Wireless Broadcast with optimal labelling

1st Gewu BU  
Sorbonne University, UPMC, LIP6, CNRS UMR 7606  
Paris, French  
gewu.bu@lip6.fr

2nd Maria POTOP-BUTUCARU  
Sorbonne University, UPMC, LIP6, CNRS UMR 7606  
Paris, French  
maria.potop-butucaru@lip6.fr

Abstract—In this paper we study the broadcast problem in wireless networks when the broadcast is helped by a labelling scheme. Our contribution is twofold. First, we propose label optimal broadcast algorithms in a class of networks issued from our recent studies in wireless body area networks. We refer to these networks (level separable networks). We focus on two variants of broadcast: broadcast without acknowledgement (i.e. the initiate of the broadcast is not notified at the end of broadcast) and broadcast with acknowledgement. We propose an acknowledgement-free broadcast strategy using 1-bit labelling and broadcast with acknowledgement using a 2-bit labelling. In the class of level-separable networks our algorithms are memory optimal and finish within 2D rounds for both broadcast with and without acknowledgement, where D is the eccentricity of the broadcast initiator. In the broadcast with acknowledgement, we trigger the acknowledgement message during the broadcast propagation such that no extra rounds are needed for sending the acknowledgement back to the source once the broadcast is finished. Second, we improve a recent [1] labelling-based broadcast scheme with acknowledgement designed for arbitrary networks in terms of memory and time complexity. That is, we better exploit the encoding of labels in order to not use extra memory to carry back the acknowledgement to the source. Nodes store only a 3-bits labelling.

Index Terms—Labelling Scheme, Broadcast, Wireless Networks

I. INTRODUCTION

Broadcast is the most studied communication primitive in networks and distributed systems. Broadcast primitive ensures that once a source node (a.k.a. the broadcast initiator) sends message to all other nodes in the network should receive this message in a finite time. Limited by the transmission range, messages may not be able to be sent directly from one node to some other arbitrary node in the network. Therefore relay nodes need to assist the source node during the message propagation by re-propagating it. Deterministic centralized broadcast, where nodes have complete network knowledge has been studied by Kowalski et al. in [2]. The authors propose an optimal solution broadcast that completes within O(Dlog^2n) rounds, where n is the number of nodes in network and D is the largest distance from the source to any node of the network. The time lower bound for broadcast, Ω(log^2n), has been proved in [3] for a family of radius-2 networks. For deterministic distributed broadcast, assuming that nodes only know their IDs (i.e. they do not know the IDs of their neighbors nor the network topology), [4] propose the fastest broadcast within O(nlogDloglogD) rounds, where D is the diameter of network. The lower bound in this case proposed in [5] is Ω(nlogD), where D is the largest distance from the source to any node of the network.

In wireless networks, when a message is sent from a node it goes into the wireless channel in the form of a wireless signal which can be received by all the nodes within the transmission range of the sender. However, when a node is located in the range of more than one node that send messages simultaneously the multiple wireless signals may generate collisions at the receiver. The receiver cannot decode any useful information from the superimposed interference signals. At the MAC layer several solutions have been proposed in the last two decades in order to reduce the collisions. All of them offer probabilistic guarantees. Our study follows the recent work that addresses this problem at the application layer. More specifically, we are interested in deterministic solutions for broadcasting messages based on the use of extra information or advise (also referred as labelling) precomputed before the broadcast is invoked.

Labelling schemes have been designed to compute network size, the father-son relationship and the geographic distance between arbitrary nodes in the network (e.g. [6], [7] and [8]). Labelling schemes have been also used in [9] and [10] in order to improve the efficiency of Minimum Spanning Tree or Leader Election algorithms. Furthermore, [11] and [12] exploit labelling in order to improve the existing solutions for network exploration by a robot/agent moving in the network. Very few works (e.g. [13] and [1] exploit labelling schemes to efficiently design broadcast primitives. When using labelling schemes nodes record less information than in the case of centralized broadcast, where nodes need to know complete network information. Compared with the existing solutions for deterministic distributed broadcast the time complexity is improved. In [13] the authors prove that for an arbitrary network to archive broadcast within constant number of rounds a O(n) bits of advice is sufficient but not o(n). Very recently, a labelling scheme with 2-bits advice (3 bits for broadcast with acknowledgement) is proposed in [1]. The authors prove that their algorithms need 2n – 3 rounds for the broadcast without acknowledgement and 3n – 4 rounds for broadcast with acknowledgement in arbitrary network.

Contribution: Our work is in the line of research described in [13] and [1]. We first introduce a new family of networks, called level-separable networks issued from our recent studies in wireless body area networks [14], [15], [16], [17] and [18]. We then propose an acknowledgement-free broadcast strategy using 1-bit labelling and a broadcast scheme with
acknowledgement using a 2-bit labelling. In the class of level-separable networks our algorithms are memory optimal and finish within $2D$ rounds for both types of broadcast primitives, where $D$ is the eccentricity of the broadcast source. Second, we address the arbitrary networks and improve the broadcast scheme with acknowledgement proposed in [1] in terms of memory and time complexity by efficiently exploiting the 3-bits labelling encoding. Our solution, use no local persistent variables except the 3-bits labelling.

II. Model and Problem Definition

A. Communication Model

We model the network as a graph $G = (V, E)$ where $V$, the set of vertices, represents the set of nodes in the network and $E$, the set of edges, is a set of unordered pairs $e = (u, v)$, $u, v \in V$, that represents the communications links between nodes $u$ and $v$. In the following $d(u)$ denotes the set of neighbors of node $u$.

We target wireless networks where due to the limitation of the transmission power, a node may not have connections with the other nodes in the network (i.e., $|d(u)| \leq |V| - 1$). However, we assume that the network is connected, i.e., there is a path between any two nodes in the network.

We assume that nodes execute the same algorithm and are time synchronized. The system execution is decomposed in rounds. When a node $u$ sends a message at round $x$, all nodes in $d(u)$ receive the message at the end of round $x$. Collisions occur at node $u$ in round $x$ if a set of nodes, $M \subseteq d(u)$ and $|M| > 1$, send a message in round $(x - 1)$. In that case is considered that $u$ has not received any message successfully.

In the following we are interested in solving the Broadcast problem: when a source node sends a message it should be received by all the nodes in the network in finite bounded time.

B. Level-Separable Network

In this section, we present a family of networks we called Level-Separable Network. We say an arbitrary network is a Level-Separable Network if the underlay communication graph $G = (V, E)$ of the network verifies the Level-Separable propriety defined below.

To define the Level-Separable propriety, we introduce some preliminary notations.

Let $G(V, E)$ be a network and let $s$, a predefined vertex $s \in V$, be the source node of the broadcast. Each vertex $u \in V$ has a geometric distance with respect to $s$ denoted $d(s, u)$. $\varepsilon_G(s)$ is the eccentricity of vertex $s$, that is the farthest distance from $s$ to any other vertex. In the rest of the paper we denote $\varepsilon_G(s)$ by $D$.

Definition 1 (Level). Let $G(V, E)$ be a network. For any vertex $u$ in $G(V, E)$, where $s$ is the source node, the level of the node $u$ is

$$l(u) = d(s, u)$$

i.e., the level of $u$ is its geometric distance to $s$. Let

$$S_i = \{u \mid u \in V, l(u) = i\}$$

denote the set containing all the vertices at level $i$.

Definition 2 (Parents and Sons). Let $G(V, E)$ be a network. A vertex $u$ is parent of vertex $v$ (a vertex $v$ is son of vertex $u$) in graph $G$ with the root source node $s$: if

$$l(v) - l(u) = 1 \land \{u, v\} \in E$$

Let $S(u)$ ($P(v)$) be the set of sons (parents) of $u$ ($v$). If $v \in S(u)$ ($u \in P(v)$), we say that $u$ ($v$) has $v$ ($u$) as son (parent).

Level-Separable propriety focus on how to separate nodes in the same level $i$ into two disjoint subsets of $S_i$. We therefore define Level-Separable Subsets below.

Definition 3 (Level-Separable Subsets). Given $G(V, E)$ a network and the set $S_i$ (the set of all vertices in the same level $i$ of $G$), the level-separable subsets of $S_i$ are $S_{i,1}$ and $S_{i,2}$, such that

$$S_{i,1} \cap S_{i,2} = \emptyset, S_{i,1} \cup S_{i,2} = S_i$$

There may be many possible pairs of $S_{i,1}$ and $S_{i,2}$ for a level $i$. Let $T_i$ be the set of all possible pairs of Level-Separable Subsets:

$$T_i = \{(S_{i,1}^{(1)}, S_{i,2}^{(1)}), (S_{i,1}^{(2)}, S_{i,2}^{(2)}), ..., (S_{i,1}^{(2^i-1)}, S_{i,2}^{(2^i-1)})\}$$

where $(m)$ represent the index of pairs in $T_i$.

Definition 4 (Multi Parents Set). Let $G(V, E)$ be network and let $l_1 > 0$ and $l_2 > 0$ two successive levels in $G(V, E)$ ($l_2 - l_1 = 1$) and let the sets $S_{l_1}, S_{l_2}$ be such that $S_{l_1}$ contains all parents of all vertices in $S_{l_2}$, and $S_{l_2}$ contains all sons of all vertices in $S_{l_1}$. The Multi Parents Set $F_{l_1}$ at level $l_2$ is a subset of $S_{l_2}$. It contains all the vertices at level $l_2$ who have more than one parents in level $l_1$. For any level $i > 0$, we define $F_i$ as:

$$F_i = \{u \mid u \in S_i, l(u) = i \land |P(u)| > 1\}$$

Definition 5 (Level-Separable Propriety). Given an arbitrary graph $G(V, E)$, for all level $i \in [1, D - 1]$, where $D$ is the eccentricity of source node, $G$ verifies the Level-Separable propriety, if there exists pairs for every $T_i$ (the set of all possible pairs of Level-Separable Subsets at level $i$), $(S_{i,1}^{(k)}, S_{i,2}^{(k)})$, such that:

$$|P(u) \cap S_{i,1}^{(k)}| = 1, \forall u \in F_{i+1}$$

i.e., for every vertex $u$ at level $i + 1$ having multi-parents at level $i$, $u$ has only one parent in $S_{i,1}$.

Note that if $F_{i+1} = \emptyset$, then $S_{i,1} = \emptyset$. When $S_{i,1}$ is fixed, $S_{i,2}$ is $S_i \setminus S_{i,1}$.

Definition 6 (Level-Separable Network). A network $G(V, E)$ is a Level-Separable Network, if its underlay graph verifies the Level-Separable propriety.

Note that Level-Separable Graph has similar flavor with Bipartite Graph [19]. A graph $G = (V, E)$ is said to be Bipartite if and only if there exists a partition $V = A \cup B$ and $A \cap B = \emptyset$. So that all edges share a vertex from both sets $A$.
and \( B \), and there is no edge containing two vertices in the same set. A bipartite graph separates nodes into two independent sets. In a level-separable network we aim at separating nodes of the same level. Moreover, we are interested in the relation between the two separated sets at level \( i \) and nodes in level \( i + 1 \), i.e., node’s father-son relationship.

Note that a level-separable network is not necessarily a tree network. However a tree is a level-separable network. A simple example of level-separable network is a tree network, where the root of the tree is the source node \( s \) who begins the broadcast. In a tree topology all non-source nodes have only one parent, i.e. \( \forall u \in V - s, \ |P(u)| = 1 \), so that in each level, the \( F_i = \emptyset \). So that all \( S_{i,1} = \emptyset \) and \( S_{i,2} = S_i \setminus S_{i,1} = S_i \).

Figure 1 shows an example of a level-separable network that is not a tree. In this network, 16 nodes are connected: one source node (i.e. the node that starts the broadcast) and 15 non-source nodes. Note that this network is not a tree: nodes may have more than one parent (e.g., node 12 has two parents: node 5 and node 6). This network is represented by levels for easy of the observation. For any level \( i > 0 \) all three nodes at that level can be separated into two level-separable sets: \( S_{i,1} = \{2\} \) and \( S_{i,2} = \{1, 3\} \). That is true because the Multi Parents Set \( F_2 = \{6\} \) and the parents set of node 6 is \( P(6) = \{2, 4\} \). Therefore \( |P(u) \cap S_{i,1}| = 1, \forall u \in F_2 \) holds. According to Definition 5, \( S_{i,1} = \{2\} \) and \( S_{i,2} = \{1, 3\} \) verify the level-separable propriety. From the same reason, at level 2, \( S_{2,1} = \{5, 8\} \) and \( S_{2,2} = \{4, 6, 7\} \) also verify the level-separable propriety.

Studies conducted recently in wireless body area networks [14], [15], [16], [17] and [18] show that various postural mobilities can be model as graphs that fit our definition of level-separable network.

In [15], authors studied the cross-layer broadcast in wireless body area network and model the network as graphs for different human postures. In this case each graph is a level-separable network, see Figure 2.

In the next section we propose a broadcast algorithm without acknowledgement with optimal labelling in separable networks. Then, we improve the broadcast algorithm proposed in [1]. Finally, we propose a solution for broadcast with acknowledgement using only 2 bits in level-separable networks.

III. BROADCAST IN LEVEL-SEPARABLE NETWORK

In this section we propose a 1-bit constant-length labelling broadcast algorithm \( \beta^{LS} \) detailed in Algorithm 1. The algorithm needs \( 2D \) rounds, where \( D \) is the eccentricity of the broadcast source node.

A. Broadcast with 1-bit Labelling

Given a level-separable network whose root is the source of the broadcast, we propose Algorithm \( \beta^{LS} \) (shown as Algorithm 1) to archive the wireless broadcast, when a 1-bit labelling scheme \( \lambda^{LS} \) is used. Each node in the network has a 1-bit label, \( X_1 \). \( X_1 \) is set to 1 or 0 following the labelling scheme \( \lambda^{LS} \) described below. The idea of the broadcast algorithm is to separate nodes at each level into two independent sets. Nodes in the first set transmit at round \( x \) and nodes in the second set transmit at round \( x + 1 \) (the next round), so that they will not generate valid collisions. Note that collisions that occur at a node who has already received the message successfully are not considered valid collisions. The broadcast Algorithm \( \beta^{LS} \) using the labelling scheme \( \lambda^{LS} \) is as follows: the source node sends the message, \( \mu \), at round 0. Nodes at level 1 receive \( \mu \) at the end of round 0.

When nodes with \( X_1 = 1 \) receive \( \mu \) at round \( 2i - 3 \) or \( 2i - 2 (i > 1) \), they send \( \mu \) at round \( 2i - 1 \). When nodes with \( X_1 = 0 \) receive \( \mu \) at round \( 2i - 3 \) or \( 2i - 2 \), they send \( \mu \) at round \( 2i \). That is, nodes at level \( i > 1 \) will receive \( \mu \) from their parents (nodes at level \( i - 1 \)) at round \( 2i - 3 \) or \( 2i - 2 \), and they will send \( \mu \) at round \( 2i \) or \( 2i - 1 \). In other words, at each level \( i \), nodes take two rounds to propagate \( \mu \) to all nodes at level \( i + 1 \).

Figure 3 presents the propagation of the message. The left side shows the level of a level-separable network, from level 0 to 2. It shows three rounds during the execution from 0 to 2. The right side shows that at which round nodes at a level receive (denoted \( R \)) or transmit (denoted \( T \)) a message. At round 0, source \( s \) sends message \( \mu \) to all the nodes at level 1. Nodes at level 1 have been already separated into two sets, blue ones and white ones by the labelling scheme \( \lambda^{LS} \). At
round 1, nodes in the white set send $\mu$, and two nodes at level 2 receive the message. At round 2, the nodes in the blue set send $\mu$ and the remaining nodes at level 2 receive the message.

1-bit Labelling Scheme $\lambda^{LS}$. To archive collision free transmission, 1-bit Labelling Scheme $\lambda^{LS} X_1$ of all nodes in $S_{i,1}$ for level $i > 0$ is 1, and $X_1$ of all nodes in $S_{i,2}$ for level $i > 0$ is 0 where $S_{i,1}$ and $S_{i,2}$ are the sets identified in Definition 5.

**B. Correctness and Complexity of Algorithm $\beta^{LS}$**

In the following we prove that Algorithm $\beta^{LS}$ is correct. First we show that the previously described scheme when used by Algorithm $\beta^{LS}$ do not generate collisions.

**Lemma 1.** Let $G = (V, E)$ be a level-separable network such that each node has a label according to the labelling scheme $\lambda^{LS}$. If nodes with $X_1 = 1$ at the same level $i \in [1, D - 1]$, where $D$ is the eccentricity of the source node, send a message concurrently they do not generate collisions at nodes level $i + 1$. If nodes with $X_1 = 0$ at the same level $i \in [1, D - 1]$, send message concurrently they do not generate collisions at nodes having only one parent at level $i + 1$.

**Proof.** At level $i$, nodes with $X_1 = 1$ are the nodes in the subset $S_{i,1}$. According to Definition 5, each node in level $i + 1$ has at most one parent in $S_{i,1}$. Therefore, when nodes in $S_{i,1}$ send a message, none of nodes in level $i + 1$ will receive more than one message. So sending of nodes holding $X_1 = 1$ will not generate any collision at level $i + 1$. Nodes with $X_1 = 0$ are nodes in $S_{i,2}$. As $S_{i,1}$ contains at least one parent for all the nodes at level $i + 1$ having multi-parents, $S_{i,2}$ contains therefore all parents of each node at level $i + 1$ having only one parent. That is, when nodes with $X_1 = 0$ send, all nodes having only one parent can receive the message without collisions.

**Note 1.** Note that 1-bit labelling scheme is optimal for broadcast in a level-separable network. That is, with 0-bit labelling (i.e. without using any labelling) it is possible that some node in the network does not receive the broadcasted message due to the collisions since nodes are synchronized and transmit in the same time.

---

**Algorithm 1 $\beta^{LS}(\mu)$ executed at each node $v$**

Each node has a variable $sourcemsg$. The source node has this variable initially set to $\mu$. All other nodes have it initially set to $null$. A variable $k$ initially set to 0 to ensure each node sends $\mu$ only once.

for each round $r$ from 0 do
  if $v$ is the source node and $r = 0$ then
    transmit $sourcemsg$
  else if $v$ is not source node and receives $\mu$ then
    if $k = 0$ then
      $sourcemsg \leftarrow \mu$
    else if $r$ is odd number then
      transmit $sourcemsg$ at round $r + 3$
    else if $X_1 = 1$ then
      transmit $sourcemsg$ at round $r + 2$
    else if $r$ is even number then
      if $X_1 = 0$ then
        transmit $sourcemsg$ at round $r + 2$
      else if $X_1 = 1$ then
        transmit $sourcemsg$ at round $r + 1$
    set $k = 1$

---

**Theorem 1.** Algorithm $\beta^{LS}$ with 1-bit constant Labelling Scheme $\lambda^{LS}$ implements broadcast in a level-separable network within 2$D$ rounds.

The proof of this theorem is a direct consequence of Lemma 2, 3, and 4.

**Lemma 2.** Given a level-separable network whose root is the source node by applying $\beta^{LS}$ and $\lambda^{LS}$, nodes in level $i > 0$ receive message $\mu$ at round $2i - 2$.

**Proof.** We begin from the base case where $i = 1$, nodes at level $i = 1$ means nodes that are only one hop away from the source node. At round $2 \times 1 - 1 = 0$, the source sends the message. All nodes at level 1 will receive the message at the end of round 0. For $i = 2$, as nodes at level 1 can receive message at round 0, they will begin to send at round 1 and round 2 for nodes in $S_{i,1}$ and $S_{i,2}$, respectively. According to Lemma 1, no collision occurs at level 2. Therefore all the nodes in level 2 can receive the message at round $2 \times 2 - 2 = 2$. For the general case, we assume that all nodes at level $i$, $i > 2$, receive the message at round $2i - 2$. So that nodes begin to send the received message at round $2i - 2 + 1 = 2(i + 1) - 1$ and $2i - 2 + 2 = 2(i + 1) - 2$, and nodes at level $i + 1$ receive the message at $2(i + 1) - 1$ and $2(i + 1) - 2$.

**Lemma 3.** Given a level-separable network whose root is the source node by applying $\beta^{LS}$ and $\lambda^{LS}$, the broadcast finishes in 2$D$ rounds.

**Proof.** From Lemma 2, nodes having the longest distance to the source will receive the message at round $2D - 2$, where $D$ is the source eccentricity. After receiving the message, these nodes will send it according to the broadcast algorithm, even though they are already the ending nodes in the network which takes two more rounds. So the broadcast finishes at round $2D$.

**Lemma 4.** Given a level-separable network whose root is the source node by applying $\beta^{LS}$ and $\lambda^{LS}$, the algorithm finishes within 2$D$ rounds.

**Proof.** According to Lemma 3, the broadcast finishes at round $2D$. Therefore, our algorithm terminates at round $2D$.

The idea of the proof is as follow. Consider the execution of the Algorithm $\beta^{LS}$ in a level-separable network with labelling.
scheme $\lambda_{LS}$, where nodes in level $i$ have been separated into two sets $S_{i,1}$ and $S_{i,2}$ verifying level-separable property at level $i$. Nodes in $S_{i,1}$ have $X_i = 1$, and nodes in $S_{i,2}$ have $X_i = 0$. The main idea of $\beta_{LS}$ is that, nodes in each level $i$ separated into two different sets transmit their received messages $\mu$ in different execution rounds to reduce the collisions impact at nodes in level $i + 1$.

According to Algorithm $\beta_{LS}$, the message $\mu$ will be propagated from level to level. Each propagation from a level to the next one takes two execution rounds. In the first round all nodes in $S_{i,1}$ send the received message $\mu$. At the end of this round all the nodes that are the sons of nodes in $S_{i,1}$ receive $\mu$, without collision, see Lemma 1. As sons of nodes in $S_{i,1}$ contain all the nodes at level $i + 1$ who have multi-parents, that means it remains only nodes at level $i + 1$ who have only one parent that haven’t received message $\mu$ yet. In the second round, all nodes in $S_{i,2}$ send $\mu$, and the remaining part of the nodes at level $i + 1$ can therefore receive $\mu$ from their unique parent. So that after these two rounds of transmission from level $i$, all the nodes at $i + 1$ can successfully receive the message $\mu$. It takes therefore $2D$ rounds to finish the broadcast. Note that nodes will only send once according to $\beta_{LS}$. Therefore the algorithm terminates.

C. Labeling Preinstall

In this section we propose a strategy to select $S_{i,1}$ for each level $i$ in a level-separable network. Note that this strategy is executed off line before the execution of the broadcast algorithm. Given two arbitrary successive levels, $i$ and $i + 1$, let $F_{i+1}$ be the set of all nodes at level $i + 1$ that have multi parents at level $i$ (see Definition 2). Let $SF$ be the set of all fathers of nodes in $F_{i+1}$, such that:

$$SF = \bigcup_{u \in F_{i+1}} P(u)$$

The main idea to select $S_{i,1}$ is to select from the Power Set of $SF$, i.e., the set of all the subset of $SF$. The set $S_{i,1}$ we should chose from the power set of $SF$ should verify: 1) nodes in $S_{i,1}$ do not have the same son nodes; 2) nodes in $S_{i,1}$ contain all parents of nodes in $F_{i+1}$. Assume that the mean number of nodes in each level is $x$, then in each level, we need to chose $S_{i,1}$ from at most $\sum_{i=0}^{x} C_x^i = 2^x - 1$ possible choices. The offline time complexity of choosing $S_{i,1}$ at each level is $O(2^x - 1)$.

IV. BROADCAST WITH ACK FOR ARBITRARY NETWORKS

In Section 2, the authors propose a broadcast with ACK algorithm using a 3-bits labelling scheme. At the end of the broadcasting, the last informed node generates and sends back to the source node an ACK message. In a 3-bits labelling, there are 8 states: 000, 001, 010, 011, 100, 101, 110 and 111 available. The algorithm in [1] uses only 5 of them: 000, 001, 010, 100 and 110. In this section, we propose an optimal labelling scheme, $\lambda_{oACK}$ and a broadcast with ACK that uses all the 8 states of the 3-bits labelling in order to improve the memory complexity of the solution proposed in [1].

The optimization of our solution is as follows: instated of only using the last bit $X_3$ (the third bit) as a mark to point who is (one of) the last informed node(s) during the broadcast, we use also this third bit to show a path back to the source node $s$ from the last informed node. Differently from the solution proposed in [1] nodes do not need to keep additional variables in order to send back to the source the $\beta_{ACK}$ during the execution. Our proposition can therefore save node’s memory and computational power. In the following we present our $\lambda_{oACK}$ labelling scheme.

A. 3-bit Labelling Scheme $\lambda_{oACK}$

The first two bits of the labelling scheme $X_1$ and $X_2$ have the same function as in the $\lambda_{ACK}$ scheme of [1] (see [1] for more details and proof). The intuitive idea is as follows: 1) $X_1 = 1$ for nodes who should propagate the message when they receive it; 2) $X_2 = 1$ for nodes that need to send stay message back to their parent to continue send the message in the next round; 3) $X_3 = 1$ for one of the last receiving node to generate ACK and send it back to the source node. In our scheme $\lambda_{oACK}$ we set $X_3$ (the third bit) to $1$ for all nodes on the path back from the last informed node (who holds 001) to the source node. Note that, nodes on that path could have four kinds of different labels: 101, 011, 111 and 001, where 001 is the label of the last informed node. As the three states 101, 011 and 111 have not been used in the original $\beta_{ACK}$, nodes can easily recognize if they are on the path to transmit the ACK message back to the source node.

B. Broadcast Algorithm $\beta_{oACK}$

Our broadcast algorithm $\beta_{oACK}$ that uses the $\lambda_{oACK}$ is described in Algorithm 2.

Given an arbitrary network applying the labelling scheme $\lambda_{oACK}$ execute $\beta_{oACK}$. Nodes with $X_1 = 1$ receiving a message at round $i - 1$ send it at round $i$. Then nodes who sent at round $i$ wait the stay message, at round $i + 1$, from other nodes with $X_2 = 1$. If nodes who sent at round $i$ receive stay at round $i + 1$, then they continue to send at round $i + 2$. Otherwise, they will stay silent. When nodes with label 001 receive the message, they generate the $ACK$ message and send it. Since $\lambda_{oACK}$ already marked the path back from this node.
to the source node, in Algorithm $\beta_{ACK}$, the $ACK$ message will only be re-propagated by nodes with $X_3 = 1$, i.e., node with label 101, 111 and 011.

Note that our proposed Algorithm $\beta_{ACK}$ does not need additional variables to reconstruct the path back to the source during the broadcast execution. In Algorithm $\beta_{ACK}$, two additional variables $informedRound$ (type int) and $transmitRounds$ (type table of int) are needed to rebuild the back-way path. $informedRound$ is used to record the round number in which a node received $\mu$; $transmitRounds$ is a table used to record all the round numbers in which one node transmits $\mu$. However, by using $\beta_{ACK}$, the $ACK$ message transfer processing can be completed only by checking the third bit, $X_3$. Our Algorithm $\beta_{ACK}$ does not need any extra local storage for directing the $ACK$ message.

C. Labeling Preinstall

In the following we propose a strategy to decide the back-way path in arbitrary network. According to the idea of $\lambda_{ACK}$ in [1], the last informed node, the 001 node, can be detected easily. If $v$ is the last informed node, let $u = P_r(v)$ is the father node of $v$ from whom $v$ received $\mu$. Since the computation is done offline, the $P_r(u)$ of any node $u$ (if it exists) can always be computed offline. The members of the back-way path belong to the set:

$$Bp = \{u, P_r(u), P_r(P_r(u)), ..., s\}$$

where $u$ is the last informed node and $s$ is the source node. To mark the back-way path, $\lambda_{ACK}$ sets the $X_3$ bit of the labels of all nodes in $Bp$ to 1.

Note that we do not change the main architecture of $\beta_{ACK}$ algorithm in [1] therefore the correctness proof of our algorithm is very similar to the one in [1].

V. BROADCAST WITH ACK IN LEVEL-SEPARABLE NETWORK

In this section, we combine the Broadcast algorithm $\beta_{ACK}$ and the labelling scheme $\lambda_{ACK}$ to propose an algorithm of broadcast with $ACK$, $\beta_{ACK}$, and the Labelling Scheme, $\lambda_{ACK}$, for level-separable networks. Our algorithm $\beta_{ACK}$ (Algorithm 3) uses only 2-bits labelling and the broadcast finishes within 2D rounds. In our solution $ACK$ goes back to the source node at rounds $2(D - 1)$ or $2D$, where $D$ is the eccentricity of $s$ (the broadcast source node).

A. 2-bit Labelling Broadcast with ACK

According to Theorem 1 the broadcast finishes in a level-separable network within 2D rounds where $D$ is the eccentricity of the source node. If the source node has the knowledge of $D$, then it automatically can decide if the broadcast is finished. However, when an $ACK$ message is necessary to inform the source node to trigger some additional functions then the source waits for the reception of this message. In order to avoid that the $ACK$ message takes addition time after the end of the broadcast, we propose to send in advance the $ACK$ message at the halfway of the transmission during the broadcast execution. Since in a level-separable network, informing nodes from level to level takes exactly 2 rounds, then $ACK$ also takes 2 rounds to inform one level above. Therefore, when the last node receives $\mu$, the source node receives the $ACK$ message at the same round. Interestingly, compared with non-ACK broadcasting, our solution uses one extra bit for labelling and no additional rounds for forwarding back to the source the $ACK$ message.

Figure 4 gives the intuition of how to send in advance the $ACK$ message: the half-way $ACK$ mechanism. In Figure 4 the network is represent in abstract levels to simplify the problem. Packets flow shown in the figure represent the propagation of messages $\mu$ and $ACK$.

B. 2-bit Labelling Scheme $\lambda_{ACK}$

We use $\lambda_{ACK}$ to set $X_2$ in $\lambda_{ACK}$ in order to verify Lemma 1. Let $X_2$ be the second bit of the $\lambda_{ACK}$ labelling scheme.

Note 2. Note that 2-bit labelling scheme is optimal to archive broadcast with acknowledgement in a level-separable network. From Note 1 1-bit labelling for broadcast without acknowledgement is optimal. When an acknowledgement has to be sent back to the source node, at least one additional bit is necessary to indicate who should be the node to generate the acknowledgement message and send it back to the source node. Without this additional bit no node can decide (unless it uses extra local memory) if it is the last receiving node, and who should send back the $ACK$.

C. Correctness and complexity of Algorithm $\beta_{ACK}$

Theorem 2 below proves the correctness of Algorithm $\beta_{ACK}$.

**Theorem 2.** Algorithm $\beta_{ACK}$ with 2-bit labelling scheme $\lambda_{ACK}$ implements broadcast in a level-separable network. The broadcast terminates in 2D rounds. The $ACK$ message is transmitted back to the source at round 2($D - 1$), if $D$ is odd or 2$D$, if $D$ is even.
The proof of the theorem is the direct consequence of Lemma 5 and 6 below.

**Lemma 5.** Given a level-separable network whose root is the source node by applying $\beta_{ACK}^{LS}$ and $\lambda_{ACK}^{LS}$, nodes in level $i > 0$ receive message $\mu$ at round $2i - 2$.

**Lemma 6.** Given a level-separable network whose root is the source node by applying $\beta_{ACK}^{LS}$ and $\lambda_{ACK}^{LS}$, the broadcast finishes at round $2D$.

Proofs for Lemma 5 and 6

**Proof.** $\beta_{ACK}^{LS}$ follows the same idea as $\beta^{LS}$. The additional ACK transmission will not have any impact so according to Lemma 2 and 3, Lemma 5 and 6 are proved.

**Lemma 7.** Given a level-separable network whose root is the source node by applying $\beta_{ACK}^{LS}$ and $\lambda_{ACK}^{LS}$, the ACK message goes back to source node at round $2(D - 1)$, if $D$ is odd; or $2D$, if $D$ is even.

**Proof.** When $D$ is odd, ACK and the message will begin to be sent to source and to the ending nodes from levels $l_{ACK}$ and $l_{MSG}$, respectively. The distances from levels $l_{ACK}$ back to source is the same with that from $l_{MSG}$ to the ending nodes. $ACK$ arrives to the source at the same round as the broadcasted message arrives at the ending nodes. According to Lemma 5 this is round $2(D - 1)$. When $D$ is even $ACK$ needs to go no level farther compared with the broadcasted message. Therefore, it takes two extra rounds when $D$ is even. Therefore, when $D$ is even the ACK message goes back to source node in $2D$ rounds.

**Lemma 8.** Given a Level-Separable Network whose root is the source node by applying $\beta_{ACK}^{LS}$ and $\lambda_{ACK}^{LS}$, the algorithm finishes within $2D$ rounds

**Proof.** According to Lemma 7, the ACK message takes at most $2D$ rounds to go back to the source, which is the same from the ending of broadcasting from Lemma 6. Algorithm $\beta_{ACK}^{LS}$ and $\lambda_{ACK}^{LS}$ terminates at round $2D$.

The idea of the correctness proof is as follows. Consider a level-separable network with the labelling scheme $\lambda_{ACK}^{LS}$, where all nodes in level $i$ have been separated into two sets $S_{i,1}$ and $S_{i,2}$. Nodes in $S_{i,1}$ have $X_i = 1$, and nodes in $S_{i,2}$ have $X_i = 0$. A way back path is marked with $X_2 = 1$ between source $s$ and an arbitrary node at level $\lfloor D/2 \rfloor - 1$, where $D$ is the eccentricity of $s$, i.e., we only mark the way back path from the half-way level $\lfloor D/2 \rfloor - 1$ of the network in this case.

The idea is that when the message $\mu$ propagates to the half-way level of the network, a node at that level will begin the $ACK$ transmission processing, so that when the $\mu$ reaches to the ending node(s) at level $D$, the $ACK$ message reaches the source $s$ at (almost) the same round. As nodes cannot decide if they are the ones at the half-way of network who should generate and send $ACK$ message, we use a Waiting Period and an extra $pACK$ message.

Algorithm 3 $\beta_{ACK}^{LS}(\mu)$ executed at each node $v$

Each node has a variable $sourceMsg$. The source node has this variable initially set to $\mu$. All other nodes have it initially set to $null$. A variable $k$ and $k_{ack}$ is initially set to 0 to ensure each node sends $\mu$ only once.

for each round $r$ from 0 do
  if $v$ is source node and $r = 0$ then
    transmit $sourceMsg$
  else if $v$ is not source node and receive $\mu$ then
    sourceMsg ← $\mu$
  else if $r$ is even number then
    if $X_1 = 0$ then
      transmit $sourceMsg$ at round $r + 3$
    else if $X_1 = 1$ then
      transmit $pACK$ at round $r + 4$
    else if $v$ doesn't receive $pACK$ at $r + 5$ then
      transmit $ACK$ at round $r + 6$, set $k_{ack} = 1$
  else if $r$ is odd number then
    if $X_1 = 0$ then
      transmit $sourceMsg$ at round $r + 2$
    else if $X_1 = 1$ then
      transmit $pACK$ at round $r + 3$
    else if $v$ doesn't receive $pACK$ at $r + 5$ then
      transmit $ACK$ at round $r + 6$, set $k_{ack} = 1$
  set $k_{ack} = 1$

According the $\beta_{ACK}^{LS}$, when a node with $X_2 = 1$, receives $\mu$ and finishes the $\mu$ retransmission, it cannot decide its position in the way back path. Therefore, it sends a $pACK$ message and begins to wait $pACK$ message sent to him in the following rounds. When a node with $X_2 = 1$ receives a $pACK$ within the WaitingPeriod, that means it is not the ending node, because there is another node with $X_2 = 1$ that received $\mu$ and sent $pACK$ to him. When a node with $X_2 = 1$ does not receive any $pACK$ within its WaitingPeriod, this means no node in the next level has $X_2 = 1$, i.e., it is the half-way ending node, so it generates and sends the $ACK$ message. All the nodes with $X_2 = 1$ will forward ACK message from the ending node to the source $s$ according to the marked way back path. In the $\beta_{ACK}^{LS}$, the WaitingPeriod is delayed two rounds after a node sends $pACK$ message to avoid the collision between $pACK/ACK$ and $\mu$.

A node with $X_2 = 1$ that receives $\mu$ at round $x$, transmits $\mu$ at round $x + 2$, then it sends $pACK$ to its parents at round $x + 4$, then it waits a Waiting Period until round $x + 6$. If it doesn’t receive another $pACK$, then it sends $ACK$ at round $x + 8$. That means, for the half-way ending node, it needs to wait 6 rounds to begin sending $ACK$. What we want for this half-way mechanism is that the source node can receive $ACK$ as fast as possible, after the broadcast finishes. When $D$ (the eccentricity of the broadcast source $s$) is odd, then if we chose the node at level $\lfloor D/2 \rfloor - 1$ as the half-way ending node, then the $ACK$ can be received by source node at the same round as the end of the broadcast. Because after waiting 6 rounds at level $\lfloor D/2 \rfloor - 1$, message $\mu$ has already been transmitted to level $\lfloor D/2 \rfloor - 1 + 3 = \lfloor D/2 \rfloor + 2$. The distance from node sending $ACK$ to source node is $d(s, \lfloor D/2 \rfloor - 1) = \lfloor D/2 \rfloor - 1$; the distance from node sending $\mu$ to nodes at level $D$ is also
Then, we improved in terms of memory and time complexity to carry back the acknowledgement to the source. Our improvement better exploits the encoding of the labels in order to not use extra memory. Therefore the algorithm terminates.

D. Labeling Preinstall

In the following we propose a strategy to decide the backward path in a level-separable network from the halfway during the broadcast propagation. Similar to Section V-C instead of choosing the last informed node as the generator of ACK message, we chose in level \( \lfloor D/2 \rfloor - 1 \) a node \( u \) and build the set \( B_p = \{ u, P_r(u), P_r(P_r(u)), ..., s \} \) from \( u \) to \( s \). To mark the way back path, one needs only to set \( X_2 \) of all nodes in \( B_p \) to 1.

When using a 3-bits label instead of 2-bits, the last informed node can be marked directly by the labelling scheme using the third bit. That means that during the broadcast execution any Waiting Period or pACK message is unnecessary, so that during the execution of \( \beta_{ACK}^{LS} \), we can save the unnecessary pACK message transmission.

VI. CONCLUSION

We proposed solutions for implementing broadcast in wireless networks when the broadcast is helped by a labelling scheme. We studied broadcast without acknowledgement (i.e. the initiator of the broadcast is not notified at the end of broadcast) and broadcast with acknowledgement. We propose an optimal acknowledgement-free broadcast strategy using 1-bit labelling and a broadcast with acknowledgement using a 2-bit labelling in level-separable networks. The complexity of both algorithms is \( 2D \) where \( D \) is the eccentricity of the broadcast initiator. They use an optimal number of bits. Then, we improved in terms of memory and time complexity the labelling-based broadcast scheme with acknowledgement proposed in [1] for arbitrary networks. Our improvement better exploits the encoding of the labels in order to not use extra memory to carry back the acknowledgement to the source.

REFERENCES

[1] Faith Ellen, Barun Gorain, Avery Miller, and Andrzej Pelc. Constant-length labeling schemes for deterministic radio broadcast. arXiv preprint arXiv:1710.03178, 2017.

[2] Dariusz R Kowalski and Andrzej Pelc. Optimal deterministic broadcasting in known topology radio networks. Distributed Computing, 19(3):185–195, 2007.

[3] Noga Alon, Amotz Bar-Noy, Nathan Linial, and David Peleg. A lower bound for radio broadcast. Journal of Computer and System Sciences, 43(2):290–298, 1991.

[4] Ferdinando Cicalese, Fredrik Manne, and Qin Xin. Faster deterministic communication in radio networks. Algorithmica, 54(2):226–242, 2009.

[5] Andrea EF Clementi, Angelo Monti, and Riccardo Silvestri. Selective families, superimposed codes, and broadcasting on unknown radio networks. In Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms, pages 709–718. Society for Industrial and Applied Mathematics, 2001.

[6] Serge Abiteboul, Haim Kaplan, and Tova Milo. Compact labeling schemes for ancestor queries. In Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms, pages 547–556. Society for Industrial and Applied Mathematics, 2001.

[7] Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. Journal of Algorithms, 53(1):85–112, 2004.

[8] Barun Gorain and Andrzej Pelc. Finding the size of a radio network with short labels. In Proceedings of the 19th International Conference on Distributed Computing and Networking, page 10. ACM, 2018.

[9] Pierre Fraigniaud, Amos Korman, and Emmanuelle Lebhar. Local mst computation with short advice. Theory of Computing Systems, 47(4):920–933, 2010.

[10] Christian Glacet, Avery Miller, and Andrzej Pelc. Time vs. information tradeoffs for leader election in anonymous trees. ACM Transactions on Algorithms (TALG), 13(3):31, 2017.

[11] Reuven Cohen, Pierre Fraigniaud, David Ilcinkas, Amos Korman, and David Peleg. Label-guided graph exploration by a finite automaton. ACM Transactions on Algorithms (TALG), 4(4):42, 2008.

[12] Pierre Fraigniaud, David Ilcinkas, and Andrzej Pelc. Tree exploration with advice. Information and Computation, 206(11):1276–1287, 2008.

[13] David Ilcinkas, Dariusz R Kowalski, and Andrzej Pelc. Fast radio broadcasting with advice. Theoretical Computer Science, 411(14-15):1544–1557, 2010.

[14] Wafa Badreddine, Claude Chaudet, Federico Petrucci, and Maria Potop-Butucaru. Broadcast strategies in wireless body area networks. In Proceedings of the 18th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pages 83–90. ACM, 2015.

[15] Wafa Badreddine and Maria Potop-Butucaru. Peak transmission rate resilient crosslayer broadcast for body area networks. arXiv preprint arXiv:1702.05031, 2017.

[16] Gewu Bu and Maria Potop-Butucaru. Total order reliable convergentcast in whan. In Proceedings of the 19th International Conference on Distributed Computing and Networking, page 26. ACM, 2017.

[17] Wafa Badreddine, Nesrine Khemane, Maria Potop-Butucaru, and Claude Chaudet. Convergentcast in wireless body area networks. Ad Hoc Networks, 66:40–51, 2017.

[18] Gewu Bu and Maria Potop-Butucaru. Ban-gzkp: Optimal zero knowledge proof based scheme for wireless body area networks. Ad Hoc Networks, 77:28–41, 2018.

[19] Jonathan L Gross and Jay Yellen. Graph theory and its applications. CRC press, 2005.