Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 31 (2008) 1343–1353
computer communications
www.elsevier.com/locate/comcom

A channel assignment algorithm for multi-radio wireless mesh networks

Stefano Avallone [a,*], Ian F. Akyildiz [b]

[a] Dipartimento di Informatica e Sistemistica, Università di Napoli, Via Claudio 21, 80125 Napoli, Italy
[b] BWN Laboratory, Georgia Institute of Technology, Atlanta, GA 30332, USA

Available online 26 January 2008

Abstract

Wireless mesh networks (WMNs) are receiving increasing attention as an effective means to deploy ISP's wireless last mile access, wireless enterprise backbone networks and several other applications. The focus of this paper is on multi-radio wireless mesh networks, given the considerable improvement in network throughput that multiple radios allow to achieve and the availability of cost-effective wireless devices. Interesting research problems are still unsolved in this field. Due to the scarcity of non-overlapped frequency channels and available radios per node, interference is still present, which limits the bandwidth available on network links and eventually cuts the achievable throughput down. As interference depends on how channels are bound to radio interfaces, a proper channel assignment scheme is needed to reduce the interference.

In this paper we identify some key requirements of a channel assignment scheme and show the interdependence between the channel assignment and the routing problems. Accordingly, a centralized channel assignment and routing algorithm is developed for multi-radio wireless mesh networks aiming to maximize the network throughput. An integer linear programming (ILP) model is presented to evaluate the performance of our heuristic. Finally, a performance study is carried out to assess the effectiveness of our proposed algorithm.

Keywords: Wireless mesh networks; Multi-radio; Channel assignment and routing

1. Introduction

Wireless mesh networks (WMNs) consist of a backbone with mesh routers which collect and relay the traffic generated by mesh clients [1]. Mesh routers have limited (if any) mobility and are usually connected through wireless media. Mesh clients are typically mobile and rely on mesh routers to deliver data to the intended destinations. The absence of a wired infrastructure makes wireless mesh networks attractive for several applications, e.g., wireless last mile access of ISPs, wireless enterprise backbone networks, building automation, broadband home networking, community, neighborhood networks. However, wireless communication suffers from environmental noise and interference problems. Interference can be alleviated by using multiple channels in a node.

The IEEE 802.11b/g and IEEE 802.11a standards define 3 and 12 non-overlapping frequency channels, respectively. Using multiple channels in multi-radio WMNs greatly improves the network throughput [2,3]. One of the most important design questions for a multi-radio WMN is the channel assignment problem, i.e., how to bind each radio interface to a radio channel.

The channel assignment has to preserve the network connectivity, as two neighbor nodes can only communicate with each other if their radio interfaces share a common channel. At the same time, the reuse of the same channel in a neighborhood must be limited, as simultaneous transmissions over the same channel collide, leading to a decrease of the throughput. The channel assignment also determines the bandwidth available on the network links. Indeed, all the links in the interference range that have been assigned the same channel cannot transmit simultaneously

* Corresponding author. Tel.: +39 0817683902.
E-mail addresses: stavallo@unina.it (S. Avallone), ian@ece.gatech.edu (I.F. Akyildiz).

0140-3664/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2008.01.031

and have to share the common channel capacity. Consequently, the channel assignment problem needs to be jointly studied with the routing problem, i.e., the problem to find a set of flow rates for every network link that achieve an anticipated objective. Unfortunately, the joint channel assignment and routing problem is NP-complete [4]. Therefore, an approximate solution is sought by solving the two problems separately. First, a flow rate computation method is proposed which aims to maximize the network capacity and hence it is not dependent on any particular traffic profile. Then, a channel assignment algorithm is presented that attempts to make the given set of flow rates achievable while preserving the network connectivity. These objectives are achieved by splitting the algorithm in two stages, *the link-group binding and the group-channel assignment*.

The paper is structured as follows. In Section 2, we give an overview of the related work. In the next section we identify the contribution of this paper. In Section 4, we formalize the channel assignment and routing problem. In the next section we illustrate the proposed flow rate computation method. In Section 6, we introduce our proposed algorithm in detail, including a proof of correctness and a complexity analysis. In Section 7, we present simulation experiments for performance evaluation of our scheme and its comparison with other existing schemes. Finally, in Section 8 we conclude the paper.

2. Related work

The static channel assignment problem in multi-radio WMNs has been investigated in the literature recently. We can roughly distinguish between *interference-aware* channel assignment algorithms,which aim to minimize some network-wide measure of interference, and *traffic-aware* channel assignment algorithms, which aim to make a given set of flow rates schedulable. Among the first interference-aware algorithms, Draves et al. [5] proposed an *identical* channel assignment, i.e., the first radio is assigned channel 1, the second radio is assigned channel 2 and so on. Such an approach clearly preserves connectivity but does not make any effort to reduce interference. In [6], a hybrid channel assignment scheme is proposed where some radios are statically assigned a channel while the remaining radios can dynamically change their frequency channel.

In [2,7] centralized channel assignment and routing algorithms are introduced. The proposed channel assignment algorithms are based on the same concept: links are visited in some particular order and a common channel is assigned to the interfaces of both end nodes. If all interfaces of the end nodes in a link are already assigned a channel and they do not share any common channel, then it is necessary to replace one of these channel assignments. Due to the limited number of radios per node, this replacement may trigger a chain reaction and must be performed recursively.

In Fig. 1, we illustrate this drawback where we assume that all nodes have two radios. When the link between *A* and *B* is visited, there are no common channels and no radios left to assign a common channel. The algorithm then replaces one of the previous channel assignments to make *A* and *B* share a common channel. Assume that channel 7 on *B* is turned into channel 6. This replacement requires to check whether the constraint on the number of radios is satisfied for all *B*'s neighbors. In the case of Fig. 1, turning channel 7 to 6 on *B* causes *D* to have 3 channels assigned to its 2 radios. To clear this inconsistency a new channel replacement is required.

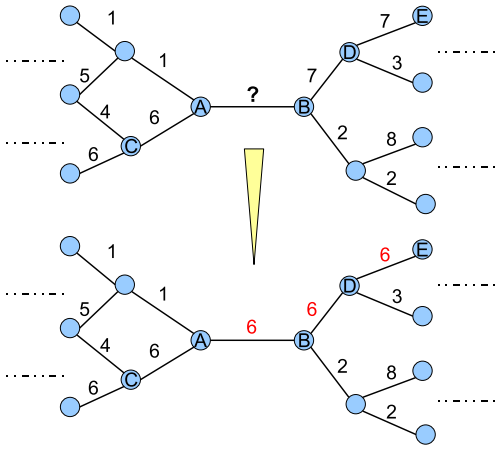

Fig. 1. Example to illustrate the need of recursive replacement in some channel assignment schemes.

The algorithms proposed in [2,7] mainly differ in the order in which links are visited and in the criteria used to select the channel to be assigned to a radio. The algorithm proposed in Tang et al. [7] is interference-aware, as it visits the links in decreasing order of the number of links falling in the interference range and selects the least used channel in that range. Assuming the knowledge of the set of connection requests to be routed, both an optimal algorithm based on solving a Linear Programming (LP) and a simple heuristic are proposed to route such requests given the link bandwidth availability determined by the computed channel assignment. The algorithm proposed in Raniwala et al. [2] is instead traffic-aware. It is assumed that the traffic profile is known which is used to determine an estimate for the expected link flow rates. The channel assignment algorithm visits all the links in decreasing order of the expected link flow rate and selects the channel which minimizes the sum of the expected flow rates of all the links in the interference region that are assigned the same radio channel. The channel assignment proposed in Raniwala et al. [2] is part of a more complex iterative algorithm aimed at routing a defined traffic profile. However, no new routing algorithm is presented, as the traffic profile is routed using either the minimum-hop path routing or a randomized multi-path routing.

In [3], distributed channel assignment and routing algorithms are developed. At any time each node joins a single gateway node and sends all the packets destined to the wired network to that gateway. Nodes advertise their cost

to reach the gateway they are currently associated with. Cost dynamically changes as it depends on residual bandwidth to achieve load balancing. If a node is notified of a less cost path towards another gateway, it starts a procedure to associate with that gateway. Such procedure involves updating the routing tables of all the nodes along the paths to the previous and the new gateways. Since the cost is dynamic, the proposed strategy may lead to route flaps and to a non-convergent network behavior, thus requiring appropriate countermeasures.

The joint channel assignment, routing and scheduling problem is investigated in [4,8]. In both papers, it is assumed that the knowledge of the traffic demands is available and that the system operates synchronously in a time slotted mode. In [4], an LP is formulated to route the given traffic demands in order to maximize the system throughput subject to fairness constraints. Constraints on the number of radios and on the sum of the flow rates for the links in the interference range are also included. Since the resulting formulation includes integer variables which make the problem NP-hard, the authors solve the LP relaxation of the problem. The result is a network flow along with a possibly unfeasible channel assignment. The proposed channel assignment algorithm then serves the purpose to adjust the flow on the graph to ensure a feasible channel assignment. The flow on the graph is further re-adjusted by a post processing and a flow scaling steps. Finally, a scheduling algorithm produces an interference free link schedule.

In [8], the traffic demands are formulated as a multicommodity flow problem, where one among several different objectives can be defined. Besides including the same constraints as in [4], the LP formulation in [8] makes use of time-indexed variables, hence solving such LP gives a solution for the entire channel assignment, routing and scheduling. However, the presence of integer variables makes the problem NP-hard and thus, the authors solve the LP relaxation of the problem. Then a channel assignment along with scheduling based on greedy coloring is used to resolve the potential conflicts. The authors present both static and dynamic channel assignment algorithms. The dynamic algorithm, assigning distinct channels to a link for different time slots, requires radios capable of fast switching between channels.

3. Our contributions

Existing channel assignment schemes as described in the previous section fail to meet the following desirable objectives:

- While selecting a channel for a radio interface, the channel assignment algorithm must take a choice based not only on information related to nodes within the interference range because the effect of such a choice propagates even further. Disregarding the interdependence among channel assignments all over the network leads to a sub-optimal assignment and gives rise to violations of the constraint on the number of radios per node that must be solved through a recursive replacement of previous channel assignments.
- The channel assignment scheme must be independent of any particular traffic profile. Otherwise, the network throughput may decrease in case the actual network load is much different than the traffic profile used to compute the channel assignment.

Unlike the existing channel assignment schemes described in the previous section, our algorithm, denoted as MCAR (Maxflow-based Channel Assignment and Routing), meets the above requirements. The interdependence among channel assignments across the whole network is taken into account by first identifying the groups of links that need to be assigned the same channel in order for the number of different channels on every router not to exceed the number of radios. Then, the actual channel assignment stage can exploit the results of the first stage to assign channels in such a way that no replacement of previous assignments is necessary. For example, this first stage may determine that links $C \rightarrow A$, $A \rightarrow B$ and $B \rightarrow D$ (see Fig. 1) must be assigned the same channel, thus avoiding to replace previous assignments.

We also propose a flow rate computation method that is independent of any particular traffic pattern and aims to maximize the network throughput. We note that, even though the MCAR algorithm has been designed in conjuction with such flow rate computation method, it is basically independent of the method used to provide the set of flow rates.

Next, we formally define the channel assignment problem addressed by our algorithm.

4. Problem definition

In this paper we consider the WMN architecture given in Fig. 2. Some of the mesh routers denoted as *mesh aggregation devices* provide network connectivity to end-user mobile wireless devices within their coverage area. Mesh aggregation devices collect user traffic and forward it to the wired network through multiple hops across the WMN. Mesh routers connected to the wired network are denoted as *mesh gateways*.

We assume that each mesh router u is equipped with $k(u) \geqslant 1$ radio interfaces and there are $|\mathcal{C}|$ available channels. For ease of exposition, we assume that the transmission and the interference radii, denoted by r_T and r_I, respectively, are the same for all the radios. Also, we assume that the transmission rate is fixed at the same value for all the radio interfaces of the same router.

The impact of the interference can be formally accounted for through either of the interference models defined in Gupta and Kumar [9]: the *protocol* model that assumes interference to be an all-or-nothing phenomenon and the *physical* model that considers the impact of interfering transmissions on the signal-to-noise ratio. In this

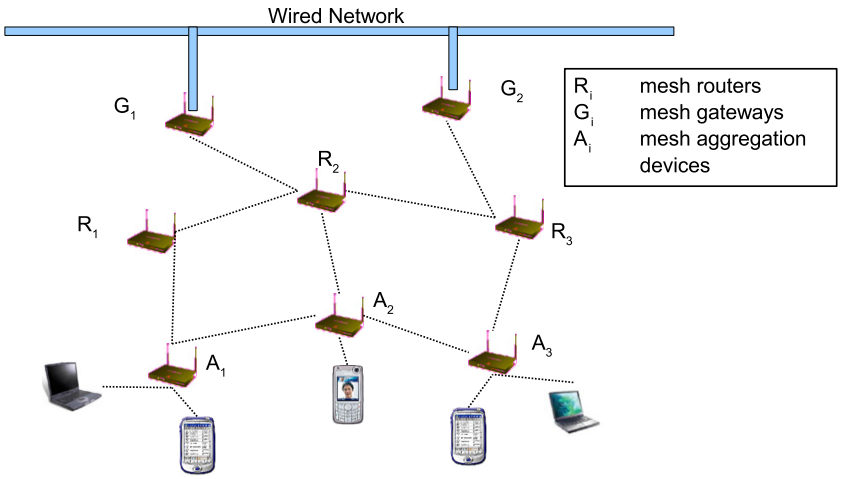

Fig. 2. Wireless mesh network reference architecture.

paper we consider the protocol model, according to which a transmission between two mesh routers u and v using the same channel is successful if both of the following conditions are satisfied:

(1) $d(u,v) \leqslant r_{\mathrm{T}}$,
(2) Any node x, using the same channel as u and v and such that $\min\{d(x,u), d(x,v)\} \leqslant r_{\mathrm{I}}$, is not transmitting,

where $d(u,v)$ is the euclidean distance between u and v.

We model the WMN as an undirected graph $G_I = (V, E_I)$, where V is a set of nodes each representing a mesh router. Given any two nodes $u, v \in V$, the undirected edge $u \leftrightarrow v \in E_I$ if and only if $d(u,v) \leqslant r_{\mathrm{T}}$. We refer to G_I as the *potential communication graph*, as in keeping with the protocol model an edge $u \leftrightarrow v \in E_I$ indicates that u and v can communicate with each other provided that they are assigned a common channel. We set the capacity $c(u \leftrightarrow v)$ of the undirected edge $u \leftrightarrow v$ to the minimum transmission rate among those of the radios of u and v. We denote by $V_A \subseteq V$ the subset of the mesh routers aggregating user traffic and by $V_G \subseteq V$ the subset of the mesh routers acting as gateways to the wired network.

According to the protocol model, we also define two links $u \leftrightarrow v \in E_I$ and $x \leftrightarrow y \in E_I$ as *potentially interfering* if $\min\{d(u,x), d(u,y), d(v,x), d(v,y)\} \leqslant r_{\mathrm{I}}$, i.e., if node x or node y are in one of the interference ranges of u and v. We denote by $\mathcal{N}(e)$ the *potential collision domain* of a link e, i.e., the set of all links that potentially interfere with it. We need this concept for the channel assignment algorithm, when channels are not assigned to links yet. We use the term "potentially" to underline that such links may interfere only if they use the same channel. When the channel assignment is known, if such links share a common channel then we simply refer to them as interfering links.

A channel assignment $\mathcal{A}$ assigns a set $\mathcal{A}(u)$ of channels ($|\mathcal{A}(u)| \leqslant k(u)$) to each node $u \in V$. Thus, $\mathcal{A}$ induces a new graph model $G = (V, E)$ where two nodes are connected if they are in the transmission range of each other *and* share at least one common channel, i.e., $u \leftrightarrow v \in E$ if and only if $d(u,v) \leqslant r_{\mathrm{T}}$ and $\mathcal{A}(u) \cap \mathcal{A}(v) \neq \emptyset$. We note that $E \subseteq E_I$, i.e., E may not contain all links belonging to E_I. This is the case when some neighboring nodes do not share any channel. Given a channel assignment, the collision domain $\mathcal{D}_{\mathrm{coll}}(e)$ of a link e is defined as the set of links interfering with it.

We can now formalize the channel assignment problem, defined as the problem to find, if any exists, a channel assignment such that a given set of flow rates are schedulable. First, we derive a *sufficient* condition for a set of flow rates to be schedulable. The constraint is that transmissions over different links in a collision domain cannot take place simultaneously due to interference. Nodes must share the common channel capacity and cannot transmit at an arbitrary rate. If we denote the flow rate for a link e_0 by $f(e_0)$, then such link has to carry an amount of data equal to $f(e_0)T$ in every time interval T. If the link capacity is $c(e_0)$, the transmission of such amount of data takes $\frac{f(e_0)}{c(e_0)}T$ time. Hence, the sufficient condition is that the sum of such amounts of time for all the links in the collision domain must not exceed T. Considering the collision domain of the generic link e, the sufficient condition can be stated as $\sum_{e_0 \in \mathcal{D}_{\mathrm{coll}}(e)} \frac{f(e_0)}{c(e_0)} T \leqslant T$ for every T. As a consequence, the following condition ensures that a set of flow rates are schedulable:

$$\forall e \in E \quad \sum_{e_0 \in \mathcal{D}_{\mathrm{coll}}(e)} \frac{f(e_0)}{c(e_0)} \leqslant 1 \tag{1}$$

Since the composition of the various collision domains in the network depends on the channel assignment, we consider the problem to find, if any, a channel assignment such that the above sufficient condition is satisfied. Such a decision problem can also be shown to be NP-complete by reducing the *Multiple Subset Sum Problem* (MSSP) [10]. Therefore, heuristics are typically proposed that first deter-

mine a channel assignment and then adjust the pre-computed flow rates to obtain a set of schedulable flow rates given the computed channel assignment. Thus, the returned set of schedulable flow rates is likely to be different than the given set of pre-computed flow rates. Our goal is to find a channel assignment and a corresponding set of schedulable flow rates that are as *close* as possible to the given set of pre-computed flow rates. In particular, we seek a channel assignment which minimizes the scaling factor $\lambda \geqslant 1$, where $\left\{\frac{f(e)}{\lambda}\right\}_{e\in E}$ is a set of flow rates satisfying the sufficient condition for schedulability given the computed channel assignment.

When the channel assignment is determined, for every link $e \in E$ we can compute $\sum_{e_0\in\mathcal{D}_{\text{coll}}(e)}\frac{f(e_0)}{c(e_0)}$, which for conciseness we refer to as the *total utilization* $U_{\text{tot}}(e)$ of the collision domain of link e. The maximum among the total utilization of all the collision domains turns out to be the minimum scaling factor which yields a set of flow rates satisfying the sufficient condition (1). It suffices to observe that, if all the pre-computed flow rates are divided by a value α, then the total utilization of every collision domain becomes α times smaller. Therefore, we tackle the problem to find a channel assignment that minimizes the maximum among the total utilization of all the collision domains:

$$\text{minimize } \max_{e\in E} U_{\text{tot}}(e);\ U_{\text{tot}}(e) \stackrel{\text{def}}{=} \sum_{e_0\in\mathcal{D}_{\text{coll}}(e)}\frac{f(e_0)}{c(e_0)}$$

This problem turns out to be the optimization version of the decision problem stated above, which is NP-complete. Thus, it is not practical to compute an optimal solution and hence we develop a heuristic algorithm described in Section 6.

5. A flow rate computation method

In this section we propose a method to compute the flow rates having the objective to maximize the achievable network throughput. We underline that the proposed method does not need the knowledge of the (expected) traffic demands. Since the flow rates are computed when a channel assignment is not known yet, the wireless mesh network is represented by the potential communication graph. Thus, to determine the flow rates used as input to the channel assignment algorithm, we may compute the maximum achievable throughput of the potential communication graph under the protocol interference model. However, the problem to determine such throughput is NP-complete [11]. Also, the maximum throughput computed in such a way is smaller than the actual capacity of the WMN, as simultaneous transmissions can take place over potentially interfering links that have been assigned different channels. Instead, if we compute the maximum throughput of the potential communication graph assuming that interference does not arise, then we clearly overestimate the capacity of the WMN. However, since the channel assignment attempts to make the given set of pre-computed flow rates schedulable, determining such values in the absence of interference enables the channel assignment algorithm to pursue the ideal situation where channels are assigned so as to eliminate the interference. Thus, for the present work we decided to base the flow rate values on the computation of the maximum throughput of the potential communication graph in the absence of interference.

Even disregarding the interference, the problem of finding the capacity of the potential communication graph, being an instance of a maximum multi-commodity flow problem, is NP-complete [12]. However, we can turn to the simpler single commodity flow problem by means of the following key observation: in the WMN given in Fig. 2 mesh routers have to forward packets towards the wired network, regardless of which particular gateway is used. In other words, mesh aggregation devices collecting user traffic do not have to forward each packet to a specific mesh gateway, but can direct it to *any* of the mesh gateways. The problem of maximizing the flow from any source to any sink is a single commodity flow problem with multiple sources and sinks. There is a standard trick to reduce this more general version to the case with a single source and a single sink, which consists in adding two extra nodes connected, respectively, to the nodes of V_A and V_G by links of infinite capacity. Formally, we consider a new directed graph $G_I' = (V', E_I')$, where V' contains the same vertices of V plus the two extra nodes which we refer to as the *supersource* s and the *supersink* t. E_I' contains the same edges of E_I plus, for each $u \in V_A$ and $v \in V_G$, edges of the form $(s \leftrightarrow u)$ and $(v \leftrightarrow t)$, with $c(s \leftrightarrow u) = c(v \leftrightarrow t) = \infty$. The maximum throughput on G_I in the absence of interference can then be computed as the maximum network flow [13] between s and t in G_I'. The maximum flow computation associates each link with the flow it must carry. We use this amount of flow as the link flow rate to be used by the channel assignment algorithm.

6. The MCAR algorithm

This section introduces the MCAR (Maxflow-based Channel Assignment and Routing) algorithm for the channel assignment problem defined in Section 4. The goal is thus to satisfy the condition (1) which ensures that a given set of flow rates is schedulable. The computed channel assignment must also ensure the connectivity of the induced graph and must be feasible, i.e., the number of channels assigned to a node must not exceed the number of available radios.

As pointed out before, other solutions [2,7], may require recursive adjustments to previous channel assignments. Our algorithm, instead, avoids this problem while ensuring connectivity and feasibility by splitting the channel assignment solution in two stages. In the first stage, links are grouped based on the flows they carry. A group may contain links from many different nodes. For each node, the first stage assures that the number of different groups

assigned to its links does not exceed the number of radio interfaces. The second stage selects a channel for each group and assigns the selected channel to all links of the group. An attempt is made to assign different channels to groups containing interfering links.

Our approach ensures connectivity by assigning a common channel on both the end nodes of every link (thus $E = E_I$). After the second stage of our algorithm, the number of channels assigned to a node does not exceed the number of radios because the first stage returns a number of groups per node not greater than the number of radio interfaces. Thus, the constraint on the number of radios per node is obeyed and no replacement of previous channel assignments is required.

Finally, we note that splitting the algorithm in two stages allows to select channels based on information on the whole network. Indeed, the first stage partitions the set of links into groups enabling the second stage to predict the impact of selecting a channel for a group on the whole network. Next we detail the two stages of our proposed algorithm.

6.1. Link-group binding

We denote by $L(e) \in \mathbb{N}$, $e \in E_I$, the group assigned to link e. Initially, the group of every link is set to zero, meaning no group has been assigned to them yet. We also denote by *neigh*(u) the set of u's neighbors in G_I. If the number of distinct groups assigned to the links incident on u is not larger than $k(u)$, $\forall u \in V$, then the grouping is said to be feasible.

The link-group binding stage of the MCAR algorithm (Fig. 3) visits all the nodes of V one-by-one. For each node u, the set $\mathcal{G}$ of the different groups which the links of u belong to is computed. If the cardinality of $\mathcal{G}$ is greater than the number of available radios on u, we need to remedy such a violation of the feasibility of the grouping. The idea is to repeatedly *merge* a pair of groups until the number of groups equals the number of radios. Clearly, the number of iterations required is given by the difference between such two numbers (lines 6–13). At each iteration, the groups to be merged (denoted by j' and j'') are selected as follows. We define the *group utilization* of a link e as $R(e) = \sum_{e0 \in \mathcal{N}(e): L(e_0) = L(e)} \frac{f(e_0)}{c(e_0)}$, i.e., the sum of the ratio of the flow rate to the capacity of all the links potentially interfering with e that are currently assigned the same group as e. Since all the links of a group are assigned the same channel in the second stage, the group utilization is an indication of the total utilization of the collision domain of link e resulting at the end of the algorithm. Hence, we associate each group with the maximum group utilization over all the links belonging to the group. The two groups to be merged at each iteration are those associated with the least such maximum. A merging reassigns all the links belonging to group j' to group j''.

The next step is to assign a group to all the links of u that are still unassociated with any group. We underline

```
LINK-GROUP(G_I(V, E_I))
 1  L(u ↔ v) = 0   ∀u ↔ v ∈ E_I
 2  g ← 1          ▷ Link group identifier
 3  for each u ∈ V
 4       do G ← {L(u ↔ w)}_{w∈neigh(u)} − {0}
 5          if |G| > k(u)        ▷ Feasibility constraint violated
 6            then for i ← 1 to |G| − k(u)        ▷ Merge groups
 7                      do for each e ∈ E_I : L(e) ∈ G
 8                               do R(e) ← Σ_{e0∈N(e):L(e0)=L(e)} f(e0)/c(e0)
 9                         j' ← argmin_{j∈G} max_{e∈E_I : L(e)=j} R(e)
10                         j'' ← argmin_{j∈G−{j'}} max_{e∈E_I : L(e)=j} R(e)
11                         for each e ∈ E_I : L(e) = j'
12                              do L(e) ← j''
13                         G ← G − {j'}
14          sort {u ↔ w}_{w∈neigh(u)} for decreasing f(u ↔ w)
15          for i = 1 to |neigh(u)|
16               do if L(u ↔ u_i) = 0
17                     then        ▷ Assign a group to the link
18                          if |G| < k(u)
19                            then L(u ↔ u_i) ← g
20                                 G ← G ∪ {g}
21                                 g ← g + 1
22                            else for each e ∈ E_I : L(e) ∈ G
23                                      do compute R(e)
24                                 L(u ↔ u_i) ← argmin_{j∈G} max_{e∈E_I : L(e)=j} R(e)
```

Fig. 3. Pseudo-code MCAR: link-group binding.

that this step does not change any previous group assignment. Preference is given to the links with the largest flow rate, as we sort the links of u in descending order of the flow they carry (line 14). We denote the neighbor of u associated with the ith link by u_i. The links that already belong to a group are skipped in this step. If the number of distinct groups assigned to the links of u is less than the number of available radios, a new group is associated with the link under consideration. Otherwise, the group associated with the least maximum group utilization is selected.

Finally, we note that at the end of the main for loop g is not the total number of groups used because each time a merging is performed, one of the two groups being merged is deleted.

(1) *Proof of correctness:* We require two properties hold at the end of the first stage:

(1) Every link is assigned a group.
(2) For every node, the number of different groups its links belong to is not larger than the number of interfaces.

The first property is clearly satisfied, as we iterate over all the nodes and for each node we assign a group to all of its links. As for the second property, there is one iteration of the main for loop for each node u. Each iteration starts by checking whether the feasibility constraint for u has been violated. In such a case, the appropriate number of mergings is performed to restore the feasibility constraint We note that a merging between two groups j' and j'' does not cause the feasibility property of other

nodes to be violated. Indeed, there are four possible cases for a generic node v other than u:

- There is no link of v belonging to either j' or j'': clearly, v is not affected by the merging.
- Some link of v belongs to j'', none to j': again, v is not affected by the merging.
- Some link of v belongs to j', none to j'': the number of different groups the links of v belong to does not change.
- Some link of v belongs to j' and some other to j'': the number of different groups the links of v belong to decreases by one unit.

Since the number of groups per node cannot increase, the feasibility property is preserved for the other nodes. Next, a group is assigned (if necessary) to the links of u in the respect of the feasibility constraint (18–24). Hence, when the iteration related to a specific node u ends the feasibility constraint for u is satisfied. Subsequent iterations do not violate such constraint.

(2) *Complexity analysis:* We denote by n the number of nodes, by m the number of links and by u the number of neighbors of the generic node. In case the feasibility constraint is violated, the most time-consuming operation to be performed is to compute the group utilization for all the links of the groups in $\mathcal{G}$, which takes $O(m^2)$ time. Since a number of mergings proportional to u has to be performed, remeding a possible violation of the feasibility constraint requires $O(m^2u)$ time.

Sorting the neighbors based on the flow on each link (line 14) takes $O(u \log u)$. Determining the group to be assigned to a link requires $O(m^2)$ in case we need to compute the group utilizations. Hence, assigning a group to all the links of a node takes $O(m^2u)$ time. Thus, the complexity of the whole link-group binding stage is $O(m^3)$.

6.2. Group-channel assignment

The first step of the group-channel assignment stage is to find the set of all the groups assigned to links in E_I. Then, we have to assign a channel to each group.

The GROUP-CHANNEL algorithm (Fig. 4) performs this task with the objective of minimizing the maximum total utilization. This is achieved by sorting the groups in decreasing order of the maximum group utilization associated with any of the links of the group and visiting them one-by-one in such an order. Hence, groups including links with possibly high total utilization are considered first and most likely are assigned channels so as they do not interfere with each other.

We denote by $\mathcal{E}_c$ the set of all links that are assigned to channel c and by $\mathcal{P}(g)$ the set of links *potentially* interfering with the links assigned to group g. Two links potentially interfere with each other if one of the end nodes of one link is in the interference radius of one of the end nodes of the other link. To compute $\mathcal{P}(g)$ we first determine the set I of the end nodes of all the links

```
GROUP-CHANNEL(G_I(V, E_I), C)
 1  for each e ∈ E_I
 2       do groups ← groups ∪ L(e)
 3  P(g) = ∅   ∀g ∈ groups
 4  E_c = ∅   ∀c ∈ C
    ▷ Explore groups in decreasing order of max_{e∈E_I : L(e)=g} R(e)
 5  for each g ∈ groups
 6       do I ← ∅
 7          for each u ↔ v ∈ E_I : L(u ↔ v) = g
 8               do I ← I ∪ {u, v}
 9          for each i ∈ I
10               do for each u ∈ V
11                        do if d(u, i) < r_I
12                                 then for all v ∈ neigh(u) :
                                           L(u ↔ v) ≠ g
13                                      do P(g) ← P(g) ∪
                                           {u ↔ v}
14          compute S(g, c) = P(g) ∩ E_c   ∀c ∈ C
15          if ∃c : S(g, c) = ∅
16            then j ← argmax_{c : S(g,c)=∅} |E_c|
17            else j ← argmin_{c∈C} max_{e∈E_I : L(e)=g} U_tot(e)
18          E_j = E_j ∪ {e ∈ E_I : L(e) = g}
```

Fig. 4. Pseudo-code MCAR: group-channel assignment.

belonging to g. Then, we consider all the nodes in V and, if $u \in V$ is in the interference range of one of the nodes in I, then we add all the links outgoing u that do not belong to g to $\mathcal{P}(g)$. For each channel c, we compute the set $\mathcal{S}(g,c)$ of all the links that are assigned channel c and potentially interfere with links of g. If there exist a channel, let it be c_0, such that the set $\mathcal{S}(g,c_0)$ is empty, it means that none of the links potentially interfering with links of g has been assigned channel c_0. Clearly, it is then a good choice to assign c_0 to the links of g. In case multiple such channels exist, we choose the channel that has already been assigned to the highest number of links (i.e., the channel c such that $|\mathcal{E}_c|$ is maximum). Clearly, among these links none potentially interferes with a link of g.

The rationale behind such a choice is not to *waste* channels. For example, suppose that the first two groups considered do not contain potentially interfering links, while the third group contains links that potentially interfere with links of both the first and the second group. Our scheme causes the second group be assigned the same channel as the first one, so there are $|\mathcal{C}| - 1$ channels that can be assigned to the third group without causing interference. If we chose different channels for the first two groups, then there would be $|\mathcal{C}| - 2$ channels left. If all the sets $\mathcal{S}(g,c)$, $c \in \mathcal{C}$, are non-empty, it means that among the links potentially interfering with links of g there is at least one link assigned to each of the channels. In this case, we choose the channel that minimizes the maximum total utilization of the collision domain of its links. At the end of each iteration of the main for (line 5) all the links belonging to g are assigned the selected channel.

(1) *Proof of correctness:* The GROUP-CHANNEL algorithm assigns a channel to every group, meaning that all the links belonging to a group are assigned the same channel.

(2) *Complexity analysis:* Determining the total number of groups requires $O(m)$ time. Due to the feasibility property of the link-group binding stage, the total number of groups is $O(n)$, where the number of radios per node is considered a constant. Computing the maximum group utilization for all the groups requires $O(m^2)$ time. The groups are then sorted (line 5) in $O(n \log n)$ time.

Finding the set I of the end nodes of links belonging to g (lines 7–8) requires $O(m)$ time, while computing $\mathcal{P}(g)$ (lines 9–13) requires $O(nm)$ time. The number of operations needed to compute the intersection of two sets is linear with their sizes, thus computing $\mathcal{S}(g,c)$ counts for $O(m)$. Selecting a channel and assigning it to all the links of a group requires $O(m^2)$ time. Thus, the main for loop (line 5) requires $O(m^2 n)$ time, which is also the complexity of the GROUP-CHANNEL algorithm.

6.3. Overall MCAR complexity

Computing the maxflow requires $O(nm \log \frac{n^2}{m})$ time if the algorithm in [14] is used. The link-group binding stage and the group-channel assignment stage require $O(m^3)$ and $O(m^2 n)$ time, respectively. Thus, we can conclude that the overall complexity of the MCAR algorithm is $O(m^3)$.

6.4. Mixed Integer Linear Programming (MILP)

Here we present a Mixed Integer Linear Programming (MILP) formulation (Fig. 5) for the channel assignment problem. The purpose is to compare the solution returned by the MCAR algorithm to the optimal solution obtained by solving the MILP. A performance comparison is reported in Section 7.

For each node u and channel c we define a binary variable $x_{u,c}$ which is equal to 1 if and only if one of u's radios is assigned channel c. For each link e and channel c we use a binary variable $y_{e,c}$ to indicate whether channel c has been selected for link e. The binary variable $z_{e_1,e_2,c}$ is set to 1 if and only if both links e_1 and e_2 are assigned channel c.

variables

$$x_{u,c} \in \{0,1\} \quad \forall u \in V, c \in \mathcal{C}$$
$$y_{e,c} \in \{0,1\} \quad \forall e \in E_I, c \in \mathcal{C}$$
$$z_{e_1,e_2,c} \in \{0,1\} \quad \forall e_1, e_2 \in E_I, c \in \mathcal{C}$$
$$M \in \mathbb{R}^+$$

minimize M

subject to

1) $M \geqslant U_{tot}(e) \quad \forall e \in E_I$
2) $U_{tot}(e) = \sum_{c \in \mathcal{C}} \sum_{e_0 \in \mathcal{N}(e)} z_{e_0,e,c} \frac{f(e_0)}{c(e_0)} \quad \forall e \in E_I$
3) $\sum_{c \in \mathcal{C}} x_{u,c} \leqslant k(u) \quad \forall u \in V$
4) $\sum_{c \in \mathcal{C}} y_{e,c} = 1 \quad \forall e \in E_I$
5) $x_{u,c} + x_{v,c} \geqslant 2y_{e,c} \quad \forall e = u \leftrightarrow v \in E_I, c \in \mathcal{C}$
6) $y_{e_1,c} + y_{e_2,c} \geqslant 2z_{e_1,e_2,c} \quad \forall e_1, e_2 \in E_I, c \in \mathcal{C}$
7) $y_{e_1,c} + y_{e_2,c} - 1 \leqslant z_{e_1,e_2,c} \quad \forall e_1, e_2 \in E_I, c \in \mathcal{C}$

Fig. 5. MILP formulation.

We recall that the channel assignment problem is to minimize the maximum among the total utilization of all the collision domains. In order to get a linear objective function, we introduce a real variable M which is to be minimized subject to the constraint that it must be larger than every total utilization (set of constraints 1). Total utilizations can be expressed as a linear combination of the ratio of the flow rate to the capacity by means of the $z_{e_1,e_2,c}$ variables (set of constraints 2). The set of constraints 3 impose that each node u has $k(u)$ radios, while the set of constraints 4 impose that each link is assigned just one channel. Each constraint of the set 5 is equivalent to the predicate $y_{u \leftrightarrow v,c} = 1 \Rightarrow x_{u,c} = 1 \wedge x_{v,c} = 1$, i.e., a necessary condition to assign a channel to a link is that a radio of both the end nodes is assigned that channel. Finally, for each pair of links, the constraints in 6 and 7 assure that $z_{e_1,e_2,c} = 1 \iff y_{e_1,c} = 1 \wedge y_{e_2,c=1}$, i.e., $z_{e_1,e_2,c}$ is set to 1 if and only if both links e_1 and e_2 are assigned channel c.

7. Performance analysis

We performed a set of simulations to evaluate the performance of the MCAR algorithm. The results are illustrated in the next subsections.

7.1. Comparing MCAR to the optimum

The MCAR algorithm provides an approximate solution to the channel assignment problem defined in Section 4, i.e., the problem to find, given a set of flow rates, a channel assignment which minimizes the maximum total utilization over all the collision domains. The optimal solution can be instead obtained by solving the MILP shown in Fig. 5. The aim of this subsection is to evaluate the approximation ratio of the MCAR algorithm through simulations.

Given the hardness of the MILP due to the presence of integer variables, we were forced to consider small topologies in order to keep the size of the model small. We considered topologies with 10, 11 and 12 nodes. For each such classes, we generated 20 different topologies having different placement of nodes (and hence number of links), number of radios and channels. The flow rates were determined as described in Section 5. For each topology, we determined the maximum total utilization resulting from the channel assignment returned by MCAR (denoted as $U_{\text{tot}}^{\max}(\text{MCAR})$) and the optimal value obtained by solving the MILP ($U_{\text{tot}}^{\max}(\text{MILP})$). The MILP model was solved by using the `lp_solve` library [15], a free mixed integer programming solver. The results of the simulations are illustrated in Fig. 6, which shows the distribution of the

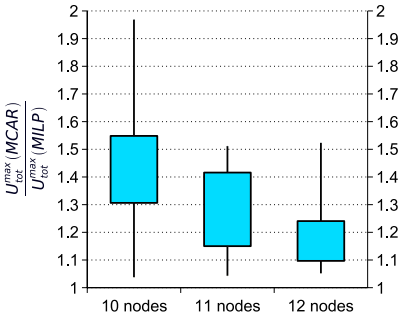

Fig. 6. Ratio of the solution returned by MCAR to the optimal one.

ratio $\frac{U_{tot}^{max}(MCAR)}{U_{tot}^{max}(MILP)}$ for each class of topologies. In particular, a vertical line spans from the minimum to the maximum ratio obtained over the 20 topologies of a class and a box extends from the first quartile to the third quartile. The results indicate that the solution returned by MCAR is at most twice the optimal solution.

7.2. MCAR: Performance evaluation

The MCAR algorithm is part of a solution to the joint channel assignment and routing problem and, as such, attempts to find a channel assignment which makes the given set of flow rates schedulable. In this section we compare our proposal to similar algorithms: LACA (Load-Aware Channel Assignment) [2] and BSCA (Balanced Static Channel Assignment) [8]. The comparison is based on their "ability" to compute a channel assignment which makes the given set of flow rates schedulable. We recall that a sufficient condition for a set of flow rates to be schedulable is that the total utilization is less than or equal to 1 for all the collision domains. We therefore compare the above mentioned channel assignment algorithms by evaluating the index $\Omega \stackrel{\text{def}}{=} \frac{\sum_{e \in E} \max(U_{tot}(e)-1,0)}{|E|}$, i.e., the average over all the collision domains of the total utilization in excess with respect to the unitary value which denotes schedulability.

We considered two classes of topologies with 25 and 50 nodes uniformly distributed in a $300 \times 300\ m^2$ field and a $400 \times 400\ m^2$ field, respectively. The probability that a mesh router be an aggregation device or a gateway is 0.15 each (such value only influences the MCAR operation through the maxflow computation). The transmission and interference ranges are 90 and 180 m. In the 25-nodes topologies, 60% of the mesh routers have 2 radios while the remaining ones have 3 radios. In the 50-nodes topologies, the distribution probability of the number of radios per node is: 20% of nodes have 2 radios, 40% have 3 radios and another 40% have 4 radios. For each class, we generated 20 different topologies having different placement of nodes (and hence number of links). We considered two scenarios: the capacity of a link is fixed at 54 Mbps (OnOff scenario) or ranges from 54 to 6 Mbps depending on the distance between the incident nodes (Step scenario). For each scenario, we considered four different cases, where the number of available channels is, respectively, 3, 6, 9 and 12. A channel assignment was computed by using the algorithms under test for all the 20 topologies of each class and for all the cases of each scenario.

The results of the simulations are shown in Figs. 7 and 8 for the 25 and 50 nodes topologies, respectively. Each figure illustrates both the OnOff and the Step scenarios. For each value of the total number of available channels, the figures show the average values of the index Ω over all the 20 topologies of each class for every channel assignment algorithm. We can observe that the best performance is achieved by MCAR. In case only three channels are available, the performance of the three algorithms is nearly the same. We can explain such a behavior because if three channels are available and nodes have 2 or 3 radios, then each channel is likely to be used for one radio of every node. However, as the number of available channels becomes larger than the number of radios per node, the performance increase achieved by MCAR becomes more evident.

7.3. Evaluation of throughput and delay

The channel assignments computed by different algorithms differ in the composition of the collision domains

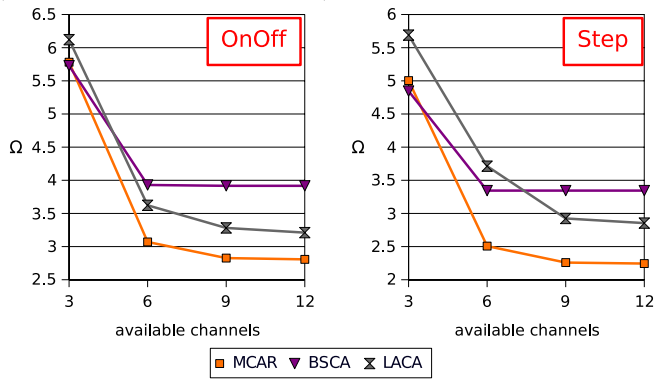

Fig. 7. Average value of the index Ω for the 25 nodes topologies.

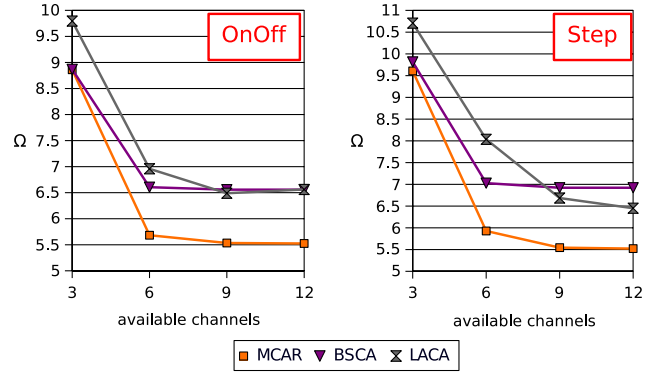

Fig. 8. Average value of the index Ω for the 50 nodes topologies.

and hence can lead to different network performances. Therefore, we carried out a simulation study using the ns-2 network simulator to evaluate the difference in the network throughput and the delay of packets achieved when assigning channels using MCAR, LACA and BSCA. Traffic is routed using the AODV (Ad-hoc On-demand Distance Vector) [16] protocol. We needed to modify ns-2 to support multiple wireless radios on mobile nodes.

For each ns-2 simulation, traffic sources generate exponential on–off TCP traffic and the average throughput over the whole simulation is measured. Such value is then averaged over all the 20 topologies of each class of topologies (25- and 50-nodes). The results are shown in Figs. 9 and 10 for different values of the number of available channels. These results clearly indicate that MCAR also enables to increase the network throughput with respect to LACA and BSCA, even by a factor of 2 in many cases.

We also show the packet delay measured for a 25-nodes (Fig. 11) and a 50-nodes (Fig. 12) topologies when using different channel assignment algorithms. The delay experienced by a group of 100 packets with consecutive identifiers (ns-2 assigns a unique id to every packet, independently of its source node) is averaged and such average value is plotted. Thus, Figs. 11 and 12 are made, respectively, by measuring the delay of more than 9000 and 12,000 packets. Such figures clearly indicate that packets experience a lower delay in case channels are assigned by the MCAR algorithm. We also measured the number of packets dropped in these experiments. The ratio of lost packets to the total number of packets sent is reported in Table 1 for the channel assignment algorithms under evaluation. Again, we can observe that MCAR achieves the best performance, as the resulting probability of packet loss is the lowest.

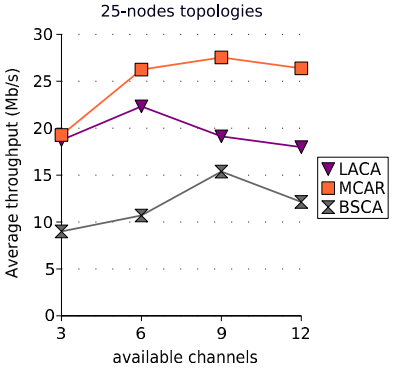

Fig. 9. Network throughput: 25-nodes topologies.

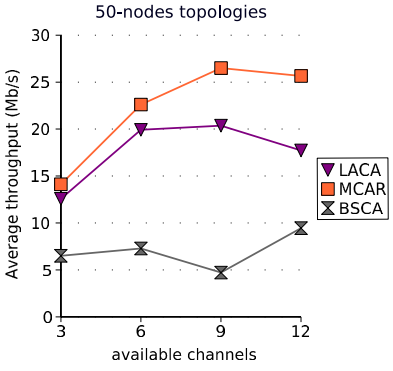

Fig. 10. Network throughput: 50-nodes topologies.

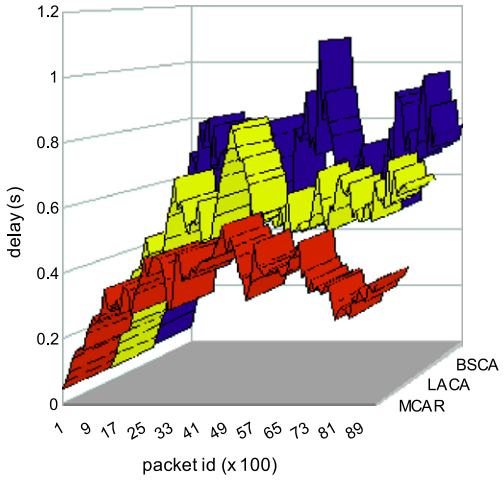

Fig. 11. Packet delay: 25-nodes topologies.

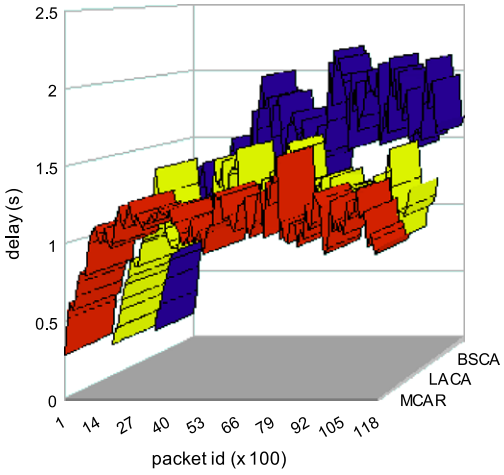

Fig. 12. Packet delay: 50-nodes topologies.

Table 1
Packet loss ratio

	MCAR	LACA	BSCA
25-nodes	0.09	0.32	0.43
50-nodes	0.15	0.30	0.40

8. Conclusions

This paper addresses a fundamental design issue in multi-radio wireless mesh networks, namely the channel assignment and routing scheme. We propose a flow rate

computation method which is independent of the traffic demands and aims to maximize the network capacity. The proposed channel assignment algorithm attempts to make a given set of flow rates schedulable. We illustrate our algorithm in detail, prove its correctness and calculate the complexity. Performance studies show that the MCAR algorithm outperforms previous proposals and leads to better network performance.

Acknowledgement

Research outlined in this paper has been partially supported by the European Union under the CONTENT Network of Excellence FP6-0384239 and by the Italian Ministry for Education, University and Research (MIUR) in the framework of the NADIR Project (PRIN Program).

References

[1] I.F. Akyildiz, X. Wang, W. Wang, Wireless mesh networks: a survey, Computer Networks 47 (4) (2005) 445–487.
[2] A. Raniwala, K. Gopalan, T. Chiueh, Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks, ACM Mobile Computing and Communications Review 8 (2) (2004) 50–65.
[3] A. Raniwala, T. Chiueh, Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network, in: Proceedings of IEEE INFOCOM 2005, vol. 3, pp. 2223–2234.
[4] M. Alicherry, R. Bhatia, E. Li, Joint channel assignment and routing for throughput optimization in multiradio wireless mesh networks, IEEE Journal on Selected Areas in Communications 24 (11) (2006) 1960–1971.
[5] R. Draves, J. Padhye, B. Zill, Routing in multi-radio, multi-hop wireless mesh networks, in: Proceedings of ACM Mobicom'04, 2004, pp. 114–128.
[6] P. Kyasanur, N.H. Vaidya, Routing and interface assignment in multi-channel multi-interface wireless networks, in: Proceedings of IEEE WCNC 2005, vol. 4, pp. 2051–2056.
[7] J. Tang, G. Xue, W. Zhang, Interference-aware topology control and QoS routing in multi-channel wireless mesh networks, in: Proceedings of ACM MobiHoc'05, 2005, pp. 68–77.
[8] M. Kodialam, T. Nandagopal, Characterizing the capacity region in multi-radio multi-channel wireless mesh networks, in: Proceedings of ACM MobiCom'05, 2005, pp. 73–87.
[9] P. Gupta, P.R. Kumar, The capacity of wireless networks, IEEE Transactions on Information Theory 46 (2) (2000) 388–404.
[10] A. Caprara, H. Kellerer, U. Pferschy, The multiple subset sum problem, SIJOP: SIAM Journal on Optimization 11 (2000) 308–319.
[11] K. Jain, J. Padhye, V. Padmanabhan, L. Qiu, Impact of interference on multi-hop wireless network performance, in: Proceedings of ACM MobiCom'03, 2003, pp. 66–80.
[12] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, second ed., MIT Press, 2001.
[13] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows: Theory Algorithms, and Applications, Prentice-Hall, Englewood Cliffs, NJ, 1993.
[14] B.V. Cherkassky, A.V. Goldberg, On implementing push-relabel method for the maximum flow problem, Algorithmica 19 (1997) 390–410.
[15] lp_solve, Available from: <http://tech.groups.yahoo.com/group/lp_solve/>.
[16] C. Perkins, E. Belding-Royer, S. Das, Ad hoc on-demand distance vector (AODV) routing, IETF, RFC 3561, July 2003.