Minimal CDMA Recoding Strategies in Power-Controlled Ad-Hoc Wireless Networks

Indranil Gupta
Department of Computer Science
Cornell University
Ithaca, NY 14853, USA
gupta@cs.cornell.edu *

Abstract

The problem of Code Division Multiple Access (CDMA) code assignment to eliminate primary and hidden collisions in multihop packet radio networks has been widely researched in the past. However, very little work has been done on the very realistic distributed, dynamic version of the CDMA transmitter-oriented code assignment (TOCA) problem in an ad-hoc network. None of the existing dynamic TOCA CDMA algorithms in literature are efficient, in terms of maximum code index assigned in the network, or number of times a mobile has to change its code. We present a set of local, distributed recoding strategies for the TOCA CDMA problem in an ad-hoc network where mobiles can arbitrarily 1) connect and disconnect, 2) move about, and 3) increase or decrease their transmission power - all these may need some mobiles to be recoded, to avoid new collisions. Our strategies, unlike those proposed earlier in literature, guarantee minimal recoding, that is, given a current network-wide code assignment and one of the above events, our strategies change the codes of the minimum number of mobiles needed to eliminate all collisions. Minimal recoding can be very important in reducing the effect of frequent code changes on the performance and criticality of distributed applications. Performance results that evaluate our dynamic minimal strategies are also presented.

1. Introduction

Ad-hoc wireless networks are characterized by lack of an established infrastructure such as an underlying wired network or base stations. However, their potential uses range widely from scenarios where an ad-hoc network could be just convenient, such as a conference where members com-municate with each other, to critical ones, such as networks formed on the fly by satellite constellations, on the battlefield etc. [13].

Transmissions in such wireless media could lead to *collisions*, where transmissions are garbled at the receiving end. This could be either a *primary* collision, where an incoming transmission is damaged by a simultaneous outgoing transmission from the receiving mobile, or a *secondary* (also *hidden*) collision, where two incoming transmissions garble each other. *C*ode *D*ivision *M*ultiple *A*ccess is a widely used technology that completely eliminates collisions by techniques such as spread spectrum and *orthogonal codes*. We consider only the case of orthogonal codes. CDMA protocols also require that either receivers, or transmitters, or both, are code-agile, that is, are able to communicate over a range of codes. We are concerned only with the first kind, which leads to the Transmitter Oriented Code Assignment (TOCA) problem [3]. In this problem, with each mobile (computer+transceiver) modeled as a node, and each code modeled as a positive integer, codes have to be assigned to different nodes in the network, one code per node, in a 1) correct manner to eliminate all collision *constraints* (i.e., all primary and hidden collisions) and 2) an efficient manner, such as say, to minimize the maximum code index assigned to any network node. This problem has been extensively studied for static multihop networks [2, 3, 9, 12, 14]. Finding an optimal TOCA code assignment in terms of the maximum code index assigned in the network has been mapped to the graph coloring problem [4], where codes are represented by colors - this has been shown to be NP-complete [3]. Several centralized and distributed heuristics have been proposed for the same [2, 3, 5, 9, 10, 12, 14].

In an ad-hoc network, nodes are free to 1) move about, 2) connect or disconnect from the network, and 3) increase or decrease their transmission ranges (such a capability is often mandatory due to the power-sensitivity of CDMA transceivers, besides other advantages such as saving power,

controlling network connectivity and throughput, security etc. [1, 8, 11]). These *events* may invalidate any statically generated code assignment by introducing new *conflicts* in the assignment, that is, by causing new collisions. A *recoding*, that is, a change in the code assignment of some network nodes, is needed to eliminate these new collisions. In general, a *recoding strategy* is a set of algorithms (one for each of the above event types) for a reassignment of codes to some of the nodes in the network to maintain the correctness of the code assignment. Centralized code assignment algorithms such as those of [2, 3, 9, 10, 12, 14] are inappropriate in an ad-hoc network as they determine a new code assignment for every node on each event. The distributed heuristics of [3, 9] are also inadequate as they assume a static network. Probably the only prior works to propose distributed solutions in a dynamic ad-hoc network-like scenario are [5, 10]. None of these papers consider the recoding problem arising out of a change in transmission range.

In this paper, we give a set of efficient *recoding algorithms* for the above events. These algorithms involve communication only local to the event and are *distributed*, i.e., they require no central coordination. Our algorithms (provably) satisfy the important goal of *Minimal Recoding*, that is, a recoding strategy must try to minimize the number of nodes that are recoded (with a new code) on any network event. More concretely, given a current code assignment and one of the above events, among all possible recoding strategies, our algorithms achieve the lowest bound on the number of nodes that need to be recoded to eliminate all conflicts in the network. This could be useful, even critical, in ad-hoc networks where frequent recoding might be costly to the applications using the communication medium. Examples include hard real-time applications [15], and applications where maintaining a persistent high data rate is critical to its performance. Moreover, our algorithms for most of the above ad-hoc network events are (provably) *optimally minimal*, that is, given a current code assignment and an event, among all possible recoding strategies (for that event) that are minimal and consider recoding only nodes one hop away from the initiating node, our strategy (for that event) achieves the optimal (least) increase in the maximum code index assigned to the network. We also present simulation results that verify our hypothesis that our strategies would indeed be practicable in an ad-hoc network in the long run, and perform better than previously suggested strategies. In addition, our strategies can also be used as orthogonal recoding algorithms to any global code assignment heuristic in a dynamic ad-hoc network.

The rest of the paper is organized as follows. Section 2 discusses the assumed network model, and our high-level and concrete goals in designing the dynamic recoding algorithms. Section 3 describes on the previous work on this problem. Section 4 presents our recoding algorithms and Section 5 presents performance results for them. We conclude in Section 6.

2. Model and Problem Statement

A power controlled ad-hoc network is modeled as a dynamic directed graph (digraph) $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$ the (current) set of nodes in the network. Each vertex v_i in V has a configuration defined by its current position coordinates ((x_i, y_i) in a 2-dimensional network) and a (variable) maximum transmission power range r_i which specifies the *maximum* distance from (x_i, y_i) that other nodes in the network can hear or are affected by interference from its transmissions. The set of edges $E = \{(v_i, v_j) : i \neq j \ \& \ d_{ij} \leq r_i\}$ consists of directed edges of the type $v_i \rightarrow v_j$ if and only if v_j is within v_i's transmission range, that is, if the distance d_{ij} between v_i and v_j is less than r_i. Note that this can be easily generalized for the non-free-space propagation case where, due to obstacles, although $d_{ij} \leq r_i$, $(v_i, v_j) \notin E$. Node i's assigned code is denoted by c_i and is a positive integer.

Nodes can arbitrarily join or leave the network, increase or decrease their power range r_i, and move about within the network; call each of these *events* or *reconfigurations* in the network. For simplicity, we make the following assumptions, the latter two of which are very realistic, and the first of which can be relaxed.

1. Network *events* or *reconfigurations* occur throughout the ad-hoc network one after the other and not simultaneously (see [7] on how to relax this assumption and parallelize our recoding algorithms).
2. Nodes move and change their ranges in discrete (and not continuous) steps.
3. **Minimal Connectivity:** A node v_i can change its configuration if and only if there are nodes v_j, v_k $(j, k \neq i)$ in the new configuration such that v_j is within v_i's transmission range, and v_i is within v_k's transmission range.

The TOCA code assignment problem [2] is to assign a code (equivalently, a color), which is essentially a positive integer, to each node in the network so that the following constraints CA1 and CA2 are satisfied throughout the network at all times.

Condition CA1 - (Primary) Collision Avoidance 1: For every edge $(v_i, v_j) \in E$, $c_i \neq c_j$.
Condition CA2 - (Secondary) Collision Avoidance 2: For every pair of edges $(v_i, v_k), (v_j, v_k) \in E$ & $i \neq j$, $c_i \neq c_j$.

Fig 1(a) shows a snapshot of an ad-hoc network containing 4 nodes $\{1, 2, 3, 4\}$, with their maximum transmission
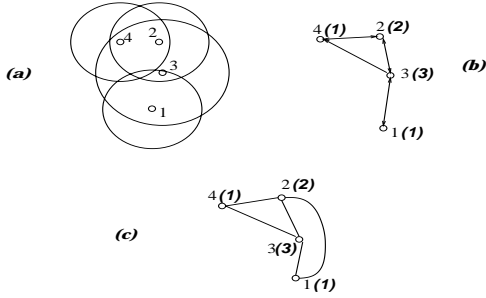
Figure 1. Example Ad-hoc Network with (a) nodes and transmission ranges, (b) induced digraph and (c) coloring constraints (two nodes connected by an edge cannot have the same code) and a correct assignment.

ranges. Fig 1(b) shows the directed graph model of this ad-hoc network and Fig 1(c) shows the constraints in the TOCA problem for the above network. The indices against the nodes show the optimal color assignment to satisfy CA1 and CA2.

Ideally, for each event, an efficient recoding strategy should attempt to (1) minimize the maximum code index used by any node in the network, (2) (*Minimal Recoding*) minimize the number of nodes that change their code, (3) minimize the overhead of recoding, and (4) keep the recoding strategy distributed and local. Goal 1 is needed because the hardware of a node can be designed to transmit on only some maximum number of codes. Goal 2 is very important for an ad-hoc network since recoding can be very costly, as mentioned in section 1. These two goals are contradictory - global coloring algorithms satisfy only goal 1 (ex. [3]) while local coloring algorithms may satisfy only goal 2. Our minimal approaches to recoding obtain a tradeoff between these two goals, while achieving the other goals. They give the best performance in terms of minimum number of nodes recoded among all possible strategies while using almost as few colors as a global coloring heuristic, in fact differing only by a few colors.

Henceforth, during the recoding for an event, for each network node, the set of colors that it cannot be assigned since it would violate either CA1 or CA2 with some other node, will be called its *constraints*. *Conflicts* will refer to the violation of CA1 or CA2 somewhere in the code assignment due to the event. In the rest of the paper, the terms "color" and "code"'are used interchangeably, as are "recoloring" and "recoding".

3. Previous Work

We have touched on related work in Section 1. Here, we elaborate on the only prior works considering the TOCA

problem in the dynamic and distributed context [5, 10]. We will compare our recoding approaches to these strategies. Both works are similar in spirit and give recoding strategies for nodes joining and leaving a dynamic network with symmetric links, but they can be extended to the asymmetric case.

The *CP* Recoding Strategy of [5] works as follows. When a new node joins the network, it contacts all its neighbors. The new node and its 1-hop neighbors exchange information about their current assigned colors and the constraints induced on each by the colors of nodes 1 and 2 hops away. All pairs of nodes 1 hop away from the new node which have the same colors violate CA2 and have to select new colors. In addition, the new node has to select a color that does not violate the constraints induced by nodes 1 and 2 hops away from it. This is achieved by having each nodes continuously check if it is the highest (or lowest)-identity node in its vicinity (defined by itself and nodes up to 2 hops away from it) that has not yet been assigned a color. The node selects the lowest available color (not yet taken by any of its 1 hop and 2 hop neighbors) when this condition is satisfied. The ordering by identities and respect for constraints ensures that no conflicts arise among nodes choosing new colors and with those whose colors will not change. When a node leaves the network, its neighbors update their lists, if any, about the constraints placed on them for future color selection. No recoding is required in this case. Node movement is handled as a pair of events consisting of a node disconnection and connection from/to all its neighbors.

4. New Recoding Strategies

This section presents our recoding strategies. Sections 4.1, 4.2, 4.3, 4.4 respectively discuss the recoding strategies for node join, power range increase, power range decrease or leave, and movement.

4.1. Handling a Node Join

Consider the event where a new node n joins the ad-hoc network, as shown in Fig 2. Fig 2 shows a partition of the old nodes into three sets, where n has incoming edges from nodes in 1_n and 2_n, outgoing edges to nodes in 2_n and 3_n and no edges to nodes in the set 4_n. We call the colors assigned to the nodes in the network just prior to n joining as their *old* colors and those assigned to them after the $RecodeOnJoin$ operation finishes as *new* colors (which, for a node, may be the same as its old color). Also, we call the constraints to be taken into account for the new coloring (recoding) as *new* constraints.

Consider the new constraints/conflicts created by this join. From CA1 and CA2, observe that all nodes in 1_n, 2_n and $\{n\}$ each need to have colors different from each other.
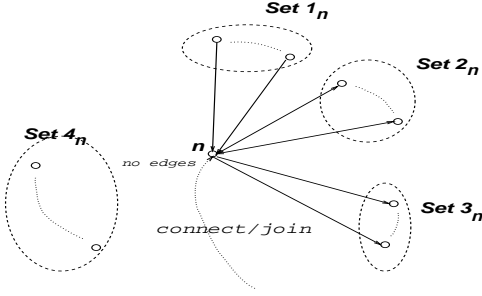
Figure 2. Node n joins the network

RecodeOnJoin(Node n**)**

1 Obtain the constraints $(u, oldcolor(v))$ of the from-neighbors u of n,
   $u \in 1_n \cup 2_n, v \notin 1_n \cup 2_n \cup \{n\}$.
2 Obtain the constraints $(n, oldcolor(v))$ for $n, v \notin 1_n \cup 2_n$.
3 Let $max =$ the maximum color seen in these constraints or
   old colors in $1_n \cup 2_n$.
4 Let $V_1 = 1_n \cup 2_n \cup \{n\}, V_2 = \{1, \ldots, max\}$.
      Draw the bipartite graph G' by joining edges from each vertex
      v in V_1 to each color k in V_2 that it can be assigned
      without conflicting with the constraints with any of the nodes
      not in $1_n \cup 2_n \cup \{n\}$.
         Assign this edge weight 3 if this is the old color assigned
         to v, otherwise assign it a weight of 1.
5 Run the bipartite matching algorithm on G'. For each edge in v
   that is matched to some edge (v, k), assign it k as the new color.
      For all nodes in V_1 not assigned a color above, say m of them,
      randomly assign them colors $max + 1, \ldots, max + m$
6 Dissipate this information to all concerned nodes, agreeing
   on when to change color.

Figure 3. $RecodeOnJoin$

However, nodes in 3_n need not change their color since n will be assigned a new color anyway and this will need to be different from any of the colors in 3_n. Recollect the goal of *Minimal recoding* we set down in section 2. Keeping this in mind, instead of recoding any nodes more than 1 hop away from n, we will attempt to minimize the total number of codes changed in the set $1_n \cup 2_n \cup \{n\}$.

Now, note that if a K-sized subset of nodes in $1_n \cup 2_n$ have the same old color, only $K - 1$ need to change their color, and one of them can maintain the same color in the new code assignment. More generally, if the set of old colors of the nodes in $1_n \cup 2_n$ is $\{C_1, C_2, \ldots, C_m\}$ and the associated number of nodes in $1_n \cup 2_n$ with these corresponding colors is $\{K_1, K_2, \ldots, K_m\}$, apart from recoding n, at least $\sum_{i=1}^{m}(K_i - 1) = \sum_{i=1}^{m} K_i - m$ of the nodes in $1_n \cup 2_n$ need to be recoded with different new colors to avoid conflicts in the new code assignment after node n joins. This is the minimal recoding bound.

Why ? Clearly CA2 tell us that all nodes in $1_n \cup 2_n$ have to have different new colors after the recoding. Suppose, by contradiction, that less than $\sum_{i=1}^{m}(K_i - 1) = \sum_{i=1}^{m} K_i - m$ of the nodes in $1_n \cup 2_n$ are recoded with new colors when node n joins. This means that at least $|1_n \cup 2_n| - (\sum_{i=1}^{m}(K_i-1)-1) = |1_n \cup 2_n| - \sum_{i=1}^{m} K_i + m + 1 = m + 1$ nodes in $1_n \cup 2_n$ retain their old colors after the recoding. However, since the old code assignment $\{C_1, C_2, \ldots, C_m\}$ to $1_n \cup 2_n$ had just m colors, this means that at least two nodes in $1_n \cup 2_n$ will have the same color after the recoding, a contradiction to CA2.

Now, the questions that arise are, which $\sum_{i=1}^{m}(K_i - 1)$ nodes from $1_n \cup 2_n$ do we chose for recoding, and what colors do we assign them ? Our solution is the following. Consider the undirected graph $G' = (V_1 \cup V_2, E')$ where $V_1 = 1_n \cup 2_n \cup \{n\}, V_2 = \{i : i \in Z^+ \,\&\, i \leq max=$ maximum color constraint in the vicinity of $1_n \cup 2_n \cup n\}$, $E = \{(u,v) : u \in V_1, v \in V_2,$ u is not constrained to be colored newly with v$\}$. By "maximum color constraint in the vicinity of $1_n \cup 2_n \cup n$", we mean the maximum integer among all constraints due to $3_n \cup 4_n$ on nodes in $1_n \cup 2_n \cup n$

(i.e., the set of old colors in $3_n \cup 4_n$ forbidden to be assigned to nodes in $1_n \cup 2_n \cup n$ due to CA1 and CA2), and all old colors in $1_n \cup 2_n$. The edges in E are weighted; edges of the kind (u,v), $u \in 1_n \cup 2_n, v \in V_2$ where v is the old color assigned to node u are assigned weight 3; all other edges have weight 1. Note that G' is an instance of a *bipartite graph*.
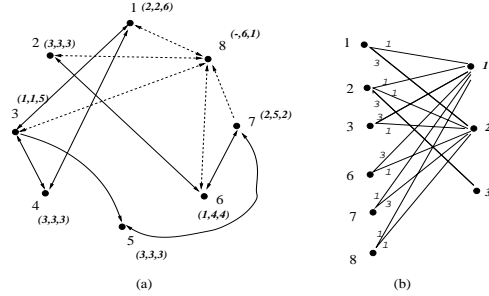

Figure 4. (a) Node 8 joins the network. Dotted edges added by the join. Against each node is shown its (old color, new color by $RecodeOnJoin$, new color by *CP*) (b) Weighted bipartite graph used by $RecodeOnJoin$; dark edges show matching constructed by $RecodeOnJoin$. $max = 3$ here.

Next, consider a maximum matching M ($\subset E$) on this bipartite graph i.e., a set of edges with no common end-vertices and with the maximum total weight among all such sets. Efficient algorithms exist to find a maximum matching on any weighted bipartite graph[6]. We shall not enumerate further on these algorithms but treat them as a black box in the ensuing discussion. Our recoding strategy is to assign a node u in V_1 to the color in V_2 that M matches it to, and for all $u \in V_1$ not matched by M, assign them consecutive col-

ors one by one starting from $|V_2|+1$ onwards. This strategy suffices to construct a minimal recoding for the event of a node n joining. The proof of this is excluded due to space constraints and can be found in [7].

The pseudo-code for the $RecodeOnJoin$ algorithm executed by a new node n, which uses the maximum weighted matching on G', is shown in Fig 3 and is self-explanatory. Note that this is a local recoding strategy since the onus of recoding the nodes in $1_n \cup 2_n \cup \{n\}$ is *locally centralized* at node n, and uses only local information. Termination and correctness properties of this algorithm are proved in [7]. For brevity, we state here the theorems regarding the claimed minimality and optimality among minimal algorithms properties of our algorithm - their proofs can be found in [7].

Theorem (Minimality): At the event where n joins the network, $RecodeOnJoin$ achieves the minimal recoding bound among all possible recoding strategies. □

Theorem (Optimality among Minimality): At the event where n joins the network, among all correct recodings of the network that achieve the minimal recoding bound and consider only nodes in $1_n \cup 2_n \cup \{n\}$ for recoding, $RecodeOnJoin$ gives us one which (re-)assigns the least maximum color to any node. □

An example for the recoding by the *CP* strategy and $RecodeOnJoin$ when a new node (8) joins an ad-hoc network is shown in Fig 4. The reader is encouraged to work through this simple example to understand the presented algorithm. Note that $RecodeOnJoin$ causes only 3 recodings while the *CP* join strategy (which uses a highest-first node ordering) causes 4 of them. Both end up using the same maximum color index after the join event (6).

Ignoring the latency of dissipation of color information in steps 1, 2 and 6, the complexity of $RecodeOnJoin(n)$ is dominated by the bipartite matching step 4. If the maximum in- and out-degree of any node in the network is k, $RecodeOnJoin$ has a complexity of $O(k^9 ln(k))$. In a planar ad-hoc network, k would be expected to be a constant (as in planar graphs), thus giving us a constant expected time complexity for $RecodeOnJoin$.

4.2. Handling Node Power Increase

Let us look at Fig 2 again and envision what happens to the constraints on the nodes in $1_n, 2_n, 3_n, 4_n$ when n increases its maximum power range r_i by some amount. Nodes which were earlier in set 4_n might now be included in 3_n and nodes earlier in 1_n might jump into 2_n. However, note that *no new constraints are induced among the nodes in $1_n \cup 2_n \cup 3_n \cup 4_n$ due to this*. In other words, *all* constraints due to CA1 *and* CA2 added by the new edges involve node n !! If n's old color can no longer be assigned to it because of a new constraint, then the minimal recoding

RecodeOnPowIncrease(Node n)
1 Obtain the (new) constraints $(n, oldcolor(v)), v \neq n$ for self (n).
2 If current color does not violate new constraints, stop.
3 Recode n with the lowest available color that does not violate any of the constraints.
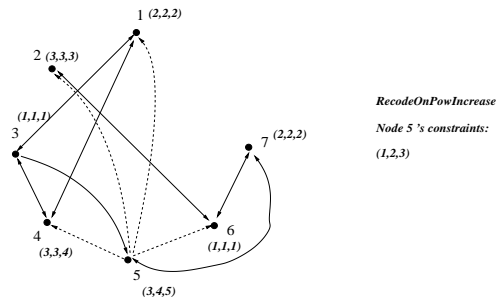
Figure 5. $RecodeOnPowIncrease$

Figure 6. Node 5 increases its range. Dotted edges added to network. Against each node is shown its (old color, new color by $RecodeOnPowIncrease$, new color by CP)

would need at least one node to change its color - we chose n to be this node, thus achieving the minimal bound. If n's color has no conflict with the new constraints on n, then the minimal recoding changes no colors. This is exactly what algorithm $RecodeOnPowIncrease$ does (Fig 5).

We now extend the CP strategy to account for recoding on power range increase. When a node n increases its power range, all nodes up to two hops away from n that now have a new constraint (due to either CA1 or CA2) with n and the same old color as n (and thus have a conflict with n), consider themselves for recoding. These nodes, along with n, do so in a distributed fashion in increasing or decreasing order of their identities, in a manner similar to the algorithm presented in section 3.

An example comparing the performance of the CP and $RecodeOnPowIncrease$ strategies is shown in Fig 6, where node 5 increases its maximum transmission range to now include node $1, 2, 4, 6$ within its receiving range, thus setting up new constraints. $RecodeOnPowIncrease$ causes only 1 new recoding while the CP strategy causes 2 nodes to be assigned different new colors. $RecodeOnPowIncrease$ ends up with a lower maximum color index in the network (4) as against CP's 5.

The termination, correctness and minimality properties of this algorithm are formally stated and proved in [7]. However, note that $RecodeOnPowIncrease$ may not always achieve the optimal bound among all minimal recoding strategies for recoding when a node increases its r_n.
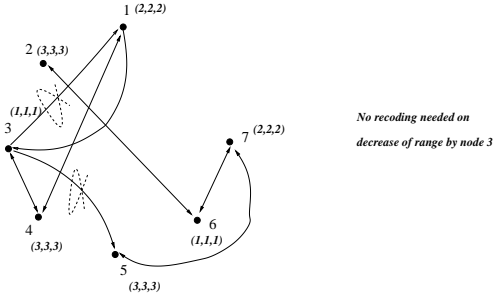
Figure 7. Node 3 decreases its power range. Crossed out edges deleted and old code assignment is valid. Against each node is shown its (old color, new color by $RecodeDecreasePowOrLeave$, new color by CP)

RecodeOnMove(Node n**)**

0 Define $1_n, 2_n, 3_n, 4_n$ for the node n in its new position as in section 4.1.
1 Obtain the constraints $(u, oldcolor(v))$ of the from-neighbors u of n, $u \in 1_n \cup 2_n, v \notin 1_n \cup 2_n \cup \{n\}$.
2 Obtain the constraints $(n, oldcolor(v))$ for $n, v \notin 1_n \cup 2_n$.
3 Let $max =$ the maximum color seen in these constraints and in the old colors of nodes in $1_n \cup 2_n$.
4 Let $V_1 = 1_n \cup 2_n \cup \{n\}$, $V_2 = \{1, \dots, max\}$.
   Draw the bipartite graph G' by joining edges from each vertex v in V_1 to each color k in V_2 that it can be assigned without conflicting with the constraints with any of the nodes not in $1_n \cup 2_n \cup \{n\}$.
   Assign this edge weight 3 if this is the old color assigned to v, otherwise assign it a weight of 1.
5 Run the bipartite matching algorithm on G'. For each edge in v that is matched to some edge (v, k), assign it k as the new color.
   For all unmatched vertices in V_1, say m of them, randomly assign them colors $max + 1, \dots, max + m$
6 Dissipate this information to all concerned nodes, agreeing on when to change color.
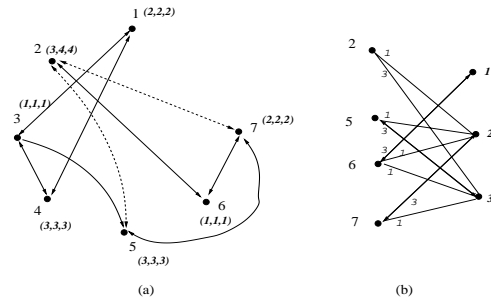
Figure 8. $RecodeOnMove$

Consider the example of n having only one new constraint with another node m. If n has lots of old constraints (which still hold) and m very few, recoding only m might be more optimal in terms of maximum color index assigned to the network, while achieving the minimal recoding bound.

$RecodeOnPowIncrease$'s time complexity, ignoring the constraint collection step 1, is $O(k^2)$, where $k =$ maximum in- and out-degree of any node in the network.

4.3. Handling Node Leaves and Power Decreases

As in [5], our approach for $RecodeDecreasePowOrLeave$ adopts the passive strategy of no recoding when a node leaves the network or decreases its power, since no new conflicts are introduced by these events and thus the minimum number of codes to be changed in the network to maintain no conflicts is zero. An example is shown in Fig 7. The termination, correctness, minimality, and optimality among minimality properties of this algorithm follow from the above discussion and are stated in [7].

4.4. Handling Node Movement

As mentioned in section 3, the CP strategy for handling recoding on node movement is to treat it as a pair of consecutive events where the moving node n leaves and joins the network. Such an approach can be very costly as mobility is inherent to ad-hoc networks.

Our strategy for recoding on a node move, $RecodeOnMove$, is shown in Fig 8 and is very similar to $RecodeOnJoin$. In fact, were the moving node n to leave the network and then join it immediately, this would be the exact sequence of steps executed for the recoding (see [7] for a proof). Notice, however, that our

Figure 9. (a) Node 2 moves. Dotted edges added by the move. Against each node is shown its (old color, new color by $RecodeOnMove$, new color by CP) (b) Weighted bipartite graph used by $RecodeOnMove$; dark edges show matching constructed by $RecodeOnMove$.

strategy is more advantageous than the CP strategy since a moving node disconnects from and connects to only some nodes, not the entire network.

The termination, correctness, minimality and optimality among minimality properties of this algorithm are formally stated in [7]. An example for the working of $RecodeOnMove$ is shown in Fig 9. Both $RecodeOnMove$ and the CP (which uses a highest-first node ordering) strategies cause 1 new recoding and end up with 4 as the maximum color index in the network after this event.

5. Simulation Results

The recoding algorithms presented in the last section are provably minimal only for the individual events considered.

In this section, we address the question: how well do the set of our minimal recoding strategies, call them *Minim*, of section 4 perform for a long sequence of events in an ad-hoc network ? We evaluate the performance of the *Minim* strategies against (1) a strategy that uses a centralized coloring heuristic: the *BBB* algorithm of [2], to recolor the entire network at every event, and (2) the *CP* recoloring strategies. The performance metrics we are concerned with are 1) the maximum color index assigned in the network (the lower, the better is the code reuse) and 2) the number of nodes recolored (recoded with a new color different from its old one). This section handles this issue via discrete event simulations.

In the following discussion, we describe experiments on our strategies for recoding on a node join (section 5.1), node power increase (section 5.2) and node movement (section 5.3). The other two strategies of section 4.3 are trivial and the same as [5], hence we did not evaluate them. The experimental plots are collected in Fig 10, and discussed below. All points on these plots were derived from 100 random runs of the experiment for that setting of parameters.

5.1. Node Join

N nodes were consecutively allowed to join and construct the ad-hoc network. Their position was determined by choosing their x and y coordinates independently and uniformly from the interval $[0, 100]$. Their transmission ranges were chosen uniformly in the interval $(minr, maxr)$. For this experiment, Fig 10 shows the two metrics for the three different algorithms with increasing N (Figs 10(a-c) with $minr = 20.5, maxr = 30.5$).

Fig 10(a-c) shows that the *Minim* approach performs better than the *CP* approach in both the maximum color index (Fig 10(a)) and number of recodings (Fig 10(c)). The fewer colors used by *BBB* (Fig 10(a)) may not be worth the extra recodings this centralized approach requires (Fig 10(b)).

5.2. Node Transmission Range Increase

To measure the effectiveness of $RecodeOnPowIncrease$, we started with the ad-hoc networks and the code assignment thereof generated in the last section (with $N = 100, minr = 20.5, maxr = 30.5$). A parameter $raisefactor$ was introduced, and half of the N nodes in the ad-hoc network were randomly chosen and their power ranges increased by a factor of $raisefactor$. The change (Δ's) in maximum color index assigned in the network and the total number of recodings were measured for this sequence of range increase events and are shown respectively in Fig 10(d) and Figs 10(e,f).

The *CP* approach performs better than the *Minim* minimal approach in terms of maximum color index assigned

to the network (Fig 10(d)). This is because unlike the modified CP strategy to handle range increases, the *Minim* $RecodeOnPowIncrease$ strategy is very simple and does not care about minimizing the number of maximum color index after reassignment. However, *Minim* requires much fewer of recodings compared to the *CP* (and the *BBB*) strategies (Figs 10(e,f)). For example, at $raisefactor = 4$, *Minim* performs worse than *CP* in the maximum color index metric by only 6 colors but outperforms it by around 50 recodings (Figs 10(f)).

5.3. Node Movement

To evaluate the performance of $RecodeOnMove$, we started with the networks generated in section 5.1 (with $N = 40, minr = 20.5, maxr = 30.5$). Each run of this experiment had $RoundNo$ rounds, with all N nodes being displaced in a random direction by a uniformly chosen distance in the interval $[0, maxdisp]$, in each round. The two metrics - change in maximum color index in the network, and total number of recodings, were measured for this sequence of node move events (Figs 10(g-j)).

The *Minim* strategy performs far better than either *CP* or *BBB* in terms of the number of recodings involved, for different values of $maxdisp$ (Fig 10(g)) and as rounds progress (Fig 10(i,j) where $maxdisp = 30$). For example, at $RoundNo = 10$, the *Minim* achieves 400 fewer recodings than *CP* !). Fig 10(h) shows that the *Minim* strategy performs only slightly worse than the *CP* strategy in the increase in the number of colors in the network.

The overall conclusion from these experiments is that the *Minim* strategies result in the use of a few more extra colors (codes) for a vast reduction in the number of recodings.

6. Conclusions

The problem of CDMA code assignment to eliminate collisions in packet radio networks has been widely researched in the past, but none of the algorithms proposed for code assignment in a dynamic scenario guarantee any strong performance bounds. In this paper, we have presented a set of *recoding* strategies *Minim* for TOCA CDMA recoding in an ad-hoc network where mobiles can arbitrarily 1) connect and disconnect, 2) move about, and 3) increase or decrease their transmission power. Our strategies, unlike those proposed earlier in literature, have been proved to guarantee *minimal recoding*, that is, given a current code assignment and one of the above events, our strategies change the codes of the minimum number of mobiles needed to eliminate all collisions in the network after the event. Simulation results reveal that our *Minim* approaches trade off a relatively small loss in terms of maximum color

index assigned in the network to obtain a significant gain in terms of the total number of instances where a node has to change its code. The proposed *Minim* strategies can be very practical in scenarios such as hard real-time systems and high data rate applications running on an ad-hoc network, where it is much more preferable to use a few more codes in the network than to suffer the (possibly) critical loss incurred by changing the codes of several mobiles.

References

[1] M. Adler and C. Scheideler. Efficient communication strategies for ad-hoc wireless networks. In *Proc. SPAA 98: 10th Annual ACM Symp. Parallel Algorithms and Architectures*, pages 259–268, 1998.

[2] R. Battiti, A. A. Bertossi, and M. A. Bonucceli. Assigning codes in wireless networks: bounds and scaling properties. *Wireless Networks*, 5(3):195–209, August 1999.

[3] A. A. Bertossi and M. A. Bonuccelli. Code assignment for hidden terminal interference avoidance in multihop packet radio networks. *IEEE/ACM Trans. Networking*, 3(4):441–449, August 1995.

[4] D. Brelaz. New methods to color the vertices of a graph. *Commun. ACM*, 22:251–256, 1979.

[5] I. Chlamtac and S. S. Pinter. Distributed nodes organization algorithm for channel access in a multihop dynamic radio network. *IEEE Trans. Computers*, c-36(6):728–737, June 1987.

[6] Z. Galil. Efficient algorithms for finding maximum matching in graphs. *ACM Computing Surveys*, 18(1):23–38, March 1998.

[7] I. Gupta. Minimal CDMA recoding strategies in power-controlled ad-hoc wireless networks. *Technical Report, Dept. of Computer Science, Cornell University*, 2001.

[8] L. Hu. A novel topology control for multihop packet radio networks. In *Proc. INFOCOM 1991*, volume 3, pages 1084–1093, 1991.

[9] L. Hu. Distributed code assignments for CDMA packet radio networks. *IEEE/ACM Trans. Networking*, 1(6):668–677, December 1993.

[10] S. Khuri, W. M. Moh, and F. Chung. Code assignments in CDMA networks: distributed algorithms and genetic algorithm heuristics. In *Proc. 6th IEEE Singapore Intl. Conf. Networks*, pages 91–105, 1998.

[11] L. M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. In *Proc. STACS 97: 14th Annual Symp. Theoretical Aspects of Computer Science*, pages 363–374, 1997.

[12] T. Makansi. Transmitter-oriented code assignment for multihop packet radio. *IEEE Trans. Communications*, com-3(12):1379–1384, December 1987.

[13] C. E. Perkins. Mobile networking in the internet. *Mobile Networks and Applications*, 3(4):319–334, January 1999.

[14] S. Ramanatham. A unified framework and algorithm for channel assignment in wireless networks. *Wireless Networks*, 5(2):81–94, March 1999.

[15] K. G. Shin and P. Ramanathan. Real-time computing: A new discipline of computer science and engineering. *Proc. IEEE*, 82(1):6–24, January 1994.

(a) Node Join I

(b) Node Join II

(c) Node Join III

(d) Node Power Increase I

(e) Node Power Increase II

(f) Node Power Increase III

(g) Node Movement I

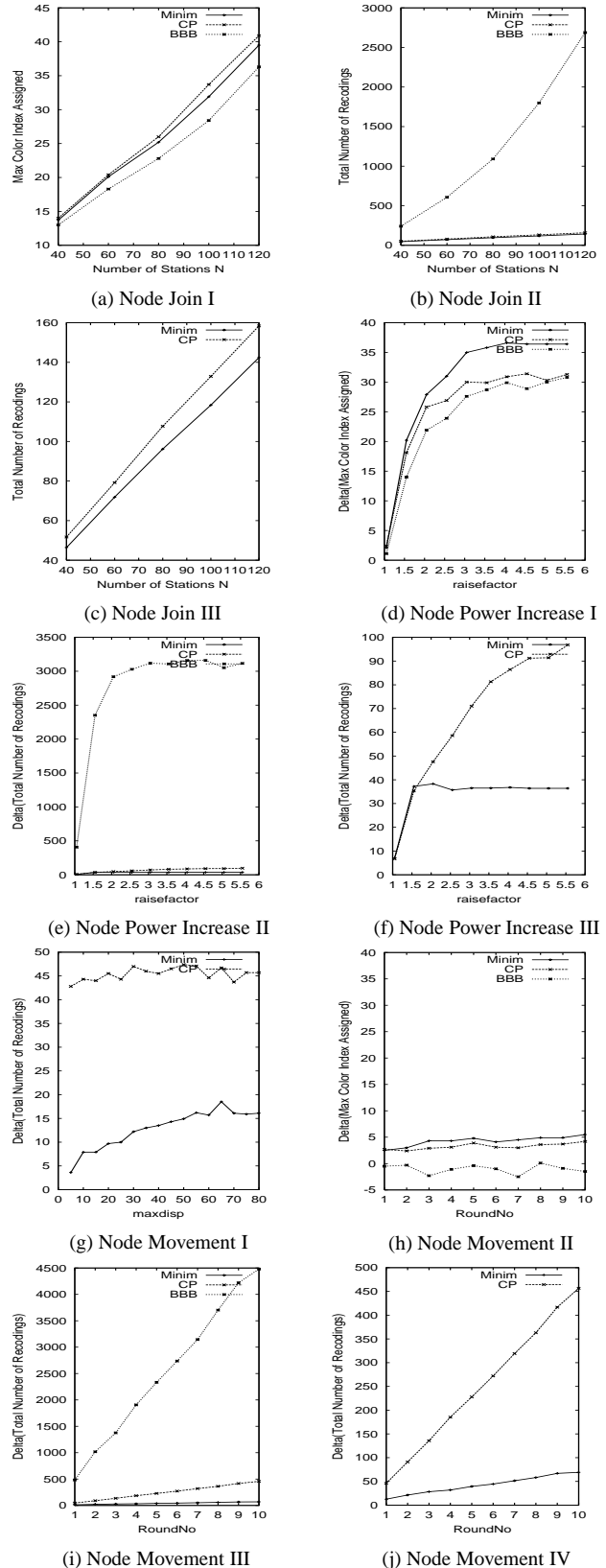(h) Node Movement II

(i) Node Movement III

(j) Node Movement IV

Figure 10. Simulation Results - see text for explanation.