can tune to only one channel at a time; (b) radio links are bidirectional; (c) each node is endowed with a GPS receiver and knows its own geographical location, but it is unaware of the terrain dimensions in which the network operates; (d) time is slotted with time slots having a fixed duration; and (e) any pair of nodes can be synchronized at the time-slot level. For simplicity, the rest of the paper also assumes that all the nodes have the same constant transmission, reception and interference ranges; and that the network uses a single channel. However, CSIR can be applied to more general cases with variable radio ranges and multiple channels as well.

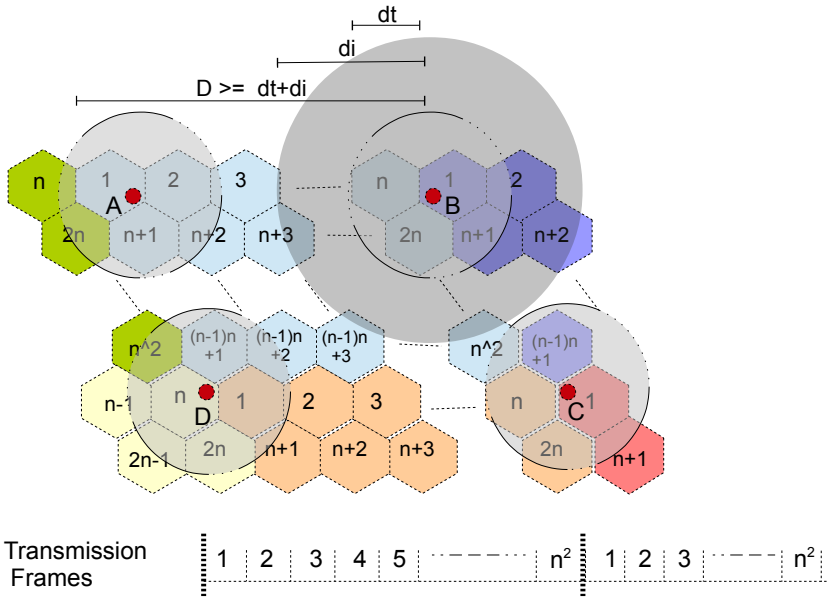

Fig. 2: Blocks of size $n \times n$ cells

The objective of CSIR is to coordinate routing, scheduling, and traffic management in order to disseminate flows from a source to destination(s) across the network while meeting the bandwidth and delay requirements for the flow. Fig. 1 illustrates the components used in CSIR. The routing component of CSIR establishes and maintains interest-driven routes for the flows. The traffic management component consists of priority queues for control, real-time and elastic (non-real-time) traffic. The scheduling component coordinates with the routing component and uses *Hello* packets to schedule transmission slots and allocate additional bandwidth if necessary for real-time flows. Channel access is based on spatial classification of nodes into cells based on their GPS coordinates and provide channel-access delay guarantees for both elastic and real-time flows.

IV. Channel Access

A. Channel Structure

As shown in Fig. 2, the Euclidean space is organized into blocks of n^2 cells. Each cell is of hexagonal shape of circum-radius d_r. Each cell is assigned a location number from 1 to n^2 according to its position in the block. The latitude and longitude coordinates of a node's location are available as 'x' and 'y' coordinates in the two-dimensional Euclidean space starting from two common points of origin (*e.g.*, the north and south poles). From these coordinates, each node is able to calculate the cell number in which it resides using a modulo ($\%$) operator, as illustrated in the example in Fig. 3.

The length of a block, denoted by D, is set such that $D \geq (d_t + d_i)$ (as shown in Fig. 2), where d_i is the interference range of a node. This is to ensure that two nodes located in the same cell number in adjacent blocks do not interfere with each other when they transmit concurrently. For example,

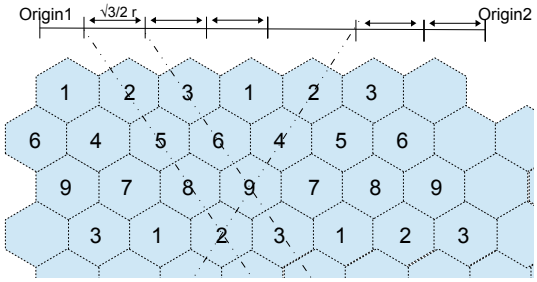

Fig. 3: Blocks of size 3×3 cells

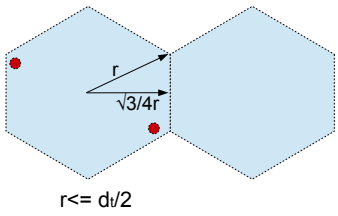

Fig. 4: Cell size

concurrent transmissions by nodes A, B, C and D in Fig. 2 do not interfere with each other. The circum-radius of a cell, d_r, should not be any greater than d_t, the transmission range of the node, as shown in Fig. 4. This guarantees that every node in a cell is within the transmission range of all other nodes within the same cell and hence nodes in the same cell can hear each other. Given that the interference range d_i is greater than the

transmission range d_t, and that $D \geq (d_i + d_t)$, it is easy to observe that $n \geq 3$ (*i.e.*, a total of at least 9 cells in a block) is necessary to avoid multiple access interference (MAI). As Fig. 2 also shows, the common channel is organized into transmission frames, with a transmission frame F_t consisting of a constant number of time slots equal to the number of cells in a block (n^2). It can be shown that the schedules are collision free, given that nodes that can potentially transmit in the same slot are either within the same cell or are located at a distance that is away from their interference regions.

Each time slot in CSIR is used for the transmission of a *Hello* and zero or more data packets. A *Hello* consists of the node identifier (ID), its own geographical coordinates and a $rank$ value that specifies total number of IDs (both real and virtual) within the cell.

Each node maintains a list of one-hop neighbors along with the geographical coordinates and *rank* reported by each neighbor. The node also has a counter that is used to keep track of the number of active one-hop neighbors that belong to the same cell as the node itself. The counter is incremented every time a *Hello* is received from a new neighbor in the same cell and it is decremented when a neighbor fails to send a *Hello* for a prespecified period of time. This value gives the number of one-hop neighbors of a node located in the same cell as the node itself.

Nodes share the same frequency band and access the common channel assuming that it is organized using the time-division multiple access frame structure illustrated in Fig. 2. Each CSIR frame is composed of N time-slots (from 0 to $N-1$), where $N = n^2$. Each time slot in the frame is assigned to the corresponding cell number. There is one special-purpose time slot used to admit new nodes to the network. These admission time slots occur every A time slots, when $A \gg N$, and are used by nodes to transmit their first *Hello* packet on a contention basis. The packets sent within a time slot follow a strict priority policy as shall be explained in Section V.

B. Transmission Scheduling

Once a node finds out its own geographical location, it automatically calculates its block and cell number in the block using its distance and location from a common point or origin. Each cell number is assigned a time slot in each transmission frame F_t. We use (k_x, k_y) to denote the Cartesian coordinates of node k.

For simplicity, in the rest of this paper, we assume $d_t = d_i$, but other choices of d_i and d_t can also be used in CSIR. This same assumption has been made previously in prior MAC protocols based on transmission scheduling [3]. This assumption leads to the spatial classification and channel structure shown in Fig. 3, where each block must consist of 3×3 cells of circum-radius d_t, with cells numbered from 1 to 9.

Algorithm 1 is used to determine the rate at which a node accesses the time slots specific to the cell in which it resides. Each node maintains a sorted list of real and virtual node IDs located within the same cell and nodes are scheduled for transmission according to that list.

Algorithm 1 Transmission Algorithm

```
Let List_i be a sorted list at node i and rank ← 0
Insert i in List_i
Let OHN_i be set of nodes in one-hop-neighborhood of i
for all k in OHN_i do
    Calculate cell number c_k from (k_x, k_y)
    if c_k == c_i then
        Insert k in sorted list List_i
        rank ← rank + 1
    end if
end for
t ← current time
Find index j of node i in List_i
Cell slot c_s ← (t % n^2) + 1
if c_s == c_k and ((t − c_s)/n^2) % rank == j then
    State_i ← TRANSMIT
else
    State_i ← RECEIVE
end if
```

In a cell c, if a node finds no one else in the cell, it feels free to transmit in all slots t for which $t \,\%\, n^2 = c - 1$. For example, in Fig. 2, node A is the only node in cell 1 and can transmit in slots $0, 9, 18, 27, etc$. If more than one node share the same cell within a block, they would take turns to access the channel in their cell-specific slot in lexicographical order. As shown in Algorithm 1, the *rank* value (aggregate number of real and virtual neighbors in a particular cell) is used to determine the transmission schedule for each node.

C. Channel Allocation

The routing algorithm described in Section VI interacts with the neighbor protocol in order to determine the MAC throughput of the node. It also uses the scheduling algorithm to allocate the required time slots in a cell for a node in response to real-time traffic requirements. The channel allocation process is triggered at node x when a Mesh Announcement (*MA*) packet is received by node x when $x \in next_S$, as shall be described in Section VI. The process is also triggered when a *Hello* packet is received with a virtual node ID and the node is serving active real-time flows. The additional slots are allocated when nodes broadcast virtual *Hello* packets with virtual node IDs in their scheduled transmit slots. The virtual node IDs are added to the one-hop neighborhood list of each node including the sender. The other nodes in the cell treat the virtual node IDs as if they were regular node IDs and count them towards the *rank* of the cell. This allows the node sending virtual node IDs to transmit in additional slots more dynamically. Because all nodes within a cell are one-hop neighbors, the additional bandwidth is granted to nodes, without having to coordinate with the two-hop neighbors.

In the following, we discuss how a node decides to advertise virtual node IDs. Let B be the physical layer bandwidth available to the node, and τ be the slot duration. We denote by δ_x the time interval between two consecutive transmission slots for node x. The *MA* packet received by node x specifies the maximum tolerable end-to-end delay Δ_ϕ required by the real-time flow ϕ. The packet also contains the number of hops to the core node (destination node for unicast flows). Node x

uses this value along with the hop-distance to the source from the Mesh Request (*MR*) packet received earlier to determine the total number of intermediate hops, h_ϕ, that flow ϕ needs to travel. Let S be the packet size in bits. A minimum of $\left\lceil \frac{S}{B \times \tau} \right\rceil$ slots are thus needed for transmitting a packet. The inter-slot duration δ_x should be chosen to satisfy

$$(\delta_x + \tau)\left\lceil \frac{S}{B \times \tau} \right\rceil \leq \frac{\Delta_\phi}{h_\phi \, \bar{f}_\phi}\,, \tag{1}$$

where $\bar{f}_\phi$ is the average number of flows served by nodes handling flow ϕ.

Assuming a slot duration large enough to avoid packet fragmentation, Eq. (1) can be simplified to the following requirement on the inter-slot interval:

$$\delta_x \leq \frac{\Delta_\phi}{h_\phi \bar{f}_\phi} - \tau\,. \tag{2}$$

Node x verifies whether the current inter-slot interval determined by Algorithm 1 satisfies the requirement in Eq. (2). If so, no further slot allocation is needed. However, if the inter-slot interval is longer, the node calculates additional slots needed and broadcasts supplementary *Hello* packets with virtual node IDs to accommodate the specified end-to-end delay requirement. This allows the node to allocate sufficient bandwidth instantaneously without any additional coordination with two-hop neighbors.

V. Traffic Management

When a node wins a slot for transmission using the transmission algorithm described in Section IV-B, it fits in as many packets in the slot as possible. The node uses a strict priority scheduler to dequeue packets from its local FIFO transmission queue. Packets are taken from the non-empty queue with the highest priority. If more than one non-empty queues with the same priority exist, their packets are served in a round-robin fashion.

Hello packets have the highest priority (P_{hello}), since they are used for scheduling and it is necessary to provide them with collision free channel access. Network layer signaling packets have the second highest priority (P_{ctrl}), since these packets are used to establish routes and allocate additional bandwidth. This is followed by real-time data packets (P_{rt}). Finally, elastic traffic ($P_{elastic}$) has the lowest priority. To summarize, during a transmission time-slot allocated to a node, the relationships among traffic priorities is given as

$$P_{elastic} < P_{rt} < P_{ctrl} < P_{hello}\,. \tag{3}$$

VI. Interest-driven Routing

A. Overview

Routing in CSIR is based on the combined unicast and multicast routing protocol proposed in [7]. However, in addition to the data structures maintained by PRIME, CSIR maintains and updates data structures that are used by the scheduling algorithm to allocate slots dynamically. CSIR also maintains a mesh request list *MRL* and chooses forwarding nodes towards the source that is used to forward *MA* packets as will be described later in Section VI-D.

A routing mesh is established and maintained for each unicast and multicast flow in the network. The first node that becomes an active source for a unicast or multicast flow sends the first data packet piggybacked on a mesh request (*MR*) packet. The *MR* is flooded up to a horizon threshold by nodes that are able to furnish the required end-to-end delay and bandwidth guarantees for the flow. In case of a flow to a unicast destination, on receiving *MR*, the destination node sends *MA* packets along the shortest path back to the source. For multicast receivers, all nodes that belong to the multicast group participate in a core election algorithm for the group. The routing mesh is maintained by the *MA* packets generated continuously by the core.

B. Destination Nodes and Routing Meshes

Let D denote the set of destination nodes. For unicast flows, D comprises a single node, whereas for multicast data flows, D contains all nodes in the multicast group as well as the ones needed to ensure connectivity between them. D_m refers to the *routing mesh* of nodes that include destination node(s) in D and the relays connecting the source to such destination(s).

The routing meshes used are composed of paths that meet the end-to-end delay requirements for that particular flow. As discussed in Section IV-C, a routing path is constructed through a particular cell only if the nodes in the cell can allocate slots as determined by the flow requirements.

Elastic flows are simply routed using shortest paths. Nodes in the path do not allocate additional slots since the flows demand no stern end-to-end restrictions. For real-time flows, nonetheless, additional slots are dynamically allocated in participating cells along the path as required.

The meshes are also used to limit the propagation of signaling packets only to the regions of the network with interest in the given data flow. It has been shown [7] that such a routing scheme guarantees instantaneous loop-freedom. Meshes are constructed and maintained by the unicast destination or by a dynamically elected delegate for the multicast group. Presence or absence of data packets entails activation or deactivation of meshes.

C. Information Stored and Packets Exchanged

When a node becomes an active source for a unicast or multicast flow, it floods the data packet piggybacked on a *MR* up to a horizon threshold. For elastic traffic, all nodes in the network forward the *MR*. However, for real-time packets, only nodes that are able to guarantee a certain per-cell delay get to forward the *MR*. Nodes forwarding the *MR* store state information of the source. Such information is used to route *MA* back to the source later on.

The *MR* is a eight-tuple of the form: ⟨*type, hops, horizon, more-data, src, dest, id, data*⟩, where *type* is the type of message; *hops* is the hop count from source; *horizon* is the application-specific horizon threshold; *more-data* is the persistence of interest at the source; *src* is the source address; *dest* is the destination address (unicast or multicast); *id* is the packet ID; and *data* is the piggybacked data payload.

For a given source S, nodes maintain a neighborhood mesh request list MRL_S that stores an ordered set composed of the *MR*'s that they have received recently from that source. The requests stored at MRL_S ordered under a strict total order relation $\prec$ defined by Eq. (4), where d_S^A is the distance in hops from node A to source S and $rank^A$ is the *rank* of node A as defined in Section IV. The list MRL_S may contain multiple *MR*'s of the same order.

$$MR_S^B \prec MR_S^A \Leftrightarrow \left(d_S^B > d_S^A\right) \vee \left((d_S^B = d_S^A) \wedge (rank^B < rank^A)\right) \quad (4)$$

The proof that $\prec$ is anti-reflective, asymmetric and transitive is straightforward and results from the fact that sequence numbers, distances and node identifiers can be seen as natural or real numbers and that node identifiers are unique. The details are omitted in the interest of space.

When the *MR* reaches the intended unicast destination or a multicast subscriber, the node starts sending *MA*'s. The *MA*'s are used to establish routing meshes in order to interact with the neighboring protocol, allocate resources locally, meet end-to-end schedules for real-time flows and elect the multicast core in case of a multicast flow.

When a node x receives a *MA* with preferred next-hop to destination set as node x, it calculates the required bandwidth for the flow and allocates it instantaneously by sending a *Hello* packet. Therefore, when data packets are sent, the route to destination is established and the channel allocations are already in place to satisfy the delay and bandwidth requirements.

MA's are sent periodically along the mesh with monotonically increasing sequence numbers. A mesh announcement MA_D^{*B} transmitted by node B for destination D is a seven-tuple of the form: $\langle\ id^{*B}, core_D^{*B}, sn_D^{*B}, d_D^{*B}, mm_D^{*B}, next_D^{*B}, next_S^{*B}\ \rangle$, where id^{*B} is the identifier of B; $core_D^{*B}$ is either the identifier of the core of the multicast group D known by B or the identifier of the unicast destination; sn_D^{*B} is the largest sequence number known by B of destination D; d_D^{*B} is the distance of B to the core of D or to the destination itself in case of a unicast flow; mm_D^{*B} is a multicast-specific flag that indicates whether B is a member of the mesh, multicast group, both or none; $next_D^{*B}$ is the identifier of the preferred next hop of B towards the core of D; and $next_S^{*B}$ is one or more identifiers of the preferred next hop of B towards the source S that is used to allocate channel and forward *MA*.

Algorithm 2 is used to update the routing information at nodes from the *MA*.

For a given destination D, nodes maintain a neighborhood multicast announcement list MAL_D that stores an ordered set composed of the *MA*'s that the node has recently received from each of its neighbors for that destination. The announcements stored at MAL_D are also augmented with a time-stamp (*ts*) obtained from the local clock and are ordered under a strict total order relation $\prec$ which is anti-reflective, asymmetric and transitive and is defined as follows.

$$MA_D^B \prec MA_D^A \Leftrightarrow \left(sn_D^B < sn_D^A\right) \vee \left((sn_D^B = sn_D^A) \wedge (d_D^B > d_D^A)\right) \vee \left((sn_D^B = sn_D^A) \wedge (d_D^B = d_D^A) \wedge (id^B < id^A)\right). \quad (5)$$

Algorithm 2 UpdateRoutingState(x, MA_D^{*B})

if $\left(core_D^{*B} = core_D^x\right) \wedge \left(sn_D^x \leq sn_D^{*B}\right)$ **then**

$$MAL_D \leftarrow \begin{cases} MAL_D \cup \{MA_D^{*B}\} & \text{if } MA_D^B \notin MAL_D \\ MAL_D - \{MA_D^B\} \cup \{MA_D^{*B}\} & \text{if } sn_D^x \leq sn_D^{*B} \end{cases}$$

$$fd_D^x \leftarrow \begin{cases} d_D^{*B} & \text{if } sn_D^{*B} > sn_D^x \\ \min\{fd_D^x, d_D^{*B}\} & \text{if } sn_D^{*B} = sn_D^x \\ fd_D^x & \text{otherwise} \end{cases}$$

$$sn_D^x \leftarrow \max\{sn_D^x, sn_D^{*B}\}$$

$$d_D^x \leftarrow \begin{cases} d_D^i + lc_i^x : \max\limits_{i \in MAL_D : sn_D^i = sn_D^x} \{i\} & \text{if such } i \text{ exists} \\ \infty & \text{otherwise} \end{cases}$$

$$CF_D^x \leftarrow \{i : \left(i \in MAL_D\right) \wedge \left(fd_D^x = d_D^i\right) \wedge \left(sn_D^i = sn_D^x\right)\}$$

$$next_D^x \leftarrow \begin{cases} id^i : \max\limits_{i \in CF_D^x} \{i\} & \text{if such } i \text{ exists} \\ \text{NIL} & \text{otherwise} \end{cases}$$

$$next_S^x \leftarrow \begin{cases} id^i : \max\limits_{i \in MRL_S^x} \{i\} & \text{for all } i \\ \text{NIL} & \text{otherwise} \end{cases}$$

else

if $core_D^{*B} > core_D^x$ **then**

$MAL_D \leftarrow \{MA_D^{*B}\}$; $core_D^x \leftarrow core_D^{*B}$; $fd_D^x \leftarrow d_D^{*B}$; $sn_D^x \leftarrow sn_D^{*B}$; $d_D^x \leftarrow d_D^{*B} + lc_B^x$; $next_D^x \leftarrow id^B$

D. Processing Mesh Announcements

When a node x receives a MA_D^{*B} from neighbor B, it updates its routing information using Algorithm 2.

As seen in Step 1 of Algorithm 2, node x accepts the *MA* if it contains a sequence number equal or larger than the current largest sequence number stored at x, or if it is the first time that a *MA* is received from B (see Eq. (6)). If the *MA* is accepted, then:

$$MAL_D \leftarrow \begin{cases} MAL_D \cup \{MA_D^{*B}\} & \text{if } MA_D^B \notin MAL_D\,, \\ MAL_D - \{MA_D^B\} \cup \{MA_D^{*B}\} & \text{if } sn_D^x \leq sn_D^{*B}\,, \\ MAL_D & \text{if } sn_D^x > sn_D^{*B}\,. \end{cases} \quad (6)$$

The feasible distance to the core of x, denoted by fd_D^x (Step 3), is a non-increasing function over time that is used to maintain loop-free routes to destination. It can only be reset by a change of core or by a new sequence number (Eq. (7)). Feasible distances are used to select a *feasible set* of next hops towards the core of the destination. Not all nodes part of the feasible set have the required slots allocated to them to guarantee end-to-end flow metrics. However, it has been shown [7] that the way in which nodes select their next hops suffices to guarantee instantaneous loop-freedom.

$$fd_D^x \leftarrow \begin{cases} d_D^{*B} & \text{if } sn_D^{*B} > sn_D^x\,, \\ \min\{fd_D^A, d_D^{*B}\} & \text{if } sn_D^{*B} = sn_D^x\,, \\ fd_D^x & \text{otherwise}\,. \end{cases} \quad (7)$$

The sequence number sn_D^x stored at node x for the core of destination D is a strictly increasing function over time that can only be reset by a change of core (Eq. (8)). The max function of Step 4 and Eq. (8) is the traditional max function defined over the natural numbers. The core-specific sequence number is an unsigned monotonically increasing counter. Due to the numerical limits of the integer representation of the counter, a time-stamp taken from the real-time clock of the core node is attached to the sequence number. When the counter reaches the maximum unsigned integer value, a new time-stamp is used and the counter is reset to zero. Hence,

$$sn_D^x \leftarrow \max\{sn_D^x, sn_D^{*B}\} \tag{8}$$

The distance to the core of destination D of node x, denoted by d_D^x (Step 5), is computed using Eq. (9). By definition, the core of the group has a 0 distance to itself and its feasible distance is always 0. We assume that the link cost lc is always equal to 1.

$$d_D^x \leftarrow \begin{cases} d_D^i + lc_i^x : \max\limits_{i \in MAL_D : sn_D^i = sn_D^x} \{i\} & \text{if such } i \text{ exists} \\ \infty & \text{otherwise} \end{cases} \tag{9}$$

The address of the next hop to the core of D, denoted by $next_D^x$ (Step 7), is also computed using the relation $\prec$ defined by Eq. (5), the current values of the feasible distance and sequence number:

$$next_D^x \leftarrow \begin{cases} id^i : \max\limits_{i \in CF_D^x} \{i\} & \text{if such } i \text{ exists} \\ \text{NIL} & \text{otherwise} \end{cases} \tag{10}$$

where CF_D^x is set to $F_D^x = \{i : (i \in MAL_D) \wedge (fd_D^x = d_D^i) \wedge (sn_D^i = sn_D^x)\}$, which is the set of x's feasible neighbors for destination D.

The set of nodes that are chosen to forward the MA_D^{*x} towards the source S are also determined by relation $\prec$ defined in Eq. (4) over MRL_S.

$$next_S^x \leftarrow \begin{cases} id^i : \max\limits_{i \in MRL_S^x} \{i\} & \text{for all } i \\ \text{NIL} & \text{otherwise} \end{cases} \tag{11}$$

If a node x receives a *MA* advertising a core with a larger identifier (Step 9) then $\{MA_D^{*B}\}$ is added to MAL_D, $core_D^x$ is set to $core_D^{*B}$ and the other parameters are set as follows: fd_D^x to d_D^{*B}, d_D^x to $d_D^{*B} + lc_B^x$, sn_D^x to sn_D^{*B}, and $next_D^x$ to id^B (Steps 10-12). Otherwise, if $core_D^{*B} < core_D^x$, then the *MA* is simply discarded.

In addition to the fields updated as described in Algorithm 2, the mesh membership flag $mm_D^x \in \{\text{REG, RCV, MM, RM, NIL}\}$ is also updated on the reception of *MA*. The mm_D^x indicates whether x is a regular node (REG), a group receiver (RCV), a mesh member (MM), both group receiver and mesh member (RM) or if D is a unicast destination (NIL). A node x is a mesh member if and only if

$$\exists\, y \in MAL_D : \left(mm_D^y \neq \text{REG}\right) \wedge \left(mm_D^y \neq \text{NIL}\right) \wedge \left(d_D^y > d_D^x\right) \wedge \left(next_D^y = id^x\right) \wedge \left(ct \leq ts_D^y + MA\_interval\right), \tag{12}$$

where ts_D^y is the time-stamp added to y when it was stored in MAL_D, ct is the current value of x's clock and *MA_interval* is interval at which *MA* packets are sent by the core.

E. Mesh Announcement and Data Packet Forwarding

The core node generates *MA* periodically every *MA_interval* period with increasing sequence number and is forwarded by nodes that belong to the mesh establishing next hop pointers towards the core. *MA*'s are also broadcast when there is a change in membership status. In case of a multicast destination, a multicast group member declares itself the core if it received a *MR* for the group for which it has not received a *MA*.

If multiple nodes declare themselves as the core, the node with the highest core ID is adopted as the core. If a node does not receive a fresh *MA* from the core for three consecutive *MA*-periods, it detects a partition and executes the core election algorithm by declaring itself as the core. Nodes that lay in paths $p = \langle R, n, n_1, \cdots, n_k, core\rangle$ with $next_D^R = n, next_D^n = n_1, \ldots, next_D^{n_k} = core_D$ are forced to become mesh members creating a connected component referred to as *multicast destination*.

Upon reception of a data packet from a source node, nodes check if the *(sender's address, sequence number)* pair is already in the cache. If so, the packet is dropped. Otherwise, the node forwards the data packet received from neighbor y with destination D if

$$\left(mm_D^x = RM\right) \vee \left(mm_D^x = mm\right) \vee \left(\exists\, y \in MAL_D : \left(d_D^y > d_D^x\right) \wedge \left(next_D^y \leq id^x\right)\right). \tag{13}$$

Eq. (13) states that node x forwards a data packet received from node y if x is part of the multicast mesh or if x was selected by the previous relay (*i.e.*, y) as a next hop to the core.

VII. Simulation Results

We present simulation results comparing CSIR against ODMRP [9] for multicast traffic (MCBR flows), as well as AODV [15] and OLSR [8] for unicast traffic (CBR flows) operating over 802.11 DCF MAC [1]. We selected these protocols because they are commonly used as baselines for performance comparisons of multicast and unicast routing protocols, as well as channel access. Even though these protocols were not designed to support real-time traffic, they constitute a good reference point to show the performance gains of our approach.

We use the 802.11a physical layer with data rates up to 54 Mbps, given that the Qualnet simulator [21] currently does not support the 802.11n physical layer. We use packet delivery ratio and end-to-end latency as our performance metrics. We employ a Poisson distributed random topology. We use random waypoint as the mobility model with nodes moving at speeds up to 4m/s. Simulations were done for both static and dynamic topologies.

We used the discrete-event simulator Qualnet [21] version 5.0 that provides a realistic simulation of the physical layer, and well-tuned versions of ODMRP, AODV, OLSR and IEEE 802.11e DCF, which we call WiFi for simplicity. The time-slot duration for CSIR was set to 1 ms, with the protocols capable of transmitting multiple data and control packets during a single time slot. Each simulation scenario was repeated five times using distinct seeds and average values are reported.

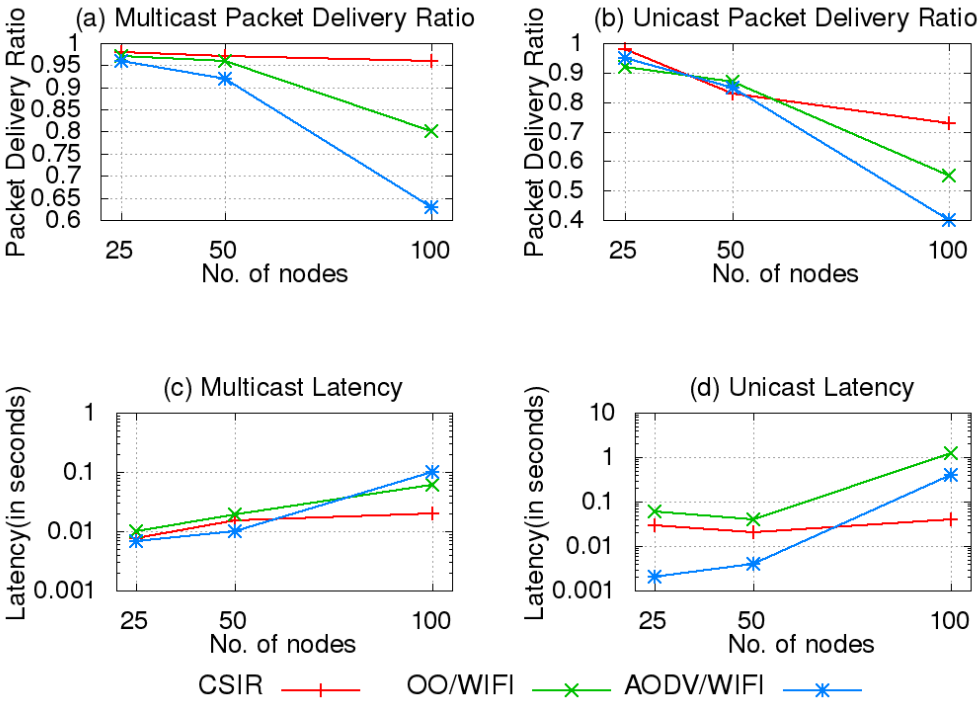

Fig. 5: Delivery Ratio and Latency Results for Load 20%, Static Topology

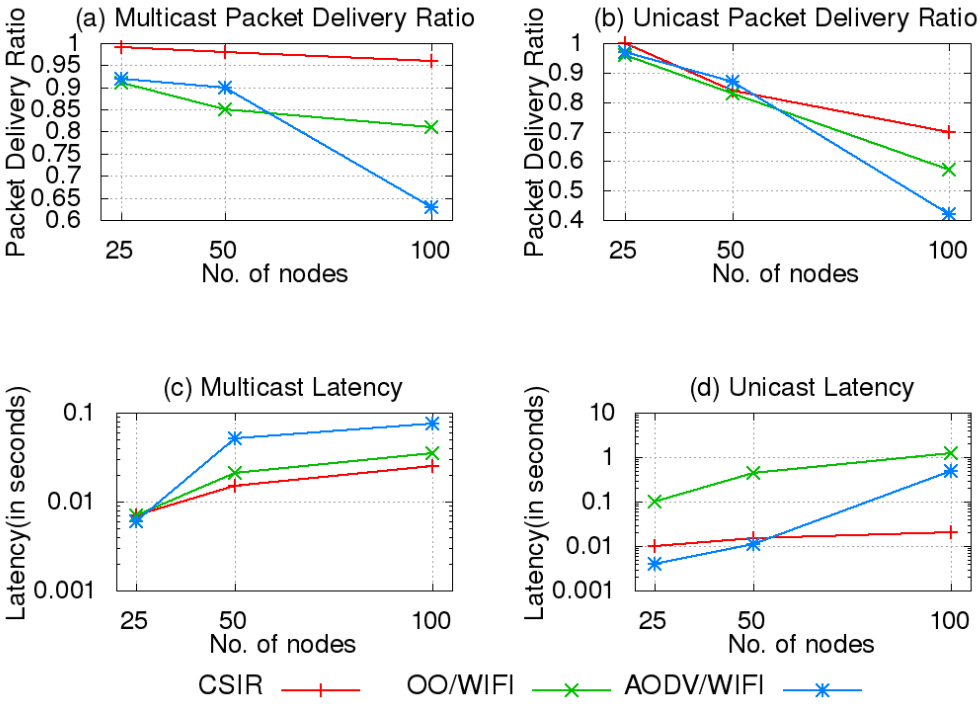

Fig. 6: Delivery Ratio and Latency Results for Load 20%, Dynamic Topology (upto 4m/s)

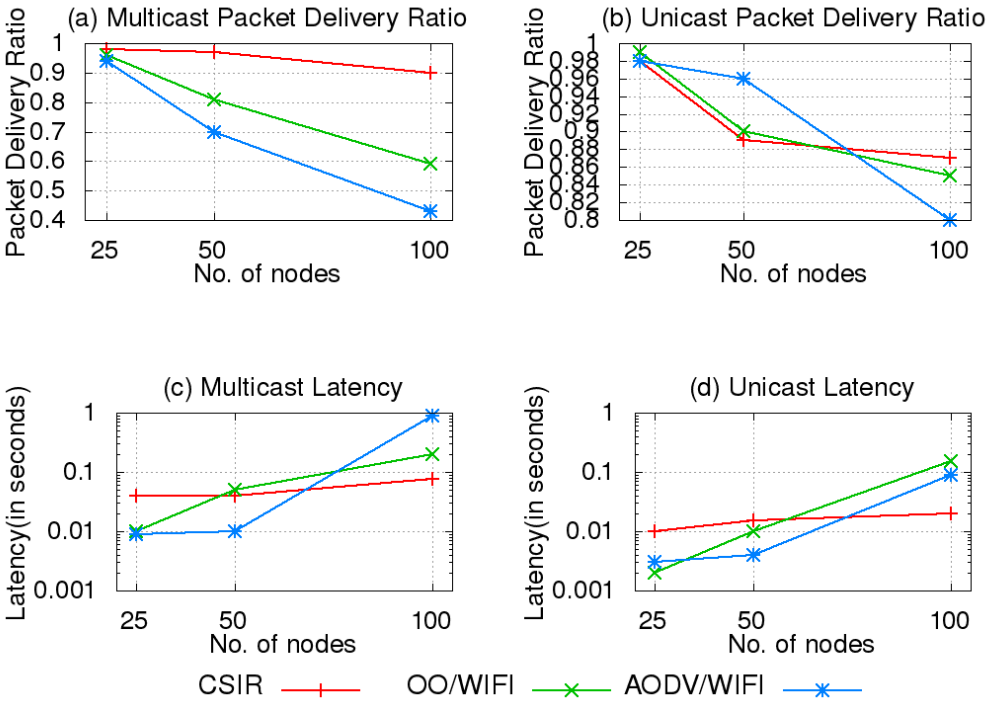

Fig. 7: Delivery Ratio and Latency Results for Load 60%, Static Topology

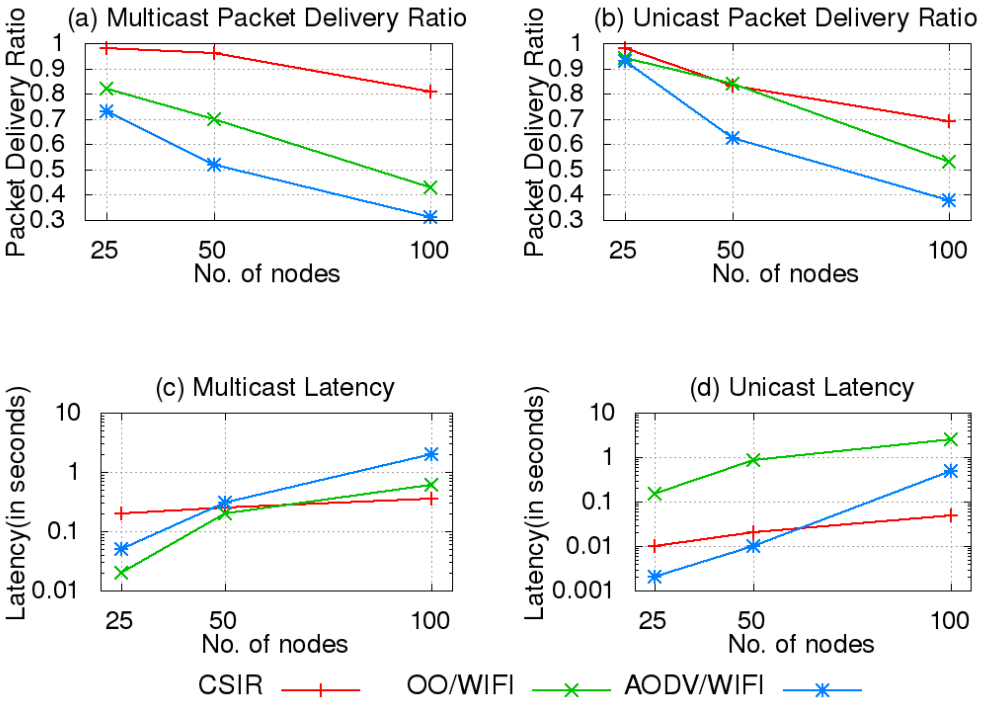

Fig. 8: Delivery Ratio and Latency Results for Load 60%, Dynamic Topology (upto 4m/s)

Nodes were distributed randomly with unicast and multicast traffic generated at rates about 20 kbps per source, using on/off busy periods. Both elastic and real-time traffics were generated in equal proportions.

Figs. 5 and 7 show the simulation results for increasing loads in static networks. Figs. 6 and 8 show similar results for nodes with mobility. Each graph shows the packet delivery ratio and latency for both unicast and multicast traffic for increasing network size. Latency is shown in log scale. The network area is expanded proportional to the network size so that the average node density remains the same in all cases. Nodes have an approximate transmission range of 300 m. The average hop-count of the flows increased from 3 to 12 with an increase in the network size from 25 to 100 nodes.

From the graphs, it is evident that OLSR/ODMRP and AODV/ODMRP do not scale well for networks containing more than 25 nodes under 802.11. There is a significant drop in packet delivery ratios for these protocols when the network size is increased. This is mainly due to the excessive signaling induced by the protocols. While the end-to-end latency for the protocols using 802.11 is in the same range as that of CSIR, an increase in the network size results in a 5 to 10 fold increase in latency, failing to comply with the ITU-T recommendation G.114 that regulates the delay requirements for supporting real-time applications, such as VOIP.

VIII. Conclusions

We introduced CSIR, a novel cross-layer protocol framework for wireless ad hoc networks that integrates deterministic scheduling based on GPS coordinates, interest-driven routing and a priority-based traffic management system that combine to provide end-to-end delay and bandwidth guarantees.

The bandwidth allocation algorithm in CSIR is controlled by the routing and the GPS coordinates of the nodes. This allows nodes to allocate bandwidth quickly and without excessive signaling. Unlike other cross-layer frameworks proposed in the past, nodes allocate bandwidth based only on the required application bandwidth and the available local transmission bandwidth without coordinating with nodes along the path.

Our simulation results demonstrate that CSIR is very scalable and robust for both unicast and multicast traffics. The results also show that the end-to-end delays attained by CSIR for unicast and multicast traffic are less than 200 ms, sufficient to support most real-time applications.

IX. Acknowledgments

This work was supported in part by the Baskin Chair of Computer Engineering at UCSC, and by the US Army Research Office under Grant SUB0700155/SC20070363.

References

[1] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Standard 802.11, Nov. 1997.

[2] L. Bao and J. J. Garcia-Luna-Aceves, "A New Approach to Channel Access Scheduling for Ad Hoc Networks," *in Proc. ACM MobiCom*, pp. 210–221, Rome, Italy, Jul. 2001.

[3] L. Bao and J. J. Garcia-Luna-Aceves, "Distributed Channel Access Scheduling for Ad Hoc Networks," *ACM Journal of Parallel and Distributed Computing*, vol. 63, no. 1, pp. 3–14, Jan. 2003.

[4] Z. Cai, M. Lu, and X. Wang, "An End-to-End Bandwidth Allocation Algorithm for Ad hoc Networks," *Telecommunication Systems*, vol. 22, no. 1, pp. 281–297, Jan. 2003.

[5] E. Carlson, C. Prehofer, C. Bettstetter, H. Karl, and A. Wolisz, "A Distributed End-to-End Reservation Protocol for IEEE 802.11-based Wireless Mesh Networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2018–2027, Nov. 2006.

[6] L. Chen and W. B. Heinzelman, "A Survey of Routing Protocols that Support QoS in Mobile Ad hoc Networks," *IEEE Network*, vol. 21, no. 6, pp. 30–38, Nov. 2007.

[7] J. J. Garcia-Luna-Aceves and R. Menchaca-Mendez, "PRIME: An Interest-Driven Approach to Integrated Unicast and Multicast Routing in MANETs," *IEEE/ACM Transactions on Networking*, vol. 19, no. 6, pp. 1573–1586, Dec. 2011

[8] P. Jacquet, A. Laouiti, P. Minet, and L. Viennot, "Performance Analysis of OLSR Multipoint Relay Flooding in Two Ad hoc Wireless Network Models," Project Hipercom, INRIA Rocquencourt, Tech. Rep. RR-4260, May. 2002.

[9] S.-J. Lee, M. Gerla, and C.-C. Chiang, "On-Demand Multicast Routing Protocol," *in Proc. IEEE WCNC*, pp. 1298–1302, New Orleans, LA, Sep. 1999.

[10] S. Giordano, I. Stojmenovic and L. Blazevic, "Position Based Routing Algorithms for Ad Hoc Networks: A Taxonomy," *Ad Hoc Wireless Networking*, pp. 103–136, 2003.

[11] C. R. Lin and M. Gerla, "Asynchronous Multimedia Multihop Wireless Networks," *in Proc. INFOCOM*, pp. 118–125, Kobe, Japan, Apr. 1997.

[12] T. Melodia, M. C. Vuran, and D. Pompili, "The State of the Art in Cross-Layer Design for Wireless Sensor Networks," *in Proc. EURO-NGI*, pp. 78–92, Vigoni, Italy, Jul. 2005.

[13] R. Menchaca-Mendez, J.J. Garcia-Luna-Aceves, "STORM: A Framework for Integrated Routing, Scheduling, and Traffic Management in Ad Hoc Networks," *IEEE Transactions on Mobile Computing*, vol. 11, no. 8, pp. 1345–1357, Aug. 2012.

[14] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 4, pp. 234–244, Oct. 1994.

[15] C. E. Perkins and E. M. Royer, "Ad-hoc On-demand Distance Vector Routing," *in Proc. IEEE WMCSA*, pp. 90–100, New Orleans, LA, Feb. 1999.

[16] E. Setton, T. Yoo, X. Zhu, A. Goldsmith, and B. Girod, "Crosslayer Design of Ad hoc Networks for Real-time Video Streaming," *IEEE Wireless Communications*, vol. 12, no. 4, pp. 59–65, Aug. 2005.

[17] Y. Wu, P. A. Chou, Q. Zhang, K. Jain, W. Zhu, and S.-Y. Kung, "Network Planning in Wireless Ad hoc Networks: A Cross-layer Approach," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 1, pp. 136–150, Jan. 2005.

[18] J. Yuan, Z. Li, W. Yu, and B. Li. "A Cross-layer Optimization Framework for Multihop Multicast in Wireless Mesh Networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2092–2103, Nov. 2006.

[19] X. Zhang and M. Haenggi, "A Location-based MAC Scheme for Random Wireless Network," *in Proc. IEEE ICC*, pp. 1–5, Kyoto, Japan, Jun. 2011.

[20] Ashok N Masilamani and J. J. Garcia-Luna-Aceves, "Scheduled channel access using geographical classification," *in 2013 International Conference on Computing, Networking and Communications (ICNC)*, pp. 790-796, San Diego, CA, Jan. 2013

[21] Qualnet 4.5, Scalable Network Technologies, http://www.scalablenetworks.com