Minimum Latency Gossiping in Radio Networks

Scott C.-H. Huang, Peng-Jun Wan, Hongwei Du, and E.-K. Park

Abstract—We studied the minimum latency gossiping (all-to-all broadcast) problem in multihop radio networks defined as follows: Each node in the network is preloaded with a message and the objective is to distribute each node's message to the entire network with minimum latency. We studied this problem in the unit-size message model and the unit disk graph model. The unit-size model means different messages cannot be combined as one message, and the unit disk graph model means a link exists between two nodes if and only if their euclidean distance is less than 1. The minimum latency gossiping problem is known to be NP-hard in these two models. In this work, we designed a gossiping scheme that significantly improved all current gossiping algorithms in terms of the approximation ratio. Our work has approximation ratio 27, a great improvement of the current state-of-the-art algorithm (which has ratio 1,947). We also discussed the single point of failure problem and its impact on our approximation ratio. We designed an amended gossiping algorithm with ratio 27 in case of a nonsource node failure. We also designed an amended gossiping algorithm with ratio 29 in case of source failure.

Index Terms—Gossip, all-to-all broadcast, latency.

1 Introduction

Broadcast is a fundamental operation in radio networks. Naïve flooding is not practical as it causes severe contention, collision, and congestion. Avoiding collision, reducing redundancy, as well as increasing reliability in radio networks are the main objectives of the broadcast storm problem [24]. Numerous network protocols are based on broadcasting: routing, information dissemination, and service/resource discovery. Since many systems have stringent end-to-end delay requirements, the design of low-latency broadcasting scheme is essential to many practical applications.

There are two basic tasks in network communication: broadcasting and gossiping. Broadcasting is distributing a single message from one source node to all other nodes. Gossiping is distributing a unique message from each node to all other nodes in the network. Essentially, gossiping can be viewed as *all-to-all broadcast*. There are three models regarding whether or not we can combine two or more messages as a single message: *unit-size*, *bounded-size*, and *unbounded-size* models.

In this paper, regarding the message models, we study the gossiping problem in the *unit-size* model in which multiple messages CANNOT be combined as a single message. As for our network model, we considered the *unit disk graph* model and designed the *Interleaved Gossiping Algorithm* that improved all previous gossiping algorithms in this model. Our algorithm is theoretically proved to be *constant approximation* with ratio 27. This is known to be the algorithm that has the lowest latency and the least ratio so far. We also discussed the single point of failure problem and its impact on our approximation ratio. We designed two amended gossiping algorithms for the case of node failure. We found that, if the failed node is not the source, then our amended gossiping algorithm still has approximation ratio 27. If the source fails, the amended gossiping algorithm has ratio 29.

The rest of the paper is organized as follows: We present the problem formulation in Section 2. Section 3 discusses about related work and a naïve gossiping algorithm is presented in Section 4. A better algorithm, namely the interleaved gossiping algorithm, is presented in Section 5. A concrete example is shown in Section 6 to help understand our algorithm. The single point of failure problem is discussed in Section 7. Finally, we conclude the work in Section 8. The proofs of all lemmas and theorems are given in the Appendix.

- *S.C.-H. Huang is with the Department of Computer Science, City University of Hong Kong, Tat Chee Ave., Kowloon, Hong Kong SAR. E-mail: shuang@cs.cityu.edu.hk.*
- *P.-J. Wan is with the Department of Computer Science, Illinois Institute of Technology, 10 W. 31st Street, Chicago, IL 60616. E-mail: wan@cs.iit.edu.*
- *H. Du is with the Department of Computer Science, Illinois Institute of Technology, 10 W. 31st Street, Chicago, IL 60616, and the Academy of Mathematics and System Sciences, Chinese Academy of Sciences, China. E-mail: hongwei.du@iit.edu.*
- *E.-K. Park is with the CSEE Department, School of Computing and Engineering, University of Missouri at Kansas City, RFH RM 560B, 5100 Rockhill Road, MO 64110. E-mail: ekpark@umkc.edu.*

Manuscript received 31 Oct. 2008; revised 14 May 2009; accepted 17 July 2009; published online 28 July 2009.
Recommended for acceptance by A. Boukerche.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2008-10-0435.
Digital Object Identifier no. 10.1109/TPDS.2009.123.

2 Problem Formulation

We consider a network of N nodes. Each node is equipped with an RF transceiver that can be used to send or receive data. We use a graph $G = (V_G, E_G)$ to represent the topology of this network in which V_G is the vertex set representing nodes and E_G is the edge set representing links. Since each node in this network corresponds uniquely to a vertex in V_G and vice versa, we do not distinguish between vertices and nodes and use them interchangeably. In convention, we use V to represent the set of nodes and V_G to represent the set of vertices in the graph, and there is a one-to-one mapping between them. We consider omnidirectional antennae only, and a node's transmission/reception range is roughly a disk centered at that node. This type

1045-9219/10/$26.00 © 2010 IEEE Published by the IEEE Computer Society

of network can be represented as a *disk graph* as follows: An arc (or directed edge) exists from u to v if and only if v lies in u's transmission area (which is a disk). For simplicity, we further assume that all nodes have the same transmission range. In such a case, we can normalize their radius to 1, and an edge exists between u, v if and only if the distance between them is less than 1. Note that in this case, all edges are bidirectional and we can simply use an undirected graph to represent the network topology. This is called the unit disk graph (UDG) model. In this work, for simplicity, we only consider the UDG model.

A node can either send or receive data at one time, and it can receive data correctly only if exactly one of its neighbors is transmitting at one moment. If two or more nodes are transmitting simultaneously and there is a node in their overlapped transmission area, then this node cannot receive the message clearly since both transmissions are interfering with each other. This type of situation is called *collision*. A node is equipped with some memory, so it can store messages received from its neighbors and forward to other nodes. Note that a node cannot forward a message unless it has already received from the node having that message.

Given a graph $G = (V_G, E_G)$, a TDMA schedule can be modeled as a function $\mathcal{W}$ from the set of natural numbers $\mathbb{N}$ (representing time) to V_G's power set (i.e., subsets of V_G) as follows: $\mathcal{W} : \mathbb{N} \to 2^{V_G}$ in which time $t \in \mathbb{N}$ is mapped to $\mathcal{W}(t) \subset V_G$, denoting that the nodes in $\mathcal{W}(t)$ are scheduled to transmit at time t. The *latency* of a TDMA schedule $\mathcal{W}$ is the last time slot such that there are still some node(s) transmitting. Formally, the latency of $\mathcal{W}$ can be defined as $lat(\mathcal{W}) = \min\{t \mid \mathcal{W}(t') = \emptyset, \forall t' > t\}$.

Minimum latency gossiping problem. Given a UDG $G = (V_G, E_G)$ such that each node has a message (depending on this node), find a TDMA schedule of minimum latency such that each node successfully distributes its own message to the entire network.

Note that, in order to make sense for this problem, we have to assume that the network is strongly connected. Essentially, the gossiping task can be viewed as *all-to-all broadcast*. Since both terms have been used extensively in the literature and they were studied in different message models in different context, we have to distinguish between the following three cases in order to clarify their problems and objectives:

1. *Messages of unit size*: In this model, all messages have the same size. If a node needs to forward two messages, it has to do two separate transmissions. Those that belong to this category include [14], [16], [22].
2. *Messages of bounded size*: In this model, nodes are allowed to send a combined message up to some limit (particularly up to $\log N$, where N is the number of nodes in the network). Those that belong to this category include [1], [5]. In [1], nodes can combine up to $\log N$ messages.
3. *Messages of unbounded size*: In this model, nodes are allowed to send a combined message including all messages it has received so far. Those that belong to this category include [27], [17] [15], [6], [8].

Although they all look alike, this little difference is vital in designing corresponding gossiping algorithms because each case has its own lower bound as well as advantages/disadvantages. The gossiping problem in these three cases should be treated as three fundamentally different problems. Note that our message model is one, the unit-size model.

Now, we introduce our terms, notations, and simple facts. Let $G = (V_G, E_G)$ be an undirected graph with $|V_G| = N$. A graph center s is a node in a graph G such that the length of the shortest path from s to the farthest node is minimized. Graph centers may not be unique (i.e., there can be more than one node satisfying this property). Let s' be a fixed node in G. The subgraph of G induced by a subset U of V_G is denoted by $G[U]$. The *depth* of a node v is the distance between v and s', and the *radius* of G with respect to s', denoted by R, is maximum distance of all the nodes from s'. They can be computed by conducting a standard breadth-first search (BFS)[10] on G. For $0 \leq i \leq R$, the layer i of G consists of all nodes of depth i.

Let X and Y be two disjoint subsets of V_G. We call X a *cover* of Y if each node in Y is adjacent to some node in X. We call X a *minimal cover* of Y if X is a cover of Y and no proper subset of X is a cover of Y. If X is a cover of $V \setminus X$, then X is called a *dominating set* of G. If X is a dominating set and $G[X]$ is connected, then X is called a *connected dominating set* of G. Suppose that X is a cover of Y. A node $y \in Y$ is said to be a *private neighbor* of $x \in X$ if and only if y is a neighbor of x but y is not a neighbor of any other node in $X - \{x\}$.[1]

A subset $U \subset V_G$ is an *independent set* (IS) of G if the nodes in U are pairwise nonadjacent, and a *maximal independent set* (MIS) U of G is an IS of G but no proper superset containing U is an IS of G. Any node ordering $v_1, v_2, \ldots, v_N$ of V_G induces an MIS U in the following first-fit manner: Initially, $U = \{v_1\}$. For $i = 2$ up to n, add v_i to U if v_i is not adjacent to any node in U. Clearly, any MIS of G is a dominating set of G. If G is a UDG, then a set U is an IS of G if and only if any pair of nodes in U are separated by a euclidean distance greater than one. In addition, each node can be adjacent to at most five nodes in any IS.

3 Related Work

Lots of works have been done about the gossiping problem in unbounded-size model [4], [6], [7], [9], [15], [19], [25], [26]. Chrobak et al. showed that deterministic gossiping can be performed in unknown directed ad hoc radio networks in $O(n^{3/2} \log^2 n)$ time [6], [8]. A constructive version of their algorithm was recently proposed by Indyk [19]. This result was improved by Gasieniec and Lingas for networks with diameter $D = n^{\alpha}$, $\alpha < 1$ [15]. They presented a gossiping algorithm that runs in $O(n\sqrt{D} \log^2 n)$ time. These results show that radio networks with a long diameter constitute a bottleneck in deterministic radio gossiping with messages of an arbitrary size. There is another approach for the radio gossiping algorithm with running time $O(Dd^2 \log^3 n)$, where d stands for the maximum in-degree of the underlying graph of connections. Chrobak et al. proposed a randomized radio gossiping algorithm with expected running time $O(n \log^4 n)$ [7]. Studies on oblivious gossiping in ad hoc radio networks can be found in [4]. Ravishankar and

1. By convention, $A - B$ represents the ordinary set minus $A \setminus B$, but it also implies $B \subset A$.

Singh studied this problem in two different models. They presented distributed gossiping algorithms for networks with nodes placed randomly on a line [25] and a ring [26].

The gossiping problem in bounded-size model was studied in the matching model [2], [3]. The results presented in [2] include the studies of the exact complexity of the gossiping problem in Hamiltonian graphs and k-ary trees, and optimal asymptotic bounds for general graphs (in the matching model with unit messages). It also contains a number of asymptotically optimal results in matching model with messages of arbitrarily bounded size. Flamini and Perennes also studied the gossiping problem in bounded-size model [11]. They focused on graphs with bounded degree. Bar-Yehuda et al. proposed a randomized gossiping algorithm in log-size model for unknown topology [1]. Christersson et al. studied deterministic $b(n)$-gossiping algorithms in ad hoc radio networks meaning that each combined message contains at most $b(n)$ single messages or bits of auxiliary information, where b is an integer function and n is the number of nodes in the network. They derived theoretical upper bounds for many functions b [5].

Gossiping in the unit-size model has also been studied in the literature. Gasieniec and Potapov studied this problem in general graph model [16]. They proposed several optimal or close to optimal $O(n)$-time gossiping procedures for various standard network topologies, including lines, rings, stars, and free trees. They also proved that there exists a radio network topology in which the gossiping (with unit-size messages) requires $O(n \log n)$ time. Manne and Xin designed a randomized gossiping algorithm in general graphs that has latency $O(n \log n)$ [22]. Gandhi et al. studied the broadcast scheduling problem in the UDG model and designed a constant approximation algorithm [14]. This algorithm, if used in conjunction with Algorithms 1 and 2, becomes a *gossiping algorithm* of constant approximation. However, this combined algorithm is too large to be practical because its approximation ratio is estimated to be 1,947. This estimation is presented in Lemma 5 in the appendix. Our interleaved gossiping algorithm is a significant improvement and has ratio 27.

4 Naïve Gossiping Algorithm

We start off with a naïve gossiping algorithm, which will later be improved by the interleaved gossiping algorithm. The naïve gossiping algorithm has three phases.

4.1 Phase I: Preprocessing

We assume that we can find a graph center s in the network. This is possible by applying the Floyd-Warshall algorithm [13]. Since it computes the lengths of the shortest paths between every pair of nodes, we can simply look at the distances between every pair of nodes and find such a node. The details can be found in [10], [13]. The graph center s has the property that the radius of the whole network with respect to s is minimized. Note that in order to running the Floyd-Warshall algorithm, the knowledge of the whole network topology is required, which may not be practical in some scenarios. In such a case, we can arbitrarily pick up a node as the graph center and run the same algorithm. Although this way the network radius may be increased, we are guaranteed that such a substitute will at most double the network radius, according to Lemma 1. Therefore, the overall gossiping latency and approximation ratio will be at most doubled.

Lemma 1. *In a graph $G = (V_G, E_G)$ with $s_1, s_2 \in V_G$. Suppose that R_1, R_2 are the radii of G with respect to s_1, s_2, respectively, then $R_1 \leq 2R_2$.*

Proof. Let $d_G(x, y)$ denote the hop distance between x and y in G. Then, according to the definition of radius, $R_1 = \max_{x \in G} d_G(s_1, x)$. Since $d_G(s_1, x) \leq d_G(s_1, s_2) + d_G(s_2, x)$, we have $\max_x d_G(s_1, x) \leq d_G(s_1, s_2) + \max_x d_G(s_2, x) \leq 2 \max_x d_G(s_2, x)$. Therefore, $\max_x d_G(s_1, x) \leq 2 \max_x d_G(s_2, x)$ and we are done. □

Now we construct a BFS tree T_{BFS} (details can be found in [10]) for the network and divide all nodes into layers (where the 0th layer is s alone and the 1st layer is its neighbors). We sort all nodes in V according to their distances to s in ascending order. Let $BLACK$ be the MIS of G induced by such a node ordering. The nodes in $BLACK$ are referred to as the *black nodes*, or *dominators*, as $BLACK$ is also a dominating set of G. The nodes not in $BLACK$ are called *white nodes*.

We construct the broadcast tree T_{BR} according to Algorithm 1. In the process of construction, we will choose blue nodes as connectors to obtain a connected tree. Note that, when this construction is completed, in T_{BR}, each black node will have a blue parent at the upper layer and each blue node will have a black parent *at the same layer or the layer right next to it above*.

Algorithm 1. Construction of the broadcast tree T_{BR}

```
Connect s to all nodes in L_1
  /* At each layer i, we pick blue nodes to connect black
  nodes at layer i and i+1 as follows: */
for each layer i from 1 to l-1
                    /* Connect BLACK_{i+1} to BLUE_i */
   for each black node v ∈ BLACK_{i+1}
      Find its parent p(v) in T_BFS
      Add p(v) to BLUE_i
      Connect p(v) to v in T_BR
   end for
                    /* Connect BLUE_i to p(BLUE_i) */
   for each blue node w ∈ BLUE_i
      Find w's dominator d_w
   /* We can prove that w must be adjacent to at least a
   black node d_w either at layer i or i-1. This property
   holds due to the MIS selection method. */
      Connect d_w to w in T_BR
   end for
end for
                    /* Connect remaining white nodes */
for each remaining white node u
   Find u's dominator d_u
   /* u must be adjacent to at least a black node d_u due to
   the maximality of the MIS. */
   Connect u to d_u in T_BR
end for
```

4.2 Phase II: Data Collection

The algorithm in this phase is based on [12]. We modified their algorithm slightly to fit our scenario here. In this phase,

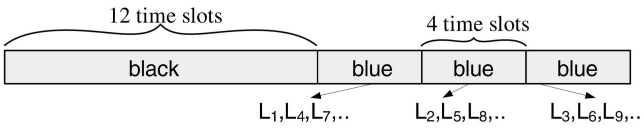


Fig. 1. Group 24 time slots in which the first 12 slots are for black nodes' transmission and the remaining 12 slots are divided into three groups for blue nodes' transmission.

each node has a message to transmit and all messages are relayed to s. We can use a simple interleaving algorithm as shown in Algorithm 2. Later we will show that there will be no collision at all and this phase terminates after $3(N-1)$ time slots, where N is the number of nodes in the network.

Algorithm 2. Data collection

/* We group L_i's according to $i \bmod 3$. */

1: Starting from $t \leftarrow 0$.
2: **repeat**
3: Pick a node $x_i \in L_i$ for each layer i with $i \equiv 1 \bmod 3$ such that that x_i either needs to transmit or forward a message to its parent in T_{BFS}. Schedule these x_i's to transmit concurrently in time slot t. $t \leftarrow t+1$
4: Pick a node $x_i \in L_i$ for each layer i with $i \equiv 2 \bmod 3$ such that that x_i either needs to transmit or forward a message to its parent in T_{BFS}. Schedule these x_i's to transmit concurrently in time slot t. $t \leftarrow t+1$
5: Pick a node $x_i \in L_i$ for each layer i with $i \equiv 3 \bmod 3$ such that that x_i either needs to transmit or forward a message to its parent in T_{BFS}. Schedule these x_i's to transmit concurrently in time slot t. $t \leftarrow t+1$
6: **until** all nodes have finished transmitting and forwarding

4.3 Phase III: Naïve Broadcast

We simply apply the EBS algorithm in [18] as follows: Starting from time 0, we schedule s to release a new message every 48 time slots by applying the EBS algorithm in [18] until all N messages have been released. All relay transmissions of different packets released by s are executed in an interleaving manner.

Now we state the following two theorems regarding the correctness of the naïve gossiping algorithm and the latency of the naïve broadcast schedule. Their proofs can be found in the appendix:

Theorem 1. *Collision will not happen in the naïve gossiping algorithm.*

Theorem 2. *The naïve broadcast schedule has latency at most* $48(N+R-2)+1$, *where* N, R *are the number of nodes in the network and the radius of the network, respectively.*

This latency can be further reduced in the next section. Now we state the approximation ratio regarding the naïve gossiping algorithm as follows:

Theorem 3. *The approximation ratio of the naïve gossiping algorithm is at most 51.*

5 Interleaved Gossiping Algorithm

The naïve gossiping algorithm has approximation ratio 51. Although it's already an improvement of [14], we can further reduce the latency by interleaving those broadcasts to the fullest extent. We present our interleaved gossiping algorithm, which simply replaces the phase III (naïve broadcast) by the interleaved broadcast (Algorithm 3).

Interleaved broadcast (to be used in Phase III). First, we color all black nodes by applying the smallest-degree-last ordering in G^2 as described in [18]. Here, the *colors* are some integers (from 1 to 12) to be used in Algorithm 3. These colors are unrelated to black, blue, or white nodes, introduced in Algorithm 1. We need 12 colors according to [18].

Now we group every 24 time slots as one unit as shown in Fig. 1. Black nodes only transmit within the first 12 slots and blue nodes only transmit within the last 12 slots. In each 24-slot unit, time slots 13-16 are reserved (i.e., will be used) only for the blue nodes at layer i such that $i \equiv 1 \bmod 3$. Similarly, time slots 17-20 are reserved only for the blue nodes at layer i such that $i \equiv 2 \bmod 3$, and time slots 21-24 are reserved only for the blue nodes at layer i such that $i \equiv 0 \bmod 3$, as shown in Fig. 2. The details of the interleaved broadcast algorithm are given in Algorithm 3. Note that, in a 24-slot unit, each black and blue node transmits exactly once.

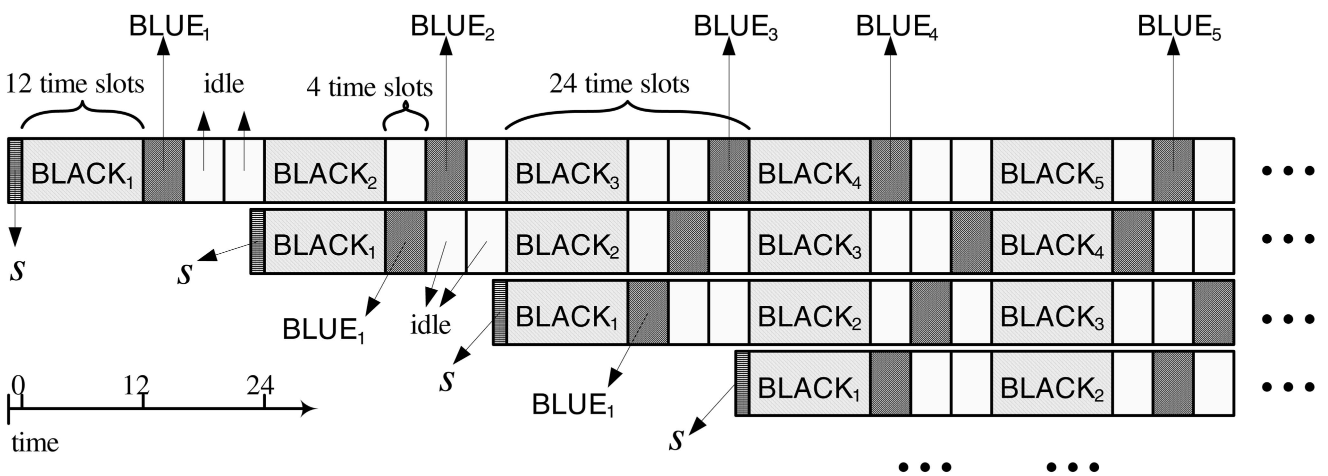


Fig. 2. Interleaved broadcast scheduling. Blank blocks represent idling time slots. The figure clearly shows that s releases four packets in the first 96 time slots and these four broadcasts are interleaved.

Algorithm 3. Interleaved broadcast schedule

Color all black nodes by applying the smallest-degree-last ordering in G^2 using 12 colors as described in [18]. Use C_{12} to denote this coloring, where C_{12} is a function that maps a node x to a number in $\{1, 2, \ldots, 12\}$.
for i from 1 to R
    apply IMC (Algorithm 4) with $X = BLUE_i$, and $Y = \{y \mid y \text{ is a child of some node } x \in BLUE_i \text{ in } T_{BR}\}$ to obtain $W_0, W_1, \ldots$. Define the function $\ell : BLUE_i \to \mathbb{N}$ as follows: $\ell(x)$ is defined as $\ell(x) = l$ if $x \in W_l$.
end for
    /* Note that ℓ is well defined on the whole vertex set $BLUE$ since $BLUE = \bigcup_i BLUE_i$. Also, $\ell(x) \leq 4$ for all $x \in BLUE$ since a blue node can have at most four black children. */
Starting from $t \leftarrow 0$, schedule s to release a new message every 24 time slots until all n messages have been released.
for each layer i from 1 to R
Schedule each node $x \in L_i$ according to the following criteria.
    if x is black, schedule x to transmit at time $t + C_{12}(x)$.
    if x is blue and $i \equiv 0 \bmod 3$, schedule x to transmit at time $t + 12 + \ell(x)$.
    if x is blue and $i \equiv 1 \bmod 3$, schedule x to transmit at time $t + 16 + \ell(x)$.
    if x is blue and $i \equiv 2 \bmod 3$, schedule x to transmit at time $t + 20 + \ell(x)$.
    After having scheduled all nodes in L_i, we update $t \leftarrow t + 24$
end for

Algorithm 4. Iterative minimal cover (IMC)

Input: A graph $G = (V, E)$, vertex subsets $X, Y \subset V$ such that X is a cover of Y in G
Output: Disjoint vertex subsets $W_0, W_1, \ldots, W_l$ such that $\bigcup_{i=0}^{l} W_i = X$.
Initialize $i \leftarrow 0, X_0 \leftarrow X, Z \leftarrow Y$
repeat
    $i \leftarrow i + 1$
    Find a minimal cover $X_i \subset X_{i-1}$ of Z.
    $W_{i-1} \leftarrow X_{i-1} - X_i$
    $Z \leftarrow Z - \{z \in Z \mid z \text{ is a private neighbor of some node } x \in X_i\}$
    /* private neighbors are defined in §2, meaning z not adjacent to any other node in X_i here. */
until $Z = \emptyset$
return $W_0, W_1, \ldots, W_i$

Algorithm 5. Minimal cover construction (sequential pruning algorithm)

Input: A graph $G = (V, E)$ with vertex subsets $X, Y \subset V$ such that X is a cover of Y in G. An ordering $x_1, x_2, \ldots, x_m$ of X.
Output: Minimal cover $W \subset X$
Initialize $W \leftarrow X$.
for each $i \leftarrow m$ to 1
    if $W - \{x_i\}$ is a cover of Y, remove x_i from W.
end for
return W

Now we state the following two theorems regarding the correctness and latency of the interleaved broadcast schedule. Their proofs can be found in the appendix:

Theorem 4. *Collision will not happen in the interleaved gossiping algorithm.*

Theorem 5. *The interleaved broadcast schedule defined in Algorithm 3 has latency* $24(N + R - 2) + 1$.

Now we state the following theorem regarding the approximation ratio of the interleaved gossiping algorithm:

Theorem 6. *The approximation ratio of our interleaved gossiping algorithm is at most 27.*

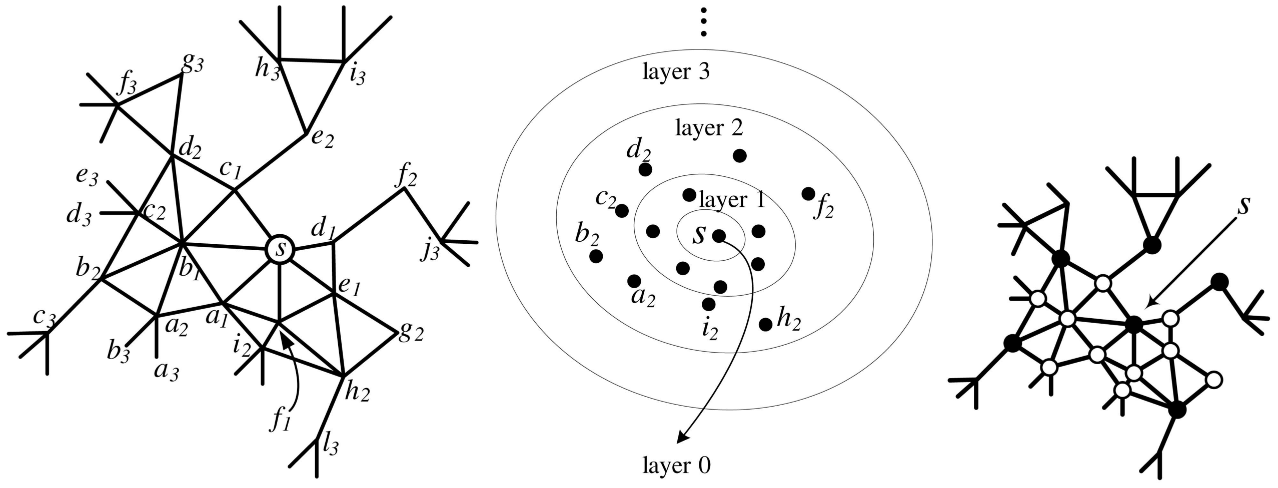
Theorems 2 and 5 are about the latency of the naïve and interleaved gossiping algorithms, respectively. Regarding the message complexity of a specific node, we use the *number of transmissions* to evaluate our algorithms. As for the whole network, we define the *maximum number of transmissions* as the maximum of the number of transmissions of a node, where the maximum is taken over all nodes. We state the following theorem and leave its proof in the Appendix:

Theorem 7 (Message complexity). *In both naïve and interleaved gossiping algorithms, the maximum number of transmissions of the whole network is at most* $2N$.

6 An Example

In this section, we present an example of the interleaved gossiping algorithm. Consider a network topology shown in Fig. 3a. We divide all nodes into layers, as shown in Fig. 3b. Then we construct the MIS layer by layer as shown in Fig. 3c. In the first step, s is selected in the MIS and added to $BLACK_0$. In the second step, since the source is black, all nodes at layer 1 must be white; otherwise, it won't be independent of s. In the third step, we select an MIS b_2, d_2, e_2, f_2, h_2 at layer 2, which must also be independent of the black nodes of the previous layer, $BLACK_1$, although there is no black node at layer 1 and this does not have any effect. Fig. 3c shows that b_2, d_2, e_2, f_2, h_2 are added to $BLACK_2$. (Note that node identifiers are only shown completely in Fig. 3a, and not shown in Fig. 3c for simplicity.) We keep doing this and select black nodes until all layers have been worked in this way. The black node selection depends solely on G and there is nothing to do with the BFS tree. Not until all blue nodes have been selected do we need to consider the BFS tree, as shown in Fig. 4a. In Algorithm 1, we are trying to add appropriate blue nodes to interconnect all black ones. Since the source s 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 v at layer 2, we find its parent $p(v)$, add $p(v)$ to $BLUE_1$, and connect $v, p(v)$ in the BFS tree, as shown in Fig. 4b.

In Fig. 4b, we see that b_1, c_1, d_1, e_1 at layer 1 are added to $BLUE_1$ and connected to some black nodes at layer 2. a_1, f_1 are not added to $BLUE_1$, so they remain in $WHITE$. We



(a) (b) (c)

Fig. 3. (a) G's topology. (b) Layers of G. (c) Layered MIS.

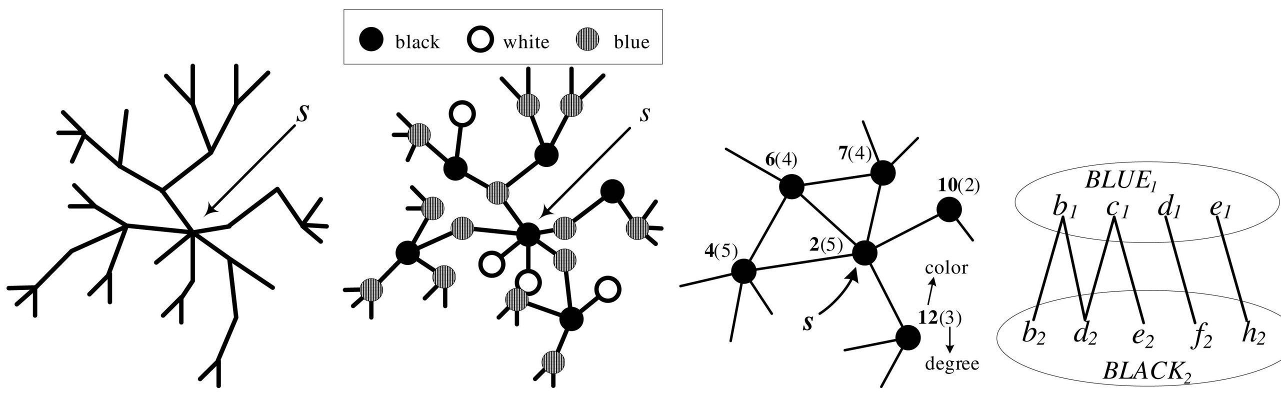
also connect $a_1, b_1, c_1, d_1, e_1, f_1$ to s since they are dominated by s. We keep working this way on layer 3. For simplicity, suppose that we have already found the black nodes at layer 3 and their corresponding blue nodes at layer 2. Fig. 3d shows that there are three blue nodes (a_2, c_2, i_2) at layer 2 connected to their black children at layer 3, g_2 remains in $WHITE$. So far, we have

$$\begin{cases} L_2 = \{a_2, b_2, c_2, d_2, e_2, f_2, g_2, h_2, i_2\}, \\ BLACK_2 = \{b_2, d_2, e_2, f_2, h_2\}, \\ BLUE_2 = \{a_2, c_2, i_2\}, \\ WHITE_2 = \{g_2\}. \end{cases}$$

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, each nodes of L_2 must be adjacent to at least one black node in $BLACK_1$ or $BLACK_2$. However, since $BLACK_1 = \emptyset$, each node of $BLUE_2$ or $WHITE_2$ must be adjacent to at least one node in $BLACK_2$, as shown in Fig. 4b. We keep doing this for all layers and the broadcast tree will be constructed in this way.

An example of data collection is described here. Since in Algorithm 2 we only need layer numbers to do scheduling, whether a node is black, blue, or white makes no difference and we only need to consider the topology in Fig. 3a as well as layer information in Fig. 3b. Nodes are named in the following way: Note that, except for s, we use the subscript to represent the layer. For example, at the first layer, we have $a_1, b_1, \ldots, f_1$. According to Algorithm 2, we randomly pick up a node in $L_1 = \{a_1, b_1, \ldots, f_1\}$, say a_1, to transmit in time slot 0. We also randomly pick up a node in $L_4, L_7, \ldots$ (not shown in Fig. 4a) and schedule them to transmit concurrently. Then we randomly pick up a node, say a_2, in L_2 to transmit in time slot 1. We also pick up a node in $L_5, L_8, \ldots$ (not shown in Fig. 4a) and schedule them to transmit concurrently. We just follow this method until all nodes have finished transmitting and forwarding.

Finally, we present an example of interleaved broadcast scheduling. First, we consider G^2 (the 2nd power graph of G, i.e., the graph with the same vertex set of G but having all two-hop paths in G as its edges) and give a 12-coloring to all black nodes by applying the smallest-degree-last ordering. Note that this 12-coloring is represented as numbers and has nothing to do with the black, blue, or white *colors*, which only refer to node sets. Fig. 4c shows (part of) G^2



(a) (b) (c) (d)

Fig. 4. (a) BFS tree. (b) Broadcast tree. (c) Smallest-degree-last coloring in G^2. (d) IMC.

TABLE 1
Interleaved Broadcast Schedule within Time Slots 0-36

slot(s)	0	1-12	13	14	15,16	17-20	21-24		
node(s)	s	-	b_1, d_1, e_1	c_1	-	-	-		
slot(s)	25-27	28	29	30	31	32,33	34	35	36
node(s)	-	b_2	-	d_2	e_2	-	f_2	-	h_2

with colors and degrees. Numbers in parentheses represent degrees and numbers without parentheses represent colors. We then run IMC layer by layer to obtain the function ℓ as follows: Starting from $X = BLUE_1$ $Y = BLACK_2$, X_0 is set to X and Z is set to Y. Now the minimal cover of Z will be X_0 itself (in this particular case only), so $X_1 = X_0$ and $W_0 = X_0 - X_1 = \emptyset$.

$\{b_2, e_2, f_2, h_2\}$ is the set of private neighbors of X_0 and Z is therefore reset to $Z \setminus \{b_2, e_2, f_2, h_2\} = \{d_2\}$. In the second iteration, the minimal covering of Z can either be $\{b_1\}$ or $\{c_1\}$, so we arbitrarily pick $X_2 = \{c_1\} \subset X_1$. Therefore, $W_1 = X_1 - X_2 = \{b_1, d_1, e_1\}$, $Z \leftarrow Z \setminus \{d_2, e_2\} = \emptyset$. So we stop and set $W_2 = X_2 = \{c_1\}$. Therefore, ℓ is defined on $BLUE_1$ as follows: $\ell(x) = 1\ \forall x \in W_1 = \{b_1, d_1, e_1\}$, and $\ell(c_1) = 2$. We keep applying IMC layer by layer this way (next one $X = BLUE_2$ and $Y = BLACK_3$). The details are omitted for simplicity.

Now we come to the scheduling part. Since white nodes are not scheduled, we can ignore them. According to Algorithm 3, s is scheduled to transmit in time slot 0 and all nodes in L_1 will receive the message collision free. Then from time slots 1 to 12, all nodes will be idling since there is no black node in L_1. The transmission between $BLUE_1$ and $BLACK_2$ will be scheduled from 13 to 16, and 17-24 are idling slots according to Algorithm 3. The actual transmission slots during 13-16 will be determined according to the function ℓ as described on lines 7-11 in Algorithm 3. Fig. 4d shows an example of how to determine those actual transmission slots. Therefore, b_1, d_1, e_1 are scheduled to transmit in slot 13, c_1 is scheduled to transmit in slot 14, and 15, 16 are both idling slots.

Now, we schedule the black nodes from slot 25 to slot 36 according to their color. As shown in Fig. 4b, $BLACK_2 = \{b_2, d_2, e_2, f_2, h_2\}$, where $color(b_2) = \#4$, $color(d_2) = \#6$, $color(e_2) = \#7$, $color(f_2) = \#10$, and $color(h_2) = \#12$. Therefore, b_2, d_2, e_2, f_2, h_2 are scheduled to transmit in time slots 28, 30, 31, 34, 36, respectively, and thus, 25, 26, 27, 29, 32, 33, 35 are idling slots. To summarize this example, the overall interleaved broadcast schedule from time slots 0 to 36 is shown in Table 1. In this table, "-" represents idling time slots.

7 Dealing with the Single Point of Failure Problem

Both the naïve and interleaved gossiping algorithms suffer from the single point of failure problem because both algorithms need to gather data to a single node s and then broadcast them later. This can be a serious problem in the systems where nodes have a nonnegligible failure probability. Now we consider the gossiping problem when a node $v \in V$ fails. We assume that the source is able to detect the failure of any black or blue node. This assumption is reasonable because each transmitter can use the watchdog mechanism [23] (i.e., each transmitter can overhear its receiver's relay message to detect any transmission failure). When a failure occurs, the transmitter can unicast it to the source node. In case of white nodes' failure, we can simply ignore it simply because they do not relay messages. Therefore, the failed node v be either blue or black, source or nonsource. The failure may occur in either phase II or phase III. We do not consider the case where v fails in Phase I because no scheduling is involved and we can reapply the interleaved gossiping algorithm once again in the new network. Note that the induced subgraph $G[V - \{v\}]$ may not be connected. However, if it is not connected, we can run all of our algorithms (to be presented in this section) in any connected component, which can be determined by performing a network-wide neighbor exploration. Note that this can be achieved by performing a distributed BFS search. For this reason, we may assume that $G[V - \{v\}]$ is connected. We distinguish between the following cases:

Case 1: v fails in Phase II, and $v \neq s$. We do the followings:

1. Construct the amended BFS tree $T_{BFS}(v)$ using Algorithm 6.
2. Apply Algorithm 2 on $T_{BFS}(v)$ for Phase II.
3. For Phase III, we do the followings. We construct the amended broadcast tree $T_{BR}(v)$ using Algorithm 7. Apply the interleaved broadcast algorithm (Algorithm 3) on $T_{BR}(v)$.

Algorithm 6. Construction of the amended BFS tree $T_{BFS}(v)$, $v \neq s$

/* Let $V_{BFS}(v)$ denote the vertex set of the subtree rooted at v in T_{BFS}. */

1: Add to $T_{BFS}(v)$ the edges that are in T_{BFS} but not incident to any nodes in $V_{BFS}(v)$.

2: For each node $w \in V_{BFS}(v), w \neq v$, perform a BFS traversal from $\{x \in V - V_{BFS}(v) |\ x$ is adjacent to some node(s) in $V_{BFS}(v) - v\}$ to $V_{BFS}(v) - \{v\}$ and add corresponding edges to $T_{BFS}(v)$ using the algorithm in [10].

3: Finally we update the layer of each node in $V_{BFS}(v) - \{v\}$ accordingly.

/*Note that the radius of $T_{BFS}(v)$ may be different than that of T_{BFS}. Also, $G[V_{BFS}(v) - v]$ may not even be connected and we may not be able to traverse it completely. In this case, we simply traverse the connected component containing s. */

Case 2: v fails in Phase III, and $v \neq s$. In this case, data aggregation has already terminated. All data have been transmitted to the source node s. Similar to Case 1, we do the followings:

1. Construct the amended BFS tree $T_{BFS}(v)$.
2. Construct the amended broadcast tree $T_{BR}(v)$ using Algorithm 7.
3. Apply Algorithm 3 on $T_{BR}(v)$.

Algorithm 7. Construction of the amended broadcast tree $T_{BR}(v)$, $v \neq s$

1: First we add to $T_{BR}(v)$ all edges in T_{BR} not incident to $V_{BFS}(v) - \{v\}$.

/* We construct the amended black set $BLACK(v)$ in $V_{BFS}(v) - \{v\}$ as described below. (Note that $V_{BFS}(v)$, defined above, stands for the vertex set of the subtree rooted at v in T_{BFS}.) */
Sort all nodes in $V_{BFS}(v) - \{v\}$ according to their distances to s in ascending order. Let $\{z_1, z_2, \ldots\}$ denote this ordering.
Initialize $BLACK(v) \leftarrow \emptyset$.
for $i = 1$ up to $|V_{BFS}(v) - \{v\}|$
  Add z_i to $BLACK(v)$ if z_i is not adjacent to any node in $BLACK(v) \cup (BLACK \backslash V_{BFS}(v))$.
end for

/* Now we choose blue nodes and use them to connect black nodes. Let $BLUE(v)$ denote the amended blue vertex set. */
Initialize $BLUE(v) \leftarrow \emptyset$
for each layer i in $T_{BFS}(v)$
  /* Connect black nodes to blue ones */
  for each black node $v \in BLACK(v) \cap L_{i+1}$
    Find its parent $p(v)$ in $T_{BFS}(v)$
    Add $p(v)$ to $BLUE(v)$
    Connect $p(v)$ to v in $T_{BR}(v)$
  end for
  /* Connect blue nodes to black ones */
  for each blue node $w \in BLUE(v) \cap L_i$
    Find w's dominator d_w
    Connect d_w to w in $T_{BR}(v)$
  end for
end for
/* Connect remaining white nodes */
for each remaining isolated white node u
  Find u's dominator d_u
  Connect u to d_u in $T_{BR}(v)$
end for

Case 3: s fails in Phase II or III. If s fails in either phase, we run the amended gossiping algorithm presented in Algorithm 8.

Algorithm 8. Amended gossiping algorithm when s fails

/* Amended data collection */
Construct an MIS in L_1, denoted by $MIS(L_1) = \{s_1, \ldots, s_{i^*}\}$
  /* Note that i^* is the size of $MIS(L_1)$. We know that $i^* \leq 5$ since L_1 is contained in a disk of radius 1, which can have at most 5 independent nodes. */
For each node in L_1, finds a dominator for it. Let $\mathcal{D}(w)$ denote the set of dominatees dominated by w.
Apply Algorithm 2 with an imaginary source s and T_{BFS} as if s still existed. /* s does not exist any more. Such schedule results in gathering data to $MIS(L_1)$. Each node $w \in MIS(L_1)$ has the messages from $\bigcup_{x \in \mathcal{D}(w)} V_{BFS}(x)$. */

for each s_i (where $1 \leq i \leq i^*$)
  Construct the multicast tree $T_{MC}^{s_i}$ with source node s_i and destination nodes $MIS(L_1) - \{s_i\}$ using Algorithm 9 (in any order).
  Schedule s_i to release a new message every 3 time slots by applying Algorithm 10 on $T_{MC}^{s_i}$ sequentially to multicast its messages from s_i to $MIS(L_1) - \{s_i\}$.
end for /* Note that each s_i has already gathered all messages from the whole network. */

/* Amended interleaved broadcast. */
for each s_i $(1 \leq i \leq i^*)$
  Schedule s_i to release all received messages in every 24 time slots by applying Algorithm 3 in parallel until done.
end for /* This behaves as if there were an imaginary source node s that releases a message every 24 time slots with its own idling slots removed. */

Algorithm 9. Construction of the multicast tree
Input: A graph $G = (V, E)$, a source node s', and a set of destination nodes $\{dst_1, dst_2, \ldots, dst_l\}$
Output: A tree T_{MC} rooted at s' with leaves $\{dst_1, dst_2, \ldots, dst_l\}$
Initialize T_{MC} to be an empty tree.
for i from 1 to l
  Find a shortest path from s to dst_i, denoted by $PATH_i = \{p_{i0}, p_{i1}, \ldots, p_{ik_i}\}$, where $p_{i0} = s'$ and $p_{ik_i} = dst_i$.
  if $i > 1$ **then**
    We find a node $p_{ih} \in PATH_i$ such that the following conditions hold: (1) p_{ih} is adjacent to some node $p_{jj'} \in PATH_j$ for some $j < i$. (2) $\forall h' > h$, $p_{jh'}$ is not adjacent to any node on $PATH_1, \ldots, PATH_{i-1}$.
    Update $PATH_i$ as follows: $PATH_i \leftarrow \{p_{j0}, p_{j1}, \ldots, p_{jj'}, p_{ih}, p_{i,h+1}, \ldots, p_{ik_i}\}$
  end if
end for
Add $PATH_i$ (both the nodes and links) to T_{MC}.

Algorithm 10. Multicast scheduling
Input: A graph $G = (V, E)$, a multicast tree T_{MC} with source node s' and destination nodes $\{dst_1, dst_2, \ldots, dst_l\}$, and starting time T_{start}.
For any node $x \in T_{MC}$ let $dist(x)$ denote the hop-distance between x and s' in T_{MC}.
for each node $x \in T_{MC}$
  Schedule x to transmit in time slot $T_{start} + dist(x)$.
end for

Theorem 8. *The amended gossiping algorithm of both cases 1 and 2 is a valid scheduling algorithm and its approximation ratio is at most 27.*

Theorem 9. *The amended gossiping algorithm of case 3 (Algorithm 8) is a valid scheduling algorithm and its approximation ratio is at most 29.*

8 Conclusion and Future Work

We studied the minimum latency gossiping problem in multihop radio networks. We first presented a naïve gossiping scheme that achieved approximation ratio 51, and then we improved it by introducing the interleaved gossiping algorithm that has ratio 27. In both algorithms, we took great advantages of UDG's geometrical properties. However, the UDG model does not completely reflect the reality, and the gossiping problem can be reinvestigated in more practical models. For example, we can consider the 2-disk model in which the transmission range is distinguished from the interference range (usually the interference range is some 2-5 times larger). In this model, we still assume that both transmission and interference ranges are disks and use two radii to represent them. We believe that our algorithms can be extended to this model by relaxing the approximation ratios as follows: We still find MIS and construct broadcast trees according to the transmission radius. However, in all interleaved transmissions, instead of separating each transmission by three hops, we need to enlarge this hop distance correspondingly according to the interference radius. This way we can still get a gossiping algorithm with a slightly large approximation ratio. This ratio may not be a constant anymore as compared to the interleaved gossiping algorithm with ratio 27. This ratio may depend on the interference radius, since obviously large interference radius may increase this ratio.

Another network model we can consider for a possible future extension is the signal-to-noise ratio (SNR) model. This model may not be an easy extension anymore, as it's not a deterministic model and randomness is involved. In this case, we believe that we can still take some advantage of its geometrical properties as we did in this work, but we need to add some randomness to it.

Finally, as another future work, we want to design the distributed version of this gossiping algorithm. This work is basically centralized, as we that assume the full topology is known and both the broadcast tree and scheduling algorithms need topology information. However, this assumption may be relaxed, since in all algorithms of this work, we only need topology information of neighboring layers. It means that we only need local topology information. For this reason, we believe that our gossiping algorithm can be designed in a distributed fashion.

APPENDIX

Here, we will provide the detailed proof of each theorem presented in the main text. We will first introduce three lemmas. Each lemma is a separate result, singled out instead of being placed within a theorem for shortening the lengthy proof. In the end, we will present a lemma regarding the approximation ratio of the state-of-the-art gossiping algorithm in the literature, which has significantly higher ratio compared with our interleaved gossiping algorithm. Therefore, our algorithm is a significant improvement.

First, we prove Theorems 1 and 2. In order to prove Theorem 1, we introduce Lemma 2 as follows:

Lemma 2. *Collision will not happen in the data collection schedule defined in Algorithm 2.*

Proof. If two nodes $u \in L_i$ and $v \in L_j$ satisfy $|i - j| > 2$, then there will be no collision between u, v for the following reason: If collision happens, then there exists a node w adjacent to both u and v, which contradicts to $|i - j| > 2$. Algorithm 2 is designed in the 3-interleaving fashion such that in any time slot, within any three consecutive layers, there can be at most one node transmitting. For this reason, collision will never happen. □

Proof of Theorem 1. According to Lemma 2, no collision will happen in Phase II. Now we prove that collision will not happen in Phase III (naïve broadcast) as follows: The naïve broadcast algorithm applied the EBS algorithm in [18], so a single execution of EBS will not cause any collision according to [18]. The naïve broadcast algorithm calls EBS as a subroutine every 48 time slots, which means that each call of EBS will be separated by at least three layers according to the properties of EBS. Therefore, collision will not happen. □

Proof of Theorem 2. The broadcast of first message will be completed in time slot $1 + 48(R - 1)$. Since the broadcast of all messages is interleaved and each message is released every 48 time slots, the following $N - 1$ messages will arrive every 48 time slots. Therefore, the last message will arrive after $48(N - 1)$ time slots and the latency of the naïve broadcast algorithm becomes $48(N + R - 2) + 1$. □

Now we introduce Lemmas 3 and 4, which will be used in the proof of Theorem 3.

Lemma 3. *The data collection schedule defined in Algorithm 2 has latency $3(N - 1)$.*

Proof. According to Algorithm 2, s receives a message every three time slots. Moreover, there are $N - 1$ nodes that have a message to be sent to s (excluding s itself). Therefore, after $3(N - 1)$, all messages will be received by s and the data collection schedule terminates. □

Lemma 4. *$N + R - 1$ is a lower bound for the gossiping problem.*

Proof. First, we claim that at least one node should transmit R times for the following reason: There are N messages. The broadcasting of each message requires at least R transmissions, so the total number of transmissions is at least $N*R$. Hence, at least one node transmits R times.

Each node has to receive $N - 1$ times. For a node that transmits at least R times, it needs at least $N + R - 1$ time slots. Therefore, we found $N + R - 1$ is a lower bound for the gossiping problem. □

Proof of Theorem 3. According to Lemma 3 and Theorem 2, we get a combined latency of $3(N - 1) + 48(N + R - 2) + 1 < 51 * (N + R - 1)$. According to Lemma 4, $N + R - 1$ is a lower bound. Therefore, the approximation ratio is clearly at most 51. □

Proof of Theorem 4. The interleaved gossiping algorithm has three phases in which phase I is preprocessing and does not involve actual scheduling. According to Lemma 2, collision will not happen in phase II. Therefore, we only have to prove that collision will not happen in phase III, which is Algorithm 3. We look at the following cases:

1. Black nodes do not cause collision to any blue nodes. This is because the first 12 time slots within a 24-slot round are reserved for black nodes only. No blue nodes are scheduled to transmit during these 12 slots and no collision will happen.

2. Black nodes do not cause collision to each other. This can be proved according to the 12-coloring property. A black node is scheduled to transmit according to its color. Therefore, any two concurrently transmitting black nodes must have the same color, and according to the geometrical property of the 12-coloring, the distance between any such pair of nodes must be at least 2. If collision happens, concurrently transmitting black nodes must have a common neighbor, which is a contradiction. More details regarding the 12-coloring can be found in [18].
3. Blue nodes do not cause collision to any black nodes. This is because the last 12 time slots within a 24-slot round are reserved for blue nodes only. No black nodes are scheduled to transmit during these 12 slots and no collision will happen.
4. Blue nodes do not cause collision to each other. Consider two blue nodes $u \in BLUE_i$ and $v \in BLUE_j$. If $|i-j| > 2$, there will be no collision because u, v cannot have a common neighbor. If $|i-j| \leq 2$ and $i \neq j$, then there will be no collision because transmissions are all interleaved for different layers and u, v will be scheduled to transmit in different time slots according to Algorithm 3. Finally, if $i = j$, there will be no collision either because blue nodes at the same layer are scheduled according to IMC (Algorithm 4). The detailed proof of this part (that IMC does not cause any collision) can be found in [18]. □

Proof of Theorem 5. The broadcast of first message will be completed in time slot $1 + 24(R-1)$. Since the broadcast of all messages is interleaved, and each message is released every 24 time slots, the following $N-1$ messages will arrive every 24 time slots. Therefore, the last message will arrive after $24(N-1)$ time slots and the latency of Algorithm 3 becomes $24(N+R-2)+1$. □

Proof of Theorem 6. According to Lemma 3 and Theorem 5, we get a combined latency of $3(N-1) + 24(N+R-2) + 1 < 27*(N+R-1)$. According to Lemma 4, $N+R-1$ is a lower bound. Therefore, the approximation ratio is clearly at most 27. □

Proof of Theorem 8. The data collection part is based on $T_{BFS}(v)$ in case 1. Therefore, according to Lemma 2, there will be no collision and the latency is $3(N-1)$. The data collection part in case 2 is Algorithm 2, which also has latency $3(N-1)$. As for the broadcast schedule in both cases, collision will not happen for the same reason as Theorem 4. However, the latency will be slightly increased as the depth of $T_{BR}(v)$ may exceed the radius of $T_{BFS}(v)$ by 1 for the boundary between the original black nodes and newly added black nodes (i.e., between $BLACK \backslash V_{BFS}$ and $BLACK(v)$). On the boundary, we do not have the property that each blue node has a black parent at the same layer or the layer right next to it above. Instead, the blue nodes on the boundary may have a black parent at the layer (w.r.t. to $T_{BFS}(v)$) below, and the overall radius may increase by 1. Therefore, the latency is at most $24(N+R(v)-1)+1$, where $R(v)$ is the radius of $T_{BFS}(v)$. The latency of the amended gossiping algorithm is therefore bounded by $27(N+R)$ and the approximation ratio is at most 27. □

Proof of Theorem 9. In Algorithm 8, line 3 takes $3(N-1)$ time slots since it is applied as if s were still there. The multicast part (lines 4-7) takes at most $5(R(s)+4)+N$. The amended interleaved broadcast (lines 8-10) takes $24(N+R(v)-3)+1$ (similar to the proof of Theorem 5). Therefore, the overall latency is bounded by $29(N+R(v)-1)-25$ and the approximation ratio is at most 29. □

Finally, we present the following lemma about the approximation ratio of the state-of-the-art gossiping algorithm in the literature, which is a combined algorithm of Algorithms 1 and 2 of this work as well as the broadcast algorithm proposed in [14]:

Lemma 5. *The combination of Algorithms 1 and 2 of this work as well as the broadcast algorithm proposed in [14] is a gossiping algorithm with approximation ratio 1,947. In other words, if we replace the EBS algorithm by [14] in the naïve gossiping algorithm, the resulted algorithm is a gossiping algorithm with approximation ratio 1,947.*

Proof. First, by straightforward calculation, we know that the latency of the single-source broadcast algorithm in [14] is $648R$, where R is the radius of the network. According to Lemma 3, the data aggregation latency is $3(N-1)$, where N is the number of nodes. Since [14] should be executed repeatedly such that each execution is separated by at least three layers, [14] should be executed every $3 \times 648 = 1{,}944$ time slots, and it should be executed N times. Therefore, the broadcast latency of these N packets is exactly $1{,}944(N+R-2)+1$ and the overall gossiping latency is $3(N-1) + 1{,}944(N+R-2)+1 = 1{,}947(N+R-1)$. According to Lemma 4, its approximation ratio is at most 1,947. It is trivial to verify that this bound is tight, so we omit this verification. □

Proof of Theorem 7. In either algorithm, in Phase II (the data aggregation part), a node needs to transmit its own packet as well as forwarding a packet for each of its descendants in the BFS tree. Therefore, a node needs to transmit at most N times. In Phase III, either the naïve or interleaved broadcast, a node needs to forward at most N packets for its parents, so a node needs to transmit at most N times. Therefore, a node needs to transmit at most $2N$ messages. □

ACKNOWLEDGMENTS

Scott C.-H. Huang was supported in part by the Research Grants Council of Hong Kong under the project GRF CityU 113807. P.-J. Wan and H. Du were supported in part by the US National Science Foundation under the project CNS-0831831.

REFERENCES

[1] R. Bar-Yehuda, A. Israeli, and A. Itai, "Multiple Communication in Multihop Radio Networks," *SIAM J. Computing*, vol. 22, no. 4, pp. 875-887, 1993.

[2] J.C. Bermond, L. Gargano, A.A. Rescigno, and U. Vacarro, "Fast Gossiping by Short Messages," *Proc. 22nd Int'l Colloquium Automata, Languages and Programming (ICALP '95),* pp. 135-146, 1995.
[3] J.-C. Bermond, L. Gargano, A.A. Rescigno, and U. Vaccaro, "Fast Gossiping by Short Messages," *SIAM J. Computing,* vol. 27, no. 4, pp. 917-941, 1998.
[4] B.S. Chlebus, L. Gasieniec, A. Lingas, and A.T. Pagourtzis, "Oblivious Gossiping in Ad-Hoc Radio Networks," *Proc. Fifth Int'l Workshop Discrete Algorithms and Methods for Mobile Computing and Comm. (DIALM '01),* pp. 44-51, 2001.
[5] M. Christersson, L. Gasieniec, and A. Lingas, "Gossiping with Bounded Size Messages in Ad Hoc Radio Networks," *Proc. 29th Int'l Colloquium Automata, Languages and Programming (ICALP '02),* pp. 377-389, 2002.
[6] M. Chrobak, L. Gasiniec, and W. Rytter, "Fast Broadcasting and Gossiping in Radio Networks," *Proc. 41st IEEE Symp. Foundation of Computer Science (FOCS '00),* pp. 575-581, 2000.
[7] M. Chrobak, L. Gasiniec, and W. Rytter, "A Randomized Algorithm for Gossiping in Radio Networks," *Proc. Seventh Ann. Int'l Computing and Combinatorics Conf. (COCOON '01),* pp. 483-492, 2001.
[8] M. Chrobak, L. Gasiniec, and W. Rytter, "Ast Broadcasting and Gossiping in Radio Networks," *J. Algorithms,* vol. 43, no. 2, pp. 177-189, 2002.
[9] A.E.F. Clementi, A. Monti, and R. Silvestri, "Selective Families, Superimposed Codes, and Broadcasting in Unknown Radio Networks," *Proc. 12th ACM-SIAM Symp. Discrete Algorithms (SODA '01),* pp. 709-718, 2001.
[10] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms,* second ed., chapter 25. McGraw-Hill, 2001.
[11] M. Flamini and S. Perennes, "Lower Bounds on the Broadcasting and Gossiping Time of Restricted Protocols," technical report, Inst. Nat'l de Recherche en Informatique et en Automatique, 1999.
[12] C. Florens, M. Franceschetti, and R.J. McEliece, "Lower Bounds on Data Collection Time in Sensory Networks," *IEEE J. Selected Areas Comm.,* vol. 22, no. 6, pp. 1110-1120, Aug. 2004.
[13] R.W. Floyd, "Algorithm 97: Shortest Path," *Comm. ACM,* vol. 5, no. 6, p. 345, 1962.
[14] R. Gandhi, S. Parthasarathy, and A. Mishra, "Minimizing Broadcast Latency and Redundancy in Ad Hoc Networks," *Proc. ACM MobiHoc,* pp. 222-232, 2003.
[15] L. Gasieniec and A. Lingas, "On Adaptive Deterministic Gossiping in Ad Hoc Radio Networks," *Information Processing Letters,* vol. 83, pp. 89-93, 2002.
[16] L. Gasieniec and I. Potapov, "Gossiping with Unit Messages in Known Radio Networks," *Proc. IFIP 17th World Computer Congress—TC1 Stream/Second IFIP Int'l Conf. Theoretical Computer Science (TCS '02),* pp. 193-205, 2002.
[17] L. Gasieniec, I. Potapov, and Q. Xin, "Time Efficient Gossiping in Known Radio Networks," *Proc. 11th Colloquium Structural Information and Comm. Complexity (SIROCCO '04),* pp. 173-184, 2004.
[18] S.C.-H. Huang, P.-J. Wan, X. Jia, H. Du, and W. Shang, "Minimum-Latency Broadcast Scheduling in Wireless Ad Hoc Networks," *Proc. IEEE INFOCOM,* pp. 733-739, 2007.
[19] P. Indyk, "Explicit Constructions of Selectors and Related Combinatorial Structures, with Applications," *Proc. 13th ACM-SIAM Symp. Discrete Algorithms (SODA '02),* pp. 697-704, 2002.
[20] X. Jia, "A Distributed Algorithm of Delay Bounded Multicast Routing for Multimedia Applications in Wide Area Networks," *IEEE/ACM Trans. Networking,* vol. 6, no. 6, pp. 828-837, Dec. 1998.
[21] D. Li, X. Jia, and H. Liu, "Energy Efficient Broadcast Routing in Static Ad Hoc Wireless Networks," *IEEE Trans. Mobile Computing,* vol. 3, no. 2, pp. 144-151, Apr.-June 2004.
[22] F. Manne and Q. Xin, "Optimal Gossiping with Unit Size Messages in Known Topology Radio Networks," *Proc. Third Workshop Combinatorial and Algorithmic Aspects of Networking (CAAN '06),* pp. 125-134, 2006.
[23] S. Marti, T.J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," *Proc. ACM MobiCom,* pp. 255-265, 2000.
[24] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," *Proc. ACM/IEEE MobiCom,* pp. 151-162, 1999.
[25] K. Ravishankar and S. Singh, "Asymptotically Optimal Gossiping in Radio Networks," *Discrete Applied Math.,* vol. 61, pp. 61-85, 1995.
[26] K. Ravishankar and S. Singh, "Gossiping on a Ring with Radios," *Parallel Processing Letters,* vol. 6, no. 1, pp. 115-126, 1996.
[27] Y. Xu, "An $O(n^{1.5})$ Deterministic Gossiping Algorithm for Radio Networks," *Algorithmica,* vol. 36, no. 1.

Scott C.-H. Huang received the BS degree from National Taiwan University and the PhD degree from the University of Minnesota, Twin Cities. He is currently a lecturer in the Computer Science Department, City University of Hong Kong. His research interests include wireless ad hoc/sensor network, network security, communication theory, as well as combinatorial optimization.

Peng-Jun Wan received the BS degree from Tsinghua University, the MS degree from the Chinese Academy of Sciences, and the PhD degree from the University of Minnesota. He is currently a professor of computer science in the Department of Computer Science, Illinois Institute of Technology, Chicago. His research interests include wireless networks and algorithm design and analysis.

Hongwei Du received the BSc degree from the Central China Normal University in 2003 and the PhD degree from the City University of Hong Kong in 2008. He is currently a senior research associate in the Department of Computer Science at the Illinois Institute of Technology and also a postdoctor in the Academy of Mathematics and System Sciences, Chinese Academy of Sciences. His research interests include wireless networks, mobile computing, and algorithm design.

E.-K. Park received the PhD degree in computer science from the Northwestern University, Evanston, Illinois. He is a professor of computer science at the University of Missouri at Kansas City. His research interests include data mining, bioinformatics, information and knowledge management, computer communications and networks, optical networks, distributed systems, and object-oriented methodology. Currently, he is on an assignment serving as the program director of the Division of Computing and Communications Foundations at the US National Science Foundation.