Distributed Channel Assignment for Multi-radio Wireless Networks

Minho Shin, Seungjoon Lee, Yoo-ah Kim

Abstract— **We consider the channel assignment problem for multihop wireless networks in which nodes have multiple interfaces. Given the number of interfaces at each node and available channels in the system, we find a feasible channel assignment to improve network performance. Even when routing is given, finding a channel assignment for optimal performance is NP-hard. We present the SAFE (Skeleton Assisted partition FrEe) channel assignment scheme, which uses randomized channel assignment in a distributed manner while maintaining network connectivity. SAFE can utilize all independent channels in the system while attempting to distribute edges sharing a particular channel evenly throughout the network. To handle topology change and incremental deployment better, SAFE decouples the channel assignment problem from routing. Our simulation results show that SAFE significantly improves network performance in terms of throughput and delay and is comparable to the best prior centralized scheme that jointly considers routing and channel assignment.**

I. INTRODUCTION

In this paper, we consider the problem of assigning a channel to each radio (or network interface) when a multihop wireless network consists of nodes with multiple radios [1–7]. Suppose that all nodes in Figure 1 have two interface cards. In this example, node A is using two independent channels: channel 1 for e_1 and channel 2 for e_2. With this channel assignment, we can use all three wireless links at the same time. If all nodes have packets to transmit, then the network throughput using the above channel assignment will be three times as high as that of the single-channel case. However, such an ideal channel assignment is not always possible. It is because the number of channels a node can use is bounded by its interface count as well as by the number of available channels in the system. For example, if A has more than two neighbors, then A cannot communicate with each of its neighbors at the same time since A has only two interfaces. The goal of this paper is to develop a scheme that uses independent channels available to wireless devices to enhance the performance of wireless networks.

One of our design goal is to *use currently available hardware and MAC protocols* without modification.

M. Shin and S. Lee are with Department of Computer Science, University of Maryland, College Park, MD 20742. E-mail: {mhshin,slee}@cs.umd.edu. Y. Kim is with Department of Computer Science and Engineering, University of Connecticut, Storrs, CT 06269. E-mail: ykim@engr.uconn.edu.
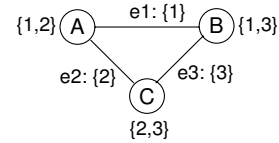
Fig. 1. In this example network, each node has two interface cards. We can assign a distinct channel to each link as shown above. With the above channel assignment, there is no interference among wireless links.

This makes the proposed scheme readily deployable using current commodity wireless equipments. We also want our scheme to be *incrementally deployable*. For this feature, the proposed scheme should inter-operate with existing upper-layer protocols, such as routing protocols. An alternate approach is to consider a joint problem of channel assignment and routing for better performance [2, 7]. With our approach, however, even when only a subset of nodes are aware of the channel assignment scheme, both aware and unaware nodes can inter-operate using a well-known routing scheme. Also, we can easily embrace later improvement in upper-layer protocols such as new routing protocols like [8].

We want our scheme to operate in *dynamic networks* such as mobile ad-hoc networks or community wireless mesh networks with frequent topology changes. Most existing channel assignment schemes focus on static mesh networks and develop centralized algorithms using the knowledge of traffic pattern and global topology [2, 6, 7]. However, such schemes are not suitable for mobile ad-hoc networks because of frequent changes in topology and traffic pattern due to mobility. Even in a community mesh network where nodes are static, some nodes may leave the system unexpectedly, which results in topology change and potential traffic pattern change throughout the network. Our scheme is independent of such change in traffic pattern and network topology and achieves high performance in dynamic scenarios.

We also want our scheme operate in a *distributed manner using local information only*, which is more suitable to dynamic environment. One difficulty in designing distributed channel assignment schemes is to maintain network connectivity. Since a node cannot use more channels than the number of interface cards, a pair of nearby nodes may use disjoint sets of channels and not be able to communicate through the wireless link that would exist in the single-channel case. Then, without careful coordination, a local channel

assignment scheme may lead to network partition. We address this issue in detail in Section IV.

This paper makes the following contributions:

- We consider throughput and delay as network performance metrics. Even when paths are given, finding an optimal channel assignment for either metric is *NP-hard*.
- We design SAFE (Skeleton Assisted partition FrEe), a distributed channel assignment heuristic, which utilizes all independent channels available in the system while maintaining network connectivity.
- Using simulations, we show that SAFE significantly improves network performance and is comparable to a constant-approximation scheme that jointly considers channel assignment and routing in a centralized manner [7].

The rest of this paper is organized as follows. We review related work in Section II and present network model and some analysis results in Section III. We describe our distributed heuristic in Section IV. We present our simulation results in Section V.

II. RELATED WORK

Many schemes have been proposed to exploit multiple channels for performance improvement in wireless networks. One class of work is to enable only one network interface card to operate in multiple channels. Such works include Multinet [9], SSCH (Slotted Seeded Channel Hopping) [10], MMAC (Multichannel MAC) [11], and multichannel CSMA [12]. However, these schemes require change in MAC protocols or capabilities that current wireless cards do not have. Also, in all the above schemes using a single interface, frequent channel switching introduces extra overhead. Our approach focuses on multi-interface scenarios and does not require modification of hardware or MAC protocols.

Multi-radio Unification Protocol (MUP) [1] uses multiple interfaces. In MUP, when a node has K network interface cards, it only uses K channels (channels $1, 2, \cdots, K$) even when there are more channels. Each node statically assigns a channel to each interface card, and when a node needs to transmit a packet, it checks the channel condition and uses the channel with the best condition at that time. Raniwala et al. [2, 13] consider the joint problem of channel assignment and routing in the context of mostly static mesh networks. Based on the assumption of the knowledge of long-term traffic load, they present a centralized heuristic [2] and a distributed algorithm [13] for throughput improvement. However, their distributed routing scheme is based on a tree rooted at a gateway, and each node has to learn 3- or 4-hop neighborhood information. Consequently, the algorithm may not work well in more dynamic networks. In contrast, our proposed scheme only uses one-hop neighborhood information and is aimed for dynamic networks. We also isolate the channel assignment problem from routing for incremental deployment.

A few recent papers study theoretical aspects of multi-radio networks. Kyasanur and Vaidya [5] extend the work of Gupta and Kumar [14] and analyze the network capacity when there are multiple channels and interfaces. They show that given a number of interfaces, the use of more channels can decrease the network throughput. Kodialam and Nandagopal [6] develop a necessary condition for the feasibility of a given link flow set and derive an upper bound on the achievable throughput and propose two greedy channel assignment schemes based on linear programming. Alicherry et al. [7] also study the joint channel assignment and routing problem assuming that traffic demands and network topology are known. They present LP formations to maximize the aggregate throughput. They also propose a centralized algorithm that assigns channels to node radios and finds routing paths. They prove that the algorithm gives a constant factor approximation. We compare SAFE with the scheme by Alicherry et al. in Section V.

III. MODEL AND METRIC FORMULATION

A. Network Model

We represent a network as an undirected graph $G = (V, E)$, where V is a set of nodes, and E a set of wireless links. (We use link and edge interchangeably). There is an edge between two nodes if and only if they are within the transmission range. A transmission over an edge may fail when transmissions over other nearby edges simultaneously occur on the same channel. We use $I(e)$ to denote the set of edges that interfere with edge e, and assume $e \in I(e)$. There exist a number of interference models that describe whether a transmission is successful or not [15–17]. Although our proposed scheme in Section IV does not depend on a particular model, we describe the *two-hop interference model* for illustration, which reasonably approximates scenarios with the IEEE 802.11 standard [16]. In this model, two edges may interfere with each other if they are within two-hop distance. In other words, two edges e and e' cannot transmit simultaneously on the same channel if they are sharing a node or adjacent to a common edge.

Ideally, we want to assign channels so that we can use all edges at any time (i.e. $I(e) = \{e\}$ for all

$e)$[1] to achieve maximum performance gain. However, since the number of interfaces at a node limits the number of channels the node can simultaneously use, such an ideal assignment is not always possible. For example, if a node has two interface cards and three neighbors, at least two neighbors should share the same channel. Also, the number of available channels in the system can limit the result of channel assignment. We assign one channel on each interface and assume that there is no hardware or system support to enable a single interface to access/switch multiple channels. Two nodes can communicate with each other only if there is a common channel assigned to two nodes.

In this paper, we consider dynamic networks where nodes can join, leave the network, or move their positions, and present a distributed channel assignment algorithm to improve network performance while satisfying these constraints (e.g, due to the interface count and available channel count).

In the following two subsections, we discuss the metrics that we use to evaluate the performance of channel assignment.

B. Throughput-based Metric

In this paper, we consider two aspects of network performance: *throughput* and *delay*. In this subsection, we focus on maximizing the total throughput over all active flows. We discuss the delay performance in the following subsection.

A *path* p is a sequence of wireless links that connect a source and a destination. We use P to denote the set of all paths that are currently in use. Depending on the path selection, a wireless link can be used multiple times for different source-destination pairs. We define the usage of edge e, u_e, to be the total number of paths in which e is used: $u_e = \sum_{p \in P: e \in p} 1$. Since an upper-level routing protocol is independent of our channel assignment schemes, we assume that all paths for currently active flows (denoted by P) are already found, and that u_e values are accordingly given.

In general, it is known to be hard to find the maximum throughput in wireless networks due to interference [18, 19]. As shown in Appendix A, it is also NP-hard to find the total maximum throughput that can be sent over a set of fixed paths P, given $I(e)$ for each e. Thus, instead of finding the optimal solution, we find a lowerbound of the maximum throughput, using the formulation similar to the one proposed by Kumar et al. (See Section 6 in [19]). Let us define T_p to be the end-to-end throughput (e.g., in kbps) of

the flow that uses path p. If we only concentrate on maximizing the global sum of throughput values, it is possible that a small number of flows keep sending packets to increase the total throughput while others cannot send any packets. To avoid such starvation for flows, we define *fairness ratio* α (≥ 1), which bounds the ratio of maximum and minimum throughput values.

Our throughput-based formulation is as follows:

$$T_{total} = \max \sum_p T_p \qquad (1)$$

subject to

$$T_{max} \geq T_p \quad \forall p \qquad (2)$$
$$T_{min} \leq T_p \quad \forall p \qquad (3)$$
$$T_{max} \leq \alpha T_{min} \qquad (4)$$
$$\sum_{e' \in I(e)} \sum_{e' \in p} T_p \leq c \quad \forall e \in E \qquad (5)$$

Constraints (2)–(4) are to bound the ratio of maximum and minimum throughput values. In an extreme case when $\alpha = \infty$ (where we do not attempt to balance the throughputs among flows), we can omit these constraints.

Constraints (5) are the interference constraint which forces that only c edge in $I(e)$ can be used at a time. It is known that any feasible solution satisfies the constraint when c is a small constant [7]. When $c = 1$ the formulation gives a lowerbound on the optimal throughput and the metric is within a small constant factor (e.g., 5) from the optimal solution. Moreover, we can find a feasible schedule to achieve the obtained throughput when $c = 1$ [19]. We use the objective value in Eq. (1) as throughput-based metric for performance comparison. We note that the metric provides a good performance indicator through simulation results in Section V.

Let us consider a special case of $\alpha = 1$, in which we can understand the objective function better as we can reduce our formulation into a simpler form. In this case of strict fairness, $T_p = T$ for all p. Therefore, Constraints (5) becomes $\sum_{e' \in I(e)} \sum_{e' \in p} T = T \sum_{e' \in I(e)} u_{e'} \leq 1$. We then obtain the maximum throughput T for each flow.

$$T = \min_e \frac{1}{\sum_{e' \in I(e)} u_{e'}} \qquad (6)$$

Therefore, to improve the throughput capacity, it is beneficial to minimize $\max_e \sum_{e' \in I(e)} u_{e'}$. In general, it is NP-hard to find an optimal channel assignment to maximize the total throughput [2].

C. Delay-based Metric

In this subsection, we consider the end-to-end delay that a flow may experience in a network. Our goal is to

---

[1]In this paper, we assume that there is no interference between independent channels. In practice, depending on the specific hardware, packet transmissions on two independent channels may interfere with each other [1].

minimize the total delay over all active flows. Another obvious delay objective is to minimize the maximum delay among flows, which we plan to consider in our future work.

Consider a flow that uses path p. For simplicity, we assume that $T_p = T$ for all p and packet lengths are all the same. Let d_e denote the delay that a packet experiences to traverse an edge e on path p. The end-to-end delay (d_p) of a packet for the flow can be obtained by summing up d_e over all edge e in path p: $d_p = \sum_{e \in p} d_e$. The total delay of all flows can be obtained as follows.

$$D_{total} \quad = \quad \sum_{p \in P} d_p = \sum_p \sum_{e \in p} d_e = \sum_e u_e d_e. \quad (7)$$

The last equality comes from the definition of u_e.

The delay d_e on edge e depends on how we schedule edge communications. Ideally, we want to find a schedule that minimizes the total delay given in Eq. (7). However, as in the case of throughput, it is NP-hard to find an optimal schedule that minimizes the total end-to-end delay (See Appendix B). Therefore, in the following we try to capture an upperbound on the total end-to-end delay over all flows.

We consider the following feasible schedule, which gives an upperbound of the optimal end-to-end delay of the schedule. Note that the total number of competing flows over edge e is $\sum_{e' \in I(e)} u_{e'}$. Consider a feasible schedule in which we schedule edges in round-robin fashion. Then the delay d_e of a packet to traverse edge e in this schedule is upperbounded by $\sum_{e' \in I(e)} u_{e'}$. Therefore, we have

$$D_{total} \leq \sum_e u_e \sum_{e' \in I(e)} u_{e'} \quad (8)$$

We use this delay metric to evaluate the performance of channel assignment in Section V. The following theorem shows that it is difficult to find a channel assignment that minimizes the total delay even when the usage of edges are all the same.

Theorem 3.1: Even when $u_e = 1$ for all e, it is NP-hard to find an optimal channel assignment that minimizes the total delay.

Proof: See Appendix C. ∎

Discussion: In the proposed metrics, we make an implicit assumption that link utilization is the same regardless of the number of competing nodes. While this assumption may be valid with some systems (e.g., those using Time Division Multiple Access), in the case of networks based on CSMA (Carrier Sense Multiple Access), link utilization decreases with more competing nodes due to increased contention [20]. Therefore, the reduced number of competing nodes after channel assignment can lead to further performance improvement in networks using the CSMA-based IEEE

802.11. We discuss this aspect later in Section V.

IV. DISTRIBUTED HEURISTIC ALGORITHM

As discussed in the previous section, finding an optimal channel assignment is NP-hard even in simple scenarios. In this section, we develop a distributed heuristic which increases the total throughput and reduces the total end-to-end delay. Consider a simplified scenario where the expected usage of each edge is identical (i.e., $u_e = 1$, for all $e \in E$) and perfect fairness is enforced ($\alpha = 1$). Then, from Eq. 6, the throughput of each flow becomes $T = \min_e \frac{1}{|I(e)|}$. Also, the delay metric in Eq. 8 becomes $\sum_e |I(e)|$. Therefore, to increase total throughput and reduce total delay, our heuristic attempts to minimize $|I(e)|$. To reduce $|I(e)|$ for each edge, our proposed scheme (i) uses as many distinct channels as possible and (ii) assigns each channel to a similar fraction of edges.

Based on the above goals, we next present our distributed heuristic for channel assignment called SAFE. Depending on the number of available channels, SAFE takes two strategies: *random assignment* and *skeleton assisted assignment*. We describe each strategy in the following subsections. We focus on the case where all nodes have the same number of wireless interfaces. However, we can easily extend our algorithm for scenarios where nodes have different numbers of interfaces (see Section IV-C).

A. Random Assignment

Let K be the number of wireless cards on each node and N be the number of available channels. We denote the set of all available independent channels by $\{1, 2, \ldots, N\}$. In the random assignment scheme, each node v randomly chooses a set of K channels S_v, called *channel set*. If two neighbors u and v have a common channel ($S_u \cap S_v \neq \emptyset$), they can communicate over one of their common channels. If they do not share any channel, the edge (u, v) is dropped. Neighboring nodes can discover their common channels by exchanging hello messages on their chosen channels. For example, consider a network in Figure 2-(a) where four nodes A, B, C, and D have three wireless cards ($K = 3$). Figure 2-(b) shows a random assignment where each node chooses three channels out of seven channels. As shown in the figure, only edges (A, B) and (C, D) are preserved because they share a common channel to communicate.

As illustrated in Figure 2-(b), random assignment can cause network partitions. However, if $N < 2K$, any pair of nodes share at least one common channel by the *pigeonhole principle*. In Figure 2-(c), we use only five channels ($< 2K$) and all edges are connected. For network connectivity, SAFE uses the random assignment algorithm only when $N < 2K$.
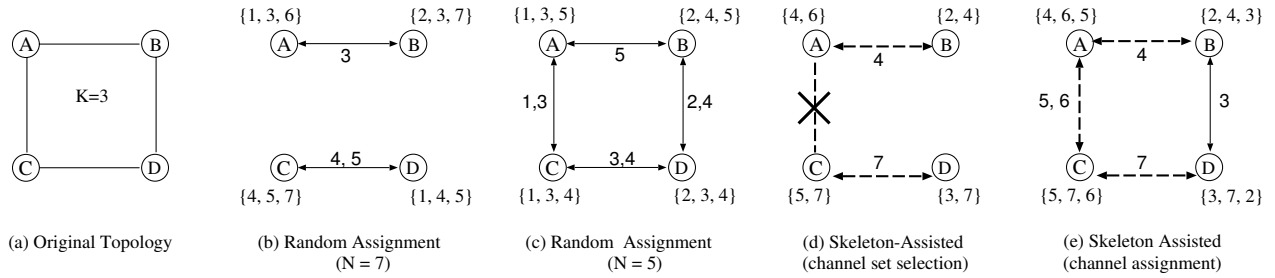
Fig. 2. Example of the random assignment and the skeleton assisted method when $K = 3$. (a) The original topology. Solid lines denote wireless links (b) Random assignment when $N = 7$ with network partition. (c) Random assignment when $N = 5$. Since $N \leq 2K - 1$, the assignment ensures network connectivity. (d) Skeleton assisted algorithm in the channel set selection phase. Skeleton edges are in dashed lines. (e) Skeleton assisted algorithm in the channel assignment phase. SAFE preserves all skeleton edges.

B. Skeleton Assisted Assignment

If we simply use the randomized channel selection when $N \geq 2K$, two neighbors may lose an edge due to lack of a common channel, which can lead to network partition, since each node selects channels in a distributed manner. To ensure network connectivity, SAFE uses a *skeleton*, a spanning subgraph which connects all the nodes in the network; all edges in the skeleton are preserved to ensure connectivity, while other edges may be dropped. Various distributed schemes have been proposed to find a spanning subgraph [21–23], and SAFE can use any of them. When needed, SAFE uses a *default* channel for some skeleton edges. However, if the default channel is used for too many edges, then the effect of interference will degrade the overall network performance. To reduce the interference effect, in SAFE, we use the default channel *only* when using a non-default channel for a skeleton edge becomes difficult. SAFE also results in many non-skeleton edges using non-default channels and can achieve significant performance improvement as we present in Section V. We next describe the detailed channel selection procedure in SAFE.

Algorithm 1 describes the skeleton assisted assignment. Let channel 1 be the default channel. In lines 1 and 2, node v chooses $(K-1)$ channels out of $(N-1)$ non-default channels and exchange with neighbors. If v is connected with all its skeleton neighbors (line 5), we choose one more non-default channel (line 6). Otherwise, if there is a common channel that all not-connected skeleton neighbors share (line 7), we can choose one of such channels (line 8). If v fails with both conditions (lines 5 and 7), it chooses the default channel. Note that the use of default channel increases as N increases. To avoid excessive use of the default channel, we assign the default channel only to skeleton edges. We revisit this topic in section V.

Figures 2-(d) and (e) illustrate the skeleton assisted algorithm when $K = 3$ and $N = 7$. Dashed lines in the figure represent skeleton edges. In Figure 2-(d), each node chooses two channels from non-default channels

---

Algorithm 1 Skeleton Assisted Algorithm at node v

1: Choose $(K - 1)$ channels from $\{2, 3, \ldots, N\}$, add to S_v
2: Broadcast S_v and collect neighbor's S_u's
3: T_v = set of skeleton neighbors u s.t. $S_v \cap S_u = \emptyset$
4: $U_v = \bigcap_{u \in T_v} S_u$
5: **if** $T_v = \emptyset$ **then**
6:     Choose a channel from $\{2, 3, \ldots, N\} - S_v$, add to S_v
7: **else if** $U_v \neq \emptyset$ **then**
8:     Choose a channel from U_v, add to S_v
9: **else**
10:     Add the default channel to S_v
11: **end if**

$\{2, 3, \ldots, 7\}$. In this example, a skeleton edge (A, C) and a non-skeleton edge (B, D) are disconnected. Since all B's skeleton edges (i.e, (A, B)) are preserved, $T_B = \emptyset$ and B randomly chooses channel 3 in Figure 2-(e). D similarly chooses channel 2. However, since A has one disconnected skeleton neighbor C, $T_A = \{C\}$ and $U_A = \{5, 7\}$. Therefore, in Figure 2-(e), A randomly chooses channel 5 from U_A. C chooses channel 6 for the same reason. As a result, all skeleton edges are connected. Also non-skeleton edges (B, D) is connected unintentionally.

In summary, SAFE uses the random assignment method when $N < 2K$ and the skeleton assisted method otherwise.

C. Discussions

1) Channel Selection between Nodes: When two neighbors have only one common channel after the channel set selection, they can use that channel for communication. When they have multiple common channels, we can possibly use multiple channels for one link similar to [1]. One potential issue with this scheme is packet reordering, which may lead to ineffi-

ciency in higher layers. For example, to avoid unnecessary TCP retransmissions due to packet reordering, we need to use a fixed channel for a given flow. In our experiments, a node always uses one channel to a given neighbor, which leads to a lower bound of the best possible network performance.

2) Algorithm Operation in Dynamic Networks: In a dynamic network, nodes can join and leave the network, or move their positions. In SAFE, node v periodically sends its channel set to its neighboring nodes. When a node detects changes (e.g., new or failed neighbors, changes in neighbors' channel set), the node performs the assignment algorithm again (random or skeleton-assisted as appropriate). Such changes include changes of its neighbor set, changes of skeleton edges, and changes of neighbor's channel set.

3) Broadcasting: Broadcasting of a message to neighbor nodes (in the original topology) is more expensive with SAFE than in the single-channel case. When SAFE uses the random assignment scheme, a node sends the message on each K chosen channels. Then, all neighbor nodes can receive the message. In contrast, when SAFE uses the skeleton assisted scheme, a node may not reach all its neighbor nodes by using K chosen channels. In this case, one simple option is to send the message to all N channels. Alternatively, Raniwala et al. [13] propose to use a more complicated scheme (e.g., using multicast), which we can adopt as well. Note that for flooding-based route discovery packets, a node always use K chosen channels, not N, even with the skeleton assisted scheme.

4) Algorithm Operation with Different Number of Interfaces: In practice, nodes can have different numbers of interfaces. We can extend SAFE for this scenario as follows. Let K_v denote the number of interfaces at node v. Let $K_{min} = \min_v K_v$ and $K_{max} = \max_v K_v$. We can use the random assignment when $N < 2K_{min}$ and the skeleton-assisted assignment when $N > 2K_{max}$. When $2K_{min} \leq N < 2K_{max}$, each node v chooses K_v channels out of $2K_v - 1$ channel pool, so that it preserves all edges. One simple way is to choose S_v such that S_v contains K_u channels in $\{1, 2, \ldots, 2K_u - 1\}$ for each neighbor u. For example, assume that a node v with $K_v = 4$ has two neighbors u and w with $K_u = 2$ and $K_w = 3$. Then, v randomly chooses two channels out of $\{1, 2, 3\}$, one more channel out of $\{1, 2, \ldots, 5\}$, and another channel out of $\{1, 2, \ldots, 7\}$.

V. SIMULATION RESULTS

In this section, we present simulation results to evaluate our proposed algorithm. We first describe simulation scenarios and then discuss simulation results.
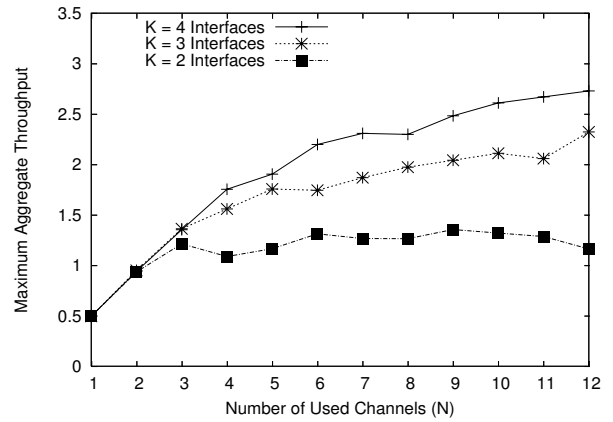

Fig. 3. Maximum aggregate throughput of SAFE when we vary the number of used channels. Three lines corresponds to different numbers of interfaces available at each node. We fix $\alpha = 1$.

A. Simulation Model

We place stationary nodes in a 1000m by 1000m square area uniformly at random. In our experiments, all nodes have the same number (K) of interface cards. We vary the system channel count up to 12 based on the IEEE 802.11a, and use a nominal transmission range of 250 meters for all nodes. We also experimented with different experiment parameters (e.g., node density, network size, longer flow duration), and the results were similar.

As described in Section IV, SAFE employs a skeleton when $N \geq 2K$. In our simulations, we use the LMST [23] for the following desirable properties: (1) it uses only local information and operates efficiently in dynamic networks and (2) it provably guarantees a small degree for each node in the resulting subgraph. The second property is beneficial to our proposed scheme because fewer skeleton edges result in more balanced channel assignment due to less usage of the default channel.

B. Experiments Using Proposed Metrics

In this subsection, we compare the performance of various schemes using the two metrics (throughput and delay) proposed in Section III. We first perform channal assignment according to respective schemes and then perform the shortest path routing using the preserved edges. Then, we solve the linear program in Section III-B (with $c = 1$) using CPLEX to obtain *maximum aggregate throughput*, T_{total} in Eq. 5. We also calculate *upperbound of total delay*, D_{total} as in Eq. 8. Later in Section V-C, we present simulation results using other routing schemes [8, 24]. In this set of experiments, we present the results when the network has 100 nodes. We use the average of 50 runs each with different placement.
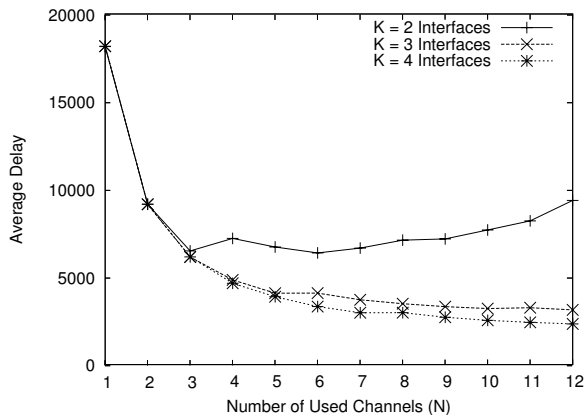
Fig. 4. Comparison of average end-to-end delay when we vary the number of used channels.

Fig. 5. Supported Data Rates ($\geq 95\%$) with various number of interfaces

Throughput of SAFE*:* We first investigate the maximum aggregate throughput by SAFE with various system parameters. In these simulations, we fix the fairness ratio $\alpha = 1$ and experiment using different numbers of interfaces. In Figure 3, we plot the maximum aggregate throughput when we vary the number of channels used. Note that the values on the Y-axis are in proportion to the link capacity $c = 1$. When $K > 2$, our results show that the use of more channels helps to increase the amount of supported traffic. For example, when each node has four interfaces, the maximum aggregate throughput of SAFE increases up to 2.73. When compared to single channel case (i.e., $N = 1$), this is more than five times performance improvement (2.73 vs. 0.49). Not surprisingly, the performance gain is larger when there are more interfaces at each node. When $K = 2$, the amount of performance gain becomes smaller as N increases. We observe that the performance starts to decrease as N becomes larger than 10. This is because with fewer interfaces at each node, the network loses more edges, thus lessening the benefit of using more channels.

Although we do not observe such degradation for $K > 2$ in Figure 3, the performance will eventually drop as N grows further (e.g., N=20, K=3). Analytical analysis on the performance with different N and K is an interesting area for future work.

Results using Delay Metric: We now present the results using the delay metric in Eq. 8. In Figure 4, we plot the average values of delay metric when we change the number of used channels. We also experiment with different numbers of interfaces at each node and observe significant performance gain. (In the figure, the time unit is the transmission time for one packet.) For example, when $K = 4$, the average delay of SAFE using 12 channels is around 13% when compared to single channel case ($N = 1$). As in Figure 3, we observe that with $K = 2$, the delay
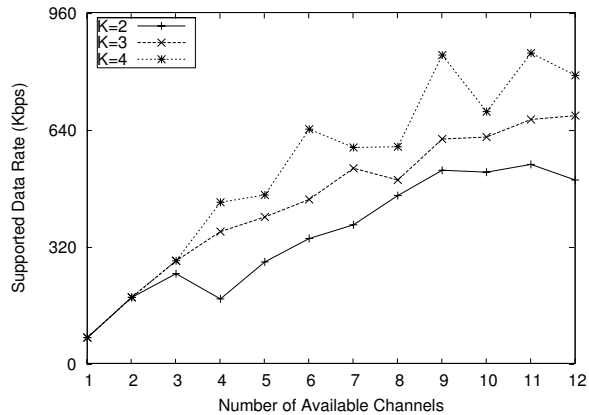
performance becomes worse after N becomes larger than a certain point.

In this subsection, we have shown that SAFE improves performance against the metrics proposed in Section III. We next report packet-level simulation results that illustrate SAFE actually leads to throughput improvement in more realistic environments.

C. Packet-level Simulations

In this section we present the results of simulation experiments using *ns-2*. Using packet-level simulations, we examine the performance of SAFE in more realistic environments. We have modified the simulation code such that each node has multiple wireless interfaces. We implemented LMST-based scheme [23] to construct the skeleton. We consider two different routing schemes: centralized shortest-path routing (using the Floyd-Warshall algorithm) and AODV.

We place 100 stationary nodes uniformly at random on a 1000m by 1000m square with the transmission range of 250 meters. We select 40 source-destination pairs uniformly at random. Each source generates a CBR (Constant Bit Rate) flow using 1024-byte UDP packets. Each flow starts between 90 and 110 seconds chosen uniformly at random and ends at 170 seconds. In a particular experiment, each source generates a same data rate. We use the IEEE 802.11 standard for the underlying MAC layer protocol [25], and the maximum data transmit rate is fixed at 1 Mbps for all nodes. We use average values of 16 runs with different node placements and different source-destination pairs.

Network Throughput: We first investigate the maximum aggregate throughput that the SAFE can support. The aggregate throughput is the sum of data rates of all active flows. In this set of experiments, to approximate the scenario of perfect fairness ($\alpha = 1$) as in Figure 3, we consider that a network can support an aggregate throughput if the *minimum* delivery ratio over all flows
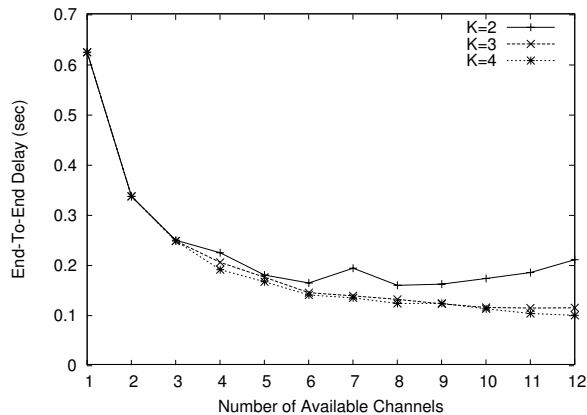
Fig. 6.   End-to-End Delay with various K and N.

Fig. 7.   The supported delivery ratio with AODV routing (K=4).

is more than 95%. To measure the maximum aggregate throughput, we increase the aggregate end-to-end data rate by an increment of 80 Kbps until the aggregate throughput is not sustainable by the network. In this set of experiments, we implement the Floyd-Warshall algorithm for shortest path routing.

In Figure 5, we present the maximum aggregate throughput that SAFE can support when we vary K and N. Each line in the figure represents the result for a different K value. When $K = 3$ or 4, we observe that SAFE significantly improves the throughput as N increases. For example, when $K = 4$, we can achieve up to 1276% throughput improvement (893 kbps when $N = 12$ vs. 70 kbps when $N = 1$). Similar to the results in Figure 3, when $K = 2$, the performance starts to decrease as N becomes larger than 11. The similarity of Figure 5 to Figure 3 illustrates that the metric in Eq. 1 is a good indicator for performance comparison. We notice that SAFE achieves larger performance improvement in packet-level simulations than in metric-based simulations in Section V. For example, SAFE achieves up to 1276% throughput improvement in packet simulation, compared to 557% in Figure 3. This is because, as discussed in Section III, the decrease in the number of competing edges leads to the increased MAC-level throughput [20].

End-to-End Delay: To measure end-to-end delay, each of 40 sources sends *one* 1024-byte packet simultaneously, and we obtain the average end-to-end delay. We send only one packet to avoid packet losses due to unnecessarily high data rates that the network cannot support. Nodes find the route to destinations by the Floyd-Warshall algorithm. In Figure 6, we present the results when we vary N and K. In the figure, when K =3 or 4, the delay decreases as we use more channels. However, when $K = 2$, the delay begins to increase after $N = 9$. The results in Figure 6 look similar to those in Figure 4, which illustrates that the metric in Eq. 8 is a good indicator for end-to-end delay.
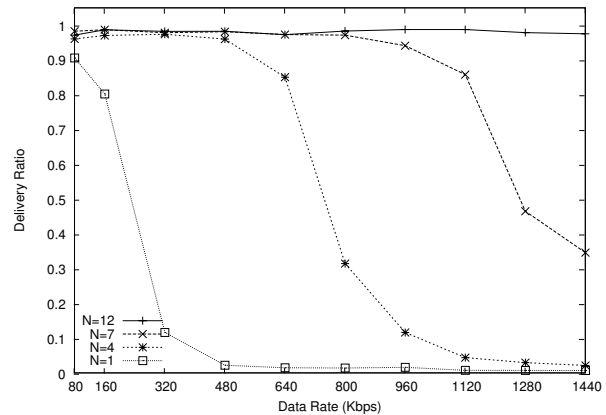
Delivery Ratio with AODV: In this set of experiments, we use AODV as routing protocol instead of the centralized scheme used in previous experiments. In Figure 7, we present the minimum delivery ratio over all flows when the aggregate data rate increases from 80 Kbps to 1440 Kbps. For fixed $K = 4$, we experiment with $N = 1, 4, 7, 12$, which correspond to different lines in the figure. In the figure, we observe that SAFE can support high delivery ratios (over 95%) even when the aggregate data rate is 1440 Kbps. However, the delivery ratio of single channel case ($N = 1$) is less than 90% even with 80 Kbps, and we can observe that SAFE achieves significant throughput improvement (up to 1700%).

In our results, the performance gain using the AODV protocol is larger than with the centralize fixed routing. For example, compared to single channel case, SAFE used with AODV achieves up to 1700% throughput improvement, while centralized routing achieves up to 1276% improvement. We explain this as follows. Suppose an edge is heavily used. Then, even though the edge is still available, transmission attempts over the edge may fail repeatedly. In that case, AODV assumes the edge is broken, and tries to find a new route, which is potentially less congested. Thus, packets in effect detour highly congested links. In contrast, the centralized shortest-path routing does not consider congestion, and the throughput performance is accordingly lower.

Experiments with Dynamic Scenarios: Due to the space constraint, we briefly report the results when we use SAFE in mobile networks. In these scenarios, nodes constantly move according to the Random Waypoint mobility model with maximum speed of 5m/s, and the use of SAFE again improves the network performance significantly. For example, when we use SAFE with AODV when $K = 4$ and $N = 7$, the minimum delivery ratio is 88% and average delivery ratio is 97% for 800 Kbps. However, with single channel, the minimum delivery ratio is 1% and the average is 21%.

	Average Throughput (in Mbps)		
	N=7	N=10	N=12
SAFE	7.08 (0.12)	8.76 (0.23)	8.90 (0.78)
Centralized [7]	6.96 (0.02)	9.82 (0.27)	10.59 (0.85)

TABLE I

COMPARISON BETWEEN SAFE AND CENTRALIZED SCHEME.
THE VALUES IN PARENTHESIS ARE STANDARD DEVIATIONS.
$(K = 4)$

Due to the mobility, the delivery ratio is slightly lower than stationary cases (minimum of 97% and average of 99%) in Figure 7, but the difference is marginal.

D. Comparison with a Centralized Scheme

In this subsection, we compare the performance of SAFE with the centralized scheme developed for mesh networks [7]. We use 50 nodes including 5 wired gateways and 10 sources with the same traffic demand. We use the LP-based simulation code by the authors of [7]. Note that while assigning channels, the centralized scheme in [7] and thus the simulation code use the global information about traffic pattern and network topology. In our experiments, we independently run SAFE on the same network setting, but assign channels in a distributed way without using the traffic information. Then we feed the channel assignment result by SAFE into the simulation code (for routing and scheduling) and compare the performance.

In Table I, we report the average throughput when we vary the number of channels available in the system. Without using the information about traffic pattern and global topology, when $N = 7$, SAFE performs the same as the centralized scheme, which is a constant-factor approximation scheme to the optimum. When $N \geq 2K = 8$, SAFE maintains the network connectivity using skeleton, and we observe that the performance gap between SAFE and the centralized scheme. However, the difference is at most 16% in this experiment, which is moderate considering the distributed operation of SAFE. This simple experiment indicates that our distributed scheme performs comparable to the best known centralized scheme.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we have considered the channel assignment problem in wireless networks with multiple interfaces. Finding an optimal channel assignment is NP-hard even when we do not consider routing. We have presented a distributed channel assignment scheme called SAFE. SAFE maintains network connectivity and achieves significant performance improvement, which is comparable to the best prior centralized scheme.

Although our current scheme does not differentiate links, it is possible in practice that we want to give higher priority to certain links, for example, due to higher link usage, or higher link quality [26]. In the future, we plan to investigate how to assign channels based on such priority constraints. We also want to develop an analytical bound on the performance of SAFE in comparison with the optimal solution.

REFERENCES

[1] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou, "A multi-radio unification protocol for IEEE 802.11 wireless networks," Microsoft Technical Report, MSR-TR-2003-41, Tech. Rep., June 2003.

[2] A. Raniwala, K. Gopalan, and T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM SIGMOBILE MCCR*, vol. 8, no. 2, 2004.

[3] N. Jain and S. Das, "A multichannel CSMA MAC protocol with receiver-based channel selection for multihop wireless networks," in *Proceedings of the 9th IC3N*, Oct. 2001.

[4] W.-C. Hung, K. L. E. Law, and A. Leon-Garcia, "A dynamic multichannel MAC for ad-hoc LAN," in *21th Biennial Symposium on Communications*, Apr. 2002.

[5] P. Kyasanur and N. H. Vaidya, "Capacity of multi-channel wireless networks: impact of number of channels and interfaces," in *ACM MobiCom*, 2005.

[6] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *ACM MobiCom*, New York, NY, USA, 2005.

[7] M. Alicherry, R. Bhatia, and L. E. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *Proceedings of ACM MobiCom*, 2005.

[8] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proceedings of ACM MobiCom*, 2004.

[9] R. Chandra, P. Bahl, and P. Bahl, "Multinet: Connecting to multiple IEEE 802.11 networks using a single wireless card," in *Proceedings of Infocom*, March 2004.

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

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

[12] A. Nasipuri, J. Zhuang, and S. R. Das, "A multichannel CSMA MAC protocol for multihop wireless networks," in *Proceedings of IEEE WCNC*, September 1999.

[13] A. Raniwala and T. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network," in *Proceedings of Infocom*, March 2005.

[14] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[15] R. Rajaraman, "Topology control and routing in ad hoc networks: a survey," *ACM SIGACT News*, vol. 33, no. 2, pp. 60–73, 2002.

[16] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "End-to-end packet-scheduling in wireless ad-hoc networks," in *Proceedings of SODA*. SIAM, 2004, pp. 1021–1030.

[17] S. Yi, Y. Pei, and S. Kalyanaraman, "On the capacity improvement of ad hoc wireless networks using directional antennas," in *ACM MobiHoc*, 2003.

[18] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proceedings of MobiCom*, 2003.

[19] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, "Algorithmic aspects of capacity in wireless networks," in *ACM SIGMETRICS*, 2005.

[20] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE JSAC*, vol. 18, no. 3, pp. 535–547, Aug. 2000.

[21] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," *IEEE JSAC*, vol. 17, no. 8, pp. 1333–1344, Aug. 1999.

[22] R. Wattenhofer, L. Li, P. Bahl, and Y.-M. Wang, "Distributed topology control for power efficient operation in multihop wireless ad hoc networks," in *IEEE Infocom*, 2001.

[23] N. Li, J. Hou, and L. Sha, "Design and analysis of an MST-based topology control algorithm," in *IEEE Infocom*, 2003.

[24] C. E. Perkins and E. M. Belding-Royer, "Ad hoc on-demand distance vector (AODV) routing," in *IEEE Workshop on Mobile Computing Systems and Applications*, Feb. 1999.

[25] IEEE, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications," IEEE 802.11 Standard, 1999.

[26] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," in *ACM SIGCOMM*, Sept. 2004.

[27] M. R. Garey and D. S. Johnson, *Computer and Intractability: A Guide to the Theory of NP-completeness.* Freeman, 1979.

[28] E. Kubicka and A. Schwenk, "Introduction to chromatic sums," in *Proceedings of ACM Computer Science Conference*, 1989, pp. 39–45.

[29] J. Erickson, S. Thite, and D. Bunde, "Distance-2 edge coloring is np-complete," Manuscript, In preparation as of March 1, 2002.

APPENDIX

A. Hardness of Throughput Computation

Theorem 1.1: Suppose that we are given a set of paths P and $I(e)$ for each e. When $\alpha = \infty$, it is NP-hard to find the maximum throughput.

Proof: We prove this by reduction from the maximum independence set problem, which is known to be NP-hard [27]. An independence set of a graph $G = (V, E)$ is a subset of $V' \subseteq V$ such that no two vertices in V' are adjacent to each other. Given a graph G, we construct a network N as follows (for network N, we use the terms "nodes" and "links" instead of "vertices" and "edges" which are used for graph G). For each vertex v in V, we create nodes a_v and b_v, and connect two nodes with link l_v (we denote these set of links as $L(V)$). In addition, we have a source node s and a sink t. There are links from s to all nodes a_v and from all nodes b_v to t. We also have an edge between a_v and a_u if v and u are neighbors in G. We assume that the capacity of each link is 1.

We now define $I(l)$ for each link l. For all links $l_v \in L(V)$, we assign the same channel so that $I(l_v) = \{l_u : u$ and v are adjacent to each other in $G\}$. For all other links, we assign different channels so that $I(l) = \{l\}$.

Consider all paths P from s to t of length 3. Note that each link in $L(V)$ belongs to exactly one path in P. We want to find the maximum of $\sum_{p \in P} T_p$ with constraints given in (2)–(5) and $\alpha = \infty$. It can be easily verified that the maximum throughput of the network N gives the maximum independence set of G. Therefore, the problem is NP-hard. ∎

B. Hardness of Delay Computation

Theorem 1.2: In a wireless network $G = (V, E)$, we are given a set of flow paths P and $I(e)$ for each e. It is NP-hard to find a schedule to minimize the total end-to-end delay for all flows over P.

Proof: We reduce SUM COLORING [28] to the problem. In the SUM COLORING problem, we are given a graph $G = (V, E)$ and map a vertex $v \in V$ to an integer i_v so that for any neighboring vertices u and v, $i_u \neq i_v$. The objective is to minimize $\sum_v i_v$. Given an instance $G = (V, E)$ of SUM COLORING, we construct $G' = (V', E')$ where for each vertex $v \in V$, we create two vertices and an edge $e'(v) \in E'$ connecting them, and for any neighboring vertices u and v in V, create an edge connecting any endpoints of $e'(u)$ and $e'(v)$. We assume that for each $e'(v)$, $I(e'(v))$ includes all edges $e'(u)$ such that u is adjacent to v in G (we assume two hop interference model and all edges use the same channel). For each edge $e'(v)$ in G', we want to send a packet. In other words, P includes all edges $e'(v)$. If we find a minimum total delay of all paths P, then the delay of $e'(v)$ corresponds to i_v in G and therefore gives an optimal solution of SUM COLORING. ∎

C. Proof of Theorem 3.1

Let K be the number of wireless cards on each node and let N be the number of available channels. We show that it is NP-hard to find an optimal channel assignment for general K and N by reduction from DISTANCE-2 EDGE COLORING [29]. In the DISTANCE-2 EDGE COLORING problem, we are given a simple graph $G = (V, E)$ and assign colors to edges so that any two intefering edges are not assigned the same color It is NP-hard to find an edge coloring using the minimum number of colors [29]. Suppose that we can find an optimal channel assignment when each edge (u, v) want to send a packet from u to v (so $u_e = 1$ for all edges). Given an instance of DISTANCE-2 EDGE COLORING, we find an optimal channel assignment starting from $K = N = \Delta$ and increase $K(= N)$ by one until the delay of all edges is exactly one. The solution gives a valid edge coloring with the minimum number of colors. Therefore, it is NP-hard to find an optimal channel assignment for general K and N.