Gossiping in Multihop Radio Networks

Neeraj Gupta
Motorola India Electronics Ltd
Bangalore 560 042
e-mail: `neeraj@miel.mot.com`

D. Manjunath
Dept of Elecl Engg, IIT, Bombay
Powai Mumbai 400 076
e-mail: `dmanju@ee.iitb.ernet.in`

ABSTRACT

Gossiping is when each node in the network has some information that it needs to communicate to every other node in the network. Updating of routing information in communication networks and exchange of data in control networks that use a form of distributed control, where status data from every node needs to be made available at every other node in the network, are some of the practical applications of gossiping. A gossip scheduling algorithm is used to schedule the transmissions times of the nodes and the message to be transmitted by them at these times. A good gossip scheduling algorithm should be computationally feasible **and** provide a schedule that achieves gossip quickly. In this paper we investigate and propose gossip algorithms for multihop radio networks.

Assuming slotted operation of a single frequency multihop radio network with unit length messages and a node transmitting at most one message in each slot, we propose three gossip scheduling algorithms for general topologies - Collision-Free, Centralised-Spanning-Tree and Gather-Scatter gossip scheduling algorithms. Of these three our experimental results indicate that the Gather-Scatter algorithm is possibly among the best gossip scheduling algorithm for multihop radio networks. For all these three algorithms we have measured the performance (the number of slots to achieve gossip) using an experimental model that is a realistic representation of real-life radio networks. We also overview our results for gossip schedule lengths in complete graph, star, ring and bus topologies.

I. Introduction and Problem Statement

Gossiping is when each node in the network has some information that it needs to communicate to every other node in the network. In this paper we consider gossiping in a multihop packet radio network with N arbitrarily connected nodes. It is assumed that time on the network is slotted, the network is synchronised and the slot length is exactly equal to the message transmission time plus the maximum propagation delay between any two neighbours in the network and is the same for all nodes. All nodes begin a transmission in the beginning of a slot and during a slot a transmitter can communicate with a set of receivers, exactly one message out of the several possible messages available at the node at that time. (The messages that may be available are those that have come from other nodes in the network and need to be communicated to those nodes that have not already received it.) Our problem is to develop gossip scheduling algorithms that deliver the gossip schedule in polynomial time and provide as small schedule length (time taken to achieve gossip) as possible.

This work was done while the authors were with the Deptt of Elecl Engg, IIT,Kanpur India

A. Solution Approach

It is known that for a multihop radio network consisting of N nodes, optimal scheduling of a k-to-all broadcast is NP-hard for any $k \geq 1$ [1]. Since gossiping is a special case of k-to-all broadcast where $k = N$, finding an optimal gossip schedule in multihop radio networks is NP-hard. Therefore we look for approximate algorithms that deliver a "good schedule" in polynomial time.

In developing the gossip scheduling algorithms we first realise that a multihop radio network uses a broadcast channel on which collisions occur due to simultaneous transmissions that can render into noise the information to be received by all or some of the intended recipients. This means that for a transmission to be correctly received by a receiver, the receiver and its neighbours should not be transmitting at that time. To achieve gossip quickly, we need to obtain a good time and spatial reuse of the single channel available to the network. Efficient reuse is achieved by dividing the radio network into several areas such that the needs of transmissions by nodes in a given area are not affected by the needs of the transmissions by nodes in other areas. This may result in collisions at some nodes at some times. These are allowed because they may help in reducing the schedule length. Since we are also concerned about the running time complexity of the scheduling algorithm we limit the number of nodes and links considered in the division process.

In the schedules that we present in this paper, gossip is achieved by scheduling the transmissions in a sequence of slots called the gossip frame. (A good schedule will minimise the number of slots in this frame.) The connectivity of the radio network is represented by a connected graph $G = (V, E)$, V is the set of nodes in the network and E is the set of edges of the graph representing the connectivity of the nodes, i.e., an edge $(u, v) \in E$ means that the nodes u and v can hear each others' transmissions. It is well known that a scheduling algorithm is essentially a graph colouring algorithm with some constraints. Thus a gossip scheduling algorithm assigns a colour c to nodes in a set $S(c) \subset V$ ($S(c)$ depends on the gossip scheduling algorithm). The sets $S(c)$ for $c = 1, 2, \ldots$ form a partition of V. The slots correspond to the colours and nodes with the same colour transmit in the same slot - the slot corresponding to that colour. Thus the colouring process gives us a "primary frame" consisting of a number of time slots, each corresponding to a colour. Repetition of this primary frame with improvements during the repetition gives us the gossip frame. In addition to deciding when a node should transmit, we also need to specify what should be transmitted in the assigned slot. To decide what is transmitted by the nodes in their assigned slots we can use any of the following rules to select one of the several messages they may possibly have at the beginning of the slot

FCFS select an untransmitted message that arrived earliest

LCFS select an untransmitted message that arrived latest
IO (In Order) select an untransmitted message with the smallest source node number
RS (Random Order) select an arbitrary untransmitted message.
The rules can either be strictly followed or the nodes may give first priority to their own messages and then adhere to these rules giving us seven selection methods that the nodes may use to select a message to transmit in their assigned slot. However, some times, like in the Gather-Scatter algorithm to be described later in the paper, a different specification for the transmission sequence can reduce the schedule length.

B. Performance Analysis Method

We assume a noiseless, immobile radio network in which all nodes have the same transmission radius. Thus we may represent any radio network by the 3-tuple (N, R, P) where N is the number of nodes, R is the transmission range of each node and $P = \{(x_i, y_i) : \quad 1 \le i \le N\}$ is the set of geographical locations of each of the nodes. This set is generated randomly by using a uniform distribution for the x and y coordinates of each of the nodes. From this random network, the graph, $G = (V, E)$, corresponding to this network is obtained by applying the following rule to obtain the edge set E: "An edge (u, v) belongs to the edge set E if and only if the Euclidean distance between (x_u, y_u) and (x_v, y_v) is less than or equal to R." The performance results for the algorithms are the outputs obtained by giving as inputs to the algorithms, random graphs generated as above.

In each of the next three sections we describe a gossip scheduling algorithm for arbitrary graphs. In section V we summarise some of our results on the schedule length for gossiping on regular topologies. We conclude the paper in section VI. The pseudocodes for each of the algorithms are given and the properties of the algorithms are also listed.

II. Collision Free Gossip Scheduling Algorithm

The fundamental idea in this algorithm is to develop the primary frame required by distance-2 colouring of the graph G. The gossip frame will then be just a repetition of this primary frame with some of the colours, called the "transient colours", making their presence only once in the entire gossip frame.

A. Algorithm Description

Dis_2_colour is used for distance-2 colouring of the graph G. In a distance-2 coloured graph two nodes that are one or two hops away from each other will not have the same colour. The algorithm is the same as that reported in [3]. The main idea in this procedure is to colour the vertices in a certain order, based on the vertex degrees. We first assign a unique label to each $v \in V$ using Procedure label as follows: the vertex v labelled in step l is the one with the least number of neighbours in the subgraph formed by the unlabelled vertices. After labelling, we colour the vertices in decreasing order of their labels by assigning the lowest numbered colour that can be assigned without causing a conflict with previously coloured vertices. For any vertex, all vertices having a distance less than or equal to two are said to be conflicting with it. Distance-2 colouring in this manner guarantees that the transmissions from a node in its assigned slot would reach collision free to all its neighbours. If a node has only one neighbour, it needs to be allowed to transmit only once when it will transmit its own message. If a colour c has only such nodes, then it is a transient colour and needs to be used only once in the gossip frame.

The largest colour that is assigned to a vertex u is at most one greater than the maximum size of the set Y, constructed in the Dis_2_colour procedure in each iteration. (Y is the set of colours assigned to the one hop and two hop neighbours of the node being labelled. See pseudocode at the end of the section.) For a graph with maximum degree ρ, there can be at most ρ adjacent vertices and $(\rho(\rho-1)$ distance-2 vertices. Assuming in the worst case that all of these vertices are coloured differently, number of colours used in the worst case is no more than $(1+\rho)+\rho(\rho-1) = \rho^2+1$. An improved bound is provided later.

Dis_2_colour tells the nodes when to transmit. To decide what to transmit in a given slot, any of the seven rules mentioned in Section 1 may be followed.

Procedure label(G)

```
/*
Input is an undirected graph G = (V, E)
output is a label to each vertex in G
*/
    begin
        set count ← 1
        while (there is any unlabelled vertex) do
            u ← min degree vertex in subgraph
                induced by unlabelled vertices
            label(u) ← count
            count ← count + 1
        endwhile
    end
```

Procedure Dis_2_colour(G)

```
/*
Inputs an arbitrary connected undirected graph G = (V, E)
Calls Procedure label to label vertices
Outputs a colouring scheme c : V → {1, 2, 3 ...}
and f(c) for every c. (If f(c) = 0 c is transient)
*/
    begin
        label(G)
        for (each u ∈ V) set c(u) ← 0. endfor
        for (every assignable colour c) set f(c) ← 0. endfor
        for (j = N to 1)
            pick vertex u with label j
            set Y ← φ
            for (every 1-hop or 2-hop neighbour v of u)
                Y ← Y ⋃ {c(v)}
            endfor
            c(u) ← (least natural num y such that y ∉ Y)
            if (degree(u) > 1) set f(c(u)) ← 1 endif
        endfor
    end
```

B. Properties

The following properties are stated without proof. For details of the proofs see [2].

1. The execution time complexity of procedure Dis_2_colour is $O(N^2 + N\rho^2)$.
2. Dis_2_colour does not assign more than $\rho(6t-1) + (6t-2)(\rho - 6t + 1) + 1$ colours to any graph G of thickness t.
3. The gossip frame length is upper bounded by $ND(\rho(6t-1) + (6t-2)(\rho - 6t + 1) + 1)$ where D is the diameter of the graph G.

III. Centralised Spanning Tree Algorithm

The basic idea in this algorithm is to construct a spanning tree T on the graph G which assigns to each node in the network a parent that will convey all messages in the network to that node (and its other children). The scheduling also ensures that when the children transmit the parent does not encounter any collision in this slot. This is to allow the children to transfer the messsage from the nodes below them in the tree towards the parent. Thus the colouring scheme, procedure Cent_Tree, works on T and assigns the colours such that no node's transmission will collide with its parent or children in the tree. As in the Dis_2_colour algorithm, the set of assigned colours gives a primary frame which is repeated in time according to certain rules described later to yield the gossip frame.

A. Algorithm Description

Cent_Tree constructs T from G to establish the parent-child relationships, assigns colours to the nodes in G and identifies the transient colours. It starts by choosing an arbitrary node in cthe graph as the root node, constructs a spanning tree, T, on this graph and assigns the colours to the nodes while progressively constructing T. After running the algorithm, each node in the graph has, associated with it, a data structure that contains the colour of the node and its parent. Any node in the network could be chosen as the root node without affecting the correctness.

Cent_Tree begins with an empty T. In the first iteration the root node r is picked and assigned colour 1. This is then added to the tree with the subsequent adoption of all its neighbours as its children. In each subsequent iteration an uncoloured node v is chosen, a colour assigned according to the colouring rules described below, added to T and finally it is assigned as the parent of all its "orphan neighbours". The algorithm stops when all the nodes have been assigned a colour. For each iteration, the node selected to be coloured should be the first uncolured node to have a parent in terms of the iteration number in which the assignment was done. Since, the algorithm requires each node to be coloured the whole process can be completed in N iterations - N denoting the number of nodes in the radio network.

A node v is coloured with the lowest natural number, c, that satisfies all three conditions below. The first two conditions ensure that each parent will be successfully able to get its children transmissions while the third one ensures that every transmission of a parent is received collision free by its children.

c1a: c is not the colour of v's parent $p(v)$.
c1b: c is not the colour of any of neighbour of $p(v)$.
c1c: c is not the colour of any of v's neighbour parent.

A colour c is transient if all the nodes having this colour have no children. As in the previous algorithm, to decide what to transmit in a given slot, any of the seven rules mentioned in Section 1 may be followed.

Procedure Cent_Tree(G)

```
/*
Inputs an arbitrary connected undirected graph G = (V, E)
Outputs a colouring scheme c : V → {1, 2, 3 ...}
and f(c) for every c. (If f(c) = 0 c is transient)
*/
 begin
  for (all v ∈ V) set c(v) ← 0 endfor
  for (every assignable colour c) set f(c) ← 0 endfor
  select a root node r
  set T ← φ
  set c(r) ← 1; set f(1) ← 1; T ← T ⋃{r}
  for (each neighbour, v, of r) set p(v) ← r endfor
  while (there is an uncoloured v ∈ V) do
     select v, first uncoloured node assigned a parent
     T ← T ⋃{v}
     select lowest natural number i such that
        c.1: i is not assigned to v's parent
        c.2: i is not assigned to v's parent's neighbour
        c.3: i is not assigned to parent of v's neighbour
     set c(v) ← i.
     for (each neighbour u of v without a parent)
           set p(u) ← v endfor
     if (at least one such neighbour) set f(i) ← 1 endif
  endwhile
 end
```

B. Properties

The following properties are stated without proof. For details of the proofs see [2].

1. The execution time complexity of procedure Cent_Tree is $O(\rho N)$.
2. For any graph $G = (V, E)$ with a maximum degree ρ the total number of distinct colours assigned by the procedure Cent_Tree is upper bounded by 2ρ.
3. The gossip frame length in the centralised spanning tree algorithm is upper bounded by $2ND\rho$, where D is the diameter of the graph.

IV. Gather-Scatter Algorithm

In this algorithm we first gather the messages from all the nodes in the network at a root node through an accumulation algorithm and follow with a dissemination algorithm to scatter these messages to all the nodes in the network. The root node can be chosen arbitrarily but the performance would depend on this choice and we expect that the best performance will be achieved if the central node in the graph is chosen as the root node.

A. Accumulation Algorithm

First choose the central node in G to be the root node. Assign to each node in G (excluding the root node) a colour c and a number f. Let the set $L = \{1, 2, \ldots, l\}$ be the primary frame, where every $c \in L$ is assigned to vertices in set $S(c) \subset V$. Every $c \in L$ is also assigned a counter $C(c)$ that depends on the values of f assigned to the nodes of the set $S(c)$. (The assignment of f to the nodes will be discussed later.) Assignment of $C(c)$ gives rise to a set $I = \{C(1), C(2), \ldots, C(l)\}$. Based on these two sets L and I, an accumulation frame is built as follows. Repeat the primary frame L with each colour c making its appearance in the accumulation frame exactly $C(c)$ times. Based on the above approach, it can be seen that length of the accumulation frame is exactly $\sum_{i=1}^{l} C(i)$ slots, l denoting the cardinality of the set L. This bound can be achieved by having the transmissions strictly adhere to the **relay rule** "in each times lot c, all nodes scheduled to transmit in c, should in the first chance transmit its message and then should transmit only those messages that have come from its children. Each node should transmit a message only once in the accumulation frame."

As can be seen the basic tasks performed by this algorithm are the assignment of colours to the nodes (performed by Acc_colour) and evaluate the counter corresponding to each colour $c \in L$ (performed by Counter_assignment). Acc_colour constructs a spanning tree T directed from the root node and establishes the parent-child relationship between the nodes. It also assigns colours to guarantee reception by a parent, the messages transmitted by its children. Acc_colour starts with T being empty. In iteration 1 the root node r is added to T. This root node now becomes the parent of all its neighbours in G and these neighbours are assigned r as their parent. In subsequent iterations, Acc_colour colours that node $v \in V$ which has not been added to T **and** was assigned a parent earliest among those not added to T, adds it to the tree T and assigns v as the parent of all its "orphan neighbours". The colouring rule is to assign the lowest natural number, c, satisfying the following rules

c2a c is not assigned to parent, $p(v)$ of v.

c2b c is not assigned to any neighbour of $p(v)$.

Counter_assignment assigns to each colour c of the primary frame a counter $C(c)$. Then the accumulation frame will be a repetition of the primary frame with each colour c appearing in the accumulation frame exactly $C(c)$ times. The counter assignment is done as follows: for each node $v \in V$, let $f(v)$ be the number of "future generation nodes" in its family consisting of v, its children, their children and so on. To calculate $f(v)$ for any v, the procedure visits every v', a child of v, sums these $f(v')$s and finally adds one corresponding to itself. Counter_assignment finds the entry $f(v)$ for all $v \in V$. The process of selecting the nodes to assign the $f(v)$ by Counter_assignment should be strictly in the reverse of the order in which they were assigned a parent in the Acc_colour procedure which is also the order in which they were entered into the tree T. After this assignment $C(c)$ is obtained as the maximum of the $f(v)$'s of all the nodes $v \in S(c)$.

It is easily seen that each node v has to transmit exactly $f(v)$ messages for accumulation of all messages at r. The **relay rule** guarantees that each node gets a new message to transmit after every primary frame which it sends on the path towards the root node in a subsequent frame. This means that each node gets a chance to transmit $f(v)$ times in an accumulation frame and this chance can come in consecutive frames. The number of slots used in the accumulation frame is thus $\sum_{i=1}^{l} C(i)$ where l is the number of colours assigned by the procedure Acc_colour.

Procedure Acc_colour(G)

```
/*
Inputs an arbitrary connected undirected graph G = (V, E)
Outputs a colour assignment to the nodes of G and an array
O where O[i] is the node coloured in iteration (i + 1)
*/
  begin
    for(all v ∈ V) set c(v) ← 0. endfor
    set T ← φ; set count ← 1
    T ← T ⋃{r}
    assign r as parent of all its neighbours
    while (there is an uncoloured vertex in V − {r}) do
      select u, first node in V − {r} to be assigned a parent
      T ← T ⋃{u}
      O[count] ← u
      colour u with lowest natural number i satisfying
        c2a: i is not assigned to u's parent
        c2b: i is not assigned to u's parent's neighbour
      for all of u's orphan neighbours assign u as the parent
      count ← count + 1
    endwhile
  end
```

Procedure Counter_Assignment(G)

```
/*
Inputs an arbitrary connected coloured graph G = (V, E) and
the array O constructed in procedure Acc_colour
Outputs a counter assignment to every colour assigned by Acc_colour
*/
  begin
    for (all v ∈ V − {r}) set f(v) ← 1. endfor
    for (each colour c assigned to nodes in V) set C(c) ← 0 endfor
    set count ← N − 1
    while (count > 0) do
      set u ← O[count]
      for (each neighbour n, a child of u) f(u) ← f(u) + f(n) endfor
      let c be the colour of u
      if (C(c) < f(u)) then
        set C(c) ← f(u) endif
      count ← count − 1
    endwhile
  end
```

B. Properties

The following properties are stated without proof. For details of the proofs see [2].

1. The execution time complexity of procedures Acc_colour and Counter_assignment is $O(\rho N)$.
2. The set constituting the colours present in the primary frame developed by the algorithm can have a maximum cardinality of $\rho + 1$ in a graph having a maximum degree of ρ.
3. If d is the degree of the root node, then the length of the accumulation frame used for accumulating the network information at the root node is upper bounded by $\lceil \frac{N-1}{d} \rceil (\rho + 1)$ time slots.

C. Dissemination Algorithm

Having gathered all of the network information at the root node, Dissem_colour is used to get the schedule for the dissemination part of the gossip frame. In this procedure we use the spanning tree T constructed in the Acc_colour and the parent child relationship developed therein. Starting from the root r Dissem_colour begins by assigning to it the first colour from the set $\{1, 2, 3, \dots, \}$ i.e., colour 1. Subsequently, it chooses one of the uncoloured nodes available and colours it according to the colouring rules **c3** described below. Like in procedure Acc_colour, in each colouring iteration this procedure chooses an uncoloured node which is the first uncoloured node to have been assigned a parent until this colouring iteration. In this colour assignment nodes having no children are not assigned any colour at all since they would have nothing useful to relay to other nodes in the network. The algorithm will stop when all nodes having at least a single child in the tree T would have been coloured. This algorithm also gives a primary frame consisting of the colours assigned to the nodes which is repeated to yield the dissemination frame that can completes the gossiping. Note that in contrast to the centralized spanning tree algorithm in which a node need not transmit a message if it has transmitted it once, in this algorithm it is required that irrespective of whether a node has transmitted a message in the accumulation phase or not it needs to transmit it again in the dissemination phase.

The colouring rules for Dissem_colour is that in each iteration, the least natural number, c satisfying the following conditions should be assigned to the chosen node v
c3a c is not the colour of v's parent.
c3b c is not the colour of any of v's neighbour parent.

Procedure Dissem_colour(G)

```
/*
Inputs an arbitrary connected undirected graph G = (V, E)
Outputs assignment of colours c(v) to nodes v ∈ V.
*/
  begin
    for (all v ∈ V) set c(v) ← 0. endfor
    set c(r) ← 1
    while (there is an uncoloured vertex with ≥ 1 child in T) do
      let v be first uncoloured node assigned a parent
      if (v has ≥ 1 child in G) do
        colour v with the lowest natural number c satisfying
              c3a: c is not assigned to v's parent
              c3b: c is not assigned to v's neighbour's parent
      endif
    endwhile
  end
```

D. Properties

The following properties are stated without proof. For details of the proofs see [2].
1. The execution time complexity of procedure Dissem_colour is $O(\rho N)$.
2. The total number of distinct colours assigned by the Dissem_colour algorithm is upper bounded by $\rho + 2$.
3. If r be the maximum distance from root node to any other node in the network, then the dissemination frame length is upper bounded by $(N + r - 1)(\rho + 2)$.

V. Gossiping Regular Topologies

We now present our results on the gossip schedule lengths for some regular topologies. It may be fairly easily derived that gossip can be achieved in a complete graph of N nodes in N slots. Likewise for a star graph, gossip can be achieved in $2N - 1$ slots.

Now let us consider a ring network with $N = 3m + k$, m a positive integer and $k = 0, 1, 2$, nodes. Distance-2 colouring on this graph will require $3 + k$ colours. If the nodes are directed to first transmit their message in their assigned slot and then in subsequent slots transmit the message that has come from the rightmost (leftmost) vertex then it can be shown that gossiping can be completed in exactly $(3 + k)(N - 2)$ slots. If $k = 2$, we have an improved schedule that can achieve gossip in $4N - 4$ slots if N is odd and $4N - 5$ slots if N is even. This reduction is achieved by organising the nodes as a tree and doing gossip in a manner similar to that in the Gather-Scatter algorithm.

For bus networks too, we pick the central node as the root node and develop an algorithm similar to the Gather_Scatter algorithm. If $N > 5$, this helps us achieve gossip in $3(\lfloor \frac{N}{2} \rfloor) + 3(N - \lfloor \frac{N}{2} \rfloor - 1) + (\lfloor \frac{N}{2} \rfloor)$. For $N \leq 5$ gossip can be achieved in fewer slots.

VI. Conclusion

In this paper we have introduced the problem of gossiping in multihop radio networks and presented some polynomial time centralised scheduling algorithms to schedule for gossip in these networks. In the development of these algorithms our interest is to reduce the running time complexity of these algorithms and at the same time deliver the best possible schedule.

From our experimental results for the Collision Free and the Centralised Spanning Tree we observe that the shortest gossip schedule is achieved when the nodes use the RS rule to select a message to transmit in the assigned slot. The FCFS, IS and LCFS rules give increasingly longer schedules. This is because in the LCFS and IS disciplines, messages may be consistently blocked and this delays their dissemination in the network. This, in turn increases the number of primary frames in the gossip frame resulting in a poor performance of the algorithm. The observation that RS is better than FCFS is similar to that made by Topkis [4] for gossiping in wireline networks. It was also observed that a node giving priority to its own message yielded a better performance than when they strictly follow the relay rules.

Among the algorithms presented we have found the Gather_Scatter algorithm to have the best inpractice performance. Our experimental results also indicate that this is probably close to the best polynomial time algorithm to achieve gossip in arbitrary graphs representing multihop radio networks. We base this conclusion on the result that the gossip schedule length approaches N as the graph becomes dense and approaches a complete graph.

REFERENCES

[1] Reuven Bar Yehuda, Technion Institute of Technology, Israel, Personal Communication,

[2] Neeraj Gupta, "Gossiping in Multihop Radio Networks," *M. Tech Thesis,* Dept of Elecl Engg, Indian Inst of Technology, Kanpur, Feb 1998.

[3] E. L. Lloyd and S. Ramanathan, "On the complexity of link scheduling in multihop radio networks," *Proceedings of the 26th Conference on Information Sciences and Systems,* pp. 491-495, March 1992.

[4] D. M. Topkis, "All-to-all Broadcast by Flooding in Communication Networks," *IEEE Transactions on Communications,* vol. 38, no. 9, pp. 1330-1333, 1989.