

# A Multi-radio 802.11 Mesh Network Architecture

Krishna Ramachandran · Irfan Sheriff ·  
Elizabeth M. Belding · Kevin C. Almeroth

Published online: 19 March 2008  
© Springer Science + Business Media, LLC 2008

**Abstract** The focus of this paper is to offer a practical multi-radio mesh network architecture that can realize the benefits of multiple radios. Our architecture provides solutions to challenges in three key areas. The first is the construction of a *split wireless router* that enables *modular* wireless mesh routers to be constructed from commodity hardware. The second is the design of a centralized channel assignment algorithm that considers the inter-dependence between channel assignment and routing in order to create high-throughput channel-diversified routes. Third is the design and implementation of several communication protocols that are necessary to make our architecture operational. Our system is comprehensively evaluated on a 20-node multi-radio wireless testbed. Results demonstrate that our architecture makes feasible the deployment of large-scale high-capacity multi-radio mesh networks built entirely with commodity hardware. Our implementation is available to the community for research and development purposes.

---

This work was completed while Krishna Ramachandran was a PhD student at UCSB.

---

K. Ramachandran  
Citrix Online, 6500 Hollister Ave,  
Santa Barbara, CA 93117, USA  
e-mail: krishna.ramachandran@citrix.com

I. Sheriff (✉) · E. M. Belding · K. Almeroth  
Department of Computer Science, University of California,  
Santa Barbara, CA 93106, USA  
e-mail: isheriff@cs.ucsb.edu

E. M. Belding  
e-mail: ebelding@cs.ucsb.edu

K. Almeroth  
e-mail: almeroth@cs.ucsb.edu

**Keywords** multi-radio · wireless mesh network · architecture · channel assignment

## 1 Introduction

Static multi-hop wireless networks, or “mesh” networks, are seeing prolific deployment. In the US alone, there are 146 working wireless mesh deployments that provide *metro-scale* wireless connectivity.<sup>1</sup> A capacity problem exists in these networks because 802.11 radios, when in each other’s carrier-sense range, interfere when simultaneously transmitting [8]. This problem is severe enough to prevent mesh networks from effectively handling a large number of users and covering large geographic areas.

Fortunately, 802.11 provides multiple orthogonal channels. Mesh routers can be equipped with multiple radios. By tuning the radios to orthogonal channels, the routers can communicate simultaneously with minimal interference. Therefore, the capacity problem can be alleviated.

This paper offers a *practical* system architecture that can realize high-capacity wireless mesh networks. We set two goals in its design. First, we must be able to build modular wireless mesh routers that can be used to construct *easily extensible* wireless mesh networks. This goal is inspired by the state-of-the-art in wired router hardware where a single router unit can support different line card technologies, such as fiber, Ethernet, or ATM. Modularity will enable wireless mesh routers to

---

<sup>1</sup>MuniWireless September 2006 Update, <http://muniwireless.com/municipal/1359>.

be equipped with heterogeneous wireless technologies, such as 802.11 and WiMAX, which can be *mixed* in various configurations. Furthermore, a modular design will allow additional radios to be added to existing routers when it becomes necessary to scale network capacity. Old radios can be easily replaced with newer technologies without changing the entire router unit. This flexibility can significantly reduce hardware costs. To the best of our knowledge, there exists no wireless router architecture that satisfies this goal.

Second, it is not only enough to have multiple radios, but it is critical that channels are assigned intelligently to them. This step is essential because channel assignment and routing are inter-dependent: channel assignments to links influence which routers are neighbors, the reliability and bandwidth of the links between routers, and also the interference relationship between the mesh links; this clearly impacts routing. Careless assignment can result in poor routes, which can be detrimental to network capacity. A channel assignment solution, therefore, must assign channels to mesh links such that high-throughput, channel-diversified routes are available in the network. Although theoretical solutions [1] have been proposed to meet this goal, they are impractical because they require network-wide coordinated link scheduling. In this paper, we describe our work to create a rigorous design, implementation, and evaluation of a high-capacity, modular, multi-radio mesh network architecture, which we call *Mcube*, that meets the above two goals. We satisfy the first goal with the *split wireless router*. A split wireless router is composed of multiple physically separated processing nodes, each equipped with a radio. The split router also alleviates the detrimental self-interference problems of board cross-talk, near-field effect, and radiation leakage that can occur between commodity radios [4, 7, 18]. This is because physical separation of radios, which is possible in a split router architecture, permits the radios to operate with reduced inter-radio interference.

With our split router in place, the next challenge is how to best accomplish channel assignment. We decompose this monolithic function into three steps: collecting information about the network topology; executing our proposed algorithm, the *topology and interference aware channel assignment algorithm (TIC)*; and disseminating channel assignments to mesh routers. Our goal in solving this particular problem is to design protocols and systems that minimize the impact of each of these steps on the network, e.g. avoid disrupting active flows, and avoid network partitioning during channel assignment.

In order to validate our proposed architecture, we have implemented a fully functional 20-node *Mcube*

network consisting of 46 802.11a/b radios. This testbed spans five floors of a typical office building. Our evaluation focuses on multiple aspects of *Mcube*, such as its performance gains in the presence of different traffic patterns, and the impact of short-term variations in link characteristics on system performance.

Our evaluation results indicate that the *Mcube* makes feasible the deployment of high-capacity modular multi-radio mesh routers built entirely using commodity hardware. An 802.11a dual-radio split router is able to forward aggregate TCP traffic over 15 Mbps. In contrast, a single-unit multi-radio router is able to operate at only 2 Mbps because of inter-radio interference. Compared to two channel assignment schemes, TIC's channel selection technique delivers TCP performance improvement in the 29–100% range.

Specifically, the contributions of this paper are as follows: to the best of our knowledge, *Mcube*'s TIC algorithm is the first *practical* channel assignment algorithm that considers the inter-dependency between channel assignment and routing during channel selection. The split wireless router is the first router architecture to enable the construction of modular, high-capacity multi-radio mesh networks. Finally, we offer a comprehensive performance study of our *Mcube* architecture in a large-scale multi-radio mesh testbed setting.

## 2 Related work

There exists a vast amount of research that focuses on the capacity problem in wireless mesh networks. The theoretical underpinnings of capacity maximization in multi-radio wireless mesh networks has been extensively studied [11, 12]. These solutions require network-wide coordinated packet scheduling in order to successfully operate, which make them impractical. Draves et al. use redundant channel assignment for a testbed-based evaluation of the WCETT metric [7]. We show in our evaluations that redundant channel assignment leads to sub-optimal network performance.

Raniwala et al. propose sophisticated centralized and distributed algorithms that assign channels on a per-flow basis to adapt to changing load conditions [17]. These two solutions require that anticipated traffic loads and the routes traversed by flows be known before channel assignment occurs. Because channel assignment and routing are inter-dependent, we note that it is challenging to predetermine the paths that will be used after channel assignment. The authors evaluated the technique over a prototype testbed spanning 9 nodes across two rooms.

Cherreddi et al. [5] propose system extensions for linux kernel to support channel switching for a multi-radio network. However, they do not look at channel assignment and system evaluation of a multi-radio mesh deployment.

Several solutions aim to assign channels in order to minimize interference between the mesh links [14, 16, 20]. TIC also reduces interference between mesh links. However, there is one key difference: TIC assigns channels in order to create high-throughput, channel-diversified paths between routers and gateways; the interference-aware solutions fail to achieve this goal.

In addition to minimizing interference between mesh links, the channel assignment solution proposed by us in an earlier work [16] assigns channels in order to minimize interference with external networks. We believe TIC can be easily extended to support the external interference detection technique used in their solution. Alicherry et al. [1] mathematically solve the interdependence between channel assignment and routing. Their solution requires that the mesh links support interference-free scheduling of each packet. This requirement makes their solution impractical. To the best of our knowledge, our proposed solution is the first *practical* approach that considers the inter-dependence between channel assignment and routing during channel assignment. We demonstrate its practicality on a large-scale multi-radio mesh testbed.

A promising alternative to using multi-radio routers is to equip a router with a single radio that either can operate on multiple simultaneous channels [9, 19] or is capable of rapid switching between channels on the order of microseconds [2, 13]. Such radios are as yet unavailable.

### 3 Design goals

#### 3.1 Modular network architecture

Our goal to incorporate modularity and flexibility in the construction of wireless mesh networks is inspired by the state-of-the-art in wired networks. Commercial wired routers from router vendors, such as Cisco Systems and Juniper Networks, ship with a modular line card architecture. Such an architecture provides the flexibility to deploy a router that supports different physical/MAC layer technologies, such as ATM, Fiber, and Ethernet.

We want the ability to build wireless mesh routers from *commodity hardware* that can offer similar modularity and flexibility. For example, we might want to deploy a wireless mesh router that uses heterogeneous

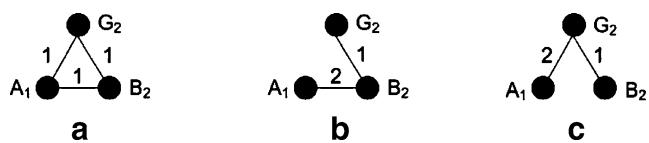
technologies, such as WiMAX and 802.11n, within the mesh, and Bluetooth for client access. The motivation for this is obvious: the growing user demand and expanding coverage coupled with the impressive pace of innovation in wireless technologies makes old technologies obsolete very quickly and necessitates newer and better technologies to be integrated into existing deployments.

#### 3.2 High-capacity provisioning

To take advantage of multiple channels, the radios in a mesh router need to be configured to operate on orthogonal frequencies. To achieve this requirement, three objectives must be satisfied. First, for a link to exist, the two end-point radios on the link must be tuned to the same channel. Currently available radios cannot switch between channels on micro-second timescales. This inability precludes per-packet channel switching. Therefore, a link is configured to operate on a particular channel for a period of a time on the order of several minutes or hours.

Second, mesh links in carrier sensing range of each other should be tuned to orthogonal channels so that they can transmit simultaneously with minimal interference.

Our third objective has to do with the interdependence between channel assignment and routing. To motivate this point further, consider the simple network illustrated in Fig. 1. Here, nodes  $G$  and  $B$  are dual-radio routers, and  $A$  is a single-radio router. The number of radios is indicated as a subscript with the router name.  $G$  is the gateway. Assume that each link in this topology has unit cost. Figure 1a illustrates the connectivity when all radios are tuned to channel one. Figure 1b and c illustrates two alternate channel assignments. In Fig. 1b, the network results in a higher cost route to  $G$  from  $A$  whereas the network in Fig. 1c optimizes the routes to both routers. Careless channel assignment can adversely impact the quality of routes between the routers and gateways. Channel assignment, therefore, should be intelligently performed so that high-throughput channel-diversified routes are available in the mesh.



**Figure 1** Connectivity with different channel assignments

### 3.3 Avoiding disruption during channel assignment

Channel tuning in an 802.11 radio is not an instantaneous operation. The task of tuning to a new frequency requires that a software application first sets appropriate hardware registers; then the base-band processor is tuned to the new frequency; and finally a series of MAC-layer control-packet exchanges need to be exchanged in order to re-associate the radio with the mesh. Packets scheduled for transmission are queued during the tuning operation to avoid unnecessary packet loss.

The actual process of notifying radios of their channel assignments and then scheduling the task of switching channels is complex. The frequency switch must occur in a well-coordinated, time-synchronized manner. Otherwise, routers can become disconnected.

Another consideration is that routing protocol state maintained by the mesh routers likely becomes stale after channel switching. This is because the neighbor set is likely to be different on the new channel. Therefore, existing routes may no longer be valid after channel assignment. The routing protocol needs to handle such cases appropriately.

## 4 Mcube design

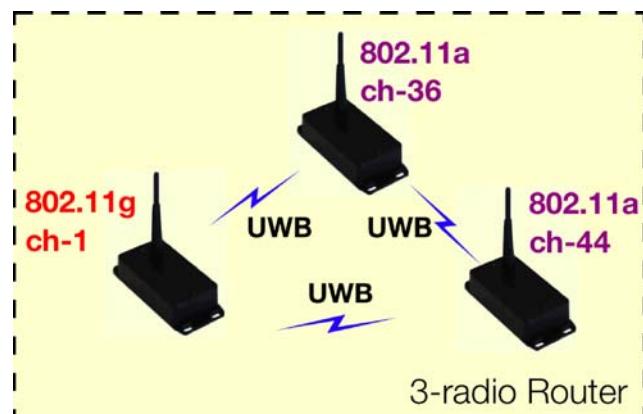
Mcube consists of two architectural components: the *split wireless router* and the *channel management server* (CMS). The CMS co-ordinates with each split router to select and assign channels.

### 4.1 Split wireless router

#### 4.1.1 Hardware architecture

The hardware architecture of a split wireless router is motivated from the problem of self-interference. Each radio in a split wireless router exists on a separate processing node. We term a radio and its processing node as a radio unit (RU). Figure 2 illustrates a 3-radio router consisting of three RUs. The nodes are connected to each other via a backhaul network. A packet that needs to be sent by an adjoining RU is sent over the backhaul to that RU, which then transmits it over the wireless medium. In the figure, the RUs are connected via an ultra wide band (UWB) backhaul. We connect the RUs in our testbed using a 100 Mbps switch.

The above architecture alleviates the self-interference problem in commodity 802.11 radios [7, 18].



**Figure 2** Split wireless router hardware architecture. The figure illustrates a 802.11 router with three radio units (RUs) where each RU is tuned to an orthogonal channel

Self-interference occurs because commodity radios are susceptible to the near-field effect [4], inter-radio board cross-talk, and radiation leakage [18]. The near-field effect is because of radio propagation characteristics [4], and the remaining causes are due to hardware imperfections.<sup>2</sup> Raniwala and Chiueh [17] use antenna separation between radios to reduce the effect of interference. However, we found that this technique is not effective when there are more than two radios per node operating on the same frequency band.

We empirically observed that with commodity 802.11a radios installed on a single processing unit and one inch separation between the radios, the throughput obtained as a percentage of the expected throughput is only 49% with a 40 MHz band separation between the channels and less than 75% with the maximum band separation of 625 MHz.

A small physical separation of approximately 0.5 meters or more, which is easily achievable with the split router architecture, effectively alleviates the self-interference problem. With some physical separation, energy leaked because of board cross-talk and radiation leakage becomes too weak to cause interference. Physical separation also reduces the near-field effect problem [4]. Through empirical measurements, we observed that the throughput obtained with simultaneously transmitting 802.11a radios that are separated by 0.5 meters and at least a 40 MHz band separation is greater than 90% of the expected throughput. Therefore, our split router architecture enables construction

<sup>2</sup>Some mesh hardware vendors, such as Bel Air networks, claim to have addressed the hardware imperfections using specialized hardware. Unfortunately, we do not have access to their hardware because of their prohibitive cost. Hence, we are unable to say conclusively about the effectiveness of their solution.

of multi-radio routers equipped with radios of the same band.

#### 4.1.2 Software architecture

We set two goals in the design of the split router's software architecture. First, each split router should appear as a single-unit router equipped with multiple interfaces. This abstraction is important; otherwise, routing protocols and network management tools, would require modification to recognize the individually visible RUs as belonging to a split wireless router. Our second goal is to support the operation of existing software, such as routing protocol implementations, without modification.

We satisfy our two goals using the software architecture illustrated in Fig. 3. One RU of a split wireless router is denoted as the *designated RU*. Software, such as the routing protocol and network management tools, are hosted exclusively on the designated RU. The *hardware abstraction layer* is executed by each RU in a split wireless router. It operates in kernel-space in order to expose the discovered RUs as local interfaces to any user-space applications. The discovery of RUs occurs dynamically as follows. Each RU in a split wireless router periodically broadcasts its identity over the backhaul in order to advertise its presence. In our implementation, an advertisement is broadcast every minute. The list of RUs is maintained as soft-state and times out every 3 min.

The hardware abstraction layer also exposes two basic forwarding primitives—for unicast and broadcast transmissions—that are essential for software, such as routing protocols, to operate without modification on a split wireless router. The unicast forwarding primitive ensures that packets destined to a next hop router are correctly forwarded over a split wireless router's backhaul to an adjoining RU, which can then transmit the packet over the wireless medium to the next hop.

To support this forwarding function, all RUs in a split router periodically exchange their observed neighbor set over the backhaul network. In our implementation, this exchange occurs every second. Each node maintains the neighbor information as soft-state. This state expires after a neighbor timeout period, which we set to 3 s. Similarly, the broadcast forwarding primitive ensures that packets that require mesh-wide dissemination are broadcast and received by each RU.

#### 4.2 Channel selection and assignment

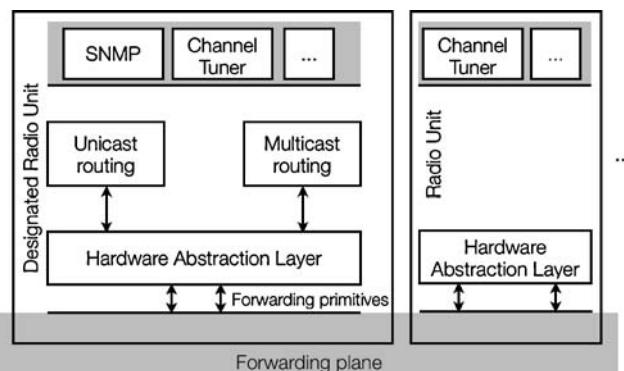
Channel selection in the Mcube architecture is performed using our Topology and Interference-aware Channel selection algorithm (TIC). In order for TIC to create high-throughput, channel-diversified routes, it needs to consider the impact of channel assignment on the network connectivity, which will in turn influence the route choice. Therefore, for TIC's operation, the topology needs to be discovered. We define topology discovery as the identification, for each router in the mesh network, of the *band-specific set of neighboring routers* and the measurement of the *quality of the link* to each of these neighbors. It is critical that band-specific discovery occur because the network topology depends on the physical layer band.

##### 4.2.1 Default channel creation

Band-specific topology discovery requires that the mesh radios be reconfigured to operate on a common channel for each supported band. This requirement, however, results in a critical consideration: what effect does topology discovery have on active flows in the mesh network at the time of invocation? Presumably, active flows can be diverted over a band-specific common channel until the topology discovery completes. However, this may adversely influence the topology discovery results in case the traffic load is high.

Therefore, we adopt the following strategy: we mandate that each mesh router designate one of its radio that is of the same physical layer type throughout the mesh as a *default radio*. This radio is switched to a *default channel* orthogonal to the one used for topology discovery. This configuration results in a single-radio mesh. Active flows are then redirected over this mesh. Flow redirection is stopped after channel assignment completes.

The single-radio mesh creation occurs through the network-wide broadcast of a DEFAULT-SWITCH message that is issued by the CMS. A DEFAULT-SWITCH message contains the channel number, which is either selected by the network operator or is randomly chosen



**Figure 3** Split wireless router software architecture

by the CMS. When the *channel tuner* module at a mesh router receives this message, it first broadcasts the message multiple times (5 in our implementation) on each of the router's radios. Multiple broadcasts are used for redundancy. The *channel tuner* then tunes the default radio to the selected channel.

Note that we assume the presence of at least one radio with a common physical layer type at each node on the mesh. For instance, in a situation where some nodes only have 802.11a radios, others only 802.11b radios, and a few others have radios of both kinds, the above technique will fail.

#### 4.2.2 Topology discovery

Once the single-radio mesh is created, it is time for topology discovery. The CMS notifies each mesh router to tune its *non-default radios* to the channel on which a band's topology is to be discovered.<sup>3</sup> The notification is sent over the single-radio mesh.

In order to discover the network topology, each router measures the link quality to its neighbors using the expected transmission time (ETT) [7] metric. ETT is an estimate of the time to transmit a packet on a link. It is derived from the link's bandwidth and loss-rate. We use packet pair probing [10] to estimate bandwidth and the expected transmission count (ETX) [6] for loss rate. Section 5 describes our implementation of ETT and the period of estimation in more detail.

#### 4.2.3 Channel selection

Our algorithm, called the topology and interference-aware channel selection algorithm (TIC) is executed by the channel management server (CMS). TIC uses the Dijkstra shortest-path algorithm to discover frequency-diversified routes between the gateway and routers that have co-located access points (APs). In this paper, we refer to a joint AP and mesh router unit simply as an AP. While discovering a route, the route's quality is evaluated using the weighted cumulative estimated transmission time metric (WCETT) [7]. The WCETT of a route is an estimate of the time a packet will take to traverse that route. The estimate is computed using the bandwidths, reliabilities, and channel assignments of all links on the path. Due to space constraints, we omit a detailed description of this metric. Other multi-radio routing metrics [21] can be used instead of

WCETT. TIC supports an alternative metric as long as the alternative can guarantee that it (1) satisfies the Dijkstra constraint that the cost increases with increasing hop count, and (2) discovers high-throughput channel-diversified routes.

TIC captures the interference relationship between mesh links using the well-known conflict graph model [8] so that it can assign interfering mesh links to orthogonal channels. Interfering mesh links can be identified using two approaches. The first approach is to use neighbor connectivity information to construct the conflict graph. This approach has been widely adopted in past work [16, 17]. A second more accurate approach is to use the pair-wise broadcast probing approach proposed by Padhye et al. [15]. Mcube supports both approaches. For the first approach, topology information collected during the topology discovery phase can be used to construct the conflict graph. To support Padhye's approach, the pair-wise broadcast probing can be invoked immediately after topology discovery completes.

**Algorithm** TIC is summarized in Algorithm 1. The input to the algorithm is a list of APs, the conflict graph (CG), and the mesh topology. The Dijkstra search begins in Line 3 by considering the AP at the head of the list,  $P$ . It then begins a neighbor search from the current router under consideration (the gateway in the first Dijkstra run) to evaluate the cost of reaching its neighboring routers via one of its radios (Lines 9–21). When considering a neighbor of the current router, TIC chooses a non-conflicting *channel* for the neighbor link. This channel is simply any channel that is *not already selected* in the conflict graph (Line 11). If all channels are selected by neighboring links, a channel is randomly selected (Lines 12–14).

Upon choosing the link's channel, TIC evaluates the cost of the path to the neighboring router given this link and its current channel choice (Line 15). If the cost of this route is lower than any previously discovered route, TIC visits the neighbor and then *tentatively* sets the link to the chosen channel (Lines 16–20). The channel selections are not finalized until the least cost route to the destination is found. The neighboring router is then added to the priority queue,  $PQ$  (Line 19).

Once all the neighbors of the current router are considered, the Dijkstra search advances the destination search by considering the least cost router (Line 23). If this router is the destination, the search for that destination ends. This condition implies that the least cost recorded path to this router is the best path found by TIC. Therefore, the tentative channels assigned to the links on this path are finalized (Line 25). The above

<sup>3</sup>Most commodity radios available today are multi-band radios. Such radios can be made to operate at a particular channel using a simple software configuration.

**Algorithm 1** TIC Algorithm

```

1: INPUT:
   P = list of APs; m = Mesh Conflict Graph; T = Neighbor
   Connectivity Graph
2: while notAllAPsFound{P} do
3:   d = head(P)
4:   n = findGateway(d)
5:   Let PQ be priority queue used in Dijkstra search from
   n to d
6:   while true do
7:     makePermanent(n)
8:     for all  $r_i$  such that  $r_i$  is a radio of n do
9:       for all  $r_j$  such that  $r_j$  is a neighbor of  $r_i$  do
10:        neigh = getRelayContainingRadio( $r_j$ )
11:        use m to select non-conflicting channel c for link
        ( $r_i, r_j$ )
12:        if c does not exist then
13:          select any random channel for link ( $r_i, r_j$ )
14:        end if
15:        cost = computePathCostWithNewLink(d,
        ( $r_i, r_j$ ), c)
16:        if cost < currentCost(neigh) then
17:          visit(neigh)
18:          setTentativeChannelForLink(( $r_i, r_j$ ),
        c)
19:          addToPriorityQueue(PQ, neigh)
20:        end if
21:      end for
22:    end for
23:    r = findMinimumInPriorityQueue(PQ)
24:    if r == d then
25:      finalizeChannelsOnPathTo(r)
26:      break
27:    end if
28:  end while
29: end while
30: permanently assign channels to radios that are not as-
  signed a permanent channel.

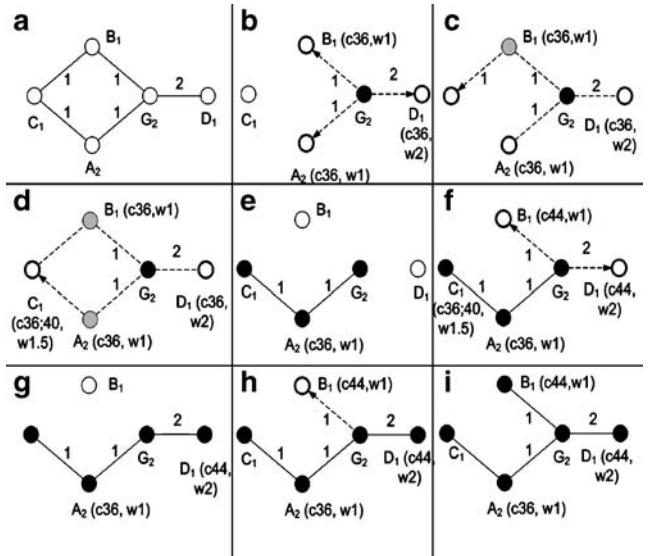
```

described process then continues until routes to all APs have been found.

**Algorithm Illustration** We illustrate TIC's operation on a simple 5-node network given in Fig. 4a. Each node has a subscript indicating its number of radios. The gateway, G, and the router, A, have two radios each. B, C, and D are single-radio routers with co-located APs to which hosts connect. The link costs are indicated in the figure. Assume all mesh links interfere with each other and that the AP list is C, D, B.

TIC starts the search for C from G by visiting each neighboring radio of G. G tentatively chooses channel 36 for the links to B, A, and D because it is an unused channel. This step is illustrated in Fig. 4b.

Once all neighboring radios are visited, TIC selects B as the least cost node (Fig. 4c). Hence, the neighbor-



**Figure 4** TIC example. The *number next to each link* indicates the link cost. The selected channels and the estimated path cost are indicated for each node using the *labels c* and *w*, respectively. A node's subscript indicates its number of radios

ing radios of B are now visited. When C is visited, it is assigned the same channel as B because B is a single radio router, and it was already assigned a channel when the search started from G.

In Fig. 4d, TIC chooses the dual-radio router, A, as the least cost node and explores all of A's neighbors. In this process, it traverses the link, AC, and assigns it the unused channel 40.

In Fig. 4e, C is the least cost visited node and its search to C ends. Now that it has found the least cost path, it finalizes the channels for the links on this path. It then invokes the Dijkstra search for each of the remaining destinations that are yet to be found. The remaining TIC steps are illustrated in Fig. 4f–i. For destination, D, a one hop path is found via link GD. TIC assigns GD channel 44 because the channels 36 and 40 are already assigned to links in its neighborhood. At this point, all of G's radios have been assigned channels. Hence, the only option is to randomly select one of the already assigned channels to reach the final destination, B. In this example, channel 44 is chosen.

**Discussion** The ordering of APs in the list provided to TIC does not influence the channel diversification of the route to each AP, i.e., the links on each AP route have the same interference relationship between one another regardless of the AP ordering. This property is because TIC utilizes the Dijkstra algorithm to discover routes. Given an unchanging set of link weights, Dijkstra's algorithm guarantees to discover the same route to a destination regardless of the order in which destinations are considered. Note that although

the channel diversification for an AP route does not change with AP ordering, AP ordering does influence the channel numbers assigned to links on routes. This effect is because the sequence in which APs are selected influences which links in the mesh are given channel assignment priority. Therefore, if multiple APs are expected to transmit data towards the gateway simultaneously, an optimal AP ordering may prevent the flows from interfering with each other. To illustrate this point further, assume in Fig. 4i that nodes  $B$  and  $D$  transmit to  $G$  simultaneously. Links from  $B$  and  $D$  to  $G$  are assigned the same channel 44. If the AP order given as input to TIC is  $B, D, C$  instead of  $C, D, B$ , the channel assignments on the link  $BG$  and  $DG$  could be 36 and 40 respectively. As a result, the flows from  $B$  and  $D$  to  $G$  will not interfere with each other.

The above discussion suggests that AP ordering can reduce inter-flow interference in the mesh. As future work, we plan to explore techniques that can exploit AP ordering to further optimize TIC's performance.

Finally, WCETT when used with Dijkstra can sometimes pick sub-optimal routes [7]. This behavior is because Dijkstra when used with WCETT does not have the property that if a whole path has the minimum WCETT value then each subpath also has the minimum WCETT. However, we believe that this sub-optimality occurs in a very small number of cases. Routing metrics [21] have been designed that address the Dijkstra-WCETT limitation. TIC supports such metrics as well.

#### 4.2.4 Channel assignment

The CMS first communicates the selected channels to each mesh router using a unicast CHANNEL-INFO message sent over the default mesh. Each router acknowledges this message and starts a timer set to 3 min. The CMS attempts communication up to five times in case of failed acknowledgments.

The CMS then issues a CHANNEL-SWITCH broadcast message in the network. Each node rapidly re-broadcasts the message over each of its radios a total of five times and then immediately deletes its cached routes. It then tunes its radios to the selected channels. This process attempts to rapidly switch all radios in the network to their respective channels, otherwise uncoordinated switching can disconnect the network, causing active flows to be disrupted. Any routes required for packet delivery are discovered using reactive route discovery after the switch completes. If the CHANNEL-SWITCH message is lost during the broadcast phase, routers that did not receive the message time out and automatically switch their radios to the assigned channels.

The CMS can eliminate the above reactive route discovery by informing APs of routes that should be used immediately after channel assignment. The CMS can compute these routes because it can infer the network connectivity that will result after channel assignment. Our implementation supports this optimization.

## 5 Implementation

Our split wireless router implementation is a Linux net-filter kernel module. The kernel module also supports ETT metric [7] collection. ETT is calculated from a link's loss rate and its bandwidth. To estimate a link's loss rate in terms of its forward delivery ratio ( $d_f$ ) and reverse delivery ratio ( $d_r$ ), a 524 byte HELLO message is issued every second by the link's end-points. The ratios are calculated from the count of delivered HELLO messages in a 10 s period. Link bandwidth ( $bw$ ) is computed by issuing *packet-pair* unicast probes of sizes 134 and 1134 bytes every 10 s. The ETT for a byte of data is then computed using the formula:  $1,134/(d_f * d_r * bw)$ . The computed ETT value is reported every 10 s to the CMS, which is co-located with the gateway in our multi-radio network deployment.

We use SRCR [3] for routing within our testbed along with WCETT [7] as the route selection metric. We set WCETT's  $\beta$  parameter to 0.5 in our evaluations, which gives equal weight to a path's channel diversification and its packet delivery delay [7].

Our CMS and TIC implementation is in Java. The CMS invokes topology discovery for a period of 5 min. We find that estimation over a 5 min period is sufficient to capture a link's long-term performance characteristics.

## 6 Evaluation

Our evaluation goal is to provide empirical results that show our proposed architecture can be used to build high-capacity modular multi-radio mesh networks. We satisfy our goal by investigating the following three aspects of the Mcube architecture on the UCSB MeshNet. UCSB MeshNet is a 20-node multi-radio 802.11 a/b network testbed deployed on five floors of the engineering building at UCSB and constructed using the split wireless router architecture.

First, we characterize the time taken by our channel assignment protocol to configure the mesh radios in our testbed. We measure this time in loaded and unloaded network scenarios.

Second, we examine the effectiveness of TIC's channel selections by investigating end-to-end throughput performance in the presence of single and multiple TCP flows in our network testbed. Third, we investigate the impact of TIC's channel re-selection period on its ability to adapt to short-term variations in link characteristics. If TIC re-selects channels over small intervals, such as every 10 min, the network topology would have to be frequently discovered, as described in Section 4.2.2. During each topology discovery phase, the mesh capacity is reduced because a single-channel mesh is used to deliver data packets. Therefore, TIC operates best in network deployments where re-selection of channels is only required over long time intervals (possibly hours). However, such a re-selection strategy essentially prevents TIC from being adaptive to short-term variations. Therefore, we use our testbed as a case study to investigate this tradeoff.

## 6.1 Testbed description

The layout of our 5-floor, 20-node testbed is shown in Fig. 5. The legend identifies the numbers of radios per router. The large number for each router indicates the router number. The subscript for each router indicates the floor on which the router is placed. Note that this notation holds true for the rest of the paper.  $G$  is the gateway and is a 4-radio split router.  $R1_2$ ,  $R2_2$ ,  $R3_3$ ,  $R6_4$  are 3-radio split routers. Each split router consists of one PC that is equipped with one Atheros AR5112 chipset 802.11a radio and one Prism 2.5 chipset 802.11b radio. In this combination, the radios do not interfere in close proximity. Each remaining radio is also an AR5112 chipset 802.11a radio installed on a separate laptop. The radio units in a split router communicate using a 100 Mbps Ethernet switch.

All radios operate in our testbed operate in “ad-hoc demo” mode. In this mode, 802.11 management frames

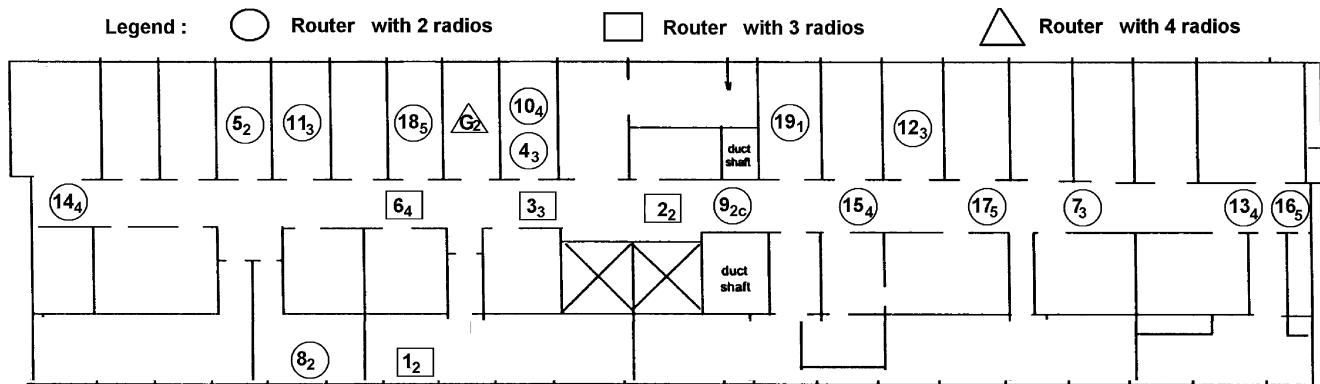
are not transmitted. The radios use auto-rate adaptation. RTS/CTS is disabled. The router placement in our testbed is well-planned in order to provide good connectivity between our 802.11a radios.

## 6.2 A simple scenario

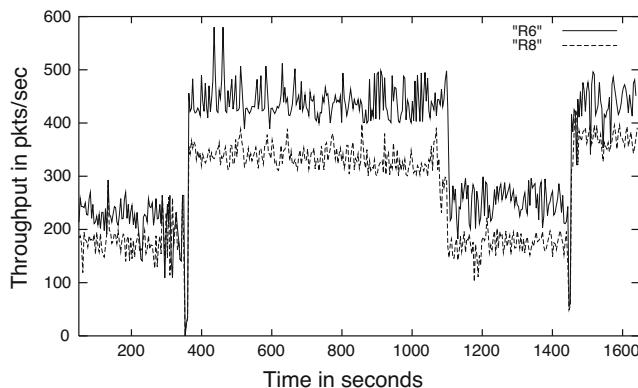
Before describing results from our set of evaluations, we first describe results from a simple scenario consisting of two flows to demonstrate the correct operation of our implementation.

Two routers,  $R6$  and  $R8$ , simultaneously send 1,500 byte packets as rapidly as possible towards the gateway throughout the experiment. At the start of the experiment, we set the mesh to operate on the default channel. In our testbed, we choose the 802.11b radios to act as the default radios. Therefore, the default mesh is an 802.11b network. Figure 6 shows the number of packets received by the gateway for the two flows. Topology discovery occurs on the non-default radios for a period of 5 min. The first channel assignment occurs at 350 s. Before 350 s, the number of packets received by the gateway per second for flows  $R6$  and  $R8$  is approximately 250 packets and 175 packets, respectively. After channel assignment, the number of packets delivered for the two flows increases. Note the period just after 350 s when the number of packets delivered for the two flows drops to zero. This outcome is because the route caches at the routers are flushed immediately after channel assignment (as explained in Section 4.2.4). There is a momentary drop in packets until the new routes are discovered by the routing protocol.

We invoke topology discovery again at time 1,130 s. The default mesh is created and the flows are redirected over the default mesh. Consequently, the number of packets reaching the gateway drops because of single-channel operation on the 802.11b band.



**Figure 5** Layout of our testbed consisting of 20 multi-radio routers



**Figure 6** A simple two flow scenario depicting channel assignment

After 300 s of topology discovery, channel assignment occurs again. This time, however, we configured the CMS to invoke the *route-installation* optimization described in Section 4.2.4. Before the CMS notifies the routers to switch their radios to the newly selected channels, the CMS notifies each mesh router of a route to the gateway. The gateway route is installed at each mesh router. When the channel switch operation is invoked, the routers skip the task of clearing their route caches. There is a momentary drop in the number of packets delivered because of buffer overflows at the router's queues because of the delay in switching channels.

### 6.3 Performance of channel assignment protocol

We performed several experiments to characterize the time taken by our channel assignment protocol to reconfigure all the radios in our testbed. The experiments were conducted in two scenarios, one where the testbed was unloaded, and the second in a loaded environment where each router initiated a 5 Mbps UDP stream to the gateway. For both scenarios, we find that our testbed radios can be reconfigured within 150 ms after the CMS initiates the first CHANNEL-SWITCH message. In the unloaded scenario, each router in our testbed switched channels after receiving the first CHANNEL-SWITCH message transmitted by a neighboring router. For the loaded scenario, the routers in our testbed typically switched on the third message transmitted by a neighbor.

### 6.4 End-to-end performance

In this section, we evaluate TIC's channel selection decision on the end-to-end throughput in the presence of single and multiple flows. To establish a baseline, we compare the throughput offered by TIC against

redundant channel assignment, used by Draves et al. [7] in their evaluation of the WCETT metric, and the *Breadth First Search Channel Assignment scheme* (BFS-CA) [16]. Redundant channel selection in our testbed setting is as follows: all the 802.11b radios are tuned to channel 1. The first 802.11a radio on each mesh router is tuned to channel 36. The routers with remaining unassigned radios tune their second 802.11a radio to channel 44. The third 802.11a radio on the gateway is randomly assigned a channel.

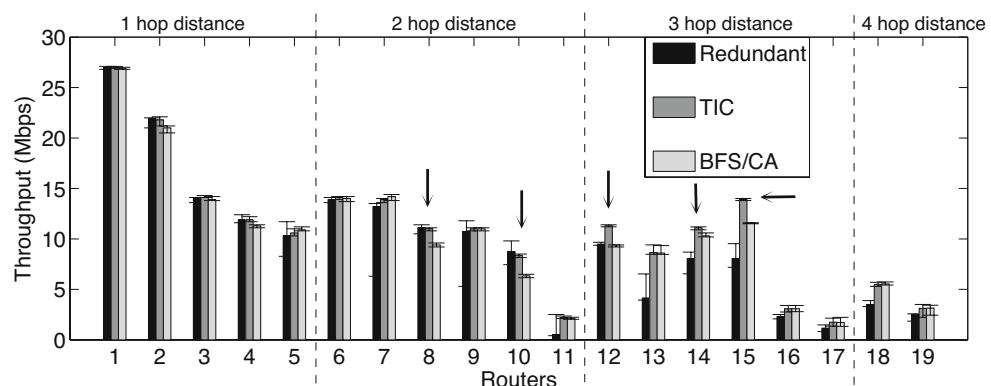
BFS-CA represents a class of several channel selection algorithms [14, 20] that minimize interference between links in a mesh network. We picked BFS-CA for our evaluation, because it, unlike other solutions in its class, channel-diversifies the AP-to-gateway routes. Therefore, any comparison with TIC would be fair. BFS-CA operates as follows: it uses a breadth first search to select channels that experience the least external interference for the mesh radios. The search begins with the highest quality (in terms of ETT) links emanating from the gateway node. As links fanning outward towards the edge of the network are progressively searched, they are assigned channels. A multi-radio conflict graph is used to prevent interfering mesh links from being assigned the same channel. The rationale behind the use of breadth first search is to give channel assignment priority to links closer to the gateway because they are likely to carry more load than links at the periphery.

With TIC and BFS-CA, channels are selected based on the network topology discovered just before the start of experiments for the single flow scenarios. The channel selection occurs again before the start of experiments for the multiple flow scenarios.

#### 6.4.1 Throughput gain in single flow scenario

To measure the throughput gain offered by TIC in the presence of single flows, three sets of TCP transfers are performed in sequence. These sets constitute one experiment run. For the first set, the channels selected by TIC are assigned to the mesh radios. All routers then initiate a 30 s TCP transfer, one at a time, to the gateway. Before a TCP transfer is established, the route tables on the sender and the gateway are flushed and a 5 s ping session is initiated so that the sender-gateway pair can discover a route to each other using reactive route discovery. The TCP session is then started. Once a TCP session ends, there is a 10 s gap before the next session is initiated. The second and third set of transfers is performed for redundant channel assignment and BFS-CA respectively. The above run is repeated a total of six times.

**Figure 7** Throughput gains with TIC in the single flow scenario. The arrows indicate instances where TIC performs better than BFS-CA



The median throughput of the TCP transfers is shown in Fig. 7. The bars in the figure are grouped by the *hop distance* of the routers from the gateway. We define the hop distance for a router as the shortest number of 802.11a hops to reach the router from the gateway. The hop distance is determined from neighbor connectivity information. The min-max bars depict the minimum and maximum throughput attained.

We make three observations from this figure. First, in general, TIC outperforms the redundant scheme. As an average of the median throughput of the single AP flows, TIC offers about 12% improvement over the redundant scheme. For the longer distances of three or four hops, TIC provides an approximately 49% throughput improvement.

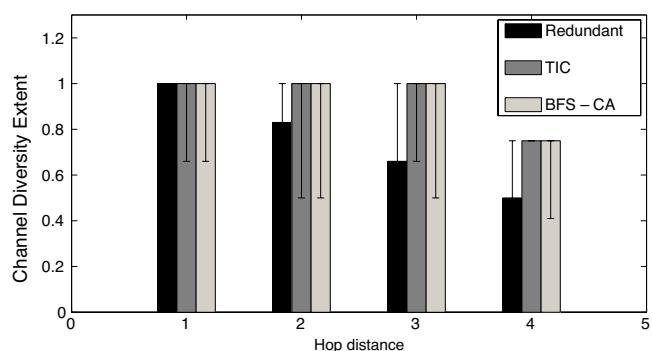
Second, TIC offers little throughput improvement for the nodes one or two hops from the gateway. This result is not surprising because the redundant scheme has the same opportunity as TIC to choose channel-diversified paths in our testbed deployment. TIC begins to outperform the redundant scheme at distances of three and four hops from the gateway. Specifically, for hop distances 3 and 4, TIC provides improvement of approximately 50% and 44%, respectively. This result suggests that TIC becomes more useful when networks grow larger and paths become longer. The channel diversity offered by TIC on the longer paths is considerably greater.

We confirm this hypothesis to be true by computing the *channel diversity extent* (CDE) of paths with TIC and the redundant schemes. The CDE of a path is defined as the ratio of the number of channels used in the path to its hop-count [16]. A path with a high CDE is generally preferred over a path with a low CDE. Figure 8 plots the median of the CDEs for all flows from routers grouped by their hop distance from the gateway. The min-max bars indicate the minimum and maximum CDE for TIC and the redundant scheme. Clearly, the flows with TIC have a higher CDE value

than with the redundant scheme. The CDE increases with increasing hop count because TIC is able to channel diversify paths in our testbed.

We observe that there is considerable variation in minimum and maximum CDE values in the redundant scheme compared to TIC. The longer bars in the redundant scheme are due to variations in link characteristics resulting in the discovery of several different paths for the TCP transfers. The number of alternate paths available with the redundant scheme is much greater than with TIC because of the increased connectivity between routers due to redundant channel assignment. On the other hand, with TIC, the TCP transfers are constrained in most cases to the paths selected by it. The result is a lower variation in the CDE in TIC. This result is also evident in Fig. 7 where the throughput variation, as indicated by the min–max bars, is smaller with TIC.

Our final observation is the performance of TIC compared to BFS-CA. In Fig. 7, BFS-CA matches TIC's performance in almost all cases except the five indicated by the line-arrows in the figure. For these five instances, TIC offered an average TCP throughput improvement of 29% over BFS-CA. BFS-CA performs poorly in these five cases because it does not consider



**Figure 8** Channel diversity extent in the single flow scenario

the inter-dependence between channel assignment and routing. TIC, on the other hand is designed to pick those channels that result in high-capacity, channel-diversified routes.

However, the fact that BFS-CA closely matches TIC's performance in the rest of the cases is worthy of discussion. This outcome is an artifact of 802.11a's propagation characteristics within an indoor environment such as ours. In most cases, 802.11a radios in our network establish connections only with radios that are placed in adjacent offices. Because of this reason, we carefully placed our multi-radio routers in order to provision for high-throughput paths between the gateway and the rest of the routers in the network. However, sparseness in 802.11a links results in TIC and BFS-CA picking the same links for channel assignment in a majority of the cases.

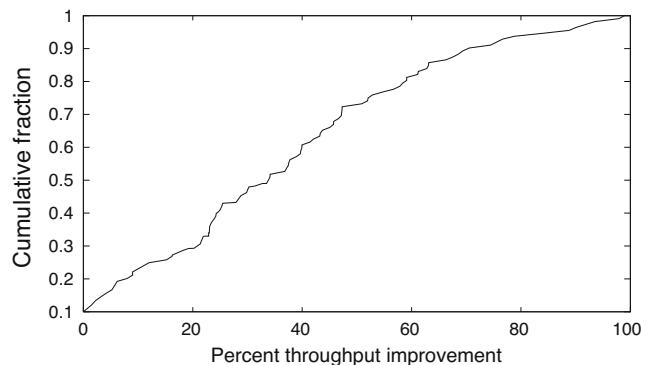
In our carefully planned testbed, TIC outperforms BFS-CA. However, we want to characterize TIC's performance vis-a-vis BFS-CA in unplanned testbed deployments. To achieve this goal, we use simulated network topologies to further our understanding.

#### 6.4.2 TIC versus BFS-CA in unplanned topologies

We next evaluate TIC with a custom-built Java simulator by using the network topology information collected from the testbed and varying the gateway position on the network. For our simulations, we use the network topology discovered by TIC in our single-flow experiments. However, instead of using the gateway position depicted in Fig. 5, we vary the location of the gateway by co-locating it with ten randomly chosen routers to create ten unplanned network topologies.

For each scenario, we feed the network topology and the gateway location to our TIC and BFS-CA implementations. Using the channels selected by the two algorithms, we modify our network connectivity to obtain connectivity graphs that would be realized if the actual channel assignments had taken place. We then compute the throughput obtainable on each AP-to-gateway route. The AP-to-gateway route is the best WCETT path chosen by executing Dijkstra on the modified connectivity graphs. Assuming a 1,500 byte packet, a route's throughput is the ratio of the packet size and the route's WCETT value.

Figure 9 plots the cumulative fraction of percentage improvement in throughput offered by TIC over BFS-CA. TIC clearly outperforms BFS-CA. If we consider the median value, TIC outperforms BFS-CA by over 34%. TIC is therefore a better channel selection algorithm. Yet it is interesting that if we consider the lower quartile, the percentage throughput improvement of



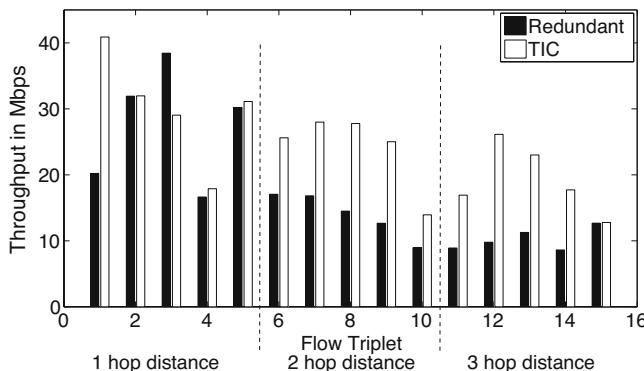
**Figure 9** Percent throughput improvement offered by TIC over BFS-CA

ferred by TIC is only up to 15%. This is because, in these cases, the majority of the routes chosen by TIC and BFS-CA are the same. We believe that this behavior is an artefact of the way our testbed is deployed. A different testbed setting could possibly result in TIC and BFS-CA selecting much different routes.

#### 6.4.3 Throughput gain in multiple flows scenario

Having established TIC's performance relative to BFS-CA in the single-flow scenario, we only resort to comparing TIC's performance against the redundant scheme for the multiple flow scenarios. For this experiment over the testbed, we choose triplets of routers at hop distances of one, two, and three from the gateway. For each hop distance, we choose five random sets of triplets. Each node in a single triplet simultaneously initiates a 5 min TCP transfer to the gateway. Before the transfers are initiated, route caches are cleared and a 5 s ping session to the gateway is initiated. The aggregate throughput at the gateway is noted after the TCP transfers complete. There is a gap of 2 min between each triplet's transmissions. The above set of experiments is performed once for TIC and once for the redundant scheme, which yields one run. A total of three runs is performed.

Figure 10 shows the average of the *aggregate throughput* attained at the gateway with TIC for each flow triplet compared to the redundant scheme. TIC clearly outperforms the redundant scheme. On average, TIC's throughput gain is over 42%. For hop distances greater than one, the median throughput improvement with TIC is over 100%. Throughput gains result due to TIC's use of routes that are more channel diversified than those with the redundant scheme. As a result, flows with TIC interfere less with each other, and, therefore, can sustain higher aggregate throughput than with the redundant scheme.



**Figure 10** Throughput gain with TIC in the presence of multiple flows

Note that for the first triplet, the redundant scheme yields less than 50% of the throughput offered by TIC. This result is in contrast to our observations in the single-flow scenario where TIC offered little throughput improvement over the redundant scheme at a one hop distance. The difference in performance can be attributed to WCETT, the routing metric we use in our evaluation. This metric selects a path based purely on its channel diversification. It does not consider the existence of other active flows that utilize the channels assigned to a path. Therefore, even if the routing protocol discovers an unloaded alternate path to the same destination, albeit of slightly lower quality, WCETT may not select it.

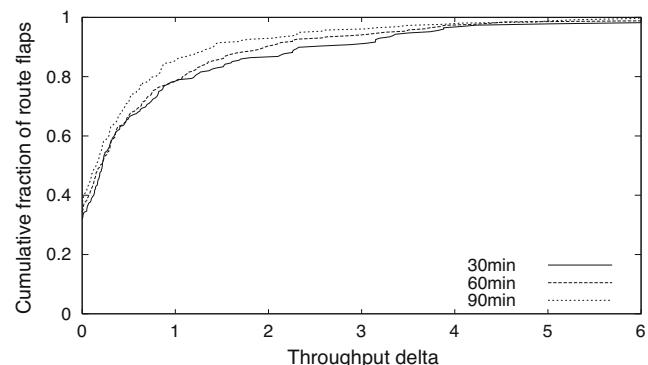
TIC, on the other hand, alters the network topology by choosing a single route for each AP that it then diversifies by assigning the links on the route orthogonal channels. Because of this choice, TIC cannot offer multiple channel-diversified routes to the same destination. Therefore, the routing protocol will likely discover routes that TIC wants it to discover. Since there is typically only one clear route choice (the one TIC decides for the AP), WCETT chooses that route.

The fact that TIC cannot offer multiple channel-diversified routes can sometimes be detrimental to its performance. For example, with the third triplet, the redundant scheme offers an approximately 32% improvement. This result is because the links from the gateway to the sources in this triplet (routers  $2_2$ ,  $3_3$ ,  $4_3$ ) were assigned the same channels by TIC because of the non-availability of radios during channel selection. Therefore, the flows from these sources interfered with each other. On the other hand, with the redundant scheme, the flow from router  $2_2$  traversed a link tuned to a channel orthogonal to the one used by links from routers  $3_3$  and  $4_3$ , therefore yielding better performance.

## 6.5 Impact of variations in link characteristics

This section investigates the adverse impact of TIC's inability to adapt to short-term variations in link characteristics. To evaluate this impact, we collected ETT statistics over a total period of 24 h from our testbed. The data was collected on three different days in order to capture day and night conditions on weekdays and weekends. In our analysis, we consider three re-selection periods: 30, 60, and 90 min. For each of these periods, we perform a static analysis with our channel selection implementation to evaluate the throughput reduction on a route due to TIC's inability to adapt to short-term link quality fluctuations.

For each re-selection period, we first compute the number of missed routes by comparing the route chosen by TIC for a destination at the beginning of the period with all the routes noted for the same destination if TIC were to be *always-adaptive*, i.e., invoked at 5 min intervals within the period. When a *route flap* occurs, i.e., a different route is observed, we compute the difference between the throughput offered by the two routes. Assuming a 1,500 byte packet, a route's throughput is the ratio of packet size and the route's WCETT value. Figure 11 plots the cumulative fraction of route flaps on the y-axis against the throughput differences noted in our analysis. We observe that for the three periods we analyzed, anywhere between 33–40% of route flaps provide no throughput improvement. These route flaps correspond to cases when the route flapped back to the route chosen by TIC at the beginning of a re-selection period. The median route flap delivers less than 0.25 Mbps improvement for the three re-selection periods considered. For the 90th percentile and the 60 min re-selection period, the improvement is less than 2 Mbps. For this route flap, in our analysis, we note the throughput with TIC at the beginning of the interval is 8.70 Mbps. If TIC were to be always-



**Figure 11** Throughput delta between TIC and an always-adaptive TIC scheme

adaptive, the throughput offered would be 10.70 Mbps. However, to be always-adaptive, TIC would have to discover the network topology every 5 min using the technique described in Section 4.2.2. Topology discovery requires that the mesh radios operate on common channels throughout the mesh. The permanent use of common channels, which is essentially the same as the redundant channel assignment scheme used in our evaluation of TIC's end-to-end performance, can result in poor mesh performance as indicated by the results presented in Section 6.4.

In conclusion, although TIC is unable to adapt to short-term variations in link characteristics, it performs well even if the re-selection of channels is done over long intervals. Although the above results are specific to our testbed setting, we believe our analysis is generally valid for static mesh deployments.

## 7 Conclusion

This paper presented Mcube, a multi-radio 802.11 mesh network architecture. Mcube's split wireless router architecture enables the construction of modular mesh networks. Mcube selects channels such that frequency diversified, high-throughput paths are available in the mesh. This goal is important because unplanned channel assignment can lead to poor routes that severely degrade mesh performance.

We foresee Mcube to be used to construct high-capacity wireless mesh networks for deployment in a city, community, or a building. As future work, we plan to extend Mcube so that it avoids channels that experience the most interference from external networks. Our current implementation is available to the community for research and deployment purposes.<sup>4</sup>

**Acknowledgements** This work was funded in part by NSF Career award CNS-0347886, NSF NeTS award CNS-0435527, and NSF CRI award CNS-0454329.

## References

1. Alicherry M, Bhatia R, Li L (2005) Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks. In: ACM MobiCom. Cologne, Germany
2. Bahl P, Chandra R, Dunagan J (2004) SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad hoc wireless networks. In: ACM MobiCom. Philadelphia, PA
3. Bicket J, Aguayo D, Biswas S, Morris R (2005) Architecture and evaluation of an unplanned 802.11b mesh network. In: ACM MobiCom. Cologne, Germany
4. Chebrolu K, Raman B, Sen S (2006) Long-distance 802.11b links: performance measurements and experience. In: ACM MobiCom. Los Angeles, CA
5. Cherreddi C, Kyasanur P, Vaidya N (2006) Design and implementation of a multi-channel multi-interface network. In: REALMAN. Florence, Italy
6. De Couto D, Aguayo D, Bicket J, Morris R (2003) A high-throughput path metric for multi-hop wireless routing. In: ACM MobiCom. San Diego, CA
7. Draves R, Padhye J, Zill B (2004) Routing in multi-radio, multi-hop wireless mesh networks. In: ACM MobiCom. Philadelphia, PA
8. Jain K, Padhye J, Padmanabhan V, Qiu, L (2003) Impact of interference on multi-hop wireless network performance. In: ACM Mobicom. San Diego, CA
9. Jain N, Das S, Nasipuri A (2001) A multichannel CSMA MAC protocol with receiver-based channel selection for multihop wireless networks. In: IEEE international conference on computer communications and networks. Scottsdale, AZ
10. Keshav S (1991) A control-theoretic approach to flow control. In: ACM Sigcomm. Zurich, Switzerland
11. Kodialam M, Nandagopal T (2005) Characterizing the capacity region in multi-radio multi-channel wireless mesh networks. In: ACM MobiCom. Cologne, Germany
12. Kyasanur P, Vaidya N (2005a) Capacity of multi-channel wireless networks: impact of channels and interfaces. In: ACM MobiCom. Cologne, Germany
13. Kyasanur P, Vaidya N (2005b) Routing and interface assignment in multi-channel multi-interface wireless networks. In: IEEE WCNC. New Orleans, LA
14. Marina M, Das S (2005) A topology control approach to channel assignment in multi-radio wireless mesh networks. In: IEEE Broadnets. Boston, MA
15. Padhye J, Agarwal S, Padmanabhan V, Qiu L, Rao A, Zill B (2005) Estimation of link interference in static multi-hop wireless networks. In: ACM/USENIX international measurement conference. Berkeley, CA
16. Ramachandran K, Belding-Royer E, Almeroth K, Buddhikot M (2006) Interference-aware channel assignment in multi-radio wireless mesh networks. In: IEEE infocom. Barcelona, Spain
17. Raniwala A, Chiueh T (2005) Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network. In: IEEE infocom. Miami, FL
18. Robinson J, Papagiannaki K, Diot C, Guo X, Krishnamurthy L (2005) Experimenting with a multi-radio mesh networking testbed. In: ICST workshop on wireless network measurements. Riva del Garda, Italy
19. So J, Vaidya NH (2004) Multi-channel mac for ad hoc networks: handling multi-channel hidden terminals using a single transceiver. In: ACM MobiHoc. Tokyo, Japan
20. Subramanian A, Gupta H, Das S (2006) Minimum-interference channel assignment in multi-radio wireless mesh networks. In: SUNY stonybrook technical report
21. Yang Y, Wang J, Kravets R (2005) Designing routing metrics for mesh networks. In: IEEE WiMesh. Santa Clara, CA

<sup>4</sup><http://moment.cs.ucsb.edu/tic/>.



**Krishna Ramachandran** is currently a Research Scientist with Citrix Online, a division of Citrix Systems, where his main research interests are in multimedia streaming systems, overlay networking, and web services. Krishna received his B.E. (2000) in Computer Engineering from Pune University, India, and his M.S. (2003) and Ph.D. (2007) degrees in Computer Science from UCSB. At UCSB, Krishna was advised by Professors Kevin Almeroth and Elizabeth Belding. Krishna's dissertation title was "Design, Deployment, and Management of High-capacity Large-scale Wireless Networks". His dissertation offers a comprehensive network architecture for building high-capacity, multi-radio wireless mesh networks.



**Elizabeth M. Belding** is an Associate Professor in the Department of Computer Science at the University of California, Santa Barbara. Elizabeth's research focuses on mobile networking, specifically mesh networks, multimedia, monitoring, and advanced service support. She is the founder of the Mobility Management and Networking (MOMENT) Laboratory (<http://moment.cs.ucsb.edu>) at UCSB. Elizabeth is the author of over 70 papers related to mobile networking and has served on over 50 program committees for networking conferences. Elizabeth served as the TPC Co-Chair of ACM MobiCom 2005, IEEE SECON 2005, and ACM MobiHoc 2007. She also served on the editorial board for the IEEE Transactions on Mobile Computing. Elizabeth is the recipient of an NSF CAREER award, and a 2002 Technology Review 100 award, awarded to the world's top young investigators. See <http://www.cs.ucsb.edu/~ebelding> for further details.



**Irfan Sheriff** received his bachelors degree in Computer Engineering from National Institute of Technology, Suratkal, India in 2002. He is pursuing his Ph.D. in Computer Science at the University of California, Santa Barbara since 2003. He is a member of the MOMENT research lab directed by Prof. Elizabeth M. Belding. Irfan's research focuses on resource control on wireless networks, QoS for VoIP and design and architecture of multi-radio wireless mesh networks. See <http://www.cs.ucsb.edu/~isheriff> for more details.



**Kevin C. Almeroth** is currently a Professor in the Department of Computer Science at the University of California in Santa Barbara where his main research interests include computer networks and protocols, wireless networking, multicast communication, large-scale multimedia systems, and mobile applications. He has published extensively with more than 150 journal and conference papers. He is also heavily engaged in stewardship activities for a variety of research outlets including journal editorial boards, conference steering committees, new workshops, and the IETF. He is a Member of the ACM and a Senior Member of the IEEE.