

# J-CAR: An Efficient Joint Channel Assignment and Routing Protocol for IEEE 802.11-Based Multi-Channel Multi-Interface Mobile Ad Hoc Networks

Hon Sun Chiu, Kwan L. Yeung, and King-Shan Lui

**Abstract**—The capacity of an IEEE 802.11-based multi-hop wireless network is limited. By effectively utilizing multiple non-overlapping channels and multiple interfaces, collision and co-channel interference can be reduced. This allows more concurrent transmissions and thus enhances the network capacity. In this paper, we introduce an efficient distributed joint channel assignment and routing protocol, called J-CAR<sup>1</sup>. Unlike existing schemes, J-CAR allows a data interface to dynamically change its working mode between send and receive on a call-by-call basis, which enhances the utilization of both interface and channel. In J-CAR, channels are negotiated and assigned to active links in conjunction with the on-demand routing process. At each hop, J-CAR conducts a local optimization by selecting the least interfered channel according to the channel interference index. The channel interference index is designed by taking both the protocol and physical interference models into consideration. To find the least interfered path for network load balancing on a global scale, J-CAR employs a length-constrained widest-path routing. The “width” of a path is determined by the interference level of its bottleneck link. With an adjustable threshold on the path length (with respect to the shortest-path), the excessively long path can also be avoided. We show that with a comparable complexity as the existing schemes, J-CAR provides much higher system goodputs and shorter end-to-end packet delays.

**Index Terms**—IEEE 802.11, multiple channels, multiple interfaces, joint channel assignment and routing.

## I. INTRODUCTION

**I**N traditional *single channel single interface* mobile ad hoc networks (MANETs), network capacity decreases with the increasing number of mobile nodes [1], due to collision and interference in the single shared medium. The problem deteriorates in multi-hop networks, e.g. only 1/7 of the channel bandwidth can be used in a chain setup [1]. By effectively utilizing multiple non-overlapping channels (3 in IEEE 802.11b/g and 12 in IEEE 802.11a), the network capacity can be significantly enhanced by allowing more concurrent transmissions.

Some protocols are designed based on a single wireless interface per node [2]–[4]. They aim at maximizing the

network capacity with minimum hardware cost. Limited by a single interface, nodes are required to switch between channels frequently. To amortize the channel switching delay, a node needs to stay in a channel for certain amount of time (e.g. 10ms in [2] and 100ms in [3]) in order to send or receive multiple packets. The accumulated stop-and-forward delay for a multi-hop path is high. Also, nodes must be properly synchronized to “meet” each other. Otherwise, they may not even know the existence of each other.

With the reduction in hardware cost [5], protocols using multiple wireless interfaces are proposed [6]–[13]. They aim at solving the joint channel assignment and routing problem. The major challenge lies at the inter-dependency of channel assignment and routing. Channel assignment determines the network connectivity/topology, which affects the routing decision. Routing determines the amount of traffic on each link, which in turn affects the channel assignment decision.

For a given set of node locations and traffic demands among the nodes, the joint channel assignment and routing problem can be solved by centralized algorithms [6]–[9]. Due to the deterministic nature of node locations and traffic demands, the centralized joint channel assignment and routing algorithms are usually performed during network planning for a static wireless mesh network. A popular heuristic technique is to decouple routing and channel assignment into two separate phases [6]. The algorithms first find the routes for the traffic demands (e.g. a load balanced shortest-path tree for a wireless Internet access [6]) and obtain the load estimation on each wireless link. Then channels are assigned to the links based on the load estimation. Another approach is to formulate the joint channel assignment and routing problem mathematically [7] using Integer Linear Programming (ILP), and an optimal solution can be obtained by solving the ILP. But this approach usually incurs a higher computational complexity. In some cases, no explicit traffic profile is assumed [8], [9]. Network planning is done by assigning channels only based on the given node locations (e.g. by a centralized server [8] or by graph coloring technique [9]) with the aim of minimizing the mutual interference among all the links in the network. A distributed algorithm for the same objective, i.e. minimizing interference by channel assignment, is proposed in [10]. But their focus is at enhancing the hop based communications [8]–[10], no path-based routing is considered. For all centralized

Manuscript received February 5, 2008; revised July 4, 2008 and October 16, 2008; accepted November 25, 2008. The associate editor coordinating the review of this paper and approving it for publication was W. Lou.

The authors are with the Department of Electrical and Electronic Engineering, The University of Hong Kong (e-mail: {hschiu, kyeung, kslui}@eee.hku.hk).

Digital Object Identifier 10.1109/TWC.2009.080174

<sup>1</sup>Previous version of J-CAR has been published in IEEE Globecom 2006 [18]

algorithms [6]–[9], the channel assignments and routes found are installed at each node. The installed states will usually last for a long period of time until the traffic pattern changes.

In a dynamic mobile ad hoc network (MANET), the network topology is usually not known in advance (due to nodal mobility) and the traffic demands are even harder to predict. Centralized algorithms for joint channel assignment and routing cannot be applied. Many distributed algorithms [11]–[13] are proposed to optimize the network performance in real-time and on a call-by-call basis. (Note that a distributed algorithm can co-exist with a centralized algorithm, where the centralized algorithm is for long-term network planning based on the predicted traffic profile, and the distributed algorithm for real-time network optimization.) Among various distributed algorithms, some MAC layer-based protocols [11] are designed for packet-level link-based channel assignment using channel reservation in the common control channel. But the control channel can be easily saturated [3], and throttles the utilization of the data channels. To further minimize the coordination and signaling overhead among nodes, a usual practice is to assign each node with a fixed receiving channel [12], i.e. receiving channel pre-assignment. By making the pre-assigned receiving channels known by their neighbors in advance, communication sessions can be easily set up.

However, receiving channel pre-assignment tends to lower the network efficiency. Consider the case that a node only acts as a receiver. It can only receive through its fixed receiving interface/channel [12], other interfaces/channels of the node will be left unnecessarily idle. As nodes move around, even if there is no active connections, extra power is consumed for keeping track the number of nodes in each channel [12]. The channel diversity may also suffer as consecutive nodes along an active path may use the same receiving channel. Among the distributed algorithms, [11], [12] focus on improving channel assignment performance and only adopt simple shortest-path-oriented protocol (AODV or DSR) for routing. The impact of channel diversity and interference level along a path is ignored. Unlike [11], [12], MCR [13] adopts a non-shortest-path routing protocol, where the path cost is determined by summing up all the link (delay) costs along the path. However, the importance of available path bandwidth has been overlooked.

In this paper, we focus on distributed algorithms where channel assignment and routing are carried out jointly when a new call arrives. An efficient distributed algorithm called J-CAR is proposed for IEEE 802.11-based multi-hop mobile ad hoc networks. Unlike existing schemes, J-CAR allows a data interface to switch its working mode between send and receive on a call-by-call basis. This extra flexibility boosts the utilization of both interface and channel. J-CAR does not use receiving channel pre-assignment. At each hop along a candidate route for a new call, J-CAR conducts a local search for the channel with the smallest channel interference index. The channel interference index is designed to capture the impact from both protocol and physical interference models. To find the least interfered path for load balancing in the global network, J-CAR employs a length-constrained widest-path routing, where the “width” of a path is determined by the interference level of its bottleneck link. With an adjustable

threshold on the path length (with respect to the shortest-path), the excessively long path can also be avoided. We show that with a comparable complexity as the existing schemes, J-CAR provides much higher system goodputs and shorter end-to-end packet delays.

The rest of the paper is organized as follows. The general operation of J-CAR and the notion of interface assignment are introduced in the next section. The detailed design of J-CAR is given in Sections III and IV, where Section III focuses on channel assignment and Section IV devotes to routing. Some implementation considerations of J-CAR are given in Section V. The performance of J-CAR is studied in Section VI and we conclude the paper in Section VII.

## II. J-CAR AND INTERFACE ASSIGNMENT

### A. J-CAR Overview

The general operation of J-CAR is similar to the AODV routing protocol [14] and can be briefly described as follows. First, periodic broadcast HELLO packets are exchanged (on the control channel) among neighboring nodes to identify each other. When a call arrives, the source node checks its routing table for the route to the destination. If no route can be found, a route request (RREQ packet) is initiated, carrying its proposed sending and receiving channels to be used for this route. Each intermediate node checks the availability of the proposed channels, appends its own selection to the RREQ packet and broadcasts it to the next hop. When the destination node receives the first RREQ, it starts a timer for capturing other RREQs going through different paths. When the timer expires, the destination node identifies the widest path and sends a route reply (RREP) packet to the source via the reverse path. This not only confirms the selected path, but also the channels to be used at each hop along this path.

### B. Interface Assignment

Before the detailed description on J-CAR is given, we first introduce the notion of interface assignment. Among multiple wireless network interfaces a node has, one is assigned to operate (only) at the pre-defined common control channel, called *control interface*. It is used (mainly) for carrying control and broadcast packets. The remaining interfaces are *data interfaces*, for carrying data packets on different data channels. Since RTS-CTS handshaking in the control channel [11] has been shown [3] to be inefficient, J-CAR performs the four-way 802.11 handshaking of RTS-CTS-DATA-ACK in the selected data channels. Accordingly, the traffic load on the control channel is expected to be light (consisting of routing and HELLO packets [14]). The spare capacity can be used to carry data packets for increased channel utilization.

On the other hand, the control channel should be protected from congestion. Therefore, a *data sending probability*  $p$  is associated with the control channel, governing the probability that it will also be used for carrying data. Specifically, when  $p=0$ , the control channel is not used to carry data packets. When  $p=1$ , the control channel and data channels have equal probability of being selected for data transmission. If a node only has a single interface, then data and control packets must share the same control interface/channel.

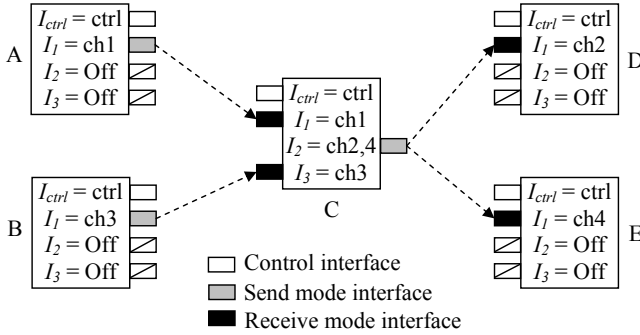


Fig. 1. The use of multiple interfaces for two active connections

When a call arrives and a route is to be set up, a data interface becomes active (where a data channel is assigned to it) and switches to either *send mode* or *receive mode*, depending on the direction of the traffic flow. To save energy, an inactive data interface can be turned off, or put in *sleep mode*. Note that the control interface is not assigned with any working mode, as it always operates in the pre-determined common control channel for both receiving and sending. Send mode and receive mode differ as follows. In send mode, an interface can switch to different data channels to send data packets to different neighbors (so effectively it works on multiple channels), but it is prohibited from receiving data packets. In receive mode, an interface can only work on a *single* data channel, mainly for data receiving, and limited data sending is also allowed (e.g. there is only one data interface in an intermediate node for data forwarding, or when the send mode interface is overloaded; to be detailed later). When a call/route finishes/expires, the data interface returns to sleep mode, and may switch to an appropriate working mode when a new route request arrives. It should be noted that since the 802.11 handshaking is performed in the data channel, a send/receive mode interface can also receive/send CTS and ACK packets.

J-CAR restricts a receive mode interface to operate on a single channel in order to minimize the *sender-receiver synchronization overhead*. Consider two nodes send data to the same receive interface via different channels. A node does not know when it can send data again while another node is sending. To properly co-ordinate the sending/receiving among the nodes, considerable amount of synchronization overhead is involved. (This also explains why a send mode interface is not used to receive data.) By restricting receiving and sending on the same channel, the status of the receive mode interface is always known by its upstream (by monitoring/overhearing the RTS/CTS packets on that channel).

### C. An Example

The potential gain of using switchable working modes can be illustrated by Fig. 1. The network consists of five nodes, each has four interfaces labeled as  $I_{ctrl}$ ,  $I_1$ ,  $I_2$ , and  $I_3$ , where  $I_{ctrl}$  is the control interface listening on the control channel (ctrl),  $I_1$ ,  $I_2$ , and  $I_3$  are data interfaces. There are two paths, A-C-D and B-C-E, intersect at node C. Consider the path A-C-D. With J-CAR, data channel 1 (ch1) is selected by link A-C. The two nodes communicate by having  $I_1$  at node A in

send mode and  $I_1$  at node C in receive mode. Similarly, ch2 is selected by link C-D, occupying  $I_2$  at C (send mode) and  $I_1$  at D (receive mode). We can see that the two links use different channels. This eliminates mutual interference.

Consider the second path B-C-E. With J-CAR, ch3 is selected by link B-C, occupying  $I_1$  at B (send mode) and  $I_3$  at C (receive mode). In order not to interfere with the data receiving at  $I_1$  and  $I_3$ , and assume  $I_{ctrl}$  is not preferred, node C selects  $I_2$  and ch4 for link C-E. Since  $I_2$  at node C is in send mode, it can switch between ch2 and ch4 for data sending. Finally, node E switches its  $I_1$  to ch4 (receive mode). It should be noted that node C receives data using two interfaces, the throughput performance is better than the receiving channel pre-assignment schemes [12], [13], due to the reduced packet collision probability.

### D. Interface Load Estimation

J-CAR selects the least loaded interface for a newly selected channel. Let the number of bytes passing through interface  $i$  over a window of  $t$  seconds be  $l_i$ . (Without loss of generality,  $t=0.5$  is used in order to be consistent with the AODV timer for updating routing table entries.) Then  $L_i^w$  the estimated load at interface  $i$  at the  $w$ -th window of time is given by the exponentially weighted moving averaging function below

$$L_i^w = \alpha L_i^{w-1} + (1 - \alpha)l_i \quad (1)$$

where  $\alpha$  is the weighting factor. (In Section VI,  $\alpha=0.7$  is used.)

## III. CHANNEL ASSIGNMENT AND INTERFERENCE INDEX

### A. Channel Interference Index

At each hop along a candidate path for a new call, J-CAR selects the best channel to use based on the *channel interference index*, which is a heuristic measure of the "goodness" of a channel as perceived by the node under consideration. We define the interference index of channel  $i$  as perceived by the node under consideration as:

$$Index_i = \sum_{j=1}^k \frac{usage(i, j)}{j^\gamma} \quad (2)$$

where  $k$  is the interference range,  $\gamma$  is the path loss exponent [15], and  $usage(i, j)$  is the *channel usage table*.  $usage(i, j)$  is maintained at each node for keeping track the amount of data sent by nodes using channel  $i$  at the  $j$ -th hop, where  $i \in [0, num\_channel-1]$  and  $j \in [0, k+1]$ . When  $j=0$ ,  $usage(i, 0)$  is the channel usage of the node under consideration. Each node estimates the amount of data it sent (send load) using (1), and exchange its send load with neighbors (within  $k+1$  hops) via the periodic broadcast HELLO packets. Note that due to mobility and possible packet corruption/collision, HELLO packets may be lost or carry obsolete information. But this would have little impact to the overall system performance because the channel selection is based on  $Index_i$ , where the contribution by lost/obsolete HELLO packets will be minor.

At each hop along a candidate path for a new call, J-CAR carries out its local optimization by always selecting the channel with the smallest channel interference index. If  $Index_i=0$ ,

then channel  $i$  is said to be idle as observed by the node under consideration. In case of a tie, J-CAR randomly selects the winner. The design rationale of the channel interference index is explained below.

Due to carrier sensing and RTS-CTS handshaking, a node cannot transmit when there is another node transmitting within its interference range. Based on this protocol interference model, a multi-hop MANET with an interference range of  $k$  hops requires that at most one node can send within the  $k$ -hop neighborhood of the target receiver. Hence, the number of interference sources (and their loading) directly affects the chance of successful transmission in the channel. This explains why  $Index_i$  is directly proportional to  $usage(i, j)$ .

Next consider the physical interference model where whether a packet is received correctly is determined by the signal-to-noise ratio (SNR) as perceived by the intended receiver. The received interference strength is largely determined by the power attenuation with distance to the interfering sources [16]. The loss characteristic can be represented by a path loss exponent  $\gamma$  [15], its value varies under different propagation models. To capture the effect due to the actual signal propagation, channel interference index in (2) is designed to be inversely proportional to  $j$ , the *hop distance* from the node under consideration, with a path loss exponent  $\gamma$ . We do not use the actual distance for two reasons. First, measuring the actual distance involves non-negligible amount of overheads and yet the result may still be inaccurate. Second, in the selection process, it is the *relative* performance of different candidate channels matter. Also notice that we have assumed that the value of the path loss exponent  $\gamma$  is known a priori in (2). In practice, its value can be obtained by efficient estimation algorithms [15].

**B. Comparison with Other Channel "Goodness" Measures**

In the literature, some schemes [2]–[4], [7], [11] only use a "binary index" to indicate whether a channel is being used by others within a node's interference range. A channel can only be selected if it is idle. If all channels are occupied, the call (packet) is rejected (delayed for later retry). Although this binary index is suitable for the packet-by-packet based channel selection (e.g. channel reservation in RTS-CTS [3], [11]), it is too conservative in the sense that less channel resources will be available for accepting new calls (e.g. for the path setup in [4]). Note that at the MAC layer, IEEE 802.11 protocol can properly resolve contention by its RTS-CTS mechanism. That means two adjacent paths having some links assigned with the same channel can still function properly.

For schemes in [6], [9], [10], [12], [13], they consider the protocol interference model when designing the index. They rank the channels by the *amount of carried load* [6] or by the *number of interfering nodes* [9], [12], [13]. Since partially overlapping channels are allowed in [10], the distance between corresponding channels in the frequency spectrum is also considered. Then the channel with the least load/interference (as measured by the number of interfering nodes) is selected. This may be a good index for protocol interference model, because the number of interfering nodes (loading) directly affects the chance of successful transmission in a channel. However, the

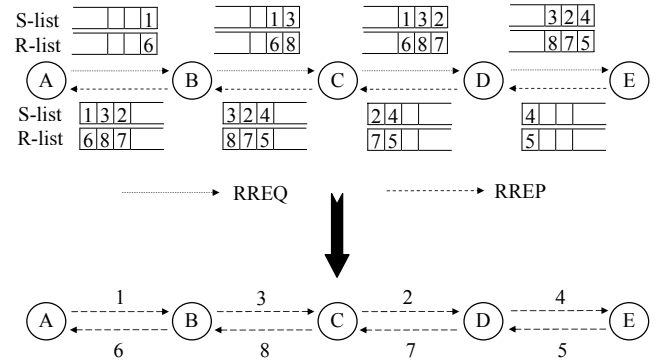


Fig. 2. A propose-and-approve process for bi-directional path setup ( $k=2$ )

impact of distance and thus the interference strength (as in the physical interference model) is not considered. Since whether a packet is received correctly is indeed based on the SNR as perceived by the receiver, the distance of the interference nodes plays an important role.

A more realistic ranking strategy is used in [8], in which the nodes sense the interference levels of the channels periodically. Then the channels are *ranked according to their interference levels*. However, channel decision made by the centralized server is based on the *average* ranking of the nodes, which may not reflect the location dependent interference level. Also, an interface becomes unresponsive during the sensing period, because it is in the "RFMon" mode [8]. This gives non-negligible amount of overhead (where the node cannot communicate with others).

Unlike [8], our proposed channel interference index in (2) does not actively sense the channel, but relies on the periodic HELLO message for channel status estimation. This allows J-CAR to fully utilize the interfaces for transmitting data packets (with the tradeoff on accuracy in estimating the interference level). In addition, our interference index is designed by taking both the protocol and physical interference models into account. It can make a better channel selection than those relying on a single interference model at a time. Besides, our interference index can adapt to different propagation conditions by using the appropriate path loss exponent.

**IV. WIDEST PATH ROUTING BASED ON PROPOSE-AND-APPROVE**

**A. Bi-directional Path Setup**

In J-CAR, the per-hop channel selection, as discussed in Section III, is carried out jointly with the on-demand route setup process. Since a send mode interface is not allowed to receive data packets, a link is unidirectional. While it does no harm to a UDP connection, a TCP receiver has to initiate another route request to set up the reverse path. This extra flooding of RREQ (route request) packets can degrade the system performance and increase the path setup delay. Thus, J-CAR adopts a propose-and-approve based bi-directional path setup mechanism.

The operation of a propose-and-approve based bi-directional path setup is illustrated by the example shown in Fig. 2, where node A is setting up a path to node E. Node A initiates a route setup to node E by broadcasting a RREQ packet, which

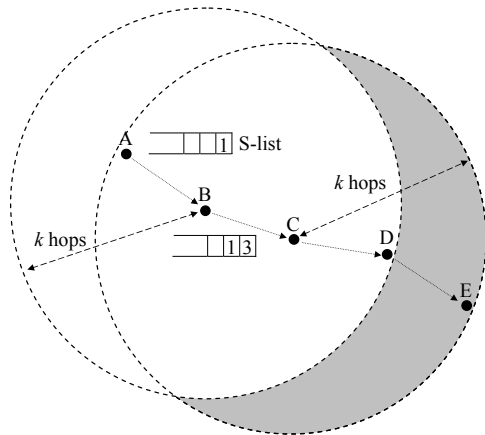


Fig. 3. Region affecting the channel selection decision

carries a *send channel list* (S-list) and a *receive channel list* (R-list). Let  $k$  be the interference range in hops. Each channel list contains  $k+1$  entries. In order to minimize the intra-path co-channel interference, channel lists are used to inform the subsequent nodes (within interference range) not to select the proposed channels (which may appear as idle in their channel usage tables). In Fig. 2, node A proposes a send channel and stores its identity (ch1) in the S-list. To set up a bi-directional path, node A also proposes a receive channel for the reverse flow and its identity (ch6) is stored in the R-list.

Since RREQ is delivered by broadcasting, when node B receives the RREQ, it first checks whether it is a duplicated RREQ (by examining its broadcast ID). If yes, the packet is dropped. If it is a new request, then node B checks its channel usage table to see whether the proposed channels are acceptable. Recall that a receive mode interface is not switchable, the proposed *receive* channel should not be rejected, unless it wants to use the control interface/channel for the reverse flow, e.g. if there is no available send mode interface for ch6. On the other hand, the proposed *send* channel is acceptable if it also has the smallest interference index as seen by node B (as interference affects receiving, not sending) and there is an available interface at node B to receive on the proposed send channel. Otherwise, a *channel conflict* occurs, and we address it in the next sub-section. Suppose node B finds that both ch1 and ch6 are acceptable. Then it proposes ch3 (send channel) and ch8 (receive channel) for using at the next hop. The proposed channels are *pushed* into the S-list and R-list respectively. The updated RREQ packet is then broadcasted to the next hop. The same operation will be carried out at nodes C and D.

It should be noted that upon receiving a RREQ, a node only needs to check the first entry in each proposed channel list. The reason is illustrated by Fig. 3, where only the S-list is considered. Assume node A proposes ch1, which is also the least interfered (or free) as seen by node B. Then node B accepts A's proposal and proposes ch3 to node C. There is no need for node C to check ch1, the second entry in the S-list. This is because nodes using ch1 as seen by node C, are either already considered by node B, or residing outside the interference range of node B (i.e. in the shaded area in Fig. 3).

Referring back to Fig. 2, when node E receives the RREQ, it checks the availability of the channels proposed by node D and sends a RREP (route reply) to node A through the reverse path. The *confirmed* S-list and R-list are piggybacked onto the RREP. When node D receives the RREP, it records the *confirmed* send channel (ch4) and receive channel (ch5), and creates a routing entry for the route to node E. Then node D confirms ch2 and ch7, and forwards the updated RREP to node C. Likewise, the RREP is delivered back to node A and the multi-channel bi-directional path from A to E is determined.

There are three points to be noticed. First, all the routing packets are transmitted in the control channel. Since RREQ packets are delivered by broadcasting, they may reach the destination via multiple paths. If a path is not confirmed (by receiving a RREP), the proposed channel lists cached at a node will be purged when the timer expires. (In our simulations, a 6-second timer is adopted as that in AODV [14] for an unconfirmed path.) Second, the above bi-directional path setup procedure is robust with uni-directional UDP connections. As in AODV [14], the routing entries (including the assigned channels and interfaces) will be purged if the connection is deemed to be inactive. (In our simulations, a 10-second timer is used as that in AODV for a confirmed path.) Third, the number of routing packets transmitted by J-CAR (including RREQ, RREP and HELLO) is the same as AODV. As the proposed channel lists (within  $k$  hops) are piggybacked onto the routing packets, their sizes will be slightly larger than that in AODV.

### B. Handling Channel Conflict

During the propose-and-approve route setup process, there are two scenarios that a proposed channel (in RREQ) may be rejected. We say a *channel conflict* occurs. J-CAR resolves the channel conflict by allowing the downstream node of the rejected channel to nominate a more suitable channel on behalf of the upstream node<sup>2</sup>. Since a receive mode interface is not switchable, channel conflict occurs in the proposed receive channel (R-list) can only be resolved by choosing the control channel (for the reverse path).

The first channel conflict scenario is due to the lack of available interfaces, and is illustrated by the example in Fig. 4. To improve the readability, only the S-list is shown. Assume  $k=2$  and 3 interfaces per node. In Fig. 4, node F is setting up a route to H, which crosses an existing route A-B-C-D-E. With J-CAR, node F proposes ch4 and broadcasts the RREQ to the network. Node G accepts the proposed ch4 and then proposes ch2 to the next hop. When node B receives the RREQ, it detects a channel conflict as all its three interfaces are occupied. Specifically, the first/control interface is occupied by control channel ch0 (not shown), the second/receive mode interface by ch5, and the last/send mode interface by ch1 and ch7. Since a send mode interface is not allowed for receiving, node B can only receive in ch0 or ch5. To resolve this conflict, node B selects ch5 on behalf of node G and updates the

<sup>2</sup>Alternatively, a node can just inform its upstream node that a channel conflict occurs and then waits for another proposed channel from its upstream. This incurs extra delay in route setup. Note that subsequently proposed channels may also be rejected.

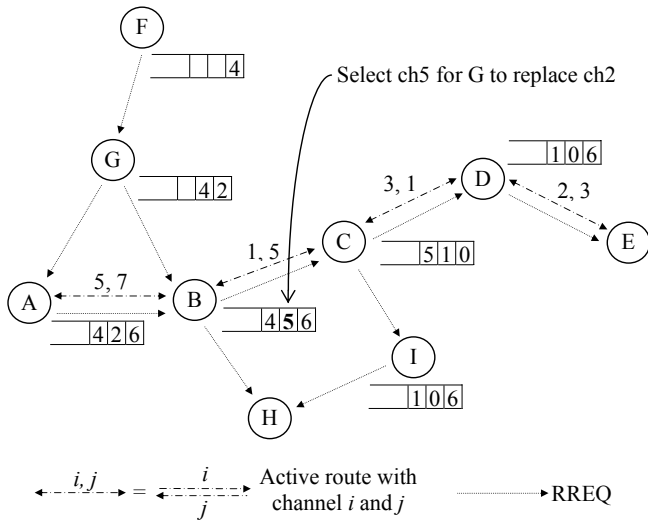


Fig. 4. Channel conflict happens at node B

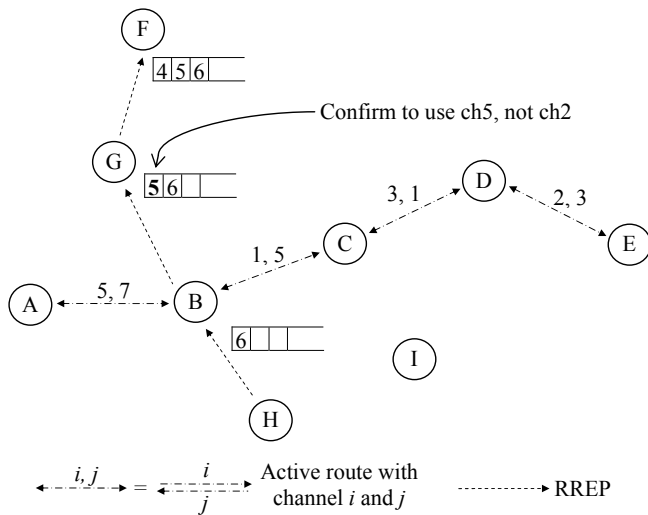


Fig. 5. Route reply confirms the channel selection

corresponding entry in the S-list. It further selects ch6 for its next hop and broadcasts the updated RREQ. At a later time, node G receives the RREP (Fig. 5). It notices that the confirmed channel is ch5, instead of its proposed ch2. Since a send mode interface is allowed to switch between channels, ch5 is accepted.

On the other hand, if ch2 is proposed for data sending by a receive mode interface, it cannot be changed (because a receive mode interface is not switchable). In this case, node G must set the *force\_bit* in the RREQ to 1, indicating that the proposed channel must be honored. If node B cannot support ch2, it falls back to the control channel (as in resolving the R-list channel conflict).

Another scenario of channel conflict occurs if the proposed channel is not the best choice as seen by the receiving node (even if it has an available interface). This happens when the receiving node finds another channel having a smaller interference index than the proposed one. The probability of this kind of channel conflict is generally low. As shown in Fig. 3, when node B proposes a channel to node C, only the

nodes within the shaded region can affect its decision. Here node C selects a more suitable channel based on the  $(k+1)$ -hop information in its channel usage table, which completely covers the interference range of node B. Therefore, the channel counter-proposed by node C is appropriate for both nodes.

In the two scenarios of channel conflict above, the conflict is resolved by the downstream node. One may think designating the channel selection decision to the downstream node may be more efficient, as it can bypass the propose-and-approve process (in Fig. 2). This is incorrect. Refer back to Fig. 3 for the channel selection of link B-C. If C decides a channel using  $k$ -hop information, some nodes within B's interference range are not considered. On the other hand, if  $(k+1)$ -hop information is used, the resulting channel selection will be too conservative because the majority of nodes in the  $(k+1)$ -tier will not be interfered by B and not affect C's receiving. With similar reason, node B does not use  $(k+1)$ -hop information in its first proposal. Therefore, in our propose-and-approve route setup process, a better channel selection should be made based on the channel usage of nodes residing in the union of B's and C's  $k$ -hop neighborhood.

### C. Length-Constrained Widest-Path Routing

Bandwidth is probably the most important performance indicator in routing, but determining the residual bandwidth of a wireless link is non-trivial. Instead of simply finding the shortest/least-delay path as in most distributed joint channel assignment and routing algorithms [11], [12], J-CAR uses the channel interference index as a heuristic measure of link bandwidth in routing. The path bandwidth is determined by the (bottleneck) link with the largest index value. For a bi-directional path, the unidirectional path from the source (the node initiated the connection) to the destination is referred as forward path, and the reverse unidirectional path as backward path. J-CAR can give different priorities to finding a good forward/backward path. Without loss of generality, we focus on finding the widest forward path.

During the routing process, the interference index of the bottleneck link is recorded inside the RREQ. Since RREQ is broadcasted, multiple RREQs (each corresponds to a different path) will arrive at the destination node. When the first RREQ arrives, instead of immediately replying a RREP (as in AODV), the destination node starts a *delay\_RREP* timer to capture the later RREQs. RREQs arrive after the timer expires are dropped due to long propagation delay (including the processing/queuing delay along the path). Among all received RREQs, the path with the least hop-distance<sup>3</sup> ( $h_{min}$ ) is identified. In general, a long (hop-distance) path tends to have higher probability of packet collision, and causes more interference to other paths. To avoid selecting excessively long paths, an adjustable threshold ( $T_H$ ) is used. All the paths with length less than  $h_{min} + T_H$  hops become the candidates for widest path selection. The destination node compares the "widths" of all the candidate paths, and sends the RREP along

<sup>3</sup>Due to the contention-based MAC protocol and the actual physical distance of each hop, the least hop-distance path may not have its RREQ packet arrived at the destination node first.

the “widest” one (i.e. with the smallest interference index) to confirm the channel selection.

Note that the widest path selection can also take place at each intermediate node, such that each node only relays the RREQ arrived from the widest path. Although this can further enhance the widest path performance, a very large setup delay will be incurred as every intermediate node has to wait for a certain amount of time for identifying the RREQ from the widest path.

## V. SOME IMPLEMENTATION ISSUES

In MANETs, the mobility of nodes causes variation on path quality. To keep track of such changes, the source node may periodically (every  $t_{update}$  seconds) send an *update\_pkt* to the destination via the control channel, with its *path\_interference\_index* field initialized to 0. When an intermediate node receives the *update\_pkt*, it calculates the interference index of its outgoing channel towards the source, and updates the *path\_interference\_index* field if its index is larger. The corresponding (reverse path) entry of the routing table is then updated. When the destination node receives the *update\_pkt*, it updates its routing table and bounces back the *update\_pkt* to the source, with the *path\_interference\_index* field reset to 0. Every intermediate node performs the same procedure as before, but this time is for updating the interference level of the forward path. Note that to avoid extra channel switching delay and synchronization overhead, this update mechanism is not used for channel re-selection of on-going calls.

Another issue caused by node mobility is route breaks. J-CAR re-establishes a broken route by local route recovery. When a node detects a route break, it simply initiates a route request and re-connects itself to the destination to bypass the failed links/nodes. When an intermediate node receives the RREQ and its routing table has an entry to that destination, it can directly reply with a RREP that carries the current “width” of the path. This *fast* RREP from intermediate nodes (also a feature of AODV) allows the connection to be resumed faster. Nevertheless, RREQs through other paths still arrive at the destination for widest path selection. Despite of the initial fast RREP, the connection can switch to a better path later on based on the result of widest path routing. The overhead in this path switching is minimal as it only involves one entry update in each affected routing table.

In our design, we do not implement the “fast RREP” in the receiver, i.e. immediately reply the first RREQ, as it will lock the channels and interfaces for a longer period of time along the *whole* path if it is not the optimal choice. This is because, due to the soft-state nature of the cached routing entry, the lifetime of a (RREP) confirmed path is typically longer than that of an unconfirmed one.

## VI. PERFORMANCE EVALUATIONS

In this section, we study the performance of J-CAR by ns-2 simulator. The network topology is generated by randomly placing a given number of nodes onto a 2-dimensional open area (without any building or mountain). As the received signal is mainly contributed by the direct line-of-sight transmission

TABLE I  
IEEE 802.11A PARAMETERS USED

Parameter	Value
SlotTime	9 $\mu$ s
SIFS	16 $\mu$ s
PreambleLength	120 bits
PLCPHeaderLength	24 bits
PLCPDataRate	6Mbps
basicRate	6Mbps
dataRate	6Mbps
CWmin	15
CWmax	1023
Freq	5GHz

and the reflection from the ground, a simple two-ray-ground propagation model [16] is used, and the path loss exponent is set to  $\gamma=4$  in determining the channel interference index in (2). At the MAC layer, the IEEE 802.11a with RTS/CTS collision avoidance is implemented and the corresponding parameters [17] are summarized in Table I. The channel switching latency is set to 100 $\mu$ s [2].

We examine the performance of carrying both UDP and TCP flows. As similar conclusions are drawn from both scenarios, we only present UDP results in this section due to the limited space. All UDP flows carry CBR traffic of 1000 packets/sec with packet size of 512 bytes, i.e. 4Mbps. Different numbers of flows (with non-overlapped source/destination nodes) are simulated with each subsequent flow starts at an interval of 10 seconds. Both single-hop and multi-hop wireless networks are simulated. A single-hop network consists of 16 nodes using 4 non-overlapping channels and nodes are randomly placed within an area of 200x200m<sup>2</sup>. A multi-hop network consists of 50 nodes randomly placed within a 750x750m<sup>2</sup> area, and using 12 non-overlapping channels. Simulation data is collected for 50 seconds when the network is stable, i.e. 10 seconds after all flows started. Each point of simulation results in the figures below is an average of 20 independent runs. We use the aggregated goodput and average packet end-to-end delay as performance metrics. Goodput excludes packet headers and signaling overhead, which is useful for measuring performance as seen by applications (e.g. video streaming).

### A. Data Sending Probability $p$

In J-CAR, the control channel can be used for carrying data and that is governed by a data sending probability  $p$ , as discussed in Section II. When  $p=0$ , the control channel is not used to carry data packets. When  $p=1$ , the control channel and data channels have equal probability of being selected for data transmission. Since the value of  $p$  directly affects the network performance, we aim at determining an appropriate value for  $p$  in this section.

Without loss of generality, the following parameters are used in our proposed widest-path routing:  $k=2$  hops,  $T_H=3$  hops,  $delay\_RREP=20$ ms, and  $t_{update}=1$ s. We simulate five randomly generated flows in both single-hop and multi-hop networks with each node equipped with three wireless interface cards. The aggregated goodput and average end-to-end delay performance for different  $p$  are given in Fig. 6. We can see that when  $p$  is small (and less than 0.5), the goodput (end-

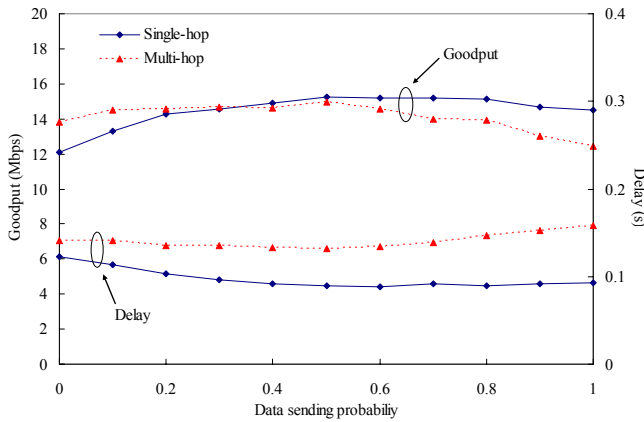


Fig. 6. Aggregated goodput and average end-to-end delay of J-CAR

to-end delay) increases (decreases) with the increasing value of  $p$ . This is because using a small value of  $p$  is equivalent to using fewer channels for data transmission. Since there are 4 channels in single-hop network and 12 channels in multi-hop networks, setting  $p=0$  cuts down the data sending capacity by 25% and 8.33%, respectively. On the other hand, we also see that a large  $p$  (over 0.5) lowers the performance. This is because it allows more data packets to compete with the control/broadcast packets. The deterioration is more serious in multi-hop networks due to its higher packet collision probability (as a larger number of control/broadcast packets are generated by more nodes). From Fig. 6, it is observed that the best performance can be obtained by setting the data sending probability to about  $p=0.5$ . Hence, in the following simulations, we fix  $p=0.5$ .

**B. Single-hop Networks**

Now we compare the performance of J-CAR with other existing schemes using single-hop networks in this sub-section, and multi-hop networks in the next sub-section. Specifically, PCAM [12] and MCR [13] are implemented for comparison because they also employ multiple interfaces and on-demand routing. Since there is no routing protocol proposed in PCAM, we simulate it together with AODV (denoted as PCAM/AODV) and our proposed length-constrained widest-path routing (denoted as PCAM/widest). For comparison, we also implement a J-CAR variant (J-CAR/AODV) that uses the original AODV. For MCR, its originally proposed multi-channel routing protocol (based on link delay) is used. Although J-CAR supports heterogeneous number of wireless interfaces per node, in comparing with PCAM and MCR, we assume a fixed three wireless interfaces per node.

In single-hop networks, the impact of routing is minimal, and thus the network performance is largely determined by the channel assignment strategy. The performance of J-CAR, PCAM/AODV and MCR is shown in Fig. 7, where the aggregated goodput and average end-to-end delay performances are plotted against the number of flows. A single channel network is also simulated for observing the impact of using multiple channels. As expected, the goodput of the three multi-channel protocols increases with the number of flows, accrediting to their multi-channel nature. Among them, PCAM/AODV and

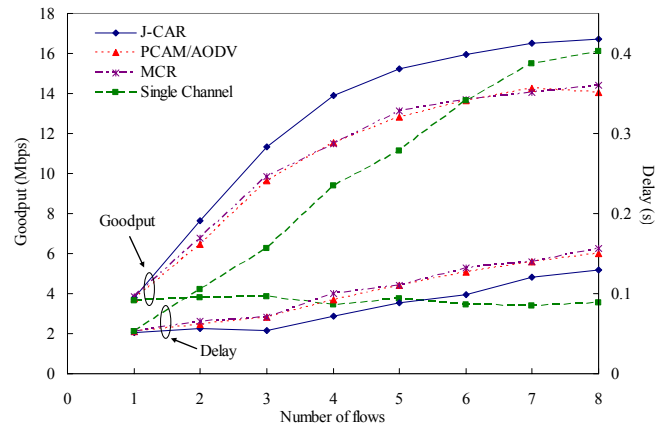


Fig. 7. Aggregated goodput and average end-to-end delay in single-hop networks

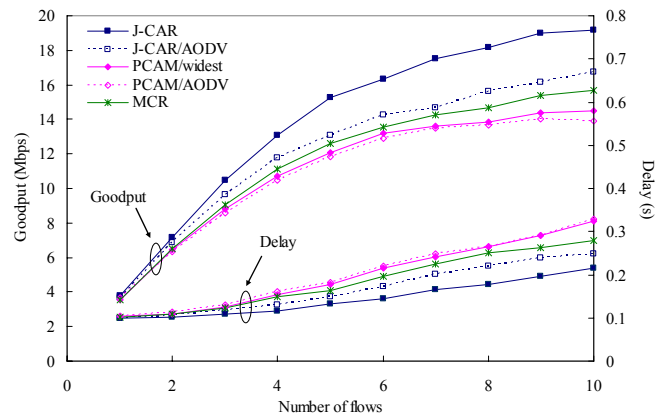


Fig. 8. Aggregated goodput and average end-to-end delay in multi-hop networks

MCR employ receiving channel pre-assignment. Since several receiving nodes may be pre-assigned with the same receiving channel, their performance gain is limited. In contrast, J-CAR distributes the flows evenly among the channels, by selecting the best channel (using channel interference index) on a call-by-call basis, and renders the best overall performance. When there are 8 active flows, J-CAR outperforms PCAM/AODV and MCR by 16.3% gain in goodput and 13.3% cut in end-to-end delay.

**C. Multi-hop Networks**

Multi-hop networks allow us to study the combined effect of routing and channel assignment. The aggregated goodput and average end-to-end delay performances are shown in Fig. 8. We can see that J-CAR always outperforms other protocols, even with the shortest-path routing (J-CAR/AODV). There are two main reasons. First, by always selecting the least interfered channel/path during the on-demand route setup, J-CAR achieves better channel diversity and load balancing than the receiving channel pre-assignment in PCAM and MCR. Second, by assigning interfaces to work in appropriate working modes, J-CAR maximizes concurrent data transmissions. Note that even PCAM is equipped with our widest-path routing, its performance gain is marginal. Since PCAM pre-assigns channels to every node in advance, it cannot fully

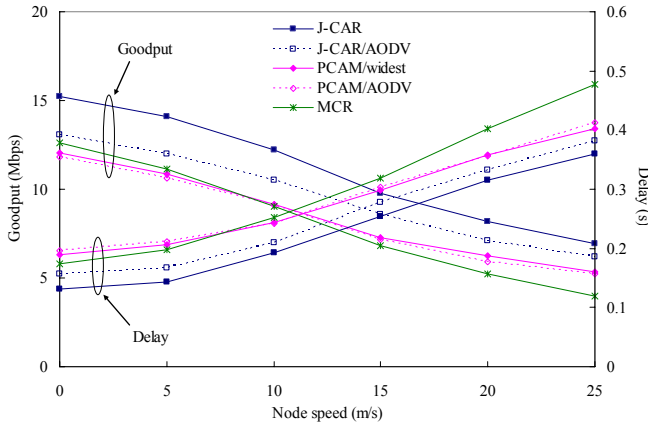


Fig. 9. Aggregated goodput and average end-to-end delay with nodal mobility

enjoy the channel diversity benefit from our propose-and-approve routing mechanism.

#### D. Impact of Mobility

We then examine the impact of nodal mobility in multi-hop networks, where node movement is determined using the random waypoint mobility generator of ns-2. The pause time is chosen between 0 to 2 seconds and the node speed is chosen between  $v-1$  to  $v+1$ , both based on a uniform distribution. In each simulation, five randomly generated flows are considered with varying speed of  $v$  from 5 to 25 m/s. From Fig. 9, we can see that for all protocols, the goodput (delay) decreases (increases) with the node speed. This is because higher speed leads to more route failures, and subsequent extra overheads in route maintenance and/or new route discovery. Nevertheless, J-CAR gives the best performance. Upon a route break, J-CAR initiates a route recovery and selects the best channels for the recovered links. In contrast, PCAM cannot change the pre-assigned channel and results in poor channel diversity. MCR performs the worst due to the large broadcasting overhead (to every channel) for route maintenance, and the large synchronization overhead involved when a node changes its fixed channel to a less utilized channel with probability 0.4, a feature designed in MCR [13] for minimizing co-channel interference.

#### E. Impact of Number of Interfaces

Since the number of interfaces determines the number of parallel transmissions, we investigate the performance of J-CAR with different number of interfaces per node in a multi-hop network. The aggregated goodput and end-to-end delay performance are shown in Figs. 10 and 11, where "NIC" stands for network interface card, and "random" denotes that the number of NICs at a node is uniformly distributed between 1 and 5. With 1 interface per node (1 NIC), the network is equivalent to a single channel network, where all nodes can only use the control channel for communications. The performance is slightly enhanced by equipping 2 interfaces per node (2 NICs). The enhancement is limited because the majority of data interfaces in the network are in receive mode, data forwarding by receive mode interface forces consecutive links of a path to use the same channel. The intra-path interference

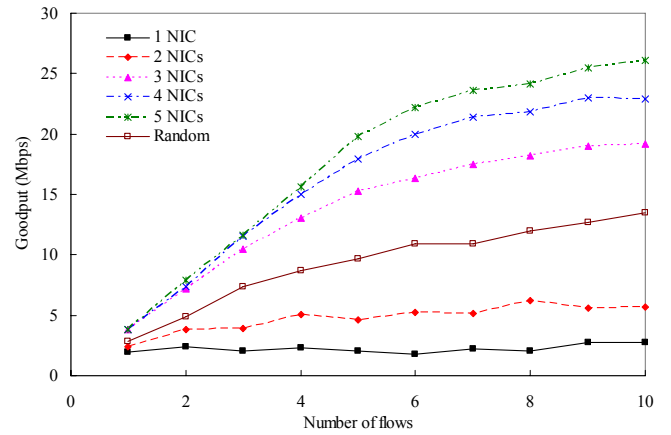


Fig. 10. Aggregated goodput of J-CAR with different numbers of NICs

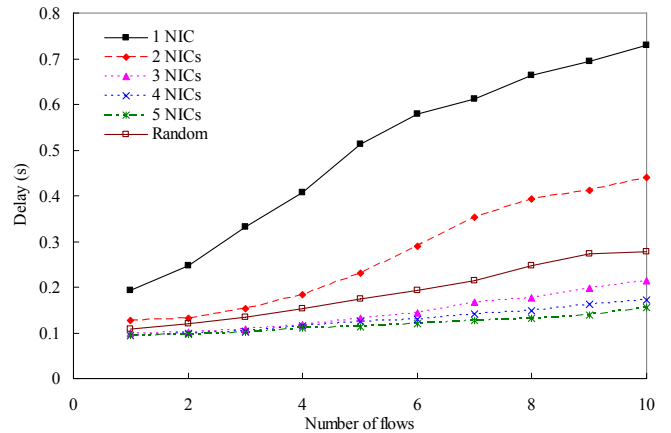


Fig. 11. Average end-to-end delay of J-CAR with different numbers of NICs

limits the system performance. With 3 or more interfaces, an intermediate node can spare a send mode interface to forward data and leads to higher channel diversity. When there are 10 flows, using 3 interfaces achieves 5.97 and 2.36 times increase in goodput, and 70.5% and 51.2% cut in end-to-end delay, than using 1 and 2 interfaces, respectively.

Figs. 10 and 11 also show that using more interfaces than 3 interfaces per node cannot obtain proportional performance gain. Since the number of channels is fixed, using more interfaces increases the number of interfering sources within the interference range, thus a lower performance gain. Finally, randomly assigning 1-5 interfaces to nodes does not yield as good performance as using 3 interfaces. Roughly, 40% of nodes have less than 3 interfaces. When a path passes through those nodes, consecutive links along the path tend to use the same channel for data transmission. The intra-path interference limits the overall network performance.

## VII. CONCLUSION

By effectively utilizing non-overlapping channels in an IEEE 802.11-based multi-hop wireless network, collision and co-channel interference can be reduced. This allows more concurrent transmissions, and enhances the network capacity. In this paper, an efficient distributed joint channel assignment and routing algorithm called J-CAR is proposed, where channel assignment and routing are carried out jointly on an on-

demand call-by-call basis. Unlike existing schemes, J-CAR allows a data interface to switch between send and receive modes for each call. This extra flexibility allows J-CAR to have better utilization of both interface and channel. At each hop along a candidate route for a new call, J-CAR conducts a local optimization where the channel with the smallest channel interference index is selected. The channel interference index is defined to capture the impact from both the protocol and physical interference models. To balance the load in the network, J-CAR employs a length-constrained widest-path routing, where the "width" of a path is determined by the interference level of its bottleneck link. With an adjustable threshold on the path length (with respect to the shortest-path), the excessively long path can also be avoided. Simulation results showed that with comparable complexity as existing schemes, J-CAR yields much higher system goodput and lower end-to-end packet delay, due to the improved load balancing and channel selection performance.

REFERENCES

[1] J. Li, C. Blake, D. S. De Couto, H. I. Lee, and R. Morris, "Capacity of ad hoc wireless networks," in *Proc. ACM MobiCom*, 2001, pp. 61-69.

[2] P. Bahl, R. Chandra, and J. Dunagan, "SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks," in *Proc. ACM MobiCom*, 2004, pp. 216-230.

[3] J. So and N. Vaidya, "Multi-channel MAC for ad hoc networks: handling multi-channel hidden terminals using a single transceiver," in *Proc. ACM MobiHoc*, 2004, pp. 222-233.

[4] M. X. Gong, S. F. Midkiff, and S. Mao, "Design principles for distributed channel assignment in wireless ad hoc networks," in *Proc. IEEE ICC*, 2005, pp. 3401-3406.

[5] P. Bahl, A. Adya, J. Padhye, and A. Wolman, "Reconsidering wireless system with multiple radios," *ACM SIGCOMM Computing Commun. Review*, vol. 34, no. 5, Oct. 2004.

[6] J. He, J. Chen, and S. H. G. Chan, "Extending WLAN coverage using infrastructureless access points," in *Proc. IEEE HPSR*, 2005, pp. 162-166.

[7] A. H. M. Rad and V. W. S. Wong, "Joint channel allocation, interface assignment and MAC design for multi-channel wireless mesh networks," in *Proc. IEEE Infocom*, 2007, pp. 1469-1477.

[8] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Budhikot, "Interference-aware channel assignment in multi-radio wireless mesh networks," in *Proc. IEEE Infocom*, 2006.

[9] M. K. Marina and S. R. Das, "A topology control approach for utilizing multiple channels in multi-radio wireless mesh networks," in *Proc. IEEE BroadNets*, 2005, pp. 412-421.

[10] B.-J. Ko, V. Misra, J. Padhye, and D. Rubenstein, "Distributed channel assignment in multi-radio 802.11 mesh networks," in *Proc. IEEE WCNC*, 2007, pp. 3981-3986.

[11] C. Xu, G. Lui, W. Cheng, Z. Yang, "Multi-transceiver multiple access (MTMA) for mobile wireless Ad Hoc networks," in *Proc. IEEE ICC*, 2005, pp. 2932-2936.

[12] J. S. Pathmasuntharam, A. Das, and A. K. Gupta, "Primary channel assignment based MAC (PCAM) V A multi-channel MAC protocol for multi-hop wireless networks," in *Proc. IEEE WCNC*, 2004, pp. 1110-1115.

[13] P. Kyasanur and N. H. Vaidya, "Routing and link-layer protocols for multi-channel multi-interface Ad Hoc wireless networks," *Mobile Computing Commun. Review*, vol. 10, no. 1, 2006.

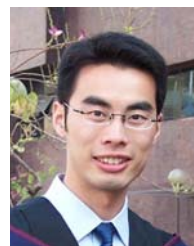
[14] C. E. Perkins, E. M. Royer, and S. Das, "Ad-hoc on-demand distance vector (AODV) routing," *IETF RFC3561*, July 2003.

[15] G. Mao, B. D. O. Anderson, and B. Fidan, "Path loss exponent estimation for wireless sensor network localization," *Computer Networks*, vol. 51, no. 10, 2006, pp. 421-436.

[16] T. K. Sarkar, J. Zhong, K. Kim, A. Medouri, and M. Salazar-Palma, "A survey of various propagation models for mobile communication," *IEEE Antennas Propagation Mag.*, vol. 45, no. 3, 2003.

[17] IEEE 802.11a Standard, [Online] Available: <http://standards.ieee.org/getieee802/download/802.11a-1999.pdf>.

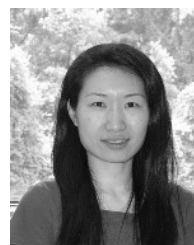
[18] H. S. Chiu, K. Yueng, and K.-S. Lui, "J-CAR: an efficient channel assignment and routing protocol for multi-channel multi-interface mobile ad hoc networks," in *Proc. IEEE Globecom*, 2006.



**Hon Sun Chiu** received his B.Eng. degree in Information Engineering in 2002, and M.Phil. degree in Electrical and Electronic Engineering (major in computer networks) in 2004, both were from The University of Hong Kong. He is currently a Ph.D. candidate in the Department of Electrical and Electronic Engineering of The University of Hong Kong. His research interests include routing, resource allocation and management, MAC layer protocol design, cross-layer optimization and security in both wired and wireless networks.



**Kwan L. Yeung** received his B.Eng. and Ph.D. degrees in Information Engineering from The Chinese University of Hong Kong in 1992 and 1995, respectively. He joined the Department of Electrical and Electronic Engineering, The University of Hong Kong in July 2000, where he is currently an Associate Professor. His research interests include next-generation Internet, active queue management, packet switch/router design, all-optical networks and wireless data networks.



**King-Shan Lui** obtained her B.Eng. (first class honors) and M.Phil. degrees in computer science from the Hong Kong University of Science and Technology. She then received her Ph.D. degree, also in computer science, from the University of Illinois at Urbana- Champaign, USA, in 2002. She joined the Department of Electrical and Electronic Engineering, the University of Hong Kong, as an assistant professor in August 2002. Her research interests include QoS issues, protocol and algorithm design in the Internet, ad hoc networks, and sensor networks. She is a member of IEEE.