Broadcast Scheduling in Interference Environment

Scott C.-H. Huang, Peng-Jun Wan, Jing Deng *Member, IEEE*, and Yunghsiang S. Han *Senior Member, IEEE*

***Abstract*—Broadcast is a fundamental operation in wireless networks and naïve flooding is not practical because it cannot deal with interference. Scheduling is a good way to avoid interference, but previous studies on broadcast scheduling algorithms all assume highly theoretical models such as the unit disk graph model. In this work, we re-investigate this problem using the 2-Disk and the signal-to-interference-plus-noise-ratio (SINR) model. We first design a constant approximation algorithm for the 2-Disk model and then extend it to the SINR model. This result is *the first result* on broadcast scheduling algorithms in SINR model, to the best of our knowledge.**

I. INTRODUCTION

Broadcast is probably the most fundamental yet challenging operation among all operations of wireless ad hoc networks. The broadcast storm problem [26] tells us that naïve flooding is simply not practical because it causes severe contention, collision, and congestion. When two or more nodes are transmitting to a node, their signals will interfere with each other, resulting in the receiving node's inability to recognize anything. In the literature, broadcast is often studied in the highly theoretical *Disk Graph* model, in which the transmission and interference range of a node equipped with an omnidirectional antenna is thought of as a disk centered at this node with some radius. Disk graphs in this case are defined as follows. The node set is the set of all transceivers. A directed edge exists from u to v if v lies in u's disk. In addition, if all nodes have the same radius, then the resulted graph is bidirectional and we can thus use an undirected graph to represent it. This is called the *Unit Disk Graph* model, which has been widely used in the literature. Others use a more generalized *General Graph* model, in which the transmission and interference topology is modeled as a general graph. However, these three models are all overly simplified and they do not match what actually happens in reality. For example, a node can interfere with a far-away node and the interference range of a node is generally much larger than its transmission range [16], [17]. None of these three models described earlier can address this issue.

In this paper we investigate the broadcast problem using two new models that are much more realistic. First we use the *2-Disk* model, in which two disks are employed to represent the transmission and interference range, respectively. Then we use the Signal-to-Interference-plus-Noise-Ratio (SINR) model, which deals directly with transmission laws in general physics. SINR is more realistic, as it actually models the case where many far-away nodes could still have the effect of interfering some nodes if they are transmitting simultaneously. This case cannot be dealt with in the 2-Disk model, as no interference whatsoever is assumed when nodes are located outside the interference range. The SINR model gives a more precise analysis in this case, in which the accumulative interference of many nodes outside the interference range are not be neglected. Surprisingly, we found that we can still use the 2-Disk model to deal with this case by carefully select the transmission and interference radii. This result is **the first result** on broadcast scheduling algorithms in SINR model, to the best of our knowledge.

The rest of this paper is organized as follows. Related work is introduced in Section II. In Section III we formally present our interference models, assumptions, and the definition of the broadcast scheduling problem in both models. We give the preliminaries of tessellation in Section IV, to be used extensively in later sections. We present our broadcast scheduling algorithms in Section V, give an example of them in Section VI, and analyze them in Section VII. Simulation results are given in Section VIII.

S.C.H. Huang is with the Computer Science Department, City University of Hong Kong, Kowloon, Hong Kong SAR (Email: shuang@cityu.edu.hk)

P.J. Wan is with the Department of Computer Science, Illinois Institute of Technology, Chicago IL 60616, USA. (Email: wan@cs.iit.edu)

J. Deng is with the Department of Computer Science, University of New Orleans, LA 70148, USA. (Email: jing@cs.uno.edu)

Y.S. Han is with the Graduate Institute of Communication Engineering, National Taipei University, Sanshia, Taipei, 237 Taiwan (Email: yshan@mail.ntpu.edu.tw)

II. RELATED WORK

Broadcast was studied extensively in the literature. Sheu *et al* [27] did empirical studies about the efficiency of broadcasting schemes in terms of collision-free delivery, number of retransmissions and latency. They also designed a centralized as well as a distributed broadcast algorithm. Basagni *et al* [4] presented a mobility transparent broadcast scheme for mobile multi-hop radio networks by using mobility-transparent schedule that guarantees bounded latency.

Minimum-latency broadcast schedule has been extensively studied in the literature. The prevailing network model in the literature is an arbitrary undirected graph. Let n be the number of nodes in the graph, Δ the maximum node-degree in the graph (i.e., the maximum number of neighbors of a node), and R the radius of the source in the graph (i.e., the number of hops from the source to the farthest node). Obviously, R is a trivial lower bound on the latency of any broadcast schedule. Alon et al. proved in [1] the existence of a family of n-node networks of radius 2, for which any broadcast schedule has latency $\Omega(\log^2 n)$. Chlamtac *et al* [6] established the NP-hardness of the minimum-latency broadcast schedule in general graphs. Recently, Elkin *et al* investigated the hardness of approximation for the same problem. They proved in [10] a logarithmic multiplicative inapproximability: unless $NP \subseteq BPTIME(n^{O(\log\log n)})$, $\Omega(\log n)$-approximation of the radio broadcast problem is impossible. They also proved in [11] a polylogarithmic additive inapproximability: unless

$NP \subseteq BPTIME(n^{O(\log\log n)})$, there exists a constant c such that there is no polynomial-time algorithm that produces, for every n-node graph G, a broadcast schedule with latency less than $opt(G) + \log^2 n$, where $opt(G)$ is the optimal broadcast latency for G. Several multiplicative approximation algorithms for minimum-latency broadcast schedule have been proposed in [6], [7], [20]. Chlamtac *et al* in [6] proposed a broadcasting schedule of latency $O(R\Delta)$. Chlamtac *et al* in [7] gave the first broadcast schedule whose latency is $O(R\log^2(n/R))$, where R (the radius of the source) is the lower bound of the broadcast latency. This algorithm is of the best possible order for networks with constant diameter due to the lower bound obtained in [1]. Kowalski *et al* [20] improved this result by constructing a broadcast schedule with latency $O(R\log n + \log^2 n)$. For $R = \Omega(\log n)$, the approximation ratio is $O(\log^2(n/R))$, which is of the best possible order unless $NP \subseteq BPTIME(n^{O(\log\log n)})$ due to the inapproximability result in [11]. Bar-Yehuda *et al* [3] obtained the same result as [20] earlier, but their solution was a randomized algorithm of Las Vegas type (which means they cannot guarantee 100% success). Although this is a serious problem in some scenarios, it does have great advantage in distributed implementation. A couple of additive approximation algorithms for minimum-latency broadcast schedule have been proposed in [13], [12]. Gaber *et al* in [13] presented a method consisting of partitioning the underlying graph into clusters. This method improves the time of broadcast, because the existing broadcast schemes can be applied in each cluster separately and the diameters of clusters are smaller than the diameter of the graph. This method can be used to construct (in polynomial time) a deterministic broadcast scheme working in $O(R + \log^6 n)$ steps by using the broadcast schedule in [7]. It can produce a broadcast scheme with latency $O(R + \log^5 n)$ by using the schedule in [20]. Recently, the clustering method in [13] was improved by Elkin *et al* in [12]. This new clustering method can be used to construct (in polynomial time) a deterministic broadcast scheme working in $O(R + \log^5 n)$ steps by using the broadcast schedule in [7], and it can produce a broadcast scheme with latency $O(R + \log^4 n)$ if the schedule in [20] is used. This result was reduced to $O(R + \log^3 n)$ by Gąsieniec *et al* in [15]. Very recently Kowalski *et al* [19], further reduced it to $O(R + \log^2 n)$ in [20], which is asymptotically optimal unless $NP \subseteq BPTIME(n^{O(\log\log n)})$.

The minimum-latency broadcast schedule in wireless ad hoc networks represented by unit-disk graphs was only considered in [14], [9]. Dessmark *et al* in [9] presented a broadcast schedule of latency at most $2400R$. Bruschi and Del Pinto [5] considered distributed protocols and obtained a lower bound of $\Omega(R\log n)$ with the assumption that no nodes know the identities of their neighbors. Kushilevitz and Mansour [21] proved that for any randomized broadcast protocol there exists a network whose latency is $\Omega(R\log(N/R))$. Chlebus *et al* [8] studied deterministic broadcasting without a-priori knowledge of the network. They considered two models (with and without collision detection), and designed algorithms for them separately. They also established a lower bound $\Omega(R\log n)$ for the scheme without collision detection. Apart from these results on upper or lower bounds, there are also some results on the hardness of approximation of this problem. Gandhi *et al* in [14] established the NP-hardness of minimum-latency broadcast schedule restricted to unit-disk graphs and presented an improved broadcast schedule of latency at most $648R$. Huang *et al* [18] studied the unit disk graph model and designed two scheduling algorithms that improved the approximation ratio of [14]. In their work, these two algorithms have approximation ratios 52, and 24 respectively. They also designed a theoretically near-optimal scheduling algorithm, whose latency is bounded by $O(R + R\log^{1.5} R)$. If R is large, then the approximation ratio is nearly 1. This algorithm is nearly optimal for all broadcast scheduling algorithms in unit disk graphs.

Our work uses the SINR model, so it is also related to those who used this model. Moschibroda and Wattenhofer [25] considered the problem of scheduling a given topology using the SINR model. In a network, for any given topology, we may not be able to realize this topology in one time slot, if interference is considered. In other words, we need to do scheduling in order to make a topology feasible, and [25] focused on the latency issue. This problem is not directly related to our work, as scheduling a topology is always a one-hop concept, in which there is no relay. In broadcast, a non-source node cannot transmit a message unless it has already received from another node. This property makes our work fundamentally different from [25]. Zheng and Barton [28] investigated theoretical limits of data aggregation. They proved that the data aggregation rates $\Theta((\log n)/n)$ and $\Theta(1)$ are optimal for systems with path-loss exponent α satisfying $2 < \alpha < 4$ and $\alpha > 4$, respectively.

III. Interference Models, Assumptions, and Problem Definition

In this section we introduce two interference models, namely the *2-Disk* and *SINR model*. The descriptions of 2-Disk model are as follows. A wireless network is modeled as a set of nodes V arbitrarily located in a 2-dimensional Euclidean space. Each node is associated with two radii, the transmission radius r_T and the interference radius r_I (where $r_I \geq r_T$). The transmission range of a node v is a disk of radius r_T centered at v, and the interference range of v is a disk of radius r_I centered at v. However, the transmission range is a concept with respect to the transmitting nodes while the interference range is a concept with respect to the receiving nodes. A node u receives a message successfully from v if and only if u is within v's transmission range and no other nodes are within u's interference range. For simplicity, we assume that all nodes have the same r_T and r_I in the 2-Disk model throughout this paper.[1] Note that the transmission range can now be considered from the receivers' point of view and the interference range can be considered from the transmitters' point of view, since they are equivalent this way.

[1]This of course limits the proposed algorithms to homogenous networks, where each node has the same transmission range and the same interference range. Interestingly, as we will show later, the same algorithms and transmission schedules can be used in the SINR model, in which received signal's power is compared to the overall interference and noise level and no fixed interference range r_I is assumed.

In SINR model, a wireless network is also regarded as a set V in a 2-dimensional Euclidean space. Each node is associated with a transmission power P. For simplicity we assume all nodes have the same P. According to general physics we know that if a node u is transmitting with power P, the theoretically received signal strength P_v at another node v is given by

$$P_v = \frac{P}{r^\alpha}$$

where r is the distance between u, v and α is a constant called the *path-loss exponent*. As commonly assumed [16] the *path-loss exponent* is greater than **two** (i.e. $\alpha > 2$). A node v receives a message successfully in a time slot from another node u if and only if the SINR at v is at least a given constant β, where β is called the minimum SINR. The SINR at v is given by

$$SINR_v = \frac{P_v}{N + I_v}$$

where N is the background noise and I_v is the total interference at v. P_v and I_v are given by

$$P_v = \frac{P}{d(u,v)^\alpha} \quad , \quad I_v = \sum_{w \in T - \{u\}} \frac{P}{d(v,w)^\alpha}$$

In the above expressions, $d(u, v)$ is the Euclidean distance between u and v, and $T \subset V$ is the set of nodes scheduled to transmit in the current time slot. Note that in order for the SINR to make sense, we need to assume that $N + I_v > 0$.

In practice, we further consider the generalized physical model in which the actually received signal strength P_A can deviate from the theoretical value by a factor of $\theta > 1$ [25], i.e.

$$\frac{1}{\theta} \cdot \frac{P}{r^\alpha} < P_A < \theta \cdot \frac{P}{r^\alpha}$$

We assume that the network is *connected*. This fundamental assumption has different representations in different models. In 2-Disk model, it means nothing more than that the disk graph generated by V and r_T (i.e. an edge exists between $u, v \iff d(u, v) < r_T$) is connected. However, in SINR model, it means more. Let u and v be any two nodes with edge between them in V that is connected. Any successful received message at v means $SINR_v \geq \beta$. Thus, we have $\theta \frac{P}{d(u,v)^\alpha} > \beta(N + I_v) \geq \beta N$. Equivalently, we can say there exists a $\gamma > 1$ such that $\frac{\theta P}{\gamma N \beta} = d(u, v)^\alpha$. Let r' be any distance between two nodes with edge on them in V, we can make the following assumption on connectivity:

> **Connectivity Assumption**: There exists a constant $\gamma > 1$ such that the disk graph generated by V and $r' = \sqrt[\alpha]{\frac{\theta P}{\gamma N \beta}}$ is connected.

Finally we also make an assumption that every node knows its location. This assumption is strong but essential since we are considering the SINR model, which is a geometrical concept.

The problem definition for either model is as follows. Given a set of nodes V and a source $s \in V$, the objective is to find a schedule $\{U_1, U_2, \ldots\}$ satisfying the following requirements. (1) for all i, $U_i \subset V$ represents the set of nodes scheduled to transmit in time slot i. (2) A node cannot be scheduled to transmit unless it has already received successfully in an earlier time slot. (Note that the conditions of successful reception are different in those 2 models.) (3) In the end, all nodes in V receive successfully. *Latency* is the first time slot such that this happens.

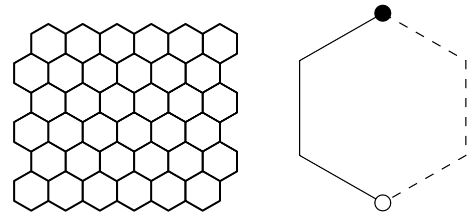

Fig. 1. (a) hexagonal tessellation (b) one hexagon

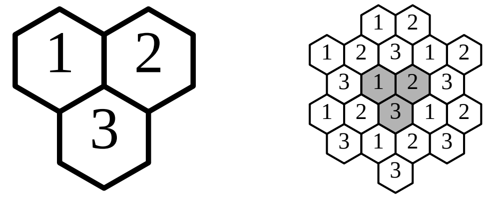

Fig. 2. (a) 3-coloring ($k = 1$) (b) 3-coloring filling up the plane

IV. Tessellation of Hexagons

Before presenting the proposed broadcast algorithm we introduce a tessellation/coloring technique. This technique will be used in our algorithm.

A tessellation of the entire plane is a way of partitioning into equal (or similar) pieces. We partition the plane into hexagons as shown in Fig. 1(a). Each hexagon has radius 1/2 and is half open, half closed, with the topmost point included and

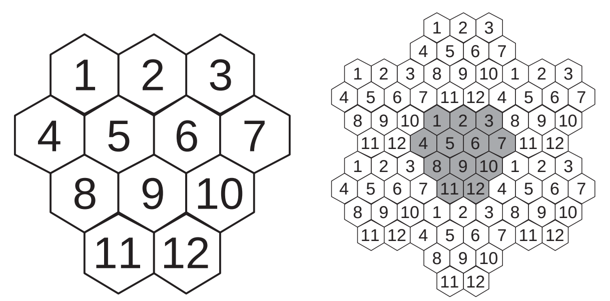

Fig. 3. (a) 12-coloring ($k = 2$) (b) 12-coloring filling up the plane

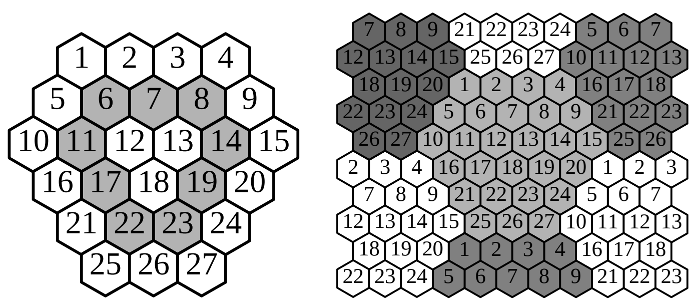

Fig. 4. (a) 27-coloring ($k = 3$) (b) 27-coloring filling up the plane

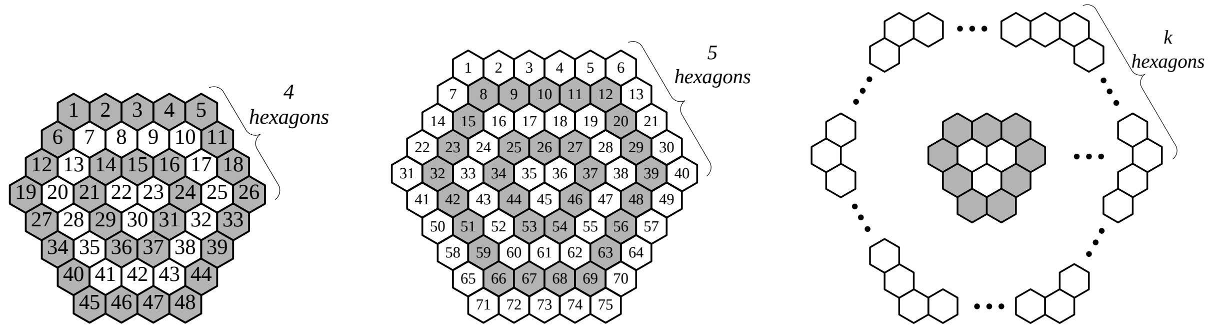

Fig. 5. (a) 48-coloring ($k = 4$) (b) 75-coloring ($k = 5$) (c) General $3k^2$-coloring

the bottommost point excluded as shown in Fig. 1(b). We can give many different colorings to this tessellation.

3-coloring is shown in Fig. 2(a)(b). Three hexagons are grouped together as shown in Fig. 2(a), and they can fill up the entire plane as shown in 2(b). Now let's look at the three hexagons in Fig. 2(a) again. If we *enclose* another layer of hexagons, we get 12 hexagons grouped together as shown in Fig. 3(a). This introduces a 12-coloring and they fill up the plane as shown in Fig. 3(b). Similarly, we can further enclose layers and layers and get a 27-coloring , a 48-coloring, a 75-coloring, as well as general $3k^2$-coloring as shown in Fig. 4(a)(b), and Fig. 5(a)(b)(c).

Note that hexagons of the same color in a 3-coloring are separated by at least the distance of one radius, which is 1/2. In a 12-coloring, they are separated by at least the distance of four radii, which is 2. They are separated by 7, 10, and 13 radii in a 27, 48, and 75-coloring, respectively. In general, hexagons of the same color are separated by at least $3k-2$ radii (or Euclidean distance $\frac{3k-2}{2}$) in a $3k^2$-coloring. This can be easily proved by Mathematical induction. There are many different ways to color these hexagons, and we just consider one of them [22].

V. Broadcast Scheduling Algorithm

In this section we first look at the 2-Disk model and design a broadcast scheduling algorithm of approximation ratio $6\left\lceil\frac{2}{3}\left(\frac{r_I}{r_T}+2\right)\right\rceil^2$, which is a constant. Later we'll show that the SINR model can be reduced to the 2-Disk model and the same scheduling algorithm can be applied.

We consider the following graph: the *transmission graph* $G_T = (V, E_T)$ generated by r_T and V. To define the broadcast schedule, we first need to construct a virtual backbone as follows. We look at G_T and its Breadth First Search (BFS) tree, and then divide V into layers $L_0, L_1, L_2, \ldots, L_R$ (where R is the radius of G_T and source s). All nodes of layer i are thus i hops away from the root. Then we construct a layered maximal independent[2] set, called $BLACK$, as follows. Starting from the 0-th layer, which contains only s, we pick up a maximal independent set, which contains only s as well. Then, at the 1st layer, we pick up a maximal independent set in which each node is independent of each other and those nodes at the 0-th layer. Note that this is empty because all nodes in L_1 (layer 1) must be adjacent to s. Then we move on to the 2nd layer and pick up a maximal independent set and mark these nodes black again. Note that the black nodes of the 2nd layer also need to be independent of those of the 1st layer. We repeat this process until all layers have been worked on. Those who are not marked black are marked white at last. Those black nodes are also called the *dominators*, and we will use these two terms interchangeably throughout this paper. The pseudocode of layered Maximal Independent Set (MIS) construction is given in Algorithm 1.

Algorithm 1 Construct an MIS in G_T layer by layer

Input: V, s, G_T

```
BLACK ← ∅
for i ← 0 to R do
    Find an MIS BLACK_i ⊂ L_i indep. of BLACK
    BLACK ← BLACK ∪ BLACK_i
end for
return BLACK
```

Now we construct the virtual backbone as follows. We pick some of the white nodes and color them blue to interconnect all black nodes. Note that $L_0 = \{s\}$ and all nodes in L_1 must be white. We simply connect s to all nodes in L_1. To connect L_1 and L_2, we look at L_2's black nodes. Each black node must have a parent on L_1 and this parent node must be white since black nodes are independent of each other. We color this white node blue and add an edge between them. Moreover, we know that this blue node must be dominated by a black node either on L_1 or L_0 (in this case L_0). We then add an edge between this blue node and its dominator.[3] We repeat this process layer by layer and finally obtain the desired virtual backbone (which is a tree) in this manner. Note that, in this tree, each black node has a blue parent at the upper layer and each blue node has a black parent *at the same layer or the layer right next to it above*. The pseudocode is given in Algorithm 2. Note that, until now, the construction of the virtual backbone is not related to the 2-Disk model and only the concept of transmission range is used. The concept of interference range is used when we schedule the time slot for each node, which will be explained

[2] The term "independent" means non-adjacent with respect to G_T.

[3] If there are more than one dominators of the blue node, only one needs to be chosen to connect to the blue node.

next, according to the tessellation of hexagons, where enough colors must be used in order to avoid interference.

Algorithm 2 Virtual backbone construction

Input: V, s, G_T

```
T_vb = (V, E_vb), E_vb ← ∅
                    ▷ /* Connect black nodes layer by layer */
∀u ∈ L_1 add an edge between u, s
for i ← 1 to R − 1 do
    for all black nodes v ∈ BLACK_{i+1} do
        Find its parent p(v) in G_T's BFS tree
        Color p(v) blue and find its dominator d_{p(v)}
         in BLACK_i ∪ BLACK_{i−1}
        Add an edge between p(v), v to E_vb
        Add an edge between d_{p(v)}, p(v) to E_vb
    end for
end for
                    ▷ /* Connect remaining white nodes */
for all remaining white nodes u do
    Find u's dominator d_u
    Add an edge between u, d_u to E_vb
end for
return T_vb
```

The broadcast scheduling algorithm based on the virtual backbone in the 2-Disk model is described as follows. Note that the layers of the BFS tree and the virtual backbone may be different. Starting from the 0-th layer containing only the source s, we schedule s to transmit in the first time slot, and obviously this transmission causes no collision and after the first time slot all nodes of the 1st layer will receive successfully. We will design a schedule such that all nodes of the $(i+1)$-th layer receive from the i-th layer successfully for $i = 1, 2, \ldots, R$. We partition the plane into half-open, half-closed hexagons of radius $\frac{r_T}{2}$[4] and give a $3\lceil\frac{2}{3}(\frac{r_I}{r_T}+2)\rceil^2$-coloring with proper scaling, as described in section IV (in which $k = \lceil\frac{2}{3}(\frac{r_I}{r_T}+2)\rceil$). Then the distance between two hexagons of the same color is at least $r_T + r_I$, which guarantees the validity of the proposed schedule. This schedule has two parts, and in the first part we schedule each blue node of layer i to transmit in the time slot according to its targeted black nodes' colors. If there are more than one targeted black nodes with the same color, those blue nodes will need to transmit multiple times.[5] For example, suppose the starting time of the i-th layer is T_i. If a blue node has 3 black children of colors #4, #9, #13, then we schedule it to transmit in time slots T_i+4, T_i+9, and T_i+13. In the second part we schedule each black node of layer $i+1$ to transmit in the time slot according to its own color. After these two parts complete, all nodes at layer $i+1$ receive the broadcast message. The pseudocode of this part is given in algorithm 3.

[4]The size of hexagons is determined by guaranteeing no more than one black node is in the same hexagon. $r_T/2$ is thus the largest radius of each hexagon we can have.

[5]When a blue node sends a message, only the targeted black node is guaranteed to received successfully, although other child(ren) may still be able to receive.

Algorithm 3 Broadcast Scheduling

Input: V, s and virtual backbone T_{vb}

```
1: Tessellate the plane and give a 3⌈(2/3)(r_I/r_T + 2)⌉^2-coloring
   by setting k = ⌈(2/3)(r_I/r_T + 2)⌉
2: Schedule s to transmit in time slot 1.
3: T_start ← 1
4: for i ← 1 to R − 1 do
5:     ∀u ∈ BLUE_i, ∀w ∈ {u's children}, schedule
           u to transmit in time slots T_start + color(w)
6:     T_start ← T_start + 3⌈(2/3)(r_I/r_T + 2)⌉^2
7:     ∀v ∈ BLACK_{i+1}, schedule v to transmit in
           time slot T_start + color(v)
8:     T_start ← T_start + 3⌈(2/3)(r_I/r_T + 2)⌉^2
9: end for
```

Note that in line 5 of Algorithm 3, each blue node has at most 4 black children and therefore we need at most 4 time slots. This is because those black children are all independent of each other in G_T, and in the transmission range of any blue node u (i.e. in the disk centered at u with radius r_T), there can be at most 5 independent nodes and one of them must be u's parent. Note that the source s does not have any parent, but s is black. So u cannot be the source. For this reason each blue node can only have at most 4 black children.

In SINR model, we simply set

$$r_T = \sqrt[\alpha]{\frac{P}{\gamma\beta\theta N}}\ ,\ r_I = \sqrt[\alpha]{\frac{24\theta P}{(\gamma-1)N}\Big(\frac{2}{\alpha-2}+\frac{1}{\alpha-1}+3\Big)}$$

and apply the broadcast scheduling algorithm for 2-Disk model.[6]

Note that since the proposed algorithm is a centralized algorithm, the source needs to inform each node its time slot to forward the message. However, this initial message forwarding is only performed once in the whole network lifetime. Any inefficient forwarding can be used without increasing overhead significantly.

VI. An Example

Figure 6(a)(b) shows the layered construction of MIS as described in Algorithm 1. Figure 6(a) shows the topology of G_T. In the first step, the source s is selected in the MIS and colored black. Note that layer 2 is represented with light gray color for the ease of understanding (this color has nothing to do with the black-blue coloring scheme). In the second step, since the source is black, all nodes at layer 1 must all be white, otherwise it won't be independent of s. In the third step, we will select an independent set at layer 2, which must also be independent of the nodes at the previous layer, layer 1, though there is no black node at layer 1 and this does not have any effect. Figure 6(b) shows that 5 more black nodes were selected at layer 2. We keep doing this and select black nodes until all layers have been worked on. The black node selection

[6]It will be explained in detail in Section VII.

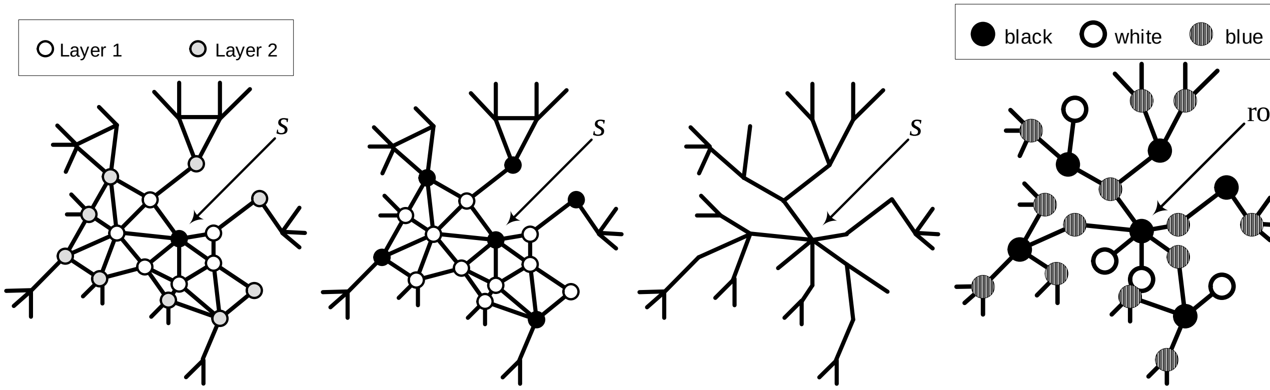

Fig. 6. (a) G_T's topology (b) Layered MIS (c) BFS tree (d) Virtual backbone

depends on G_T only and it has nothing to do with the BFS tree. Not until blue nodes are being selected do we need to consider the BFS tree, as shown in Fig. 6(c). In Algorithm 2, we are trying to add appropriate blue nodes to interconnect all black ones. Since the source does not have an upper layer and there are no black nodes at layer 1, we start from layer 2 directly. For each black node at layer 2, we color it blue and connect to its parent in the BFS tree, as shown in Fig. 6(d). In Fig. 6(d) we see 4 nodes at layer 1 are colored blue and connected to some black nodes at layer 2. Those which are not colored blue remain white, and there are 2 white nodes. We also connect these 4 blue nodes and 2 white nodes to the source s since they are dominated by s. We keep working on layer 3, for simplicity, suppose we've already found the black nodes at layer 3 and their corresponding blue nodes at layer 2. Figure 6(d) shows that there are 3 blue nodes at layer 2 connected to their black children at layer 3. Note that there are 9 nodes at layer 2, in which 5 are black, 3 are blue, and the remaining node is still white. Now, for each blue or white node at layer 2, we know that it must be adjacent to at least one black node either at layer 2 or layer 1, since $BLACK_2$ is a maximal independent set. Because of its maximality, all nodes at layer 2 must be adjacent to at least one black node at the same layer or the previous layer. Therefore, for each blue/white node at layer 2 we find a black node either at layer 1 or layer 2 and connect to it, as shown in Fig. 6(d). We keep doing this for all layers and the virtual backbone will be constructed this way.

We present an example of broadcast scheduling in the 2-Disk model, as shown in Fig. 7. Assume $r_I/r_T = 3$. $3\left\lceil \frac{2}{3}\left(\frac{r_I}{r_T}+2\right)\right\rceil^2 = 48$ colors should be used to separate the transmission schedules of these hexagon cells ($k = 4$) and we give a 48-coloring. In Fig. 7, a virtual backbone has already been constructed according to Algorithm 2. The root (source) is black and all nodes at layer 1 are either blue or white (4 blue, 2 white). The blue nodes at layer 1 are chosen to connect the black nodes at layer 2, and the remaining are white. At layer 3, there are 5 black nodes, 2 blue nodes, and 1 white node. We explain the broadcast schedule of our scheme, according to Algorithm 3, as follows.

1) The source transmits in time slot 1 and set $T_{start} \leftarrow 1$.
2) The 4 blue nodes at layer 1 are scheduled according to their black child(ren)'s color. Therefore, the first node transmits in time slots $T_{start} + 24 = 25$ and $T_{start} + 25 = 26$, the second transmits in time slot $T_{start} + 26 = 27$, the third in $T_{start} + 31 = 32$, and the last in $T_{start} + 39 = 40$. Note that the first node transmits in two time slots because it has two black children. The white nodes do not transmit at all. All other time slots between $[T_{start} + 1, T_{start} + 48 + 1]$ are idle.
3) $T_{start} \leftarrow T_{start} + 48 = 49$
4) At layer 3, there are 5 black nodes of colors #24, #25, #26, #31, and #39. Their transmission time slots are $T_{start} + 24 = 73$, $T_{start} + 25 = 74$, $T_{start} + 26 = 75$, $T_{start} + 31 = 80$, and $T_{start} + 39 = 88$, respectively.
5) Set $T_{start} \leftarrow T_{start} + 48 = 97$, and by this time all nodes at layer 3 should have already received the message successfully.
6) We keep scheduling in this manner until all nodes at layer R receive the message successfully and the broadcast finishes.

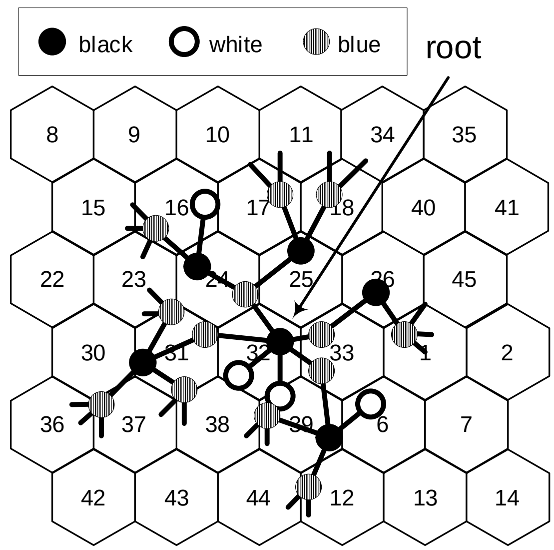

Fig. 7. An example of broadcast scheduling in 2-Disk model with $r_I/r_T = 3$

VII. Analysis

Theorem 7.1: Algorithm 3 is a valid scheduling algorithm.

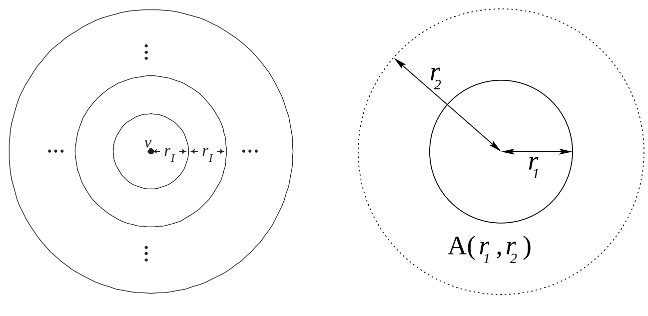

Fig. 8. (a) Concentric disks at v (b) Annulus $A(r_1, r_2)$

(*Proof.*) We prove two things. (1) Each node will receive successfully before it is scheduled to transmit. (2) In the end all nodes receive successfully. Algorithm 3 begins with the source's transmission and since there's only one node transmitting there will be no collision and all nodes of L_1 will receive successfully. Now, we prove that all nodes of L_{i+1} will receive successfully from L_i for all $1 \leq i \leq R-1$. First, we show that all nodes of $BLACK_{i+1}$ will receive successfully from $BLUE_i$. This is straightforward. Assume the contrary, if there exists a receiver $v \in BLACK_{i+1}$ such that another node $w \in BLUE_i$ is interfering with the sender $u \in BLUE_i$. If this happens, we know that $d(u, v) < r_T$ and $d(w, u) < r_I$. This implies $d(v, w) < r_T + r_I$, contradicting to the fact that any two hexagons of the same color must be at least $r_T + r_I$ apart. Second, we show that all nodes of $L_{i+1} - BLACK_{i+1}$ must receive successfully from $BLACK_{i+1}$. This is also straightforward by using similar arguments. Assume the contrary, if there is a node $v \in L_{i+1} - BLACK_{i+1}$ such that another node $w \in BLACK_{i+1}$ is interfering with the sender $u \in BLACK_{i+1}$, then similarly $d(u, v) < r_T$ and $d(w, u) < r_I$ implying $d(v, w) < r_T + r_I$ and we get a contradiction. □

Theorem 7.2: Algorithm 3 has latency (the total number of time slots to complete the broadcast procedure) $1 + \big(6\big\lceil \frac{2}{3}\big(\frac{r_I}{r_T} + 2\big)\big\rceil^2\big)(R-1)$.

(*Proof.*) We study the 'for' loop in Algorithm 3. Inside the loop, first we schedule the blue nodes according to their black children's colors, which takes $3\big\lceil \frac{2}{3}\big(\frac{r_I}{r_T} + 2\big)\big\rceil^2$ time slots since we use $3\big\lceil \frac{2}{3}\big(\frac{r_I}{r_T} + 2\big)\big\rceil^2$ colors to construct the tessellation. Then, we schedule the black nodes to transmit according to their colors. Therefore it takes $3\big\lceil \frac{2}{3}\big(\frac{r_I}{r_T} + 2\big)\big\rceil^2$ time slots as well. As a result, each iteration of the for loop takes $6\big\lceil \frac{2}{3}\big(\frac{r_I}{r_T} + 2\big)\big\rceil^2$ time slots and there are $R-1$ iterations. Along with the source's time slot in the beginning, the overall latency is $1 + \big(6\big\lceil \frac{2}{3}\big(\frac{r_I}{r_T} + 2\big)\big\rceil^2\big)(R-1)$. □

Having the above latency bound and that R is itself a lower bound for any broadcast schedule, we can get the following corollary.

Corollary 7.1: The broadcast scheduling algorithm for 2-Disk model is a constant approximation algorithm with ratio $6\big\lceil \frac{2}{3}\big(\frac{r_I}{r_T} + 2\big)\big\rceil^2$.

It is easy to see that the approximation ratio of the proposed algorithm is only related to the physical transmission characters. That is, the approximation ratio of the proposed algorithm only depends on the ratio of interference range to transmission range. When these two ranges are similar, the approximation ratio becomes 24 no matter how many nodes are in the network. In a large network, the proposed algorithm can broadcast the message efficiently.

It is obvious that there are many idle time slots in the proposed scheduling algorithm. In practice, we can delete all idle time slots and re-index all scheduling of nodes. We will show by simulation that it can reduce the latency up to 86%.

Theorem 7.3: In SINR model, if we set r_T, r_I as follows

$$r_T = \sqrt[\alpha]{\frac{P}{\gamma\beta\theta N}}\ ,\ r_I = \sqrt[\alpha]{\frac{24\theta P}{(\gamma-1)N}\Big(\frac{2}{\alpha-2} + \frac{1}{\alpha-1} + 3\Big)}$$

and we use Algorithm 3 to schedule the transmissions, then the overall interference at any *intended* receiver (i.e. the node that is scheduled to receive at this time) at any time is strictly less than $(\gamma - 1)N$.

(*Proof.*) Since we use Algorithm 3, we know that at any time the distance between two simultaneously transmitting nodes is at least $r_T + r_I$ because any two hexagons of the same color must be at least $r_T + r_I$ apart. Moreover, let u be a sender and v be its intended receiver at any time in Algorithm 3, then there will be no other sender that is transmitting simultaneously and whose distance to v is less than r_I. This is true because r_I is the interference radius and we've avoided this situation in Algorithm 3. Now, let's pick up an intended receiver v and consider its concentric circles of radii $r_I, 2r_I, 3r_I, \ldots$ as shown in Fig. 8(a). Here we use $A(r_1, r_2)$ to denote the annulus between two concentric circles of radii r_1 and r_2 ($r_1 < r_2$), as shown in Fig. 8(b). We define $A(r_1, r_2)$ to be inner-closed and outer-open $\big($i.e. $A(r_1, r_2)$ contains the circle of radius r_1 but does not contain the circle of radius $r_2\big)$. Now we consider $A\big((i-1)r_I, ir_I\big)$ and consider the senders scheduled to transmit simultaneously at a fixed time. Let M_i be the number of these senders in $A\big((i-1)r_I, ir_I\big)$. We know that the distance between any two black nodes is at least $r_T + r_I$. Moreover, since each blue sender is at most r_T from its black receiver, the distance between any two blue senders is at least $r_I - r_T$. Therefore, the distance between any two senders is at least $r_I - r_T$. If we draw an open disk of radius $\frac{r_I - r_T}{2}$ at each sender in $A\big((i-1)r_I, ir_I\big)$, then these disks will not overlap at all. Moreover, all of these disks will be completely contained in $A\big((i-1)r_I - \frac{r_I - r_T}{2}, ir_I + \frac{r_I - r_T}{2}\big)$. Therefore, by comparing their areas we know that

$$\pi\Big(\frac{r_I - r_T}{2}\Big)^2 \cdot M_i <$$

$$\pi\Big\{\Big[ir_I + \frac{r_I - r_T}{2}\Big]^2 - \Big[(i-1)r_I - \frac{r_I - r_T}{2}\Big]^2\Big\}$$

and that

$$M_i < \frac{4(2i-1)r_I(2r_I - r_T)}{(r_I - r_T)^2} \qquad \text{(VII.1)}$$

Since the distance between v and any point in $A\big((i-1)r_I, ir_I\big)$ is at least $(i-1)r_I$, the cumulative interference caused by sender in $A\big((i-1)r_I, ir_I\big)$ is bounded by $M_i \frac{\theta P}{\left((i-1)r_I\right)^\alpha}$ and the overall interference I_{total} at v caused by all senders on the entire plane is bounded by

$$I_{total} \leq \sum_{i=2}^{\infty} M_i \frac{\theta P}{\big((i-1)r_I\big)^{\alpha}}$$

Here i starts from 2 because, except for the intended sender, no other interfering senders are within the disk centered at v with radius r_I. Plugging in (VII.1) we know that I_{total} is less than

$$\sum_{i=2}^{\infty} \frac{4(2i-1)r_I(2r_I - r_T)}{(r_I - r_T)^2} \frac{\theta P}{\big((i-1)r_I\big)^{\alpha}} \qquad \text{(VII.2)}$$

Now, let q be defined as follows

$$q = \frac{r_I}{r_T} = \sqrt[\alpha]{\frac{24\gamma\beta\theta^2}{\gamma-1}\Big(\frac{2}{\alpha-2} + \frac{1}{\alpha-1} + 3\Big)}$$

Then (VII.2) becomes

$$I_{total} < \sum_{i=2}^{\infty} \frac{4(2i-1)q(2q-1)}{(q-1)^2} \cdot \frac{\theta P}{(i-1)^{\alpha} r_I^{\alpha}}$$

$$= \frac{4q(2q-1)}{(q-1)^2} \cdot \frac{\gamma\beta\theta^2 N}{q^{\alpha}} \sum_{i=2}^{\infty} \frac{2i-1}{(i-1)^{\alpha}} \qquad \text{(VII.3)}$$

(VII.3) is obtained by plugging in

$$r_I = q \cdot r_T = q \sqrt[\alpha]{\frac{P}{\gamma\beta\theta N}}$$

In (VII.3)

$$\sum_{i=2}^{\infty} \frac{2i-1}{(i-1)^{\alpha}} = \sum_{i=2}^{\infty} \left[\frac{2(i-1)}{(i-1)^{\alpha}} + \frac{1}{(i-1)^{\alpha}}\right]$$

$$= 2\sum_{i=2}^{\infty} \frac{1}{(i-1)^{\alpha-1}} + \sum_{i=2}^{\infty} \frac{1}{(i-1)^{\alpha}}$$

$$= 2\sum_{j=1}^{\infty} \frac{1}{j^{\alpha-1}} + \sum_{j=1}^{\infty} \frac{1}{j^{\alpha}}$$

From elementary calculus we know that

$$\sum_{j=1}^{\infty} \frac{1}{j^{\alpha}} \leq \frac{1}{\alpha-1} + 1 \quad \text{, plugging this in } \Rightarrow$$

$$\sum_{i=2}^{\infty} \frac{2i-1}{(i-1)^{\alpha}} \leq \frac{2}{\alpha-2} + \frac{1}{\alpha-1} + 3 \qquad \text{(VII.4)}$$

Also, in (VII.3) the term

$$\frac{4q(2q-1)}{(q-1)^2} \text{ is strictly increasing in } (1, \infty).$$

In practice, q, namely the ratio of interference radius to transmission radius is $3 \sim 5$, and we could assume $q \geq 2$ to obtain

$$\frac{4q(2q-1)}{(q-1)^2} \leq 6$$

Plugging in (VII.4) and the above expression into (VII.3), we obtain

$$I_{total} < \frac{24\gamma\beta\theta^2 N}{q^{\alpha}}\Big(\frac{2}{\alpha-2} + \frac{1}{\alpha-1} + 3\Big) = (\gamma-1)N$$

since $q = \sqrt[\alpha]{\frac{24\gamma\beta\theta^2}{\gamma-1}\Big(\frac{2}{\alpha-2} + \frac{1}{\alpha-1} + 3\Big)}$. This theorem is thus proved. □

Corollary 7.2: The SINR at any intended receiver at any time is strictly greater than β.

(*Proof.*) At any intended receiver, the signal strength is at least $\frac{P}{\theta r^{\alpha}}$ where r is the distance between the designated sender and its intended receiver and $r < r_T$. Therefore, the signal strength is at least $\frac{P}{r_T^{\alpha}} = \frac{P}{\theta(P/\gamma\beta\theta N)} = \gamma\beta N$. Theorem 7.3 tells as the overall interference is strictly less than $(\gamma-1)N$, so the SINR at any intended receiver is strictly greater than $\frac{\gamma\beta N}{(\gamma-1)N+N} = \gamma\beta$. Remember that we have made the connectivity assumption in Section III, in which the disk graph generated by V and $\sqrt[\alpha]{\frac{\theta P}{\gamma N\beta}}$ is connected. □

Corollary 7.2 tells us that Algorithm 3 is also a valid scheduling algorithm for SINR model.

Corollary 7.3: Our broadcast algorithm for SINR model has latency bounded by

$$1 + 6\left\lceil \frac{2}{3}\left(\sqrt[\alpha]{\frac{24\gamma\beta\theta^2}{\gamma-1}\Big(\frac{2}{\alpha-2} + \frac{1}{\alpha-1} + 3\Big)} + 2\right)\right\rceil^2 (R-1)$$

Note that the number of colors depends on r_I/r_T instead of number of nodes. Also, broadcast latency is invariant of the number of nodes. This is because we applied the technique of constructing a virtual backbone, which plays a vital role in coloring. The number of nodes in this virtual backbone directly affects the latencies and it is not affected by the number of nodes in the whole network.

VIII. Simulation Results

Simulations have been performed in Matlab to evaluate the latency of our proposed scheme. In these simulations, n nodes were distributed randomly into a square region of size X by Y, where X and Y are normalized to the transmission range r_T. The transmission latency was then measured after our proposed scheme is employed. We measured three different latencies in our simulations:

- Transmission latency based on Theorem 7.3. Such a transmission latency can be easily found when the maximum depth of the BFS tree is identified;
- Compact transmission latency when all idle time slots are removed in our transmission schedule;
- Color-compact transmission latency is the even shorter latency in which, in addition to removing all idle time slots, senders in the same depth are allowed to send their broadcasts in the same time slots when they have the same tessellation colors. Therefore, the shortened latency of color-compact transmission latency compared to the compact transmission latency represents the benefit of tessellation coloring (and therefore, exact location information of every node).

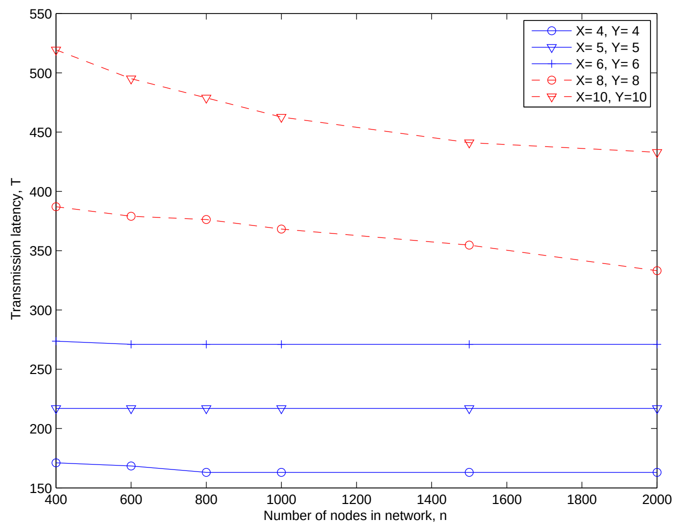

Fig. 9. Transmission latency for different network area sizes ($k = 3$).

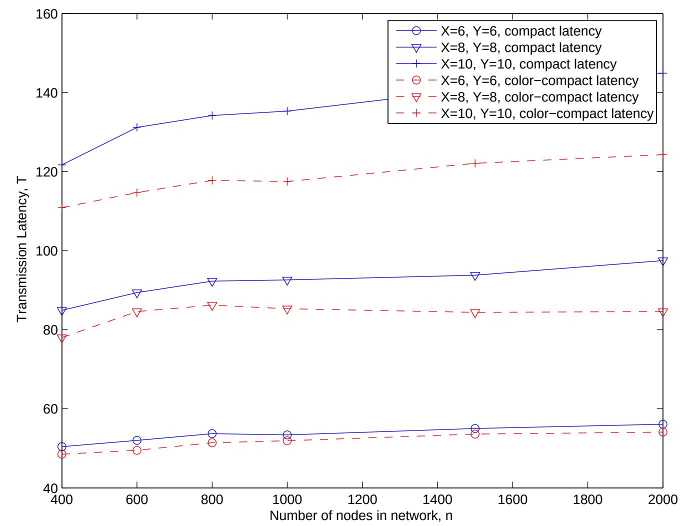

Fig. 11. Comparing the compact transmission latency and the color-compact transmission latency ($k = 3$).

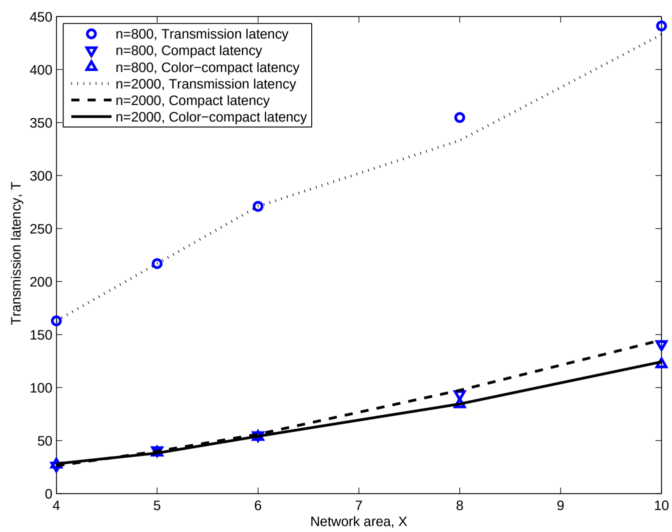

Fig. 10. Transmission latency for different number of nodes ($k = 3$).

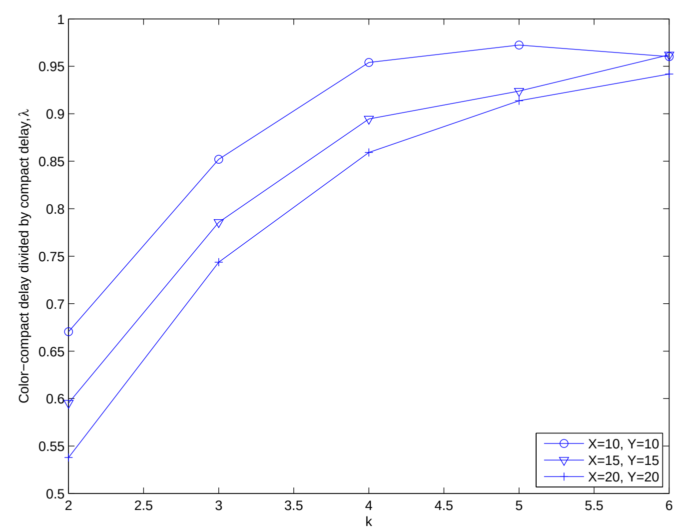

Fig. 12. Comparing the compact transmission latency and the color-compact transmission latency ($n = 2000$).

Note that the last two latency measurements were based on the assumption that such removal of idling time slots is possible, which requires some extra communication between nodes in different BFS tree depths.

Figure 9 shows the transmission latency as a function of number of nodes in the network, n, for different network area sizes, X. The value of k was set to 3 in these simulations. By Fig. 9, the transmission latencies remain almost the same when the number of nodes in network, n is larger than 1000 for each set of X and Y. This is actually expected: the increase of n does not change the transmission tessellation and its depth significantly (as discussed in Section VII). As the network size increases, the transmission latency becomes longer. This is because of the increased depth of the virtual backbone.

Figure 10 shows the three types of transmission latency as a function of network area sizes, X, for different numbers of nodes in the network, n. The value of k was set to 3 in these simulations. It can be seen that compact transmission latency is significantly shorter than the regular transmission latency. The color-compact transmission latency is even shorter, due to the scenarios where senders of the same tessellation-color are allowed to broadcast in the same time slots.

We compare the compact transmission latency and color-compact transmission latency in Fig. 11. The benefits of tessellation coloring is clearly shown.

The transmission latency of our proposed scheme with different $k = \left\lceil \frac{2}{3}\left(\frac{r_I}{r_T} + 2\right) \right\rceil$ is presented in Fig. 12. In this figure, we show the ratio of color-compact transmission latency divided by compact transmission latency, λ, as k increases from 2 to 6. Different network area sizes have been simulated. When k is smaller, more concurrent transmissions can be scheduled in the same time slot in the same depth, reducing λ.

Remarks on Distributed Implementation

Our algorithm can be modified to a distributed version for the following reason. It makes use of the following centralized information. (1) layer information in Algorithm 1 (2) MIS in Algorithm 1 (3) BFS tree in Algorithm 2 (4) color information in Algorithm 3. In (1), each node only needs to know its layer number. In (2) each node only needs to know whether or not it itself is in the MIS. In (3) each node only needs to know its parent in BFS tree. In (4), each node only needs to know its color. (1) and (2) have distributed algorithms because there are distributed BFS algorithms [2]. (3) is related to MIS and there are distributed MIS algorithms in the literature too [23], [24]. However, we need to modify those algorithms slightly and apply them layer by layer. (4) could have distributed implementations provided that each node knows its location. This may be possible if each node has a GPS device for example, or each node is given the location information when it is deployed.

Remarks on Varying r_T and r_I

Varying the values of transmission/interference ranges does not affect our algorithm; it only affects the followings. (1) Graph topologies G_T and G_I (2) Coloring (since $k = \lceil \frac{2}{3}(\frac{r_I}{r_T} + 2) \rceil$ depends on them). From a practical point of view, varying values of transmission/interference ranges only affects certain system parameters; it does not affect any algorithms/subroutines.

IX. Conclusion and Future Work

Many highly theoretical models were used in all previous works on broadcast scheduling. Instead, we have used two more practical models to re-investigate this problem. Surprisingly, we found that we can apply the same method to both models and obtain low-latency schedules. Although our proposed algorithms are centralized, we did not formulate the minimum latency problem as an optimization problem (such as linear programming) and find the optimal solution for the following reasons. First, this problem in general graph model was proposed in [6] in 1985, and so far there is still no good formulation to represent it as a linear programming problem. The main reason for that is the difficulty to represent the following condition: “a node can only transmit if it has successfully received from another node”. So far, there is still no good formulation to represent this condition even in the general graph model, so we believe it is more difficult to represent it in our more complicated 2-disk/SINR model. Second, the broadcast latency problem in disk graph has proven to be NP-hard [14], and this problem in our 2-Disk and SINR models can be regarded as a more general case and is therefore also NP-hard. For this reason, finding optimal solution is difficult.

For future work, there are two promising directions as follows. The first is to apply our techniques to directional antennae. We believe most techniques developed here can be applied to the case of directional antennae by re-investigating their geometrical properties, although the models may need to be redefined accordingly. The second direction is to apply these techniques to data aggregation (or convergecast) scheduling. In such a scenario, all nodes wish to transmit their data back to a fixed sink node. This could be regarded as a reverse-direction broadcast. The major difference is that in a broadcast a node can transmit to many nodes at the same time while in a data aggregation many nodes cannot transmit to one sink in one time slot. This property makes data aggregation fundamentally different from broadcast, but we believe that we can still apply several techniques developed in this work. For these reasons, we believe this work will be an important start that bridges the gap between theory and practice.

Acknowledgment

S.C.H Huang was supported in part by the Research Grant Council of Hong Kong under the project CERG CityU 113807. P.J. Wan is supported in part by the Research Grants Council of Hong Kong under Project Nos. CityU 114307, CityU 122105 and CityU 122807. J. Deng was supported in part by Louisiana Board of Regents RCS grant LEQSF (2005-08)-RD-A-43. We would like to give special thanks to the associate editor, Prof. M. Liu and the three anonymous reviewers for their insightful comments.

References

[1] N. Alon, A. Bar-Noy, N. Linial, and D. Peleg. A lower bound for radio broadcast. *Journal of Computer and System Sciences*, 43(2):290–298, 1991.

[2] B. Awerbuch and R. Gallager. A new distributed algorithm to find breadth first search trees. *IEEE Transactions on Information Theory*, 33(3):315–322, 1987.

[3] R. Bar-Yehuda, O. Goldreich, and A. Itai. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1):104–126, 1992.

[4] S. Basagni, I. Chlamtac, and D. Bruschi. A mobility-transparent deterministic broadcast mechanism for ad hoc networks. *IEEE/ACM Transactions on Networking*, 7(6):799–807, 1999.

[5] D. Bruschi and M. Del Pinto. Lower bounds for the broadcast problem in mobile radio networks. *Distributed Computing*, 10(3):129–135, 1997.

[6] I. Chlamtac and S. Kutten. On broadcasting in radio networks–problem analysis and protocol design. *IEEETransactions on Communications*, 33:1240–1246, 1985.

[7] I. Chlamtac and O. Weinstein. The wave expansion approach to broadcasting in multihop radio networks. *IEEETransactions on Communications*, 39:426–433, 1991.

[8] B. S. Chlebus, L. Gąsieniec, A. Gibbons, A. Pelc, and W. Rytter. Deterministic broadcasting in unknown radio networks. In *Symposium on Discrete Algorithms*, pages 861–870, 2000.

[9] A. Dessmark and A. Pelc. Tradeoffs between knowledge and time of communication in geometric radio networks. In *SPAA'01: Proc. of the 13th annual ACM symposium on Parallel algorithms and architectures*, pages 59–66, New York, NY, USA, 2001. ACM Press.

[10] M. Elkin and G. Kortsarz. Logarithmic inapproximability of the radio broadcast problem. *Journal of Algorithms*, 52:8–25, 2004.

[11] M. Elkin and G. Kortsarz. Polylogarithmic additive inapproximability of the radio broadcast problem. In *APPROX-RANDOM'04: Proc. of the 7th Int'l Workshop on Approximation Algoorithms for Combinatorial Optimization Problems*, 2004.

[12] M. Elkin and G. Kortsarz. An improved algorithm for radio networks, 2005. An earlier version appeared in *SODA'05*.

[13] I. Gaber and Y. Mansour. Centralized broadcast in multihop radio networks. *Journal of Algorithms*, 46(1):1–20, 2003.

[14] R. Gandhi, S. Parthasarathy, and A. Mishra. Minimizing broadcast latency and redundancy in ad hoc networks. In *MobiHoc'03: Proc. of the 4th ACM international symposium on Mobile ad hoc networking and computing*, pages 222–232, 2003.

[15] L. Gąsieniec, D. Peleg, and Q. Xin. Faster communication in known topology radio networks. In *PODC'05: Proc. of the 24th annual ACM symposium on Principles of distributed computing*, pages 129–137, New York, NY, USA, 2005. ACM Press.

[16] P. Gupta and P. R. Kumar. Capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000.

[17] P. Gupta and P. R. Kumar. Internets in the sky: The capacity of three dimensional wireless networks. *Communications in Information and Systems*, 1(1):33–50, 2001.

[18] S. C.-H. Huang, P.-J. Wan, X. Jia, and H. Du. Low-latency broadcast scheduling schemes in ad hoc networks. In *WASA'06: 1st Int'l Conference on Wireless Algorithms, Systems, and Applications*, August 2006.

[19] D. R. Kowalski and A. Pelc. Optimal deterministic broadcasting in known topology radio networks. to appear in *Distributed Computing*.

[20] D. R. Kowalski and A. Pelc. Centralized deterministic broadcasting in undirected multi-hop radio networks. In *APPROX-RANDOM'04: Proc. of the 7th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, pages 171–182, 2004.

[21] E. Kushilevitz and Y. Mansour. An $\Omega(D \log(N/D))$ lower bound for broadcast in radio networks. *SIAM Journal on Computing*, 27:702–712, 1998.

[22] R. A. Leese. A unified approach to the assignment of radio channels on a regularhexagonal grid. *IEEE Transactions on Vehicular Technology*, 46(4):968–980, 1997.

[23] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 21(1):1036–1053, 1986.

[24] T. Moscibroda and R. Wattenhofer. Maximal independent sets in radio networks. In *PODC'05*, pages 148–157, 2005.

[25] T. Moscibroda, R. Wattenhofer, and A. Zollinger. Topology control meets SINR: the scheduling complexity of arbitrary topologies. In *MobiHoc'06: Proc. of the 7th ACM int'l symposium on Mobile ad hoc networking and computing*, pages 310–321, New York, NY, USA, 2006. ACM Press.

[26] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. In *MobiCom'99: 5th annual ACM/IEEE int'l conference on Mobile computing and networking*, pages 151–162, New York, NY, USA, 1999. ACM Press.

[27] J.-P. Sheu, P.-K. Hung, and C.-S. Hsu. Scheduling of broadcasts in multihop wireless networks. In *The handbook of ad hoc wireless networks*, pages 483–495. CRC Press, Inc., Boca Raton, FL, USA, 2003.

[28] R. Zheng and R. Barton. Toward optimal data aggregation in random wireless sensor networks. In *Infocom'07: Proc. of the 26th IEEE Int'l Conference on Computer Communications*, pages 249–257. IEEE, 2007.