Acknowledged Broadcasting and Gossiping in ad hoc radio networks

Jiro Uchida[1], Wei Chen[2], and Koichi Wada[3]

[1] ,[3]Nagoya Institute of Technology Gokiso-cho, Syowa-ku, Nagoya, 466-8555, Japan,
[1]`jiro@phaser.elcom.nitech.ac.jp`, [3]`wada@nitech.ac.jp`
[2] Tennessee State University 3500 John A MerritBlvd, Nashville, TN 37205, USA,
[2]`wchen@tnstate.edu`

Abstract. A radio network is a collection of transmitter-receiver devices (referred to as nodes). ARB (Acknowledged Radio Broadcasting) means transmitting a message from one special node called source to all the nodes and informing the source about its completion. In our model each node takes a synchronization per round and performs transmission or reception at one round. Each node does not have a collision detection capability and knows only own ID. In [1], it is proved that no ARB algorithm exists in the model without collision detection. In this paper, we show that if $n \geq 2$, where n is the number of nodes in the network, we can construct algorithms which solve ARB in $O(n)$ rounds for bidirectional graphs and in $O(n^{3/2})$ rounds for strongly connected graphs and solve ARG (Acknowledged Radio Gossiping) in $O(n \log^3 n)$ rounds for bidirectional graphs and in $O(n^{3/2})$ rounds for strongly connected graphs without collision detection.

Key words: broadcasting, gossiping, distributed, deterministic, radio network

1 Introduction

A radio network is a collection of transmitter-receiver devices (denoted as nodes). Each node can transmit data to the nodes that exist within its transmitting capability region. A radio network can be modeled by a directed graph (we simply call it graph) $G = (V, E)$ called *reachability graph*, where V denotes a set of nodes and when a node u can transmit to a node v, there exists an edge $(u, v) \in E$. If $(u, v) \in E$, u is called an *in-neighbor* of v, and v is called an *out-neighbor* of u. If the power of every transmitter is the same, then the reachability graph is bidirectional[4], that is, if there is an edge from node u to node v, then there exists the edge from v to u, and vice versa.

We assume that all nodes in a radio network have access to a global clock (like GPS) and work synchronously in discrete time steps called rounds. At every round, each node transmits data or receives data. A node acting as a receiver in a given round gets a message iff exactly one of its in-neighbors transmits in

---

[4] bidirectional is called symmetric in [1].

this round. If at least two in-neighbors v and v' of u transmit simultaneously in a given round, none of the messages is received by u in this round. In this case we say that a conflict or a collision occurred at u. When collision occurs, two cases are considered: u notices the occurrence of a collision (i.e. it has collision detection), and u cannot distinguish between the background noise and the interference noise. It depends on the capability whether a node can detect a collision or not.

One of the fundamental tasks in network communication is radio broadcasting (RB). Its goal is to transmit a message from one node of the network, called the source, to all other nodes. The message which is disseminated is called source message. Remote nodes get the source message via intermediate nodes, along directed paths in the network. In an acknowledged radio broadcasting (ARB) the goal is not only to achieve RB but also to inform the source about the completion of RB. This may be essential, e.g., when the source has several messages to disseminate, none of the nodes should receive the next message until all nodes get the previous one [1]. Another task is radio gossiping (RG) which broadcasts the message of each node to all other nodes. We also consider the task acknowledged radio gossiping (ARG) which achieves RG and inform every node about the completion of RG.

In this paper, we consider the standard model of unknown radio networks, called the ad-hoc radio network model. We assume that each node does not know any information of the network (e.g. its neighbor, the number of nodes and the topology). The network is assumed to have a fix topology during the execution of algorithms. However, since no information of the network is used in our algorithms, they can be applied to networks with any topology. We evaluate algorithms with the number of rounds used to complete the tasks.

1.1 Previous results

The standard collision-free communication procedure for ad hoc radio networks is called *Round Robin* [2]. Round Robin contains n rounds. In the i-th round the node with identifier i transmits its whole knowledge to all its out-neighbors. In every round at most one node acts as a transmitter, hence collisions are avoided. Round Robin is used as a subroutine in many RB and RG algorithms. An RG completes in $O(n^2)$ rounds, where n is the number of nodes.

There are two situations for communication procedures in radio networks: one is that nodes have full knowledge of the network (such as the topology of the network, the number of the nodes in the network, IDs of the neighbors etc.), the other is that nodes are ignorant of the network information. Various algorithms are studied in radio networks, e.g. the centralized algorithms with the mechanism in which all nodes are concentrated and managed, and the distributed algorithms without such a mechanism; the deterministic algorithms whose process become settled uniquely, and randomized algorithms which are not so [1–10].

Under the assumption that the nodes have full knowledge of the network, in [3] the authors proved the existence of a family of n-node networks of radius 2, for which any broadcast requires $\Omega(\log^2 n)$ time, while in [4] it was proved that broadcasting can be done in $O(D+\log^5 n)$ time, for any n-node network of diameter D.

Hereafter, we assume that the nodes have neither the knowledge of the network nor the knowledge of their neighborhood.

For randomized algorithms, the lower bound of $\Omega(D\log(n/D))$ for bidirectional graphs is shown by Kushilevitz and Mansour [6], and the lower bound of $\Omega(\log^2 n)$ for constant diameter networks is obtained by Alon *et al.* [3].

For deterministic distributed algorithms, on the model without collision detection, Chlebus *et al.* have presented an optimal linear-time broadcasting protocol for bidirectional ad hoc radio networks [1]. Also, on the model with collision detection, they presented an $O(n\cdot ecc)$-time RB algorithm for strongly connected graphs, an $O(r\cdot ecc)$-time RB algorithm for arbitrary graphs, an $O(n)$-time ARB algorithm for bidirectional graphs, and an $O(n\cdot ecc)$-time ARB algorithm for strongly connected graphs, where *ecc* is the maximum distance from the source. Note that on the model without collision detection there does not exist any algorithm for ARB, even for bidirectional graphs [1]. The best $O(n^{1.5})$ time gossiping algorithm for strongly connected graphs is shown in [10].

About the lower bounds of deterministic RB, the lower bound of $\Omega(n)$ for bidirectional graphs [5] and the lower bound of $\Omega(n\log n)$ for arbitrary graphs [9] are shown.

Table 1 shows the results of these deterministic algorithms.

Problem	Collision detection	Graphs	Computation time
RB	without	bidirectional	$O(n)$ [1]
			$\Omega(n)$ [5]
		arbitrary	$O(n\log^2 n)$ [8]
			$\Omega(n\log n)$ [9]
	with	bidirectional	$O(r+ecc)$ [7]
		strongly connected	$O(n\cdot ecc)$ [1]
		arbitrary	$O(r\cdot ecc)$ [1]
RG	without	strongly connected	$O(n^{3/2})$ [10]
ARB	without	bidirectional	algorithm does not exist [1]
	with	bidirectional	$O(n)$ [1]
			$O(r+ecc)$ [7]
		strongly connected	$O(n\cdot ecc)$ [1]

(n:number of nodes, ecc:largest distance from the source, r:length of the source message)

Table 1. Previous results(Deterministic and Distributed)

1.2 Our results

In this paper, we consider the ARB and the ARG algorithms on the model of ad hoc radio networks without collision detection. As we mentioned on the model without collision detection, there does not exist any ARB algorithm even for bidirectional graphs [1], which is proved by using a special case: when the source does not receive any message about the completion of the RB, the source can not distinguish between the situations that the network has only the source node (thus the source does not receive any message) and that at least two in-neighbors of the source transmit some messages (thus collision occurs).

If we assume that the network contains at least one node other than the source node and each node knows the number of nodes or its in-neighbors, RB algorithms can be easily modified to ARB ones. It is interesting to know the weakest conditions needed for performing an ARB. In this paper, we show that if the network contains at least two nodes, we can construct algorithms which solve ARB for bidirectional graphs and strongly connected graphs under the assumption that the network has no collision detection and each node knows only its ID.

The computation time of our ARB algorithm for bidirectional graphs is the same as the existing best RB algorithm which uses $O(n)$ rounds. The computation time of our ARB algorithm for strongly connected graphs is $O(6n + \sum_{i=1}^{\lceil \log n \rceil}\{2{\cdot}RB(2^i) + RG(2^i)\})$, where $RB(n)$ and $RG(n)$ is the number of rounds which an RB and an RG requires for n-node strongly connected graphs, respectively. It becomes $O(n^{3/2})$ when using the $O(n^{3/2})$-time gossiping algorithm from [10].

In addition, we consider acknowledged radio gossiping (ARG) algorithms. We show that our ARB algorithms can be extended to ARG algorithms for both of bidirectional graphs and strongly connected graphs. Our ARB algorithm for bidirectional graphs needs a leader, and we use the source node to be the leader in the algorithm. In ARG, since no source node is given, we need to elect a leader for ARG when we extend the ARB algorithm to an ARG algorithm. For strongly connected graphs our ARB algorithm does not need a leader, therefore , in this case, the ARB algorithm can be extended to an ARG algorithm directly. The computation time of the extended ARG algorithms is $O(n + \sum_{i=1}^{\lceil \log n \rceil}\{LE(2^i)\})$ for bidirectional graphs and $O(6n + \sum_{i=1}^{\lceil \log n \rceil}\{RB(2^i) + 2{\cdot}RG(2^i)\})$ for strongly connected graphs, respectively, where $LE(n)$ denotes the number of the rounds needed to elect a leader for n-node bidirectional graphs. The computation times of ARG algorithms become $O(n \log^3 n)$ and $O(n^{3/2})$, respectively, by using the $O(n \log^3 n)$-time leader election algorithm from [8] and the $O(n^{3/2})$-time gossiping algorithm from [10].

2 Model and Definitions

In this paper, we consider the radio networks without a collision detection. We describe the model of radio networks we consider :

- The knowledge of every node is limited to its own ID.
- Each node knows whether itself is a source or not in broadcasting.
- Nodes in a radio network work per round synchronized by a global clock.
- In every round, each node acts either as a transmitter or as a receiver.
- A node acting as a receiver in a given round gets a message iff exactly one of its in-neighbors transmits in this round.

- If more than one in-neighbor transmits simultaneously in a given round, collision occurs and none of the messages is received in this round.
- A node cannot notice the occurrence of a collision (i.e. without collision detection).

For simplicity we assume that each node is labeled with distinct integers between 1 and n in an n-node network. But all the arguments hold if the labels are distinct integers between 1 and $Z = O(n)$, and we do not use the property that the labels are in $\{1, 2, \ldots, n\}$.

3 ARB and ARG in bidirectional graphs

In this section, we describe ARB and ARG algorithms for bidirectional graphs where the number of nodes in the network is at least 2. First, we describe the overview of our algorithms, secondly we show an ARB algorithm and then modify it to an ARG algorithm.

3.1 Overview of our algorithm

The main idea of our algorithm is that each node confirms all of its in-neighbors in every phase, where in the k-th phase nodes with ID at most 2^k works. In the k-th phase, first the in-neighbors of any node v whose IDs are no more than 2^k send their own IDs, thus the node v can recognize its in-neighbors' IDs that are no more than 2^k. Then in the same phase the node whose ID is the minimum one among the in-neighbors with IDs no more than 2^k, and nodes whose IDs are more than 2^k send their IDs simultaneously. If the node v receives the minimum ID (i.e. collision does not occur), it recognizes that it knows all of the in-neighbor in this phase. It is easy to perform the ARB if every node knows all of its in-neighbors. If the node v does not receive the minimum ID (i.e. collision occurs), v recognizes that it does not know all of the in-neighbors and the algorithm performs the next phase.

3.2 Algorithm bi-ARB

We show an ARB algorithm named bi-ARB for bidirectional graphs in an n node radio network, where $n \geq 2$.

Algorithm bi-ARB works phase by phase, numbered by consecutive positive integers. Phase k lasts $9 \cdot 2^{k-1}$ rounds divided into four stages. Stage A consists of 2^{k-1} rounds, Stage B consists of 2^k rounds, Stage C consists of 2^k rounds, and Stage D consists of 2^{k+1} rounds. We denote the ID of node v as ID(v). We define the following notations.

- L_k : the set of nodes with IDs $1, \ldots,$ and 2^k.
- G_k : the connected component containing the source of the network induced by L_k. $G_k = \phi$ if the ID of the source node is larger than 2^k.

- N_v^k : the set of IDs smaller than or equal to 2^k from the in-neighbors of node v.
- $\min(N_v^k)$: the minimum ID in N_v^k. If $N_v^k = \phi$, $\min(N_v^k) = \perp$.

Note that in bidirectional graphs the in-neighbors of each node v are the same as the out-neighbors of v.

Informally we show the algorithm of phase k. Stage A is a Round Robin which intends to let each node v know its in-neighbors (and out-neighbors) whose IDs are at most 2^k (N_v^k). In Stage B each node v in L_k sends $min(N_v^k)$, which will be the only node in in-neighbors of v can transmit to v in the next stage C. Stage C is used to judge whether the node v of G_k knows all of its in-neighbors or not. In Stage C the node whose ID is $\min(N_v^k)$ and nodes not in L_k send their IDs, then according to whether receiving $\min(N_v^k)$ or not every node v in G_k recognizes whether it knows all its in-neighbors or not. In Stage D the source node in G_k broadcasts the source message to every node of G_k. The stage also collects the information that whether each node in G_k knows all its in-neighbors. Thereby the source node can confirm the completion of RB. We use the broadcast algorithm shown in [1] in this stage.

bi-ARB Phase 0 consists of one round, the node with ID 1 acts as transmitter and sends its ID in this phase. The other nodes act as receivers.

Hereafter, we explain phase $k > 0$, of bi-ARB. We assume that every node is either a transmitter or a receiver in each round.

Stage A. The rounds in Stage A of phase k are numbered by integers $2^{k-1} + 1, \ldots, 2^{k-1} + 2^{k-1}$. In round number i of Stage A only the node v with ID i acts as a transmitter and sends a message ID(v).

Stage B. The rounds of this stage are numbered by integers $1, \ldots, 2^k$. In round i of Stage B only the node v with ID i acts as a transmitter and sends a message $\min(N_v^k)$. If $\min(N_v^k) = \perp$, the node v sends no message. The node w that receives $\min(N_v^k)$ stores it if ID(w)=$\min(N_v^k)$.

Stage C. The rounds in Stage C of phase k are numbered by integers $1, \ldots, 2^k$. In round i of Stage C, the node v with ID i acts as a receiver. The node with ID=$\min(N_v^k)$ acts as transmitter and sends its ID (if $\min(N_v^k)$ min()), and all the nodes whose IDs are larger than 2^k (not only in-neighbors of v) also send their own IDs in the round.

Every node v not receiving $\min(N_v^k)$ in the round ID(v), is set to the state ***warned*** which means that v does not know all its in-neighbors, or in other words, v has the in-neighbors whose IDs are larger than 2^k.

Stage D. The rounds in Stage D of phase k are numbered by integers $1, \ldots, 2^{k+1}$. The source initiates Stage D if its ID is less than or equal to 2^k. Otherwise all nodes do nothing in these 2^{k+1} rounds. We use a message called token. At the beginning of this stage every node $v \in G_k$ knows its out-neighbor N_v^k in G_k and maintains a list Q_v containing the set of its out-neighbors in G_k which were not yet visited by the token. Q_v is initialized to N_v^k.

When a ***warned*** node sends the token to an out-neighbor, it appends a ***warning*** message to the token, and the out-neighbor getting the token becomes ***warned***.

When node v gets the token, it acts as follows:

step 1. Node v sends the message `<ID(v), visited>`. If a node u receives the message, it removes v from the list Q_u.

step 2. Node v sends the token `<source message, ID(w), (warning)>` to the following node w:
  - (i) If $Q_v = \phi$, w is the node from which v got the message in step 1 for the first time.
  - (ii) If $Q_v \neq \phi$, w is the node with the smallest ID in the list Q_v.

the messages are concatenated and are sent in a single round. Node w which gets the token repeats the procedure of step 1 and step 2.

If, at the end of phase k, the source is ***warned***, it knows that the RB has not been completed, and shifts to the next phase. Otherwise the algorithm terminates.

Correctness of Algorithm bi-ARB

Lemma 1. *The following invariants are maintained after phase k of bi-ARB, for any positive integer k.*

- *Every node v knows the N_v^k, the set of IDs at most 2^k from the in-neighbors(and out-neighbors) of v.*
- *Every node in G_k knows the source message, if G_k contains the source node.*

Proof. In phase $k = 0$, G_k contains only the source node if its ID equals 1, and $N_v^k = \phi$. Therefore, Lemma 1 holds obviously in this case.

Assume that the invariants hold after phase $k-1$, $k \geq 1$. We show that the invariants are maintained after phase k.

In Stage A of phase k, the nodes whose IDs are between 2^{k-1} and 2^k transmit their IDs. In every round, exact one node acts as a transmitter and the other nodes act as receivers, hence collisions are avoided. Any node v has already known N_v^{k-1} after phase $k-1$ from the assumption, v learns $N_v^k - N_v^{k-1}$ the remaining neighbors in G_k during phase k.

In Stage D if G_k contains the source node, the token is patrolled from the source node to all nodes in G_k. At the beginning of Stage D the token is in the source node. It visits each node of G_k from the source node in depth-first order. When node v got the token, it sends the token with the source message and its ID to its out-neighbors which have not received the token yet, following the Eulerian cycle C_k of a spanning tree of G_k as follows: Q_v is the set of out-neighbors of v in G_k which were not yet visited by the token. The node v that receives the token has to send the message `<ID(v), visited>` to its in-neighbors, node w that receives the message removes v from the list Q_w. If v has the neighbors which are not visited by the token, it passes the token to the one with the smallest ID. Else, v returns the token to the node from which it got the token for the first time. In Stage A every node v in G_k knows its out-neighbors in G_k, so the token patrols every node in G_k and returns to the source finally. □

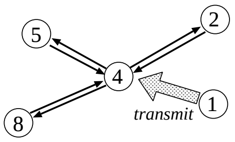

Fig. 1. knowing all in-neighbors (k=3)

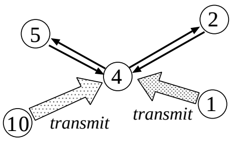

Fig. 2. otherwise (k=3)

Theorem 1. *Algorithm bi-ARB performs an ARB in time $O(n)$, for any n-node bidirectional graph with $n \geq 2$.*

Proof. Let l be such that $2^{l-1} < n \leq 2^l$. It is sufficient to show that

(1) After phase l all nodes of the network get the source message.
(2) At the end of phase l the source does not ***warned***.

In order to prove (1) consider phase l. Since G_l is the entire network, (1) follows from Lemma 1. The number of the rounds needed for this algorithm is at most $\sum_{i=1}^{l} 9 \cdot 2^{i-1} \leq 9 \cdot 2^l \leq 18n$.

We prove (2). At the end of Stage A each node v knows N_v^k. It sends $\min(N_v^k)$ in round i=ID(v) of Stage B. The node w receiving $\min(N_v^k)$ memorizes the number of the round if ID(w)=$\min(N_v^k)$, otherwise ignores the message. Thereby in round i of Stage C only the node with ID i can act as transmitter.

A node in G_k recognizes whether it knows all its in-neighbors in Stage C. In round i of this stage for the node v with ID i, the node with ID=$\min(N_v^k)$ and the nodes with IDs larger than 2^k send their own IDs. Therefore the node v having in-neighbors with ID larger than 2^k cannot receive $\min(N_v^k)$ in round ID(v) due to a collision. Then v recognizes that it does not know all in-neighbors, and becomes ***warned***. If v knows all in-neighbors, it can receive $\min(N_v^k)$ and will not become ***warned***. Figure 1 shows the case where a node knows all in-neighbors, and Figure 2 shows the other case in round 4 of phase 3, where the number of node represents its ID.

Consider phase l. Since there is no node whose ID is larger than 2^l, each node v can receive $\min(N_v^k)$ in the round ID(v) in Stage C. Therefore no node becomes ***warned*** in Stage D. Hence the source node is not ***warned*** at the end of phase l. ☐

Message size. Let S be the maximum length of the message transmitted each time and let r be the length of the source message. In Stage A,B and C each node transmits at most one ID respectively, thus $S = O(\log n)$. In Stage D each node transmits message `<ID(v), visited>` and the token `<source message, ID(w), (warning)>`, thus $S = O(r + \log n)$. Hence the maximum message size is at most $O(r + \log n)$ for algorithm bi-ARB.

3.3 Algorithm bi-ARG

The ARG algorithm bi-ARG for bidirectional graphs is obtained by changing a part of bi-ARB.

Algorithm bi-ARG works in phases, numbered by consecutive positive integers similar to bi-ARB. Each phase consists of four stages A,B,C and D. Stage A,B,C are the same as these of algorithm bi-ARB but Stage D is different. It needs a leader election procedure and an extra token patrolling. Recall that in bi-ARB, the source node is used to be the starting point of the token patrolling. Furthermore, each node knows whether itself is a source or not. But the source node does not exist for ARG. We have to elect one leader for each connected component induced by L_k so that the token patrolling can be performed in each component. We use a leader election procedure. The leader of each connected component acts as initiator and makes the token patrol twice in its connected component in Stage D. In the first patrol the leader of each connected component collects the messages which each node has and ***warning*** messages from the nodes to the leader (the same as that in bi-ARB), then in the second patrol it disseminates the messages which were collected in the first patrol to all the nodes in the component. Thereby any node knows whether RG have completed or not.

In order to use an leader election algorithm, each node must know the completion time of the algorithm, since the leader election procedure must finish in each phase of bi-ARG.

Theorem 2. *Algorithm bi-ARG performs an ARG in time* $O(n+\sum_{i=1}^{\lceil\log n\rceil}\{LE(2^i)\})$*, for any bidirectional graph with* $n \geq 2$*, where* $LE(k)$ *denotes the number of the rounds of any leader election algorithm for* k*-node bidirectional graphs in which each node knows the completion time.*

We use the algorithm FIND MAX shown in [8] as a leader election procedure. The algorithm FIND MAX elects a leader by calculating the maximum ID on a strongly connected graph under the assumption that each node knows the upper bound of IDs of nodes in the network. Moreover, if each node knows (the upper bound of) the number of nodes n in the network, it can compute the completion time of FIND MAX, which is $cn\log^3 n$ for some known constant c. Algorithm FIND MAX finds the leader based on binary search. At each step, all nodes know that the minimum ID (the node having this ID is elected as a leader) among all nodes is between a and b by broadcasting a message, where $a \leq b$. Initially $a = 0$ and $b = n$. If $a = b$, then the minimum ID is equal to a, and the computation of minimum ID is complete. In each phase we use this algorithm to elect a leader for each connected component. In phase k, the upper bound of IDs and that of the number of nodes in the connected components induced by L_k is known to be 2^k. We obtain the following corollary from Theorem 2 using the $O(n\log^3 n)$-time leader election algorithm FIND MAX.

Corollary 1. *Algorithm bi-ARG performs ARG in time* $O(n\log^3 n)$*, for any bidirectional graph with* $n \geq 2$*.*

$$\left(\because \sum_{i=1}^{\lceil\log n\rceil} 2^i\log^3 2^i \leq 2(2^{\lceil\log n\rceil}-1)\cdot(\log n+1)^3 \leq 2(2n-2)\cdot(\log n+1)^3\right)$$

Our algorithm bi-ARG is improvable if more efficient leader election algorithms can be designed for bidirectional graphs under the condition that each node knows the maximum of IDs and n.

Message size. Let S be the maximum length of the message transmitted each time and let r be the length of the message each node has. In Stage A,B and C, $S = O(\log n)$ which are the same as that of bi-ARB. In Stage D first $S = O(\log n)$ for the leader election procedure FIND MAX [8]. Next each node adds its own message to the token, $S = O(rn + \log n)$. Hence the maximum message size is at most $O(rn + \log n)$ for algorithm bi-ARG.

4 ARB and ARG in strongly connected graphs

4.1 Algorithm st-ARB

The ARB algorithm st-ARB for strongly connected graphs is obtained by changing a part of bi-ARB.

Algorithm st-ARB works in phases, numbered by consecutive positive integers. Every phase starts in the round following the end of the previous phase. Phase $k(>0)$ lasts $3\cdot 2^{k-1} + 2\cdot RB(2^k) + RG(2^k)$ rounds divided into four stages. Stage A consists of 2^{k-1} rounds, Stage B consists of $RG(2^k)$ rounds, Stage C consists of 2^k rounds, and Stage D consists of $2\cdot RB(2^k)$ rounds.

Here we show the outline of this algorithm in phase k. Stage A and C of st-ARB are the same as those of bi-ARB, and the purpose of Stage B and D also does not change. Although in bidirectional graphs a node v can transmit $\min(N_v^k)$ to its in-neighbor w whose ID=$\min(N_v^k)$ because the in-neighbors of v is also its out-neighbors, node v cannot do that in strongly connected graphs since w may not be an out-neighbor of v. To do this, v must gossip on the subgraph induced by L_k in Stage B. In Stage D each node other than the source node in L_k transmits the ***warning*** message and the source node broadcasts the source message. Thereby the source node can confirm the completion of RB.

In st-ARB we use the RB and RG in the subgraph induced by L_k (not necessarily strongly connected). In order to apply the RB algorithm for strongly connected graphs to our algorithm, it is sufficient to perform the task for all reachable nodes. About RG algorithm, it is not necessary to perform the task for all reachable nodes. Any algorithm of RB and RG can be applied to our algorithm if each node knows the completion time. We consider an extension of the RB that broadcasts from several source nodes with the same messages to all reachable nodes, and use the algorithm that performs such an extended RB in Stage D. Since the algorithm does not depend on the information of the source node, it can perform an RB in the situation such that several source nodes exist.

st-ARB Phase 0 consists of one round, the node with ID 1 acts as transmitter and sends its ID in this phase. The other nodes act as receivers.

Hereafter, we explain phase $k(>0)$ of st-ARB. Stage A and C is the same as that of bi-ARB. Every node that is not transmitter is receiver in the explanation.

Stage A. Rounds in Stage A of phase k are numbered by integers $2^{k-1} + 1, \ldots, 2^{k-1} + 2^{k-1}$. In round number i of Stage A the only node v with ID i acts as a transmitter and sends a message $\mathrm{ID}(v)$.
Stage B. Stage B consists of $RG(2^k)$ rounds. In Stage B each node v in L_k acts as a transmitter, gossiping the message `<ID(v), min(N`$^{\mathtt{k}}_{\mathtt{v}}$`)>`. If $\min(N_v^k) = \lambda$, the node v sends no message.
Stage C. Rounds in Stage C of phase k are numbered by integers $1, \ldots, 2^k$. In round number i of Stage C the node v with ID i acts as a receiver. The node with ID $\min(N_v^k)$ and the nodes whose IDs are larger than 2^k act as transmitter, sending their own IDs.

Every node v not receiving $\min(N_v^k)$ in the round $\mathrm{ID}(v)$, is set to the state ***warned***.
Stage D. Stage D consists of $2 \cdot RB(2^k)$ rounds. First, each node sends a ***warning*** message if it is ***warned***. Next, if the source does not receive the ***warning*** message, it knows that there is no node in L_k whose in-neighbors with ID$> 2^k$ and then broadcasts the source message, otherwise it knows that there still exist nodes in L_k whose in-neighbors with ID$> 2^k$ and then it becomes ***warned***, and shifts to the next phase.

Correctness of Algorithm st-ARB

Lemma 2. *If there are* ***warned*** *nodes in the strongly connected graph after phase k of st-ARG then there is a path from at least one* ***warned*** *node to the source node that contains only nodes whose IDs are not larger than 2^k.*

Proof. Let v be some ***warned*** node. In the original graph there is a path from v to the source. If there are nodes with ID$> 2^k$ in this path, let the out-neighbor of the last of them in the path be v'. The path from v' to the source proves the lemma. □

Theorem 3. *Algorithm st-ARB performs ARB in time $O(6n + \sum_{i=1}^{\lceil \log n \rceil}\{2 \cdot RB(2^i) + RG(2^i)\})$, in any strongly connected graphs with n nodes, where $n \geq 2$ and $RB(k)$ and $RG(k)$ denotes the number of the rounds of any extended RB and RG algorithm for k-node strongly connected graphs in which each node knows the completion time, respectively.*

Proof. Let l be such that $2^{l-1} < n \leq 2^l$. It is enough to show that

(1) After phase l all nodes of the network get the source message.
(2) At the end of phase l the source node does not ***warned***.

In order to prove (1) consider phase l. Since L_l is the entire network, each node considers the upper bound of the number of nodes is 2^l and does broadcasting, then every node gets the source message. The completion time of this algorithm is at most

$$\sum_{i=1}^{l}\{3 \cdot 2^{i-1} + 2 \cdot RB(2^i) + RG(2^i)\} \leq 6n + \sum_{i=1}^{\lceil \log n \rceil} \{2 \cdot RB(2^i) + RG(2^i)\}$$

We prove (2). Since Stage A is the same as that of bi-ARB for phase k, any node v knows N_v^k in the stage.

In Stage B each node v in L_k gossips `<ID(v), min(N`$^{\mathtt{k}}_{\mathtt{v}}$`)>`. If the gossiping are performed correctly, in Stage C only one node in N_v^k can act as transmitter. If L_k does not contain all nodes of the graph, the induced subgraph by L_k is not necessarily strongly connected and the gossiping of all messages is not secured. But L_l contains all node in the graph, all messages are gossiped correctly.

Stage C is also the same as that of bi-ARB, each node v recognizes whether it knows all its in-neighbors. Similar to bi-ARB the node v having in-neighbors with ID larger than 2^k cannot receive $\min(N_v^k)$ in round $\mathrm{ID}(v)$. The node v which could not receive $\min(N_v^k)$ recognizes that it does not know all in-neighbor, and becomes ***warned***. If there is no node with ID$> 2^k$ in the graph, all messages are gossiped in Stage B. It means that v can receive $\min(N_v^k)$ in Stage C and does not become ***warned***.

In Stage D each node confirms whether it receives the ***warning*** message or not, and the source node sends the source message. From Lemma 2 if there exists at least one ***warned*** node, its ***warning*** message reaches the source node. Then the source node knows that there exist the nodes in the graph with ID$> 2^k$. Consider phase l, since there is no node in the graph with ID$> 2^l$, each message of any node is gossiped to all nodes in Stage B correctly. Therefore any node does not become ***warned*** in Stage C. Hence, the source node confirms the completion of RB and is not ***warned*** at the end of phase l since L_l is the entire network and there is no ***warned*** node in the graph. □

We obtain the following corollary from Theorem 3 using the $O(n \log^2 n)$-time broadcasting algorithm from [8] and the $O(n^{3/2})$-time gossiping algorithm from [10]. The broadcasting Algorithm from [8] can perform the extended RB. The algorithm consists of stages, with each stage having $\log n + 1 = O(\log n)$ steps. For each $j = 0, \ldots, \log n$ let $\overline{S}_j = (S_{j,0}, S_{j,1}, \ldots, S_{j,m_j-1})$ be a 2^j-selector with $m_j = O(2^j \log n)$ sets, and the transmission set at the jth step of stage s is $S_{j,\ s \bmod m_j}$, where *w-selector* is defined as follow; Given a positive integer w, a family $\overline{S}$ of sets is called a *w-selector* if it satisfies the following property: For any two disjoint sets $X, Y \in \{1, \ldots, n\}$ with $w/2 \le |X| \le w$ and $|Y| \le w$ there exists a set in $\overline{S}$ such that $|S \cap X| = 1$ and $S \cap X = \phi$. Since each node does not use the information whether it is the source or not and does not depend on the message it received in the previous round, RB can be done on condition that several source nodes have the same message. Each node can compute the completion time of each algorithm under the assumption that it knows the upper bound of IDs of nodes in the network.

Corollary 2. *Algorithm st-ARB performs ARB in time $O(n^{3/2})$, for any strongly connected graphs with $n \geq 2$.*

Message size. Let S be the maximum length of the message transmitted each time and let r be the length of the source message. In Stage A and C each node transmits at most one ID, thus $S = O(\log n)$. In Stage B each node v gossips $\mathrm{ID}(v)$

and $\min(N_v^k)$, thus $S = O(n \log n)$. In Stage D each node transmits a ***warning*** message, the source node transmits the source message, thus $S = O(r)$. Hence the maximum message size is at most $O(r + n \log n)$ for algorithm st-ARB.

4.2 Algorithm st-ARG

The ARG algorithm st-ARG for strongly connected graphs is obtained by changing a part of st-ARB.

Algorithm st-ARG works in phases, numbered by consecutive positive integers as well as st-ARB. Stage A,B and C is the same as that of st-ARB. We perform ARG by changing Stage D. Stage D consists of $RB(2^k)+RG(2^k)$ rounds. First step where each node confirms whether it receives the ***warning*** message or not is the same as that of Stage D of st-ARB. If a node does not receive ***warning*** message, it knows that there is no node with ID$> 2^k$ and gossip its own message, otherwise it knows that there still exist nodes with ID$> 2^k$ and becomes ***warned***, then shifts to the next phase.

Theorem 4. *Algorithm st-ARG performs ARG in time* $O(6n+\sum_{i=1}^{\lceil \log n \rceil}\{RB(2^i)+2 \cdot RG(2^i)\})$*, for any strongly connected graph with* n *nodes, where* $n \geq 2$ *and* $RB(k)$ *and* $RG(k)$ *denotes the number of the rounds of any RB and RG algorithm for* k*-node strongly connected graphs in which each node knows the completion time, respectively.*

We obtain the following corollary from Theorem 4 using the $O(n \log^2 n)$-time broadcasting algorithm from [8] and the $O(n^{3/2})$-time gossiping algorithm from [10] as well as Corollary 2.

Corollary 3. *Algorithm st-ARG performs ARG in time* $O(n^{3/2})$*, for any strongly connected graph with* n *nodes, where* $n \geq 2$*.*

Message size. Let S be the maximum length of the message transmitted each time and let r be the length of the message each node has. In Stage A,B and C $S = O(n \log n)$ is the same as that of st-ARB. In Stage D each node v broadcasts a ***warning*** message and gossips its own message, thus $S = O(rn)$. Hence the maximum message size is at most $O(rn + n \log n)$ for algorithm st-ARG.

5 Conclusion

In this paper, on the model without collision detection we show that we can construct deterministic and distributed ARB algorithms for symmetric digraphs in time $O(n)$, and for strongly connected digraphs in time $O(6n + \sum_{i=1}^{\lceil \log n \rceil}\{2 \cdot RB(2^i) + RG(2^i)\})$, where n is the number of the nodes in the graphs and $n \geq 2$. We also show that our each ARB algorithm can be extended to ARG algorithm.

Our algorithms can be improved if we can find more efficient leader election algorithms for symmetric digraphs and if ARB can be achieved without using RG for strongly connected digraphs.

References

1. B.S. Chlebus, L. Gąsieniec, A.M. Gibbons, A. Pelc, and W. Rytter. Deterministic broadcasting in ad hoc radio networks. *Distributed Computing 15*, pages 27–38, 2002.
2. L.Gąsieniec, M.Christersson and A.Lingas. Gossiping with bounded size messages in ad hoc radio networks. *29th International Colloquium on Automata, Languages and Programming, (ICALP'02)*, pages 377–389, 2002.
3. N.Alon,, A.Bar-Noy, N.Linial, and D.Peleg. A lower bound for radio broadcast. *Journal of Computer and System Sciences 43*, pages 290–298, 1991.
4. I.Gaber and Y.Mansour. Broadcast in Radio Networks. *6th Ann. ACM-SIAM Symp. on Discrete Algorithms, SODA'95*, pages 577–585, 95.
5. R.Bar-Yehuda, O.Goldreich, and A.Itai. On the time-complexity of broadcast in radio networks: an exponential gap between determinism and randomization, *6th ACM Symposium on Principles of Distributed Computing*, pages 98–107, 1987.
6. E.Kushilevitz and Y.Mansour. An $\Omega\left(D \log \frac{n}{D}\right)$ lower bound for broadcast in radio networks. *12th Ann. ACM Symp. on Principles of Distributed Computing*, 65–73, 1993.
7. T.Okuwa, W.Chen and K.Wada. An optimal algorithm of acknowledged broadcasting in ad hoc networks, Proc. of 2nd Int'l Symp. Parallel and Distributed Computing(2003)(to appear).
8. M.Chrobak, L.Gąsieniec, and W.Rytter. Fast broadcasting and gossiping in radio networks. *st IEEE Symp. on Found. of Computer Science (FOCS'2000)*, pages 575–581, 2000.
9. D.Brusci and M.Del Pinto. Lower bounds for the broadcast problem in mobile radio networks. *Distributed Computing 10*, pages 129–135, 1997.
10. Y.Xu. An $O(n^{1.5})$ deterministic gossiping algorithm for radio networks. *Algorithmica, May 2003*, pages 93–96, 2003.