A Learning-based Approach for Distributed Multi-Radio Channel Allocation in Wireless Mesh Networks

Sofia Pediaditaki, Phillip Arrieta and Mahesh K. Marina

School of Informatics
The University of Edinburgh, UK

Abstract—We consider the distributed channel allocation problem in IEEE 802.11-based multi-radio wireless mesh networks. We develop a new scalable protocol termed LCAP for efficient and adaptive distributed multi-radio channel allocation. In LCAP, nodes autonomously learn their channel allocation based on neighborhood and channel usage information, which is obtained via a novel neighbor discovery protocol that enables neighboring nodes to efficiently discover each other even when they do not share a common channel. Extensive simulation-based evaluation of LCAP relative to the state-of-the-art Asynchronous Distributed Coloring (ADC) protocol demonstrates that LCAP is able to achieve its stated objectives of efficient channel utilization across diverse traffic patterns, protocol scalability and adaptivity to factors such as external interference. We also present a prototype implementation of the LCAP neighbor discovery module that is key to implementing the proposed approach.

I. INTRODUCTION

Wireless mesh networking is emerging as a promising technology for enabling low-cost, ubiquitous broadband Internet access via reduced dependence on the wired infrastructure. Multi-radio wireless mesh network architecture, in which each router (doubling as an access point) is equipped with multiple radios (e.g., 802.11), is commonly seen as a practical way for efficient utilization of the available spectrum and alleviate the well-known performance degradation in multihop wireless networks with increasing network size, arising from the need to share the wireless medium among neighboring transmissions and the ensuing multiple access interference.

We consider the distributed channel allocation problem in multi-radio mesh networks. Channel allocation involves assigning (mapping) channels to radio interfaces to achieve efficient channel utilization and interference reduction while ensuring network connectivity. This problem is non-trivial in the typical case where the number of radio interfaces per node is smaller relative to the number of available channels. The distributed case is even more challenging because of the channel dependency among the nodes [1], elaborated in Section II. Nevertheless, efficient and adaptive distributed channel allocation is crucial for the following reasons:

- Enable emerging large-scale deployment scenarios (e.g., city-wide mesh network deployments such as in Taipei [2]). Larger scale scenarios also make it important

to flexibly support a wide range of traffic patterns, including "intra-mesh" applications (e.g., surveillance and other neighborhood/community applications [3]).
- Adapt to spatio-temporal variations in the number of available channels and their usability — coping with external interference from other devices using same portion of the wireless spectrum [4]; compliance with regulatory requirements such as dynamic frequency selection (DFS) [5], [6]; and exploiting spectrum "white" spaces [7].

We propose a novel approach for distributed multi-radio channel allocation that is based on learning and a protocol following this approach called Learning-based Channel Allocation Protocol (LCAP). Each node in LCAP independently and iteratively learns the channel allocation using a probabilistic adaptation algorithm for efficient channel utilization while ensuring connectivity. Key enabler of the proposed approach is a novel neighbor discovery mechanism that exploits the mesh network deployment model in practice while being compliant to the 802.11 standard. This neighbor discovery mechanism enables neighbors to discover each other even when they do not share a common channel; it leverages a technique similar to channel quieting in DFS mechanism that is part of 802.11h standard [5]. We present a prototype implementation of the LCAP neighbor discovery module that is key to implementing the proposed approach. We conduct an extensive simulation-based evaluation to evaluate the effectiveness of LCAP with respect to channel utilization, network performance with diverse traffic patterns, protocol scalability and adaptivity to factors such as external interference. Our results convincingly demonstrate that LCAP delivers superior performance on these dimensions compared to the state-of-the-art.

LCAP addresses the limitations of prior work [1], [8], [9], [10], [11], [12], [13], [14] by not placing any restrictions on the use of an interface, network structure or traffic patterns, while at the same time is localized and negotiation-free for scalable operation. Another factor contributing to LCAP's ability to achieve efficient channel utilization is the fact that it does not require a common channel like some of the existing approaches (e.g., [11]). It also has an inherent adaptive quality that is key to coping with factors such as external interference and benefiting from dynamic spectrum access opportunities.

The remainder of this paper is structured as follows. Next

section discusses related work. Section III presents the network model under consideration and provides a brief tutorial on learning automata upon which LCAP probabilistic channel allocation mechanism is based. In Section IV, we give an overview of LCAP design goals, approach and its conceptual architecture. Section V describes the LCAP neighbor discovery module, whereas probabilistic channel adaption in LCAP is explained in Section VI. Section VII evaluates LCAP using a wide range of simulation experiments. We finally conclude in Section VIII.

II. RELATED WORK

A. Distributed Multi-Radio Channel Assignment

Several *distributed* channel assignment algorithms exist (e.g., [1], [8], [9], [10], [11], [12], [13], [14]). We review the prominent ones here. Broadly speaking, existing protocols can be divided into two categories:

1) Protocols that assign certain roles to interfaces (e.g., UP and DOWN in [1], Fixed (Receive) and Switchable (Send) in [8], [9], [12], [14]).

2) Protocols that rely on negotiation to perform channel assignment in a distributed manner (e.g., [10], [11], [13]).

1) Protocols based on Interface Role Assignment: Raniwala and Chiueh [1] were the first to highlight the channel dependency problem in the context of distributed multi-radio channel assignment. Due to this problem, changing the channel of an interface may cause a ripple effect of further changes in the network necessary to maintain connectivity. To bound the scope of a channel change, the authors impose a tree structure and partition the interfaces at a node into two disjoint sets (UP-NICs and DOWN-NICs) — a node only determines the channels for its DOWN-NICs, whereas the channels for its UP-NICs are determined by the parent node. Even though their routing and channel assignment solution is load-aware, the aforementioned role assignment to interfaces leads to inefficient channel utilization for traffic patterns other than the gateway-oriented traffic pattern.

A different solution called HMCP is proposed by Vaidya et al. [8], [9] in which interfaces at a node are designated as either fixed or switchable. Different nodes use different channels for fixed interfaces; this channel assignment is not load-based and infrequently changed. Fixed interfaces can be seen as interfaces used for receiving data from neighboring nodes — a node uses its switchable interface tuned "on-demand" to the channel used by the fixed interface of a neighbor to communicate with that neighbor. Clearly, this protocol requires that each node has at least two interfaces. However, the case where nodes may have more than two interfaces is left unspecified by the authors. Besides, as elaborated in [9], this protocol cannot be readily implemented in current systems and requires special kernel support because it does not conform to the usual practice of associating each network interface with exactly one channel. The need for channel switching along with receive (fixed) channel contention in HMCP lead to

inefficiencies and limited performance improvements (in terms of throughput and delay), especially for traffic patterns where multiple flows are routed via a node.

Xing et al. [12] propose an alternative approach for localized channel assignment based on s-disjunct superimposed codes to support both unicast and local broadcast. While this approach like the previous one also divides the interfaces into transmit and receive categories, it does transmitter oriented channel assignment unlike [8]. As authors themselves note, their approach may not be effective for 802.11-based mesh networks with few tens of channels because the code strength is limited by the number of available orthogonal channels. Furthermore, this approach also limits the network size, which makes its practicality questionable.

In a recent work, Gao and Wang [14] take a game-theoretic approach for multi-radio channel allocation in mesh networks, again having different sets of interfaces for transmission and reception like [8], [12]. They, however, make a number of unrealistic assumptions, including: (i) each node participates in only one communication session; (ii) the whole network is a single collision domain (i.e., a node can hear transmissions by any other node in the network using the same channel); and (iii) Available bandwidth on a channel is equally shared by all radios using that channel. Thus, this work is mainly of theoretical interest.

2) Negotiation-based Protocols: Protocols in this category employ some form of distributed mutual exclusion [15] to carry out channel assignment.

Wu et al. [10] propose a distributed channel assignment approach that incrementally adapts the channel allocation by having each node update channels for its interfaces via a protocol for negotiation with a neighboring node followed by consensus building step among neighbors of the negotiating node pair. This approach has two key drawbacks: (i) channel change for an interface is considered only when it does not cause the channel dependency problem mentioned earlier (referred instead as chain puzzle in [10]), which is quite restrictive; (ii) the overhead of the protocol for negotiation, consensus and notification can be prohibitive since it involves voting and needs to be repeated several times to ensure network connectivity in the presence of message losses. To assist in the channel adaptation, the authors in the same work also introduce a channel cost metric used by each node that represents aggregate expected transmission time (over all channels) weighted by channel utilization in the neighborhood; this metric is somewhat similar to the CATT metric proposed in [16].

Ko et al. [11] propose a channel assignment protocol (referred to as ADC in this paper) that uses a node channel selection algorithm which minimizes average potential interference within the interference range of the node. The ADC protocol alleviates the two drawbacks of the protocol proposed by Wu et al. [10] by requiring that "one interface of each node is dedicated to a default channel common to all nodes" and using the common channel for negotiation purposes. However, this leads to inefficient utilization of

available channels and interfaces. And channel negotiation in ADC still incurs substantial overhead. Nevertheless, among the protocols discussed thus far, the ADC protocol is the most easily implementable solution that can also support diverse traffic patterns.

Recently, Subramanian et al. [13] proposed a distributed greedy algorithm (DGA) for multi-radio channel assignment. Specifically, the responsibility for assigning channel to a link is given to the end point of that link with higher node ID; the "owner" of each link does the channel assignment via a negotiation and notification protocol similar to [10]. However, the DGA protocol is simplistic and can in fact cause network partitions. This can be easily seen by considering a simple tandem network with nodes in the middle having only one interface and lower IDs compared to those towards the ends.

B. Routing and Congestion Control

Several routing metrics have been proposed for multi-radio mesh networks to improve upon the traditional shortest-hop approach by accounting for link loss characteristics, transmit rate diversity and multiple access interference. These include CATT, iAWARE, MIC, MCR, WCETT. See [16], [8] and references therein.

Congestion control and fairness issues are beginning to be looked at in the multi-radio, multi-channel wireless mesh networks context. Giannoulis et al. [17] recently proposed a iterative, decomposition approach to jointly consider congestion control and channel assignment.

III. MODEL AND PRELIMINARIES

We consider a two-tier mesh network architecture, as in [18], [19], comprising of an access tier and a backhaul tier. The access tier connects end-user client devices to mesh nodes (i.e., each mesh node has WLAN AP functionality). Mesh nodes form the backhaul tier with a subset of the mesh nodes serving as gateways to the wider Internet. The multihop mesh backhaul connects clients with the Internet and other client devices. This is a fairly common model in practice as is the separation of access and backhaul tiers on different radios and frequency bands [19], and the use of multiple radios for the mesh backhaul [20], [2]. Henceforth, we refer to the radio interfaces used for backhaul communication as *mesh interfaces* and interface used for client access as the *access interface*. We assume that all radio interfaces use omnidirectional antennas.

A few observations from a real-world perspective are in order to make the aforementioned model concrete. Using multiple backhaul radios operating in the 5GHz band is particularly attractive for improved capacity scaling given the availability of more number of channels (11 channels worldwide for indoor and outdoor use in 5.470-5.725 sub band and up to 24 channels in the US [6]) and the fact that 5GHz spectrum is less crowded. On the other hand, 2.4GHz band is typically used for the access interface as most client devices have a 802.11b/g interface. Regarding the typical number of radio interfaces, multi-radio platforms in the market provide up to 4 mini-PCI slots (e.g., RouterBOARD,

Gateworks, WILIGEAR, Pronghorn). With 4 interface slots and assuming one slot is used for the access interface, up to 3 interfaces can be used for the mesh backhaul with current hardware. Also, multi-band interfaces are preferred as they are widely available and provide greater flexibility than single band interfaces while having similar cost.

We now introduce the notion of a *channel set*, which is the main tuning parameter in our proposed protocol. A channel set is 'a subset of channels' of size equal to the number of radio interfaces at a node. With S denoting the set of all channel sets, c channels and m interfaces, the number of channel sets, $|S| = \binom{c}{m}$. For example, for $c = 3$ and $m = 2$, there are $\binom{3}{2}$ = 3 possible channel sets: $\{(1, 2), (1, 3), (2, 3)\}$.

A. Learning Automata

Here we give a brief overview of learning automata [21] concepts to serve as a background for the LCAP probabilistic channel set adaptation component described in Section VI. Our discussion focuses on a specific type of learning automata called variable structure stochastic automata. A learning automaton is a mechanism intended for adapting to changes in environments with unknown characteristics via a learning process. An environment is represented by a triple $\{a, c, \beta\}$, where a represents the action set, c represents the set of penalty probabilities (each c_i corresponds to an action a_i in set a) and β represents the response set. The goal of an automaton is to choose the optimal action among the set of actions such that the average penalty is minimized (or equivalently, the average reward is maximized) after a sequence of rounds. Specifically, the automaton maintains a probability vector $p(t) = < p_1(t), p_2(t), ..., p_r(t) >$ associated with a predetermined set of actions a provided by the environment, where r corresponds to the number of actions in the action set and $\sum_{i=1}^{r} p_i(t) = 1$. In each round t, an action a_i is selected with probability p_i and the environment provides a penalty (or reward) c_i, which is used by the automaton to update the probabilities in $p(t)$. Specifically, the probability vector is updated as indicated by the used reinforcement scheme, which controls the learning behavior of the automaton.

A general form of this scheme follows the rules shown in equations (1) and (2). At time t, the automaton has chosen action $a(t)$ and receives the environmental response $\beta(t)$. All probabilities corresponding to actions other than the used one are updated according to equation (1), while the probability of the current action is updated according to equation (2). Functions g_i and h_i are linear or non-linear functions of the probability of some action a_i.

$$
\begin{aligned}
p_i(t+1) = p_i(t) \quad &- \quad (1 - \beta(t)).g_i(p(t)) \\
&+ \quad \beta(t).h_i(p(t)), \text{if} \quad a(t) \neq a_i \quad (1)
\end{aligned}
$$

$$p_i(t+1) = p_i(t) \quad + \quad (1-\beta(t)).\sum_{j\neq i}g_j(p(t))$$
$$- \quad \beta(t).\sum_{j\neq i}h_j(p(t)), \text{if} \quad a(t)=a_i \quad (2)$$

Different value sets from which the environmental response can take values define different models for the automaton. In a P-model automaton, the response is binary — 0 or 1 corresponding to favorable and unfavorable response, respectively. Q and S models differ from P-model in the sense that they neither totally reward nor totally penalize an action. Specifically, the environmental response in Q-model takes values from a finite set in [0, 1], while the response set is continuous in [0, 1] with S-model.

Moreover, depending on the functions g_i and h_i, several linear and non-linear reinforcement (updating) schemes can be obtained. Linear schemes are simplest and commonly used. They include the linear reward-penalty (L_{R-P}), linear reward-$\in$penalty ($L_{R-\in P}$) and linear reward-inaction (L_{R-I}). For r actions and binary environmental response, the general L_{R-P} scheme is shown in equations (3)–(6).

If $\beta(t)=0$

$$p_i(t+1) = (1-a).p_i(t)), \text{if} \quad a(t)\neq a_i \quad (3)$$

$$p_i(t+1) = p_i(t) + a.(1-p_i(t)), \text{if} \quad a(t)=a_i \quad (4)$$

If $\beta(t)=1$

$$p_i(t+1) = \frac{b}{r-1} + (1-b).p_i(t)), \text{if} \quad a(t)\neq a_i \quad (5)$$

$$p_i(t+1) = (1-b).p_i(t), \text{if} \quad a(t)=a_i \quad (6)$$

These equations are obtained by substituting $g_i(p(t)) = a.p_i(t)$ and $h_i(p(t)) = \frac{b}{r-1} - b.p_i(t)$ in equations (1) and (2), and noting that $\sum_{i=1}^{r} p_i(t) = 1$. In these equations, $0 < a, b < 1$ are learning parameters associated with reward and penalty response, respectively. The scheme is symmetric if $a = b$. In the case of L_{R-I}, $b = 0$, which means that this scheme ignores penalty responses from the environment. For $L_{R-\in P}$, $0 < b \leq a < 1$.

The suitability of the learning automaton approach for distributed adaptive decision making in highly uncertain stochastic environments combined with its theoretical basis has led to its application for various wireless networking problems (e.g., [22], [23]). In this paper, we present a novel application for learning automata, i.e., distributed multi-radio channel assignment problem in mesh networks.

IV. LCAP OVERVIEW

Channel allocation in multi-radio wireless mesh networks involves assigning channels to mesh interfaces to achieve efficient channel utilization and interference reduction while ensuring connectivity of the backhaul. We consider the *distributed* multi-radio channel assignment problem. In addressing this problem, we set ourselves the following design goals:

- *Efficient Channel Utilization*: The rationale behind having this as an objective is clear and it directly benefits network performance (in terms of throughput and delay). This can be realized by reducing contention (interference) on any given channel by distributing it across as many channels as possible while not compromising network connectivity.
- *Protocol Scalability*: Our solution should have low communication overhead, thereby scale well to larger network scenarios.
- *Adaptivity*: We want our solution to not only adapt to network topology changes (e.g., node joins and failures), but also adapt to spatio-temporal variations in the number of usable channels, caused by factors such as external interference.
- *Flexibility*: We aim to support arbitrary traffic patterns and the use of any routing protocol (and metric) on top of our solution. Flexibility also means not placing any restriction on the use of an interface.

To address the above goals, we propose a novel learning-based approach that is fundamentally different from existing approaches. Specifically, with our proposed protocol termed LCAP, nodes autonomously learn their channel allocation, i.e., selection of a channel set (see Section III), based on the well developed theory of learning automata [21], reviewed in Section III-A. This learning is only based on information about local neighborhood and channel utilization within that neighborhood. Each node acquires this information via a novel and lightweight neighbor discovery mechanism in LCAP that can help discover even those neighbors with whom the node does not share a common channel. This is achieved by exploiting mesh network deployment model in practice and using channel quieting and switching in a way that still ensures compliance with the 802.11 standard.

The conceptual node architecture with LCAP is shown in Fig. 1. Note that the channel set adaptation module at each node interacts only with the local neighbor discovery module. Also note that the configuration of interfaces with new channels determined by the channel set adaptation module is not explicitly shown in the figure. Routing and IP forwarding at the network layer relies on LCAP neighbor discovery module for two purposes: (1) routing related broadcast transmissions (e.g., TC messages in OLSR), which need to be sent over all M mesh interfaces; (2) to obtain the interface channel assignment information to update the IP forwarding table. Moreover, LCAP neighbor discovery module obviates the need for routing protocol neighbor discovery messages (e.g., OLSR HELLO messages). Specifically, LCAP HELLO messages can
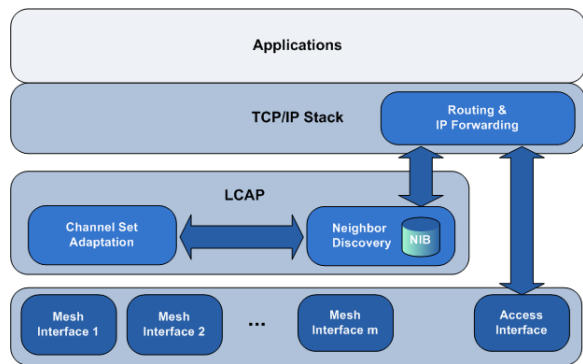
Fig. 1.   LCAP conceptual node architecture.

Fig. 2.   NAV-based channel quieting mechanism in LCAP neighbor discovery.

easily meet this need with the inclusion of additional information required by the routing protocol neighbor discovery component.

The fact that LCAP does not use a default channel like some other approaches (e.g., [11]) together with the selection of diverse channels for interface assignment at each node (elaborated in Section VI) helps in achieving efficient channel utilization. LCAP's negotiation-free property lowers the protocol overhead and contributes to its scalability. In fact, the only source of overhead in LCAP are the periodic HELLO messages used for neighbor discovery as evident from Fig. 1. Therefore, LCAP's implementability rests largely on the neighbor discovery module. We have developed a prototype implementation for this module, which is described in Section V. Probabilistic channel set adaptation (described in Section VI) helps achieve the adaptivity and flexibility goals.

V. NEIGHBOR DISCOVERY

In this section, we describe the neighbor discovery component of the LCAP protocol. At a high level, discovering neighboring mesh routers in a multi-radio wireless mesh network seems straightforward. This can be done by having each node periodically "broadcast" HELLO messages locally to announce its presence[1], thereby allowing neighboring nodes to discover it.

Looking further into the details reveals that neighbor discovery is a more involved problem. We make the following observation to make this clear. Doing a local broadcast in a multi-radio mesh network may require sending the message to be broadcasted on *all* available channels if the sending node does not share a common channel with each of its neighboring nodes. This is more so the case when the assignment of channels to radio interfaces is *being* determined. Since the number of mesh interfaces is typically smaller than the available channels, the channels they might be tuned to at a given point in time will not cover all channels. Therefore, it is necessary to somehow send each local broadcast message (e.g., a HELLO message) on every channel (even those that are not currently used by the mesh interfaces), especially when

[1]HELLO messages can additionally carry the channel utilization map in the local neighborhood as seen by the sending node to assist in finding a diverse channel assignment for interfaces.
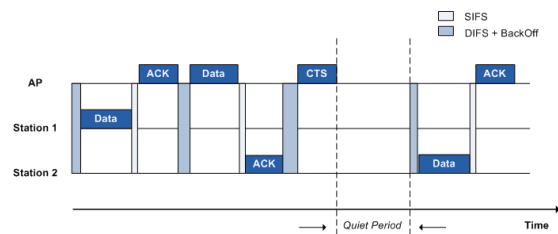
the channel assignment is being computed. This could be done naively by having each mesh interface send the message on its currently assigned channel and additionally designating one of the mesh interfaces to cycle through the rest of the "unused" channels to broadcast the message on those channels. However, this solution not only seems complicated but can also be disruptive to the distributed channel assignment process.

We propose a less disruptive solution that exploits the common two-tier mesh network deployment model described in Section III and makes use of channel quieting and switching as in the Dynamic Frequency Selection (DFS) mechanism that is part of the 802.11h standard [5]. The idea is to continue sending each HELLO message on mesh interfaces over their assigned mesh channels as in the naive solution above, but *use the access interface to send the HELLO message over unused mesh channels*. Specifically, client activity on the access interface of a mesh node is temporarily suspended for a short period every so often to "hop" through the unused channels, sending the HELLO message over each of them in the process. If the access interface is also operating over the 5GHz band like mesh interfaces, then this can be achieved by just using the channel quieting feature in the existing DFS mechanism, which is mandatory for 5GHz operation to avoid interference with radar systems.

But it is common for the access tier to use the 2.4GHz band, for which DFS functionality does not exist. For this typical case, we propose a *Network Allocation Vector (NAV) based channel quieting* mechanism to realize the channel quieting feature on 2.4GHz band with the very reasonable assumption that access interface can support multiple bands (both 2.4GHz and 5GHz bands). The idea is as follows. Whenever a mesh node needs to broadcast a HELLO message over its unused mesh channels in 5GHz band, the access interface configured as an AP generates a gratuitous CTS frame (more generally, a 802.11 MAC control frame) over the currently used channel in 2.4GHz band (say, channel 6) with the duration field in the frame set to the required *channel quieting period*. This causes clients associated with that AP to go into waiting mode until the specified NAV period elapses. This is illustrated in Fig. 2. Immediately after effecting channel quieting, the access interface switches its band to 5GHz and hops through the specified set of unused mesh channels (say, channels 100, 120 and 140) sending HELLO message over each of those channels before finally switching back to the initial channel in the 2.4GHz band (i.e., channel 6 in this example).

Note that the NAV-based solution just described is compliant with the 802.11 standard and does not require any client-side modifications. In fact, several mechanisms in the 802.11 standard exploit the NAV feature, including inter-operation of contention-free channel access with contention-based access and 802.11g protection. We would also like to point out that our approach does not require time synchronization among nodes because nodes receive HELLO messages on channels currently assigned to their mesh interfaces. We present a method for estimating the channel quiet period in Section V-B given the current channel used by the access interface and the set of mesh channels to be visited. In the event a single channel quieting period is insufficient to visit all channels, the same process can be repeated a few times to complete the process of sending the HELLO message over all unused channels. Even then, the disruption to client traffic is going to be minimal — wait for channel access will likely be below hundred milliseconds as the channel quieting period with this mechanism is upper bounded by the maximum possible NAV value (approx. 33ms). Since reliable forwarding of client traffic is anyway dependent on having a stable backhaul mesh channel assignment and neighbor discovery is a crucial part of the channel assignment, this small overhead is justified. Note that this overhead is incurred only part of the time by using our optimization to reduce the frequency of HELLO messages when the backhaul mesh is sufficiently connected (see Section VI).

A. Neighborhood Information Base (NIB) and Hello Messages

Each node maintains information about neighbors and channel utilization around the node. We collectively refer to this information as the Neighborhood Information Base (NIB). It consists of NeighborTable and ChannelUsageList. NeighborTable at node I contains an entry for each 2-hop neighbor node J along with channels used by node J's interfaces. Each entry also includes additional information such as the latest sequence number received for node J, expiry time of that entry and a quality field. The quality field is used to implement a hysteresis feature (similar to OLSR) for robustness against bursty HELLO message losses and to filter out transient neighbors.

Each node I uses the channels usage information in its NeighborTable to maintain an up-to-date ChannelUsageList; this list contains an entry for each channel with the corresponding value indicating the the count of the number of interfaces in the node's 2-hop neighborhood using that channel.

Each HELLO message broadcasted by a node I contains: channels used by I and its direct (1-hop) neighbors; a fresh sequence number generated by I; and I's ChannelUsageList.

B. Estimating Channel Quieting Period

Suppose that the access interface currently uses channel c_{init} in 2.4GHz band and also suppose that it has to visit channels $< c_1, c_2, ..., c_k >$ in 5GHz band in that order for neighbor discovery. It can be easily shown that the channel quieting period for this operation can be estimated as:

$$
\begin{aligned}
ChanQuietPeriod \quad = \quad & Switch_{2.4 \to 5}(c_{init} \to c_1) \\
& + \sum_{i=1}^{k} ChanDwellTime(c_i) \\
& + \sum_{i=1}^{k-1} Switch_5(c_i \to c_{i+1}) \\
& + Switch_{5 \to 2.4}(c_k \to c_{init}), \quad (7)
\end{aligned}
$$

where $Switch_{2.4 \to 5}(Switch_{5 \to 2.4})$ represent the delay for switching from a channel in 2.4GHz (5GHz) band to a channel in 5GHz (2.4GHz) band and $Switch_5$ represents the delay for switching between two channels within 5GHz band. And $ChanDwellTime(c)$ is the time spent in a channel c while trying to broadcast a HELLO message. It can be estimated as:

$$
\begin{aligned}
ChanDwellTime(c) \quad = \quad & DIFS \\
& + Backoff(c) * slotTime \\
& + FrameSize/TransmitRate \\
& + PropagationDelay, \quad (8)
\end{aligned}
$$

where $FrameSize$ is the size of the HELLO message size, $TransmitRate$ is typically the lowest for broadcast transmissions (6Mbps) and $Backoff(c)$ in slots can be estimated based on the work in [24] and the number of contending nodes on channel c obtained using the ChannelUsageList. Typical dwell time values are in the order of a few milliseconds. Even in the worst case when using 802.11 maximum frame size and having ten other contending nodes on the same channel (resulting in backing off for around 120 slots [24]), the dwell time is around 5.5ms.

C. Implementation

We have developed a prototype implementation for the LCAP neighbor discovery module on Gateworks multi-radio platform with 4 multi-band atheros based mini-pci cards. The implementation works in user space and is based on libpcap, 802.11-augmented libnet and ioctl system calls.

Channel quieting, the key component of the neighbor discovery module, requires that the access interface operate in monitor mode in order to function. However, performing a mode switch from AP mode to monitor mode causes the device driver to disassociate all associated clients, an undesirable situation. To avoid this, our design relies on a unique feature of the madwifi-ng driver used in our multi-radio mesh testbed. Specifically, madwifi-ng supports the creation of multiple virtual wireless interfaces which map to a single physical device, abstracting the functions provided by the hardware. For the purpose of implementing NAV-based channel quieting, we have used a virtual interface operating in AP mode to provide connectivity to clients while another virtual interface operating in monitor mode is used to perform the auxiliary functions of neighbor discovery, including channel quieting.

Fig. 3 demonstrates the working of our NAV-based channel quieting solution. For this experiment, a node A sends a very
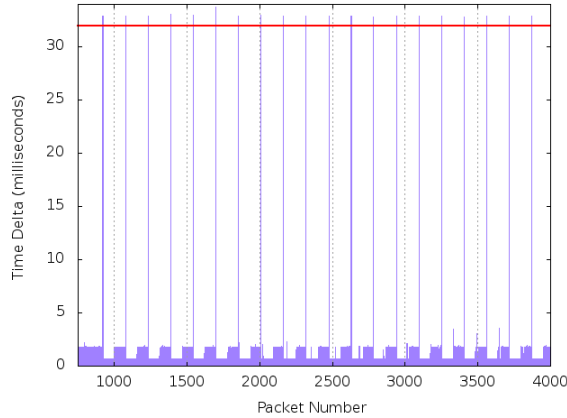
Fig. 3. Experiment demonstrating the working of NAV-based channel quieting solution with a quieting period of $32767\mu s$.

high rate UDP traffic to a node B using the iperf tool, and a third node C in range of both nodes A and B applies channel quieting every 100ms with a quieting period of $32767\mu s$ (the maximum NAV value). The plot in Fig. 3 shows the inter-frame gaps, observed at a different radio interface on node C in monitor mode. This experiment was carried out on an unused 802.11 channel. Fig. 3 shows the gap between frame receptions at node B versus the corresponding packet numbers. The periodic spikes of height around 32ms clearly show the effectiveness of the NAV-based quieting approach.

We have also carried out extensive set of measurements on our testbed to study the channel and band switching delays. A subset of these results relevant for estimating various switching delays in equation (7) are shown in Table I. These results are obtained from considering large number of channel pairs both within and across bands and conducting 1000 experimental trials per channel pair. We found that $Switch_{2.4\to5}$ and $Switch_{5\to2.4}$ are similar, so we show measurement results for both using $Switch_{2.4\leftrightarrow5}$. These results show that band switching incurs a slightly greater delay (around 0.3ms) and switching delays are fairly stable (standard deviation under $50\mu s$ whereas the mean switching delay is two orders of magnitude greater). We can estimate the switching delays in equation (7) as Mean + 4*StdDev from Table I, as with TCP retransmission timeout estimation.

VI. CHANNEL SET ADAPTATION

In this section, we describe the probabilistic channel set adaptation algorithm used at each node in LCAP that is rooted in learning automata [21]. Utility of learning automata

	Mean	StdDev
$Switch_{2.4\leftrightarrow5}$	6.54ms	$26.18\mu s$
$Switch_5$	6.24ms	$40.18\mu s$

TABLE I
CHANNEL SWITCHING DELAY MEASUREMENT RESULTS.

approach for decentralized control problems was nicely articulated by Pasquale [25]. Our distributed multi-radio channel assignment problem shares the two fundamental characteristics of decentralized control problems outlined by Pasquale [25]: state-information uncertainty and mutually conflicting decisions. The former is obvious because in a distributed system, each node only has a partial and possibly outdated view of the whole system state information. Latter is also true due to the likelihood of channel dependencies, as discussed in Section II. The application of learning automata approach to our distributed channel assignment can then be seen as a proactive and probabilistic search for good collective decisions via a series of "experiments" [25].

A. Channel Set Quality Metric

Before looking at the details of channel set adaptation algorithm, let us consider the question of evaluating the quality of a channel set as our problem is essentially that of enabling each node to autonomously converge on a good channel set by learning through feedback from the environment.

We use a simple and intuitive cost function for relative assessment of channel set qualities that "loosely" reflects the delay experienced for communication with neighboring nodes when using a channel set.

Define $\delta_i^c(j)$ for a pair of direct (1-hop) neighbors node i and node j, and channel c such that $\delta_i^c(j) = 1$, if nodes i and j share channel c, ∞ otherwise.

Now define the cost of communicating from node i to a direct neighbor j over channel c as:

$$NC_i^c(j) = \delta_i^c(j).\max(ChannelUsageList_i(c),$$
$$ChannelUsageList_j(c)), \quad (9)$$

where ChannelUsageList is part of the NIB maintained at each node (see Section V) and $ChannelUsageList_i(c)$ is the number of interfaces in the 2-hop neighborhood of node i assigned to channel c, as known by node i. Note that $\delta_i^c(j)$ in equation 9 can be seen as a connectivity check — only those neighbors sharing a common channel with node i have finite NC values at node i on common channels. The second term (i.e., the max operation) in equation 9 estimates the number of potential interfering (contending) nodes for node i when it tries to communicate with its direct neighbor j over channel c; this is obviously meaningful when $\delta_i^c(j) = 1$. Also note that nodes up to 3 hops away from node i are considered as potential interfering nodes.

Given the above, the quality of a channel set, s, at node i is defined as follows:

$$CSQ_i^s = \sum_j NC_i^s(j),$$
$$\text{where} \quad NC_i^s(j) = \min(NC_i^c(j)), c \in s. (10)$$

B. Probabilistic Channel Set Adaptation

Each backhaul mesh node has a learning automaton (see Section III-A) to help determine the channel assignment for

mesh interfaces at that node. The action set for each automaton is the set of all possible channel sets denoted by S (see Section III). Initially, all probabilities in the probability vector are set to $1/|S|$, meaning every channel set is equally likely. Afterwards, every $adaptation\_interval$[2] (referred to as an adaptation round) at node i, it first computes the quality of all channel sets using equation (10) based on equation (9) and the information obtained via the LCAP neighbor discovery module (see Fig. 1 and Section V). The automaton at i then adjusts the probabilities in the probability vector based on the following linear update scheme, where s is the currently used channel set.

If $CSQ_i^s = \min(CSQ_i^u), u \in S$

$$p_k(t+1) = (1-a).p_k(t), \text{if} \quad s \neq k \qquad (11)$$

$$p_k(t+1) = p_k(t) + a.(1-p_k(t)), \text{if} \quad s = k \qquad (12)$$

If $CSQ_i^s \neq \min(CSQ_i^u), u \in S$

$$p_k(t+1) = \frac{b}{|S|-1} + (1-b).p_k(t), \text{if} \quad s \neq k \qquad (13)$$

$$p_k(t+1) = (1-b).p_k(t), \text{if} \quad s = k \qquad (14)$$

Note that these equations are essentially similar to equations (3)–(6) — equations (11) and (12) correspond to (3) and (4), whereas (13) and (14) correspond to (5) and (6). The goal of the above update scheme is to reduce the delay to communicate with neighbors by progressively and eventually moving towards a channel set that is sufficiently diverse from channel sets used by other nodes in the neighborhood. If the channel set quality value of the currently used set is the minimum among all possible sets, the environmental response is perceived as a reward and the probability of the current set is increased (Eq. 12). The probabilities of other channel sets are uniformly decreased (Eq. 11). On the contrary, if the channel set quality value for the current set is not the minimum, then the probability of the current set is decreased (Eq. 14), while the probabilities of the remaining sets are increased (Eq. 13). The learning parameters a and b for the aforementioned scheme were empirically determined to be 0.3 and 0.08, respectively.

Note that adaptation rounds at different nodes are independent and asynchronous, determined by the adaptation interval randomly chosen from a specified range depending on the state of the channel assignment (more on this later). An attractive feature of this update scheme is that it gradually increases the probability of the best action instead of totally committing to the action inferred to be the best in one-go. This gradual approach is more suitable for non-stationary environments where the penalty probabilities change over time — when a network of automata operates on the same environment, the action chosen by one automaton can change the quality of an action previously inferred to be the best by another automaton.

We have developed several optimizations to aid in faster convergence and further reduce protocol overhead. First, we

allow exploration (i.e., continuation of the above probability vector updating scheme) until a channel set that provides connectivity to all neighbors[3]. In the event of any disruption (e.g., node join, node failure, external interference), the exploration is resumed again (somewhat akin to the way backoff counters are handled in the 802.11 MAC protocol). Second, we vary mean adaptation intervals and frequency of HELLO broadcasts in neighbor discovery based on the achieved connectivity to further improve convergence times and reduce neighbor discovery overhead. Specifically, each node evaluates its current channel set choice in terms of the connectivity. If a node is connected to more than x% of its neighbors (50% in our implementation), it increases the mean adaptation interval (from 3.5s to 16.5s in our evaluations[4]). By default, the neighbor discovery process at each node runs every HELLO interval (mean value set to 7.5 seconds in our implementation with interval values drawn uniformly from the range (0, 15s]). If, however, every neighbor of a node is connected to all its direct neighbors (kept track using a $connectivity\_status$ variable in the neighbor discovery NIB), the HELLO message frequency at the node is halved to reduce the overhead.

In the next section, we demonstrate good convergence behavior of LCAP experimentally. In on-going work, our focus is on analytically characterizing LCAP convergence properties. This is challenging in part because of the non-stationary nature of the environment in our setting. We observe that the size of the action set also plays a crucial role in the speed of convergence.

We conclude this section by noting that our probabilistic channel adaptation framework is fairly general. For instance, it can be extended to account for adjacent channel interference and partially overlapping channels by using a modified channel quality set metric.

VII. EVALUATION

In this section, we study the performance of LCAP and evaluate its effectiveness in terms of the goals stated in Section IV, viz. efficient channel utilization, protocol scalability, adaptivity and flexible support of different traffic patterns. Our evaluation is via simulation. We use the QualNet simulator version 4.0 for our evaluations. We choose ADC protocol [11] for comparative evaluation as it is the most practical solution in the literature that can also support diverse traffic patterns (see discussion in Section II). We also include the single channel case as a baseline. We have implemented both LCAP and ADC on QualNet. ADC implementation is based on the ADC paper, consultation with the authors and a longer technical report version provided by them. Our LCAP QualNet implementation includes the neighbor discovery module as described in Section V that uses the access interface for short periods periodically via channel quieting. The implementation

---

[2]The adaptation interval is a variable parameter in LCAP.

[3]We avoid useless channels that are not utilized by any neighbor.

[4]To be precise, the adaptation interval is randomly chosen from the interval [2, 5] seconds when neighbor connectivity is under 50% and it is chosen from [15, 18] seconds above that threshold.

of the LCAP protocol on a multi-radio mesh testbed is in progress.

We set the channel and physical layer parameters to reflect a urban mesh network scenario based on the earlier measurement work in [18]. Specifically, we use two-ray propagation model with pathloss exponent $\alpha = 3.3$, shadowing with standard deviation $\sigma_\epsilon = 5.9$, omnidirectional antennas with 15dBi gain and placed at 10m height [18]. Transmission power and receive sensitivity values for different transmission rates are set referring to default values for commodity hardware (specifically, Atheros-based Compex WLM54AG mini-PCI cards used in our multi-radio mesh network testbed). For routing, we use the OLSR routing protocol with the CATT metric [16]. CATT metric has been shown to outperform other routing metrics for multi-radio mesh networks, and additionally has certain attractive features such as isotonicity and a unified way of accounting both inter-flow and intra-flow interference.

A. Channel Utilization and Network Performance

As described in Section IV, efficient channel utilization without hurting network connectivity is one of our goals. Before examining LCAP's ability to achieve this goal, let us first consider the metrics to quantify channel utilization and connectivity. For channel utilization, we use a simple metric that is protocol independent yet captures the extent to which contention (interference) is evenly distributed across all channels. Specifically, we use the difference between maximum and minimum number of mesh interfaces assigned to any channel, over all channels, as a measure of channel utilization — a lower value of this measure implies a better channel utilization. Note that the maximum number of interfaces assigned to a channel is upper bounded by the network size (as would be the case in a single-interface, single-channel network, for example). We use this upper bound to normalize channel utilization measure and show it as a percentage value (see Fig. 4(a)). We measure network connectivity as the percentage of links in a single channel network that are preserved by the multi-radio channel assignment protocol. As opposed to the standard graph-theoretic connectivity measures such as k-connectivity, this measure allows a fair comparison with ADC as it also achieves connectivity like in a single channel network (via the common channel).

Fig. 4 (a) shows the key result, demonstrating that LCAP offers significantly better channel utilization than ADC while ensuring connectivity. These results correspond to a 25-node 802.11-based multi-radio mesh network with nodes randomly distributed in a 1000m x 1000m field, 11 backhaul mesh channels in the 5.470-5.725GHz band and 3 mesh interfaces per node. We have also experimented with different number of channels and interfaces. Our results also show qualitatively similarly impact from varying number of channels and interfaces as observed by other researchers in the past (e.g., [1]). Fig. 4(a) shows that LCAP provides up to 40% improvement in channel utilization over ADC. This is because LCAP does not require a common default channel (which has all nodes assigned to it in ADC) and due to its ability to select diverse channel sets for interfering set of nodes. This is also confirmed by the channel utilization distribution plot in Fig. 4(d), which shows that available channels are more evenly used with LCAP. Note that achievable channel utilization is inherently limited by the number of interfaces and the need to maintain network connectivity; this means some degree of channel reuse is inevitable, which partly explains why LCAP channel utilization stabilizes around 48%. The result in Fig. 4(a) also shows the convergence times for LCAP and ADC. Note that connectivity curve is not shown for ADC as it is always connected due to the common channel. Observe that LCAP channel assignment converges much faster than ADC because the channel assignment in the latter is via negotiations with 3-hop neighbors of a node over the common channel.

The improved channel utilization in LCAP leads to its significantly better link layer throughput and delay performance compared to ADC and the single channel case (see Fig. 4(b, e)) for the same network scenario as before. Note the log-scale in the delay plot. These results are obtained by increasing packet generation rate on each link for fixed size (1KB) packets. Fig. 4(c, f) shows the relative performance of LCAP and ADC in the case of multihop traffic with two different traffic patterns. Random traffic pattern involves 10 CBR/UDP flows with 1KB packets and varying packet rate between 10 randomly chosen source-destination pairs. On the other hand, the traffic is from an Internet gateway node to 10 randomly chosen non-gateway nodes with all else being same as the random pattern. LCAP exhibits superior performance in both traffic patterns with up to 40% throughput improvement and similar or better delays, demonstrating its ability to support diverse traffic patterns. The performance differentials drop in the case of gateway pattern because the gateway node becomes the bottleneck limited by its number of interfaces (3, in this case).

B. Protocol Scalability

We now look at communication overhead of the protocols in terms of their control messages. We consider the same 25-node random network scenario as before and show the overhead for each of the protocols in Table 5(a). The overhead for the single channel case serves as a baseline as it represents the routing protocol (OLSR) overhead in a single-interface, single channel network. LCAP overhead is marginally higher than the single channel. This is because of two reasons: (1) although in our implementation HELLO messages in LCAP neighbor discovery also serve OLSR, we can see it differently as LCAP piggybacking on OLSR messages (specifically, HELLO messages), thus there is no additional overhead with LCAP; (2) in a multi-radio network, HELLO messages need to be sent multiple times on different interfaces, which explains the increase in overhead messages compared to single interface, single channel case. Finally, ADC has about 60% greater overhead compared to LCAP, primarily because of the negotiation-based approach taken by ADC. Also given that all control traffic with ADC is carried on a common channel, higher control overhead means that the common channel has lower

(a) Channel Utilization and Connectivity   (b) Aggregate Link Layer Throughput   (c) Random Traffic

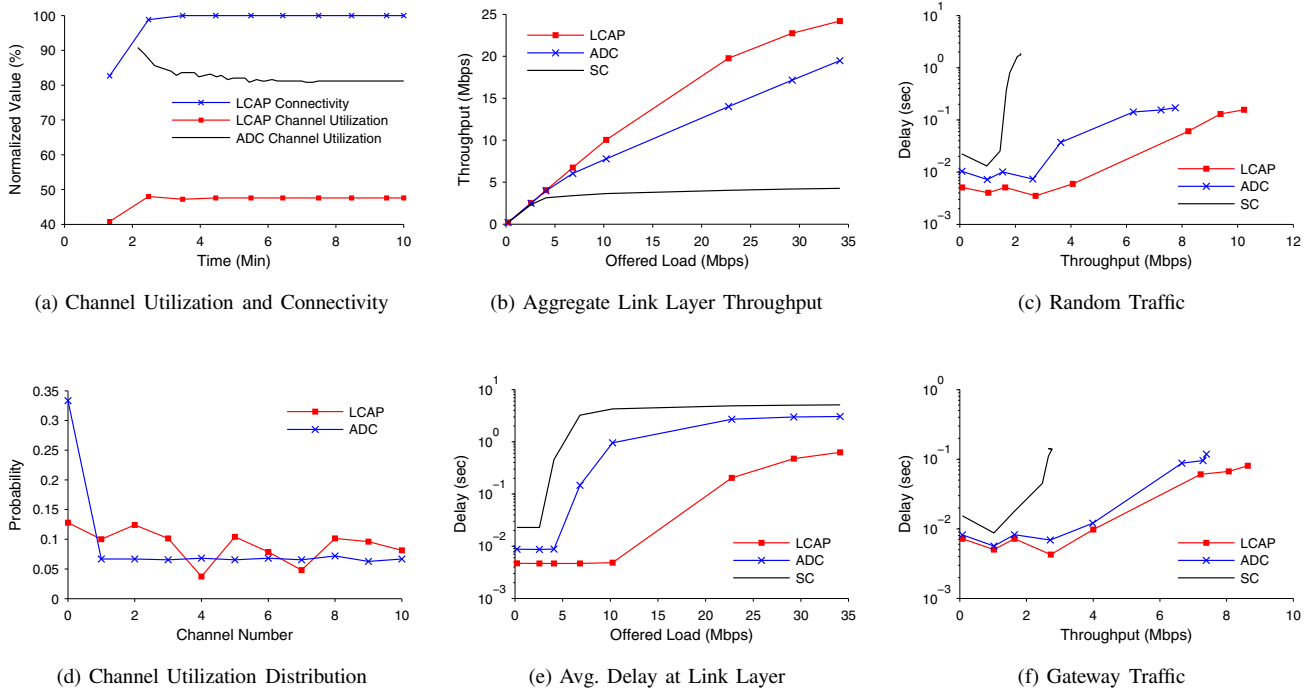(d) Channel Utilization Distribution   (e) Avg. Delay at Link Layer   (f) Gateway Traffic

Fig. 4. (a, d) Channel utilization, connectivity, protocol convergence and (b, e) link layer performance with LCAP and ADC protocols in a 802.11-based multi-radio mesh network. (c, f) Throughput and delay performance of LCAP, ADC and single channel case with two types of multihop traffic patterns: random and gateway.

Protocol	Control Overhead (msgs/node)
ADC	4486
LCAP	2777
SingleChannel	2378

(a)

NetworkSize	Control Overhead (msgs/node/s)
16	0.76
32	0.78
64	0.77
128	0.77

(b)

Fig. 5. LCAP protocol scalability in terms of communication overhead relative to ADC and single channel case.

available bandwidth for data traffic, essentially reducing the effective number of available channels. We also looked at the impact of increasing the network size on LCAP protocol overhead (not including the routing protocol overhead) while having the same node density. As is evident from Table 5(b), the LCAP overhead is pretty much constant regardless of the network size, which demonstrates its scalability property.

C. Adaptivity

Here we consider the adaptivity of LCAP to network topology changes and the presence of external interference that affects the usability of available channels. Fig. 6 shows the results. In all cases, the topology change or external interference event happens at time 800s. When a new node joins the network (Fig. 6(a)), connectivity is temporarily affected briefly because the new node and its neighbors may not share a common channel to begin with, but we can see that LCAP quickly

resolves this situation. The channel utilization (interference) metric worsens a bit because the new node forces reuse of fewer set of channels to ensure its connectivity. The opposite happens when a node fails, as seen from Fig. 6(b). Finally, we also studied the impact of external interference that affects spatio-temporal availability of usable channels. We model external interference via jamming of a subset of channels for a specified time period in a circular region of radius 300m (compare with 1000m x 1000m field), making those channels unusable during that period. This causes LCAP to adapt the channel assignment to ensure connectivity (Fig. 6(c)). There is an interesting point to note about LCAP's adaptation mechanism. The interference (channel utilization) metric gets better after external interference causes a disruption compared to its prior value, which happens because of the resumption of exploration (probability update process at each of the affected nodes) and show that such disruptions can in fact be beneficial
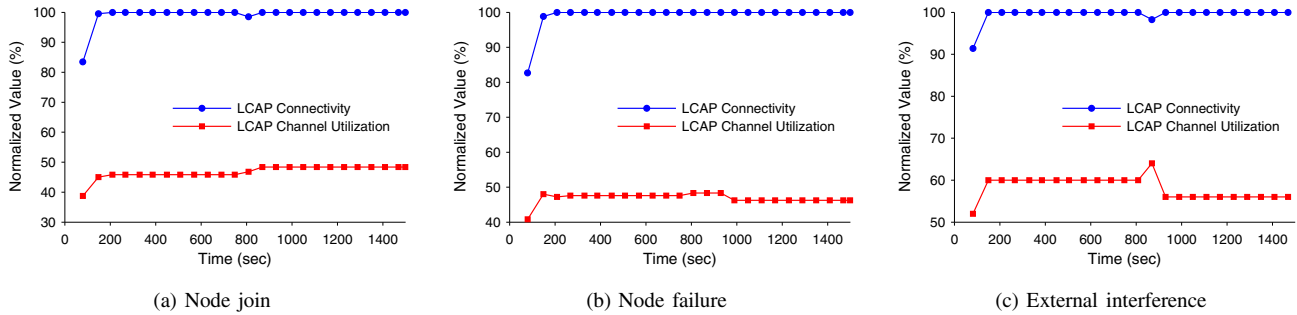
Fig. 6.   Adaptive behavior of LCAP in presence of node join, node failure and external interference.

to the channel assignment.

VIII. Conclusions and Future Work

In this paper, we have developed LCAP, a distributed multi-radio channel assignment protocol for mesh networks. LCAP takes a fundamentally different approach from existing protocols by using a probabilistic channel adaptation mechanism based on learning automata. This allows nodes to autonomously determine their channel allocation aided by a novel neighbor discovery mechanism that is based on channel quieting and allows neighbors to find each other even when not using a common channel. We have presented a prototype implementation of the neighbor discovery module and extensive simulation-based evaluation of LCAP relative to the state-of-the-art ADC protocol. Our results show that LCAP provides significant improvements in channel utilization and network performance (up to 40%) while being more scalable (with 60% less overhead) and adaptive to factors such as external interference. In on-going work, we are completing the LCAP implementation on a multi-radio mesh network testbed over which we plan to conduct extensive experimental evaluations.

References

[1] A. Raniwala and T. Chiueh. Architecture and Algorithms for an IEEE 802.11-Based Multi-Channel Wireless Mesh Network. In *Proc. IEEE Infocom*, 2005.
[2] E. Knightly. Multi-Tier Multi-Hop Wireless: The Road Ahead, Oct 2007.
[3] V. Bahl. A Crash Course in Mesh Networking. Infocom'07 Tutorial.
[4] K. Ramachandran et al. Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks. In *Proc. IEEE Infocom*, 2006.
[5] IEEE Std. 802.11h. http://standards.ieee.org/getieee802/download/802.11h-2003.pdf, 2003.
[6] Atheros. New Worldwide Regulatory Requirements for Wireless LANs, May 2008. White paper.
[7] M. Buddhikot. Understanding Dynamic Spectrum Access: Models, Taxonomy and Challenges. In *Proc. IEEE DySPAN*, 2007.
[8] P. Kyasanur and N. Vaidya. Routing and Link-layer Protocols for Multi-Channel Multi-Interface Ad Hoc Wireless Networks. *ACM SIGMOBILE MC2R*, 10(1):31–43, Jan 2006.
[9] C. Chereddi, P. Kyasanur, and N. Vaidya. Net-X: A Multichannel Multi-Interface Wireless Mesh Implementation. *ACM SIGMOBILE MC2R*, 11(3):84–95, 2007.
[10] H. Wu et al. Distributed Channel Assignment and Routing in Multiradio Multichannel MultihopWireless Networks. *IEEE JSAC*, 24(11):1972–1983, Nov 2006.
[11] B. Ko et al. Distributed Channel Assignment in Multi-Radio 802.11 Mesh Networks. In *Proc. IEEE WCNC*, 2007.
[12] K. Xing et al. Superimposed Code Based Channel Assignment in Multi-Radio Multi-Channel Wireless Mesh Networks. In *Proc. ACM MobiCom*, 2007.
[13] A. Subramanian et al. Minimum Interference Channel Assignment in Multi-Radio Wireless Mesh Networks. *IEEE Transactions on Mobile Computing*, 7(12):1459–1473, Dec 2008.
[14] L. Gao and X. Wang. A Game Approach for Multi-Channel Allocation in Multi-Hop Wireless Networks. In *Proc. ACM MobiHoc*, 2008.
[15] G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed Systems: Concepts and Design*. Addison-Wesley, fourth edition, 2005.
[16] M. Genetzakis and V. Siris. A Contention-Aware Routing Metric for Multi-Rate Multi-Radio Mesh Networks. In *Proc. IEEE SECON*, 2008.
[17] A. Giannoulis, T. Salonidis, and E. Knightly. Congestion Control and Channel Assignment in Multi-Radio Wireless Networks. In *Proc. IEEE SECON*, 2008.
[18] J. Camp et al. Measurement Driven Deployment of a Two-Tier Urban Mesh Access Network. In *Proc. MobiSys*, 2006.
[19] V. Brik et al. A Measurement Study of a Commercial-grade Urban WiFi Mesh. In *Proc. Internet Measurement Conference (IMC)*, 2008.
[20] R. Wiggins. Myths and Realities of Wi-Fi Mesh Networking, Feb 2006. Yankee Group Report.
[21] K. S. Narendra and M. A. L. Thathachar. *Learning Automata: An Introduction*. Prentice Hall, 1989.
[22] P. Nicopolitidis et al. Distributed Protocols for Ad Hoc Wireless LANs: A Learning-Automata-based Approach. *Elsevier Ad Hoc Networks Journal*, 2004.
[23] T. Joshi et al. SARA: Stochastic Automata Rate Adaptation for IEEE 802.11 Networks. *IEEE Transactions on Parallel and Distributed Systems*, 19(10), Oct 2008.
[24] M. Heusse, F. Rousseau, R. Guillier, and A. Duda. Idle Sense: An Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs. In *Proc. ACM Sigcomm*, 2005.
[25] J. Pasquale. Problems of Decentralized Control: Using Randomized Coordination to Deal With Uncertainty and Avoid Conflicts. 2001.