



## Wireless Broadcast with short labelling

Gewu Bu, Maria Potop-Butucaru, Mikael Rabie

### ► To cite this version:

Gewu Bu, Maria Potop-Butucaru, Mikael Rabie. Wireless Broadcast with short labelling. 2020. hal-01869563v3

**HAL Id: hal-01869563**

**<https://hal.science/hal-01869563v3>**

Preprint submitted on 26 Jan 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Wireless Broadcast with short labels

Gewu Bu\*, Maria Potop-Butucaru†, Mikaël Rabie‡

Sorbonne University, LIP6, CNRS UMR 7606

Paris, France

Email: \*gewu.bu@lip6.fr, †maria.potop-butucaru@lip6.fr, ‡mikaël.rabie@lip6.fr

**Abstract**—In this paper, we study the broadcast problem in wireless networks when the broadcast is helped by a labelling scheme. We focus on two variants of broadcast: broadcast without acknowledgment (i.e. the initiator of the broadcast is not notified at the end of broadcast) and broadcast with acknowledgment. Our contribution is threefold. *First*, we improve in terms of memory complexity a recent [11] labelling-based broadcast scheme with acknowledgment designed for arbitrary networks. *Second*, we propose label optimal broadcast algorithms in *level separable networks* (a class of networks issued from recent studies in Wireless Body Area Networks). In this class of networks we propose an acknowledgment-free broadcast strategy using 1-bit labels and broadcast with acknowledgment using 2-bits labels. In the class of level-separable networks, our algorithms finish within  $2D$  rounds, where  $D$  is the eccentricity of the broadcast initiator. Interestingly, the time complexity of broadcast in the case of level-separable networks does not depend on the size of the network but rather on the initiator eccentricity which makes this class of graphs interesting for further investigation. *Finally*, we study the hardness of determining that a graph is level separable. Our study shows that even though checking that a separation is a level separation can be done in polynomial time, determining that a graph has the level separable property is NP-complete. This result opens interesting independent research directions.

**Index Terms**—Labelling Scheme, Broadcast, Wireless Networks

## I. INTRODUCTION

*Broadcast* is the most studied communication primitive in networks and distributed systems. Broadcast ensures that once a *source node* (a.k.a. the broadcast initiator) sends a message, all other nodes in the network should receive this message in a finite time. Limited by the transmission range, messages might not be sent directly from one node to some other 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 [22]. The authors propose an optimal solution that completes within  $O(D \log^2 n)$  rounds, where  $n$  is the number of nodes in the network and  $D$  is the largest distance from the source to any node of the network. For deterministic distributed broadcast, assuming that nodes only know their IDs (i.e. they do not know the IDs of their neighbours nor the network topology), in [8] is proposed the fastest broadcast within  $O(n \log D \log \log D)$  rounds, where  $D$  is the diameter of the network. The lower bound in this case, proposed in [9], is  $\Omega(n \log D)$ .

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 may be received by all the nodes within the transmission range of the sender node. However, when a node is located in the range of more than one node that sends 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 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 advice (also referred to as *labelling*) precomputed before the broadcast invocation.

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. [1], [14] and [16]). Labelling schemes have been also used in [13] and [15] in order to improve the efficiency of Minimum Spanning Tree or Leader Election algorithms. Furthermore, [10] 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. [19] and [11]) exploit labelling schemes to design efficient *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 [19] the authors prove that for an arbitrary network, to achieve broadcast within a 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 acknowledgment) is proposed in [11]. The authors prove that their algorithms need  $2n - 3$  rounds for the broadcast without acknowledgment and  $3n - 4$  rounds for broadcast with acknowledgment in an arbitrary network.

**Contribution:** Our work is in the line of research described in [11]. We first improve in terms of memory complexity the broadcast scheme with acknowledgment proposed in [11]. Differently, from the solution proposed in [11], our solution does not use extra local persistent memory except the 3-bits labels. Then, we study labelling-based broadcast in a new family of networks, called level-separable networks issued from Wireless Body Area Networks (e.g. [2], [4], [6], [3] and [7]). In this class of networks we propose an acknowledgment-free broadcast strategy using 1-bit labels and a broadcast

scheme with acknowledgment using 2-bits labels. Our algorithms terminate within  $2D$  rounds for both types of broadcast primitives, where  $D$  is the eccentricity of the broadcast source. Interestingly, the time complexity of broadcast in the case of level separable networks does not directly depend on the network size which makes the study of level separable networks of independent interest. We further investigate the hardness of determining if a graph is level separable. Our study shows that even though checking that a separation is a level separation can be done in polynomial time, determining that a graph has the level separable property is NP-complete. This result opens interesting independent research directions that will be discussed in the conclusion of this document.

## II. MODEL AND PROBLEM DEFINITION

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 neighbours of node  $u$ . 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  $x$ . In that case, it is considered that  $u$  has not received any message.

In the following we are interested in solving the *Broadcast problem*: when a source node  $s$  sends a data message  $\mu$ , this  $\mu$  should be received by all the nodes in the network in a finite bounded time. We are also interested in solving *Broadcast with acknowledgment problem*: once all nodes received  $\mu$ , an acknowledgment message, called *ACK*, will be generated and sent backward to the source node  $s$  in a finite bounded time.

## III. BROADCAST WITH ACK FOR ARBITRARY NETWORKS

In [11] the authors propose a broadcast with acknowledgment algorithm  $\beta_{ACK}$  for general networks using a 3-bits labelling scheme  $\lambda_{ACK}$ . The idea of the broadcast algorithm  $\beta_{ACK}$  is an extension of algorithm  $\beta$  also described in [11] which implements the broadcast of a message  $\mu$  within bounded time. At each round, only nodes that received  $\mu$  in specified previous rounds can send it to avoid the potential collisions. Initially, the source node  $s$  sends  $\mu$  to all its neighbours. A *Frontier Set*,  $Frnt$ , is defined where  $Frnt$  contains all nodes that have not received  $\mu$  and that have direct connections with nodes received  $\mu$  at the end of that round. Then a *Minimal Dominating Set*,  $miniD$  is defined over the nodes that already have received  $\mu$  such that nodes in  $Frnt$  are dominated by nodes in  $miniD$ . Nodes in  $miniD$  then send  $\mu$ , so that some of nodes in  $Frnt$  can receive  $\mu$ .  $Frnt$  and  $miniD$  are therefore updated since some nodes will leave  $Frnt$  and may join  $miniD$  in the next round. Nodes in new

$miniD$  will continue send  $\mu$  until  $Frnt = \emptyset$ . The broadcast then finishes. Note that during the execution, a node in  $miniD$  at round  $i$  may stay in the  $miniD$  till round  $j$ , where  $i < j$ . In this case, additional notification message *Stay* is needed to be sent to nodes who need to stay in  $miniD$ .

Algorithm  $\beta_{ACK}$  extends  $\beta$  by adding an additional ACK message, that is, when the last nodes receive  $\mu$ , one of them will generate an *ACK* message that will be forwarded back to  $s$ . During the execution, nodes will store the round number at which they received and sent  $\mu$  with two variables *informedRound* and *transmitRounds*. So that nodes know which path the *ACK* should follow back to  $s$ .  $\beta_{ACK}$  is based on a 3 bits labelling scheme  $\lambda_{ACK}$ . The first bit,  $X_1$ , indicates if a node  $u$  will be in  $miniD$  at least once during the broadcast. If yes, then  $X_1$  of  $u$  equals 1; if not, it equals 0. If  $X_1$  of  $u$  equals 1, when  $u$  receives message  $\mu$ ,  $u$  can re-send it once. The second bit  $X_2$  of  $u$  equal to 1 means that  $u$  needs to send a *Stay* when it receives  $\mu$  to notify the sender of  $\mu$  to stay in  $miniD$  for the next round. Only one of the informed nodes will have the third bit  $X_3$  equal to 1. This node will generate the *ACK* to be sent back to  $s$ . At the end of the broadcasting, which finishes in  $2n - 3$  rounds, the last informed node generates and sends back to the source node the *ACK* within additional  $n - 2$  rounds, where  $n$  is the number of nodes in the network.

Our optimization with respect to the  $\lambda_{ACK}$  proposed in [11] comes from the following simple observation: in a 3-bits labelling, there are 8 possible states: 000, 001, 010, 011, 100, 101, 110 and 111. The algorithm in [11] uses only 5 of them: 000, 001, 010, 100 and 110. In this section, we propose a labelling scheme,  $\lambda_{oACK}$  and a broadcast scheme with *ACK* algorithm that use all the 8 states of the 3-bits labelling in order to improve the memory complexity of the solution proposed in [11]. The idea of our optimization is as follows: instead of only using the last bit  $X_3$  (the third bit) as a marker 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 [11], nodes do not need to keep additional variables in order to send back the *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 and  $\beta_{oACK}$  algorithm.

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

The first two bits of the labelling scheme  $X_1$  and  $X_2$  have the same functionality as in the  $\lambda_{ACK}$  scheme of [11]. The intuitive idea is as follows: 1)  $X_1 = 1$  for nodes who should propagate  $\mu$  when they receive it; 2)  $X_2 = 1$  for nodes that need to send *Stay* back to their sender neighbour to notice that they need to stay in  $miniD$  and send  $\mu$  one more time in the next round; 3)  $X_3 = 1$  for one of the last receiving nodes to generate *ACK* and send it back to the source node  $s$ . In our scheme  $\lambda_{oACK}$  we also set  $X_3$  (the third bit) to 1 for all nodes on the path back from the last informed node (who

---

**Algorithm 1**  $\beta_{oACK}(\mu)$  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*.

```
for each round  $r$  from 0 do
  if  $v$  is source node and  $r = 0$  then
    transmit sourcemsg
  if  $v$  is not source node then
    if message  $m$  is received AND  $m \neq \text{"stay"}$  then
      sourcemsg  $\leftarrow m$ 
  else if The node received  $\mu$  before round  $r$  then
    if  $v$  received sourcemsg for first time in round  $r - 2$  then
      if  $X_1 = 1$  then
        transmit sourcemsg
      else if  $v$  received sourcemsg for first time in round  $r - 1$  then
        if  $X_1 = 0$  and  $X_2 = 0$  and  $X_3 = 1$  then
          transmit "ACK"
        else if  $X_2 = 1$  then
          transmit "stay"
      else if  $v$  received "stay" in round  $r - 1$  then
        if  $v$  transmitted sourcemsg in round  $r - 2$  then
          transmit sourcemsg
      else if  $v$  received "ACK" in round  $r - 1$  then
        if  $X_3 = 1$  then
          transmit "ACK"
```

---

holds 001) to  $s$ . 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. Label states 101, 011 and 111 are not used in the original  $\beta_{ACK}$ , therefore nodes can easily recognize if they are on the path to transmit *ACK* back to  $s$ . Note that we do not change the main architecture of the algorithm  $\beta_{ACK}$  with labelling scheme  $\lambda$  proposed in [11], therefore the correctness proof of our algorithm is very similar to the one in [11]. See Section III-C for a detailed proof that follows the lines of the proof in [11].

### B. Broadcast Algorithm $\beta_{oACK}$

Our broadcast algorithm  $\beta_{oACK}$  that uses  $\lambda_{oACK}$  described above is described as Algorithm 1. Nodes with  $X_1 = 1$  receiving a message at round  $i - 1$  send it at round  $i$ . Then nodes who send at round  $i$  wait for the *stay* message, at round  $i + 1$ , from nodes with  $X_2 = 1$ . If nodes who send at round  $i$  receive a *stay* at round  $x + 1$ , they continue to send one more time  $\mu$  at round  $i + 2$ , otherwise, they stay silent. When nodes with label 001 receive the message, they generate an *ACK* and send it. Since  $\lambda_{oACK}$  already marked the path back to the source node, in Algorithm  $\beta_{oACK}$ , the *ACK* message will only be re-propagated by nodes with  $X_3 = 1$ . i.e., nodes with label 101, 111 and 011.

Note that our proposed Algorithm  $\beta_{oACK}$  does not need additional variables to reconstruct the path back to  $s$  during the broadcast execution. In Algorithm  $\beta_{ACK}$  [11], 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  $\beta_{oACK}$ , the *ACK* transmission processing can be completed only by checking the third bit,  $X_3$ . Our

Algorithm  $\beta_{oACK}$  does not need any extra local storage for detecting the path for *ACK*.

### C. Correctness of $\beta_{oACK}$

Our proposition of  $\beta_{oACK}$  with  $\lambda_{oACK}$  is based on the algorithm  $\beta$  with labelling scheme  $\lambda$  proposed in [11]. The algorithm  $\beta_{oACK}$  can be seen as the combination of two phases: *Broadcast Phase* and *ACK Phase*. The aim of the broadcast phase of  $\beta_{oACK}$  is to finish first the broadcast: every node in the network should be informed of the message  $\mu$  sent by the source node. In the second phase, one of the last informed nodes will generate *ACK* and send it back to the source node through a specific path marked according to the labelling scheme  $\lambda_{oACK}$ .

These two phases are well separated, because *ACK* will only be generated and sent to the network after one of the last informed nodes received  $\mu$  sent by the source node. Therefore, there will be no collisions between  $\mu$  and *ACK* during the execution of  $\lambda_{oACK}$ .

During the first broadcast phase, we use exactly the same idea of the algorithm  $\beta$  with labelling scheme  $\lambda$  in [11]. The correctness of this phase is given as Theorem 1 in [11], as follows:

**Theorem 1:** [11] Consider any  $n$ -node unlabelled graph  $G$  with a designated source node  $s$  with  $\mu$ . By applying the 2-bits labelling scheme  $\lambda$  and then executing algorithm *beta*, all nodes in  $G \setminus \{s\}$  are informed within  $2n - 3$  rounds.

As described in Section III, the idea is that every two rounds, if there are still nodes that have not received  $\mu$  yet, a non-null subset of these nodes will form the new *Frnt*. When the new *miniD* set of nodes send  $\mu$ , some nodes belonging to *Frnt* will receive it. Then the number of the non-informed nodes will decrease until 0. In the worst case, when the topology of the network is a line,  $\mu$  has to go through all of them one by one to reach every node. The algorithm therefore finishes within  $2n$  rounds.

We then prove that the *ACK* phase of  $\beta_{oACK}$ , finishes within  $n$  rounds.

**Lemma 1:** After the broadcast phase finishes during the execution of  $\beta_{oACK}$ , *ACK* will be sent back to the source node within  $n$  rounds.

**Proof** By using  $\lambda_{oACK}$  described in Section III-A, only one of the last informed nodes  $u$  will have its three bits equal to 001. Then  $u$  will send *ACK*, and only nodes with  $X_3 = 1$  can forward *ACK* back only when they received it. The back-forward path to the source node is chosen by  $\lambda_{oACK}$ . In the worst case, when the topology of the network is a line, then the *ACK* has to go through all the nodes to reach the source node. Therefore, during the execution of  $\beta_{oACK}$ , *ACK* will need at most  $n$  rounds to reach the source node.

The Theorem 1 and Lemma 1 therefore complete the correctness proof of  $\beta_{oACK}$ .

## IV. LEVEL-SEPARABLE NETWORKS

In this section, we define a family of networks, *Level-Separable Network*, issued from WBAN (Wireless Body Area

Networks) area (e.g. [20], [23], [25], [2], [4], [6], [3] and [7]), then we investigate the broadcast problem in these networks.

#### A. Motivation for the study of level-separable networks

The motivation of the study of level-separable networks comes from the recent studies of WBAN. WBAN is similar to WSN (Wireless Sensor Networks) in terms of devices functionalities and architecture. However, WBAN still has important differences with WSN. The deployment environment and application scenario make them totally different: WSN is usually deployed in wide range areas; WBAN on the other hand, is deployed on (or inside) the human body, to detect various physiological parameters of the human body. WBAN devices are in close contact with the body, therefore the transmission power cannot have a setting as high as in the case of WSN. Using a relatively small transmission power in WBAN might be greatly affected by the absorption, interference and refraction of the human body.

Furthermore, WBAN has to face the challenge of the human body mobility, which makes the connexion between nodes appear and disappear from time to time. The challenge in WBAN is how to improve the communication reliability of the network by taking into consideration the human mobility and the changes in the communication channels.

To our best knowledge, Naganawa *et. al* [26] proposed the first simulation-based *Data Sets* of the human mobility and the channel quality change. These data sets provide measurement results of channel attenuation between different WBAN devices deployed on different positions of the human body during different human movement actions. The data sets have been validated by comparing to massive real-human based measurement results.

The network architecture of proposed environment is composed of seven WBAN devices distributed on the body as follows: Navel, Chest, Head, Upper Arm, Ankle, Thigh and Wrist. The authors measure the connectivity between every two nodes in seven different postures: 1)Walking, 2)Running, 3)Walking weakly, 4)Sitting down, 5)Lying down, 6)Sleeping and 7)Putting on a jacket, respectively (see Figure 1).

In each posture, a continuous human action has been decomposed into a set of frames. Each single human body picture with a corresponding frame number,  $x$ , is a screenshot of this continuous human action at the  $x$ th frame. For example, in posture 1)Walking (see Figure 1), the continuous action takes 30 frame, and it uses four screenshots at 1st frame, 10th frame, 20th frame and the 30th frame, respectively to represent this action. The red diamonds in the figures represent sensors on the human body while the body is moving.

Tables I to VII show the measurement results of channel attenuation between two nodes pair in seven different human mobility postures. Values above the main diagonal represent the mean values of the random channel attenuation between any two WBAN nodes of the body. Based on data sets from [26], authors of [2] propose a channel-mobility model: for every wireless signal sent from a WBAN node, a random attenuation is added to the outgoing communication channel.

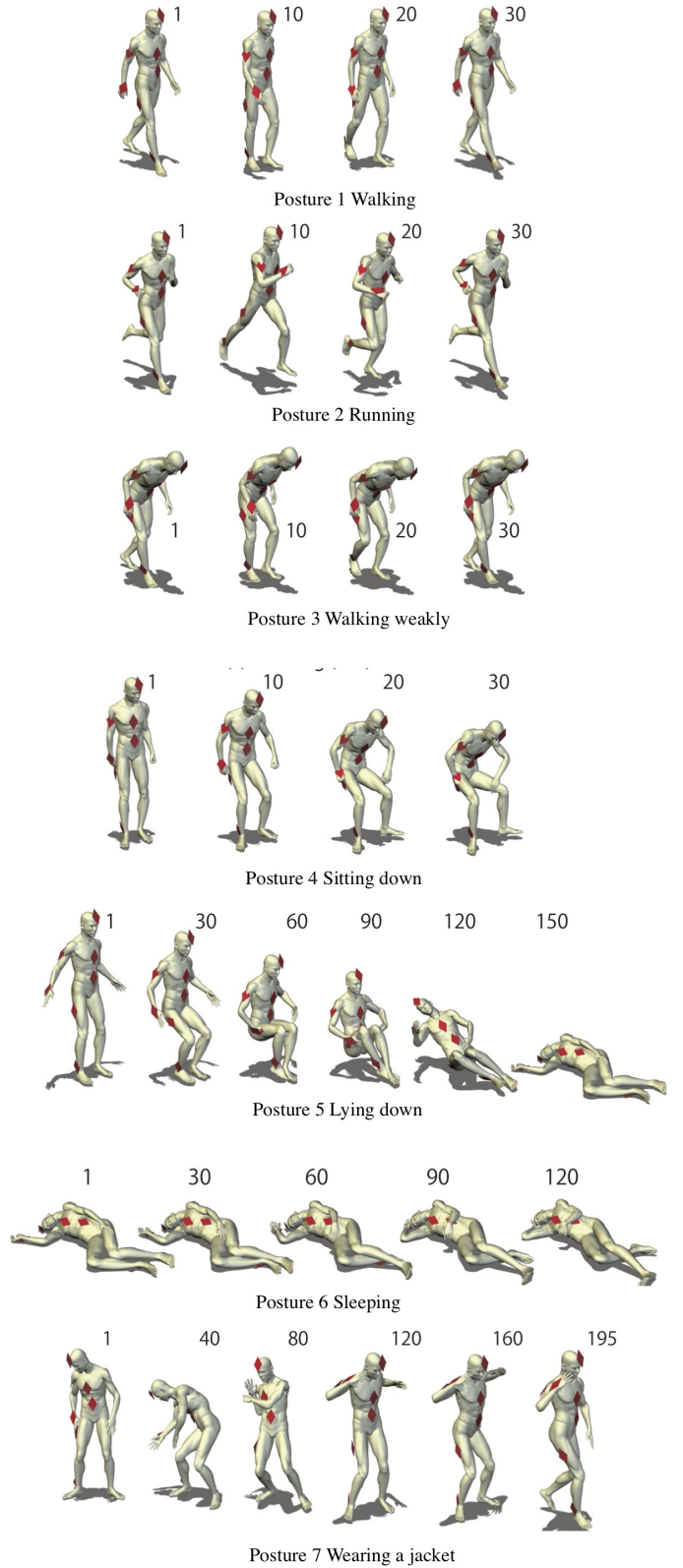


Fig. 1. 7 Different Human Postures [26]

If the signal strength after the attenuation is smaller than the sensitivity of the receiver, it will be dropped. The random attenuation is calculated by different normal distributions specified by means and standard deviations for each couple of nodes (e.g, the random channel attenuation between nodes on head and on upper arm in posture 1) Walking has the mean 45.4dB and the standard deviations 5.1dB).

TABLE I

MEANS AND STANDARDS DEVIATIONS OF PATH LOSS FOR ALL THE LINKS IN POSTURE 1) WALKING [26]

$T_X$ or $R_X$	navel	chest	head	upper arm	ankle	thigh	wrist	Mean[dB]
navel		30.6	45.1	44.4	57.4	45.8	41.0	
chest	0.5		38.5	40.6	58.2	51.6	45.1	
head	0.8	0.5		45.4	64.0	61.3	49.7	
upper arm	5.8	5.2	5.1		54.2	45.5	34.0	
ankle	4.3	3.4	5.0	3.1		40.6	48.9	
thigh	2.0	2.5	6.8	4.8	1.0		35.0	
wrist	5.0	3.6	3.8	2.5	3.8	3.3		
	Standard deviation [dB]							

TABLE II

MEANS AND STANDARDS DEVIATIONS OF PATH LOSS FOR ALL THE LINKS IN POSTURE 2) RUNNING [26]

$T_X$ or $R_X$	navel	chest	head	upper arm	ankle	thigh	wrist	
navel		31.4	47.4	54.5	57.9	44.8	45.9	Mean[dB]
chest	1.4		41.0	39.2	61.0	49.9	41.2	
head	3.5	2.9		41.3	65.6	59.3	45.5	
upper arm	9.9	8.4	8.4		58.0	52.4	33.8	
ankle	6.9	6.9	5.7	8.2		39.0	56.9	
thigh	2.0	2.5	6.8	4.8	1.0		49.6	
wrist	6.1	8.2	3.5	4.6	7.5	11.6		
	Standard deviation [dB]							

TABLE III

MEANS AND STANDARDS DEVIATIONS OF PATH LOSS FOR ALL THE LINKS IN POSTURE 3) WALKING WEAKLY [26]

$T_X$ or $R_X$	navel	chest	head	upper arm	ankle	thigh	wrist	
navel		26.1	42.4	44.3	55.4	44.9	34.0	Mean[dB]
chest	0.4		38.1	37.3	58.8	47.1	41.7	
head	1.3	0.7		44.5	52.4	60.0	42.8	
upper arm	5.5	5.5	6.8		53.7	45.1	34.5	
ankle	4.2	4.6	3.3	6.1		42.4	49.2	
thigh	2.2	5.3	5.4	4.8	2.2		37.9	
wrist	2.8	2.5	1.5	3.1	4.8	4.4		
	Standard deviation [dB]							

TABLE IV

MEANS AND STANDARDS DEVIATIONS OF PATH LOSS FOR ALL THE LINKS IN POSTURE 4) SITTING DOWN [26]

$T_X$ or $R_X$	navel	chest	head	upper arm	ankle	thigh	wrist	
navel		27.9	41.1	41.5	59.6	48.3	38.6	Mean[dB]
chest	1.0		37.0	36.0	60.0	51.0	43.2	
head	1.6	0.8		42.1	63.7	59.1	46.9	
upper arm	5.3	4.8	6.3		63.7	49.0	37.7	
ankle	8.4	8.0	8.7	8.1		40.9	60.2	
thigh	6.3	5.3	7.8	5.5	6.3		35.1	
wrist	4.6	5.3	5.5	5.7	9.6	6.9		
	Standard deviation [dB]							

Studies ([6] and [4]) conducted in WBAN show that various postural mobilities can be modeled as graphs (one for each human posture), see Figure 2. Moreover, the authors in [5]

TABLE V

MEANS AND STANDARDS DEVIATIONS OF PATH LOSS FOR ALL THE LINKS IN POSTURE 5) LYING DOWN [26]

$T_X$ or $R_X$	navel	chest	head	upper arm	ankle	thigh	wrist	
navel		30.5	45.1	54.1	65.0	55.8	49.7	Mean[dB]
chest	2.2		38.2	43.4	63.6	54.3	46.5	
head	3.3	1.3		40.0	61.8	58.6	45.5	
upper arm	5.9	4.2	4.2		58.3	50.1	38.8	
ankle	6.9	5.8	7.0	5.1		41.2	44.7	
thigh	12.4	10.1	10.1	10.1	7.2		41.6	
wrist	6.3	4.9	3.8	1.9	9.6	8.8		
	Standard deviation [dB]							

TABLE VI

MEANS AND STANDARDS DEVIATIONS OF PATH LOSS FOR ALL THE LINKS IN POSTURE 6) SLEEPING [26]

$T_X$ or $R_X$	navel	chest	head	upper arm	ankle	thigh	wrist	
navel		31.7	64.3	66.5	72.5	56.3	58.6	Mean[dB]
chest	4.3		50.9	51.9	72.4	51.3	44.1	
head	10.4	10.6		39.0	69.4	59.9	42.5	
upper arm	4.6	2.7	11.3		51.5	42.7	30.9	
ankle	5.7	7.5	9.3	0.8		35.7	56.8	
thigh	5.0	2.1	10.8	2.6	0.9		48.9	
wrist	7.8	4.1	7.2	3.6	2.8	2.5		
	Standard deviation [dB]							

TABLE VII

MEANS AND STANDARDS DEVIATIONS OF PATH LOSS FOR ALL THE LINKS IN POSTURE 6) WEARING A JACK [26]

$T_X$ or $R_X$	navel	chest	head	upper arm	ankle	thigh	wrist	
navel		27.4	43.3	56.8	62.8	45.0	52.0	Mean[dB]
chest	3.4		37.4	51.4	60.4	47.7	50.9	
head	4.9	3.6		49.2	64.0	51.7	46.8	
upper arm	6.7	5.1	9.2		52.3	52.9	31.1	
ankle	7.1	9.9	8.8	4.1		39.5	55.1	
thigh	2.5	6.3	7.0	5.1	1.7		52.3	
wrist	7.4	5.9	5.9	4.8	10.8	7.7		
	Standard deviation [dB]							

proved that the performances of any protocol for wireless body area networks strongly depend on the topology of the graph and it should be noted that none of the graphs corresponds to the classical classes (e.g. planar or minor-free).

In the case presented above (the only available to date benchmark for practical WBAN), each graph is a level-separable network defined below.

#### B. Formal definition of level-separable networks

We say that an arbitrary network is a Level-Separable Network if the underlay communication graph  $G = (V, E)$  of the network verifies the *Level-Separable* property defined below. To define the *Level-Separable* property, we introduce some preliminary notations.

Let  $G(V, E)$  be a network and let  $s \in V$ , a predefined vertex, be the source node of the broadcast. Each vertex  $u \in V$  has a geometric distance with respect to  $s$  denoted  $d(s, u)$ . The eccentricity of vertex  $s$ ,  $\varepsilon_G(s)$ , 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 and  $s$  the source node. For any vertex  $u$  in  $G(V, E)$ , the *level* of  $u$  is  $l(u) = d(s, u)$  is its geometric distance to  $s$ . Let  $S_i = \{u \mid$



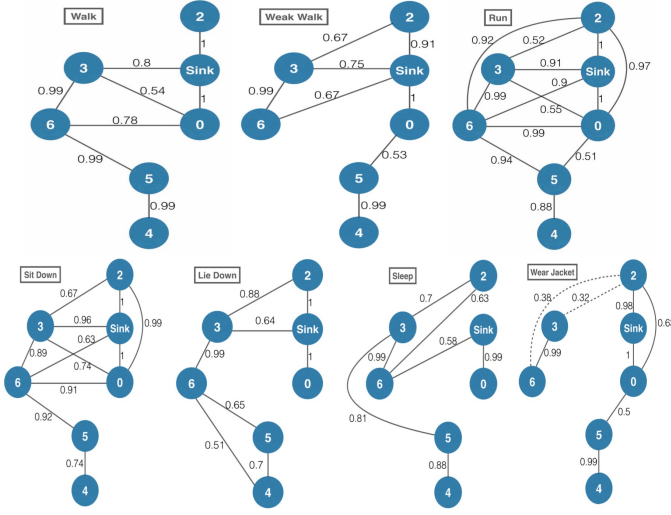


Fig. 2. Graphs that model human postures in WBAN. Numbers on the edges represent the edge reliability [4]

$u \in V, l(u) = i\}$  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  $l(v) - l(u) = 1 \wedge \{u, v\} \in E$ . 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* property below defines how to filter nodes in the same level  $i$  into two disjoint subsets.

**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*:

$$T_i = \{(S_{i,1}^{(1)}, S_{i,2}^{(1)}), (S_{i,1}^{(2)}, S_{i,2}^{(2)}), \dots, (S_{i,1}^{(2^x)}, S_{i,2}^{(2^x)})\}$$

where  $(m)$  on right-top of each pair represent the index of pairs (the  $m$ th pairs) in  $T_i$ , and  $x = |S_i|$ .

**Definition 4 (Level-Separable Property):** Given an arbitrary graph  $G(V, E)$ , for all level  $i \in [1, D - 1]$ , where  $D$  is the eccentricity of the source node,  $G$  verifies the Level-Separable property, if there are pairs for every  $T_i$ ,  $(S_{i,1}^{(k)}, S_{i,2}^{(k)})$ , such that:  $\forall u \in S_{i+1}, |P(u) \cap S_{i,1}^{(k)}| = 1 \vee |P(u) \cap S_{i,2}^{(k)}| = 1$  i.e., for every vertex  $u$  at level  $i + 1$ ,  $u$  has only one parent in  $S_{i,1}$  or  $S_{i,2}$ .

Note that when  $S_{i,1}$  is fixed,  $S_{i,2}$  is  $S_i \setminus S_{i,1}$ .

**Definition 5 (Level-Separable Network):** A network  $G(V, E)$  is a Level-Separable Network if its underlay graph verifies the Level-Separable property.

Note that *Level-Separable Graph* has a similar flavour with *Bipartite Graph* [17]. A graph  $G = (V, E)$  is said to be Bipartite if and only if there exists a partition  $V = A \cup B$  and

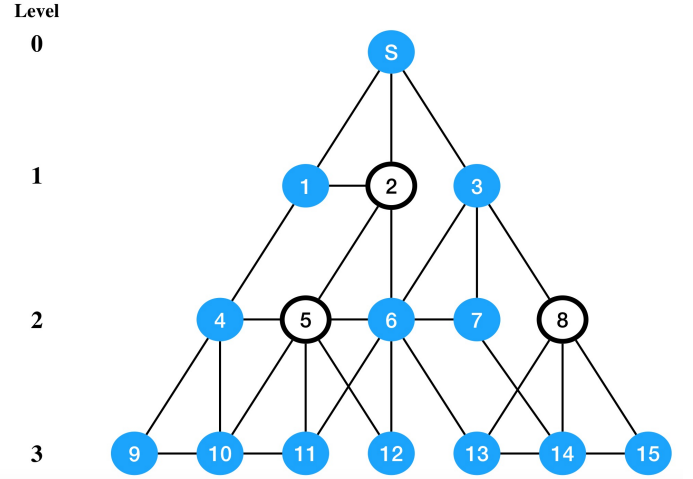


Fig. 3. Example of a Level-2 separable network, which is not a tree network

$A \cap B = \emptyset$ . 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., the node's father-son relationship. However, note that being bipartite does not necessarily means that the graph is level-separable.

Note that a level-separable network is not necessary for being a tree network. However, a tree is a level-separable network. A simple example of a level-separable network is a tree network, where the source node  $s$  can be seen as the root of the tree who begins the broadcast. In a tree topology, all non-source nodes have only one parent, i.e.  $\forall u \in V - s, |P(u)| = 1$ . Hence, we can choose  $S_{i,1} = S_i$  and  $S_{i,2} = \emptyset$ . The Level-Separable property is therefore verified. Figure 3 shows an example of a level-separable network that is not a tree.

Note that studies conducted in wireless body area networks (e.g. [2], [4], [6], [3] and [7]) fit our definition of level-separable network.

## V. BROADCAST IN LEVEL-SEPARABLE NETWORK

In this section, we propose a 1-bit constant-length labelling broadcast Algorithm  $\beta^{LS}$  detailed in Algorithm 2. The algorithm needs  $2D$  rounds to terminate, 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  $\beta^{LS}$  to achieve 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 they will not generate valid collisions<sup>1</sup>.

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

### B. 1-bit Labelling Scheme $\lambda^{LS}$

To achieve 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 4.

### C. Correctness of Algorithm $\beta^{LS}$

In the following, we prove that Algorithm  $\beta^{LS}$  is correct.

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

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

**Note 1:** Note that the 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 nodes in the network do not receive the broadcasted message due to the collisions since nodes are synchronized and transmit at the same time.

**Lemma 2:** 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  $i \in [1, D-1]$  are the only one to send a message concurrently at round  $j$  and on the next round  $j+1$  nodes with  $X_1 = 0$  at the same level  $i$  are the only one to send a message concurrently, all nodes at level  $i+1$  have received the message without collision either at round  $j$  or round  $j+1$ .

**Proof** Let  $u \in S_{i+1}$ . By construction,  $u$  has exactly one parent in  $S_{i,1}$  or  $S_{i,2}$ . In the first case,  $u$  has received the message without collision at round  $j$ , and it has received it at round  $j+1$  in the second case.

**Lemma 3:** Given a level-separable network whose root is the source node by applying  $\beta^{LS}$  and  $\lambda^{LS}$ , all nodes in level  $i > 0$  finish receiving the 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 0, which is round  $2 \times i - 2 = 2 \times 1 - 2 = 0$ , the source sends the message. All nodes at level 1 will receive the message at the end of round 0. For

<sup>1</sup>Note that collisions that occur at a node who has already received the message successfully are not considered as valid collisions.

### Algorithm 2 $\beta^{LS}(\mu)$ 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
  if  $v$  is not source node and receives  $\mu$  then
    if  $k = 0$  then
      sourcemsg  $\leftarrow \mu$ 
      if  $r$  is odd number then
        if  $X_1 = 0$  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  $k = 1$ 

```

$i = 2$ , as all nodes at level 1 can receive the 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 2, all nodes received the message without collision at round 2, which is round  $2 \times i - 2 = 2 \times 2 - 2 = 2$  and they begin to send the message at round 3 and 4. For the general case, we assume that all nodes at level  $i$ ,  $i > 2$ , finish receiving the message at round  $2i-2$ . So that nodes begin to send the received message at round  $2(i+1)-3$  and  $2(i+1)-2$ , and nodes at level  $i+1$  receive the message at  $2(i+1)-3$  and  $2(i+1)-2$ , that is nodes at level  $i+1$  finish receiving the message at round  $2(i+1)-2$ .

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

**Proof** From Lemma 3, 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. Therefore the broadcast finishes at round  $2D$ .

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 the level-separable property at level  $i$ ,  $\forall i > 0$ . Nodes in  $S_{i,1}$  have  $X_1 = 1$ , and nodes in  $S_{i,2}$  have  $X_1 = 0$ . 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 impact of the collision 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 2. Therefore 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  $i+1$  having only one parent and did not receive  $\mu$  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  $i+1$  will 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.

## VI. BROADCAST WITH ACK IN LEVEL-SEPARABLE NETWORK

In this section, we propose a broadcast algorithm with ACK,  $\beta_{ACK}^{LS}$ , and a Labelling Scheme,  $\lambda_{ACK}^{LS}$ , for level-separable networks. Our algorithm  $\beta_{ACK}^{LS}$  (Algorithm 3) uses only 2-bits labelling and the broadcast finishes within  $2D$  rounds. In our solution, *ACK* goes back to the source node in at most  $2D$  rounds, where  $D$  is the eccentricity of  $s$  (the broadcast source node). That means the *ACK* can be received by the source node at the same round of the broadcast termination.

### A. 2-bits Labelling Broadcast with ACK

According to Theorem 2 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* 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 *ACK* takes additional 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 go back one level above. Therefore, when the last node receives  $\mu$ , the source node receives *ACK* at the same round. Interestingly, compared with non-ACK broadcasting, our solution uses one extra bit for labelling and no additional rounds to forwarding *ACK* back to the source.

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

### B. 2-bits Labelling Scheme $\lambda_{ACK}^{LS}$

We use  $\lambda^{LS}$  to set  $X_1$  in  $\lambda_{ACK}^{LS}$  in order to verify Lemma 2. Let  $X_2$  be the second bit of the  $\lambda_{ACK}^{LS}$  labelling scheme.  $X_2 = 1$  for a set of nodes if they are on the way back path from a node at level  $\lfloor D/2 \rfloor - 1$  to the source node, where  $D$  is the eccentricity of  $s$  and  $s$  is the broadcast source. For the other nodes,  $X_2 = 0$ . In Section VI-C, we explain why we choose nodes at level  $\lfloor D/2 \rfloor - 1$  to begin sending the *ACK*.

**Note 2:** Note that the 2-bits labelling scheme is optimal to achieve broadcast with acknowledgment in a level-separable

network. From Note 1 1-bit is necessary for broadcast without acknowledgment. When an acknowledgment has to be sent back to the source node, at least one additional bit is necessary to indicate the node to generate the acknowledgment 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 *ACK* back.

### C. Correctness of Algorithm $\beta_{ACK}^{LS}$

Theorem 3 below proves the correctness of Algorithm  $\beta_{ACK}^{LS}$ .

**Theorem 3:** Algorithm  $\beta_{ACK}^{LS}$  with 2-bits labelling scheme  $\lambda_{ACK}^{LS}$  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  $2D$ , if  $D$  is even.

The proof of the theorem is the direct consequence of Lemmas 5, 6 and 7 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  $2i-2$ . The broadcast finishes at round  $2D$ .

**Proof**  $\beta_{ACK}^{LS}$  follows the same idea as  $\beta^{LS}$ . The additional *ACK* transmission will not have any impact according to Lemma 3 and 4. Hence the proof follows.

**Lemma 6:** Given a level-separable network whose root is the source node by applying  $\beta_{ACK}^{LS}$  and  $\lambda_{ACK}^{LS}$ , *ACK* goes back to the source node at round  $2(D-1)$ , if  $D$  is odd; or  $2D$ , if  $D$  is even.

**Proof** When  $D$  is odd, *ACK* and  $\mu$  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 the source are the same as that from  $l_{MSG}$  to the ending nodes. *ACK* arrives at the source at the same round as  $\mu$  arrives at the ending nodes. According to Lemma 5, this is round  $2(D-1)$ . When  $D$  is even *ACK* needs to go one level farther compared with  $\mu$ . Therefore, it takes two extra rounds when  $D$  is even. Therefore, when  $D$  is even the *ACK* message goes back to the source node in  $2D$  rounds.

**Lemma 7:** 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** 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  $X_1 = 1$ , and nodes in  $S_{i,2}$  have  $X_1 = 0$ . 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 *ACK* transmission processing, so that when the  $\mu$  reaches to the ending node(s) at level  $D$ , *ACK* reaches the source  $s$  at (almost) the same round. As nodes cannot decide if they are

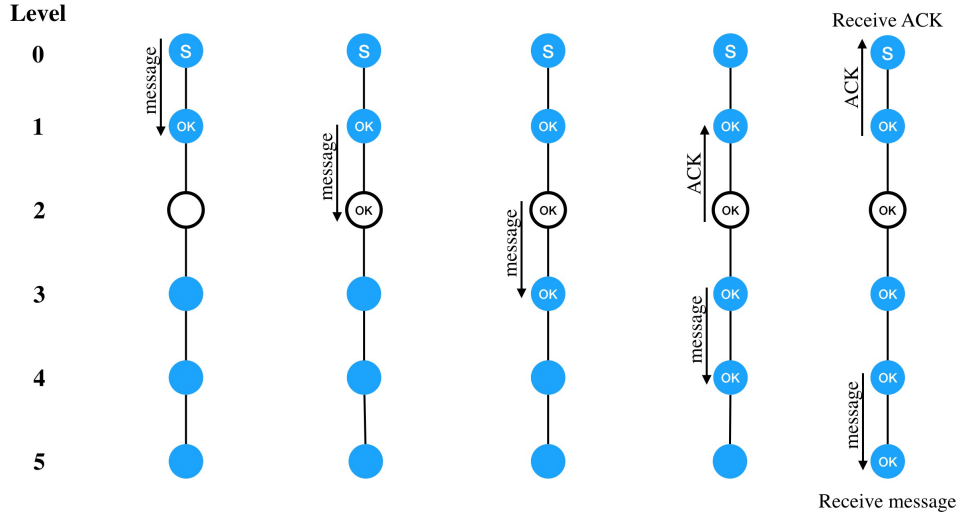


Fig. 4. Anticipating the ACK in a level-separable network

the ones at the half-way of the network who should generate and send *ACK*, we use a *Waiting Period* and an extra *pACK* message.

According to 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* and begins to wait for *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*. All the nodes with  $X_2 = 1$  will forward *ACK* 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* 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 for 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 the source node at the same round as the end of the broadcast. Because after waiting for 6 rounds at level  $\lfloor D/2 \rfloor - 1$ ,  $\mu$  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  $d(\lfloor D/2 \rfloor + 2, D) = \lfloor D/2 \rfloor - 1$ . When  $D$  is even, 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 the source node only two rounds after the round of the ending of the broadcast.

Therefore it takes  $2D$  rounds to finish the broadcast and *ACK* can be transmitted back to the source node at round  $2(D - 1)$  or round  $2D$ . Note that nodes will only send (both for  $\mu$  and *ACK*) once according to  $\beta_{ACK}^{LS}$ . Therefore the algorithm terminates.

## VII. HARDNESS OF LEVEL SEPARATION

It should be noted that checking that a separation is a Level-Separation is polynomial: it is sufficient to check that for each node  $u$ ,  $|P(u) \cap S_{l(u),1}| = 1 \vee |P(u) \cap S_{l(u),2}| = 1$ . In this section we will prove that determining if a graph has the level-separable property is NP-Hard. To do so, we will reduce 1-IN-3-SAT [21] to the level separable problem. 1-IN-3-SAT is a NP-Complete variant of the usual NP-complete problem 3-SAT, where exactly a single literal in each clause must be true. As input, we have a list of variables  $X = \{x_1, \dots, x_k\}$  and a formula  $\phi$  which is a conjunction of clauses  $c_1, \dots, c_l$  that are each composed of exactly 3 literals of the form  $x_i$  or  $\bar{x}_i$ . The goal is to find an assignation for the variables  $A : X \rightarrow \{\top, \perp\}$  such that, for every clause  $c_i$ , exactly one variable is satisfied (i.e. has the assignation  $\top$  if it appears positively,  $\perp$  if it appears negatively).

**Theorem 4:** Determining if a graph with a source has the Level-Separable property is NP-complete.

**Proof** Let  $(X, \phi)$  be an instance of 1-IN-3-SAT. We will build  $G = (V, E)$  such that  $V = \{s\} \cup S_1 \cup S_2$ ,  $S_1$  being the neighborhood of  $s$ , and  $S_2$  all the other nodes, that will actually be at distance 2 from  $s$ . We have:

- $S_1 = \{u_{n_a}, u_{n_b}, u_y\} \cup \{u_{y_i}, u_{n_i}\}_{i \leq k}$ .
- $S_2 = \{v_a, v_b\} \cup \{v_{x_i}\}_{i \leq k} \cup \{v_{c_j}\}_{j \leq l}$ .
- $\{s\} \times S_1 \subset E$ ,  $\{(u_{n_a}, v_a), (u_{n_b}, v_b), (u_y, v_a), (u_y, v_b)\} \subset E$ .

---

**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  $k_{ack}$  initially set to 0 to ensure each node send  $\mu$  only once.

```

for each round  $r$  from 0 do
  if  $v$  is source node and  $r = 0$  then
    transmit sourcemsg
  if  $v$  is not source node and received  $\mu$  then
    sourcemsg  $\leftarrow \mu$ 
    if  $k = 0$  then
      if  $r$  is odd number then
        if  $X_1 = 0$  then
          transmit sourcemsg at round  $r + 3$ 
          if  $X_2 = 1$  then
            transmit "pACK" at round  $r + 4$ 
            if  $v$  does not received "pACK" at  $r + 6$  then
              transmit "ACK" at round  $r + 6$ , set  $k_{ack} = 1$ 
        else if  $X_1 = 1$  then
          transmit sourcemsg at round  $r + 2$ 
          if  $X_2 = 1$  then
            transmit "pACK" at round  $r + 4$ 
            if  $v$  has not received "pACK" at  $r + 6$  then
              transmit "ACK" at round  $r + 6$ , set  $k_{ack} = 1$ 
      else if  $r$  is even number then
        if  $X_1 = 0$  then
          transmit sourcemsg at round  $r + 2$ 
          if  $X_2 = 1$  then
            transmit "pACK" at round  $r + 3$ 
            if  $v$  has not received "pACK" at  $r + 5$  then
              transmit "ACK" at round  $r + 5$ , set  $k_{ack} = 1$ 
        else if  $X_1 = 1$  then
          transmit sourcemsg at round  $r + 1$ 
          if  $X_2 = 1$  then
            transmit "pACK" at round  $r + 3$ 
            if  $v$  has not received "pACK" at  $r + 5$  then
              transmit "ACK" at round  $r + 5$ , set  $k_{ack} = 1$ 
      set  $k = 1$ 
  if  $v$  is not source node and received ACK then
    if  $X_2 = 1$  and  $k_{ack} = 0$  then
      transmit ACK at round  $r + 2$ 
    set  $k_{ack} = 1$ 

```

---

- $\forall i \leq k$ , we have  $\{(u_{y_i}, v_{x_i}), (u_{n_i}, v_{x_i})\} \subset E$ . If  $x_i \in c_j$ , then we have  $(u_{y_i}, v_{c_j}) \in E$ . If  $\bar{x}_i \in c_j$ , then we have  $(u_{n_i}, v_{c_j}) \in E$ .
- $\forall j \leq l$ , we have  $\{(u_{n_a}, v_{c_j}), (u_{n_b}, v_{c_j})\} \subset E$ .

An abstract graph can be seen in Figure 5 corresponding to the description above.

Let's suppose that we have a solution  $S_{1,1}, S_{1,2}$  to the problem (any partition of  $S_2$  works, as there are the farthest nodes from  $s$ ). We will call  $Y \in \{1, 2\}$  the index of the node  $u_y$ , and  $N = 3 - Y$  the index that is different from  $Y$ . Here below a list of observations:

- 1) If a node in  $S_2$  has exactly two parents, then the index of its parents must be different.
- 2)  $u_{n_a}$  (resp.  $u_{n_b}$ ) must have index  $N$ , as  $v_a$  (resp.  $v_b$ ) is only connected to it and to  $u_y$ .
- 3)  $\forall i \leq k$ ,  $u_{y_i}$  and  $u_{n_i}$  have different indexes, as they are the only parents of  $v_{x_i}$ .
- 4)  $\forall j \leq l$ ,  $v_{c_j}$  has exactly one parent of index  $Y$ , as it has at least two parents of index  $N$ :  $u_{n_a}$  and  $u_{n_b}$ .

A solution for the corresponding 1-IN-3-SAT instance is to choose, for each variable  $x_i$  such that  $u_{y_i}$  has index  $Y$ , valuation  $\top$ , and  $\perp$  for the others. Let  $c_j$  be a clause. The

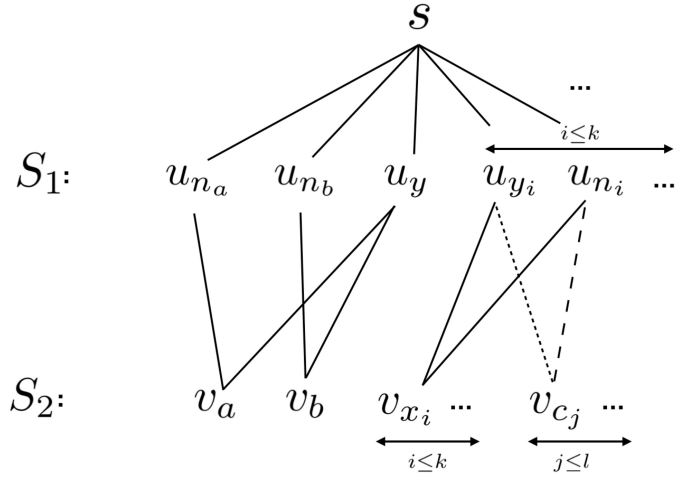


Fig. 5. Solid lines represent the edges that always exist. Dense dotted lines represent the edges that exist if  $x_i \in c_j$ . Loose dotted lines represent the edges that exist if  $\bar{x}_i \in c_j$

node  $v_{c_j}$  has exactly one parent of index  $Y$  among the ones corresponding to variables. If it is a node of the form  $u_{y_i}$ , then  $x_i$  appears positively in  $c_j$  (otherwise, it is of the form  $u_{n_i}$  and  $x_i$  appears negatively in  $c_j$ ). Let be another node corresponding to a variable connected to  $v_{c_j}$ . Its index must be  $N$ , and it appears positively in  $c_j$  iff the node is some  $u_{y_i}$  iff we chose  $\perp$  for  $x_i$ . Hence,  $c_j$  has exactly one variable satisfied.

Reversely, let's suppose that we have an assignation  $A$  to the 1-IN-3-SAT instance. We choose  $S_{1,1} = \{u_y\} \cup \{u_{y_i} : A(x_i) = \top\} \cup \{u_{n_i} : A(x_i) = \perp\}$  and  $S_{1,2} = S_1 \setminus S_{1,1}$ . Let's prove that each node in  $S_2$  has exactly one parent in  $S_{1,1}$ . For  $v_a$  and  $v_b$ , it is  $u_y$ . For a node  $v_{c_j}$ , we know that exactly one variable of  $c_j$  is satisfied. Its corresponding node is in  $S_{1,1}$  by construction, and the corresponding node of the two other variables are in  $S_{1,2}$  by construction. As  $v_{n_a}$  and  $v_{n_b}$  are also in  $S_{1,2}$ , this concludes the proof.

## VIII. CONCLUSION

We proposed solutions for implementing broadcast in wireless networks when the broadcast is helped by a labelling scheme. We studied broadcast without acknowledgment (i.e. the initiator of the broadcast is not notified at the end of the broadcast) and broadcast with acknowledgment. We first improved in terms of memory complexity the scheme proposed in [11] for arbitrary networks. Then we propose an optimal acknowledgment-free broadcast strategy using only 1-bit labelling and a broadcast with acknowledgment using a 2-bits labelling in level 2-separable networks. The complexity of both algorithms is  $2D$  where  $D$  is the eccentricity of the broadcast initiator. Level 2-separable networks have a practical interest in the large literature of WBAN.

In Section VII, we proved that the verification of the level-separable property can be done in polynomial time while determining if a graph has the level separable property is

NP-hard. This result may be considered as a serious break in exploiting the level separable property in labelling-based algorithms. However, in the case of small scale networks such as WBAN, polynomial algorithms may be of practical interest. For the case of large scale networks, since the verification of the level-separable property is NP-hard, we recommend to exploit *MIMO* antenna technology [18], [24] (wireless devices having the capability to focus the wireless transmission on several dedicated directions). Thanks to this technology the connections from a node to several of its neighbours can be disabled. This simple mechanism can help in constructing networks with built-in level separable property according to the description in Section IV. In this case, our algorithms are the best to date for labelling-based broadcast.

Independent of the practical interest of our work, an interesting theoretical research direction is opened by our study: the generalization of our results to level  $k$ -separable networks. In this framework, it would be interesting to find optimal separations for a graph and the tradeoff between the time and the bit complexity of broadcast in level  $k$ -separable networks.

## REFERENCES

- [1] 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.
- [2] 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.
- [3] Wafa Badreddine, Nesrine Khernane, Maria Potop-Butucaru, and Claude Chaudet. Convergecast in wireless body area networks. *Ad Hoc Networks*, 66:40–51, 2017.
- [4] Wafa Badreddine and Maria Potop-Butucaru. Peak transmission rate resilient crosslayer broadcast for body area networks. *arXiv preprint arXiv:1702.05031*, 2017.
- [5] Baynat Bruno, Bu Gewu, and Potop-Butucaru Maria. Markovian model for broadcast in wireless body area networks. In *Proceedings of the 17th ACM International Symposium on Mobility Management and Wireless*. ACM, 2019.
- [6] Gewu Bu and Maria Potop-Butucaru. Total order reliable convergecast in wban. In *Proceedings of the 18th International Conference on Distributed Computing and Networking*, page 26. ACM, 2017.
- [7] Gewu Bu and Maria Potop-Butucaru. Ban-gzpk: Optimal zero knowledge proof based scheme for wireless body area networks. *Ad Hoc Networks*, 77:28–41, 2018.
- [8] Ferdinando Cicalese, Fredrik Manne, and Qin Xin. Faster deterministic communication in radio networks. *Algorithmica*, 54(2):226–242, 2009.
- [9] 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.
- [10] 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.
- [11] Faith Ellen, Barun Gorain, Avery Miller, and Andrzej Pelc. Constant-length labeling schemes for deterministic radio broadcast. In *The 31st ACM on Symposium on Parallelism in Algorithms and Architectures*, pages 171–178. ACM, 2019.
- [12] Pierre Fraigniaud, David Ilcinkas, and Andrzej Pelc. Tree exploration with advice. *Information and Computation*, 206(11):1276–1287, 2008.
- [13] Pierre Fraigniaud, Amos Korman, and Emmanuelle Lebhar. Local mst computation with short advice. *Theory of Computing Systems*, 47(4):920–933, 2010.
- [14] Cyril Gavoille, David Peleg, Stéphane Pérennes, and Ran Raz. Distance labeling in graphs. *Journal of Algorithms*, 53(1):85–112, 2004.
- [15] 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.
- [16] 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.
- [17] Jonathan L Gross and Jay Yellen. *Graph theory and its applications*. CRC press, 2005.
- [18] Alexander M Haimovich, Rick S Blum, and Leonard J Cimini. Mimo radar with widely separated antennas. *IEEE Signal Processing Magazine*, 25(1):116–129, 2007.
- [19] David Ilcinkas, Dariusz R Kowalski, and Andrzej Pelc. Fast radio broadcasting with advice. *Theoretical Computer Science*, 411(14-15):1544–1557, 2010.
- [20] Nadeem Javaid, Naveed A. Khan, Mustafa Shakir, Mahmood Ashraf Khan, Safdar Hussain Bouk, and Zahoor Ali Khan. Ubiquitous health-care in wireless body area networks - A survey. *CoRR*, abs/1303.2062, March 2013.
- [21] David S Johnson and Michael R Garey. *Computers and intractability: A guide to the theory of NP-completeness*, volume 1. WH Freeman San Francisco, 1979.
- [22] Dariusz R Kowalski and Andrzej Pelc. Optimal deterministic broadcasting in known topology radio networks. *Distributed Computing*, 19(3):185–195, 2007.
- [23] Benoît Latré, Bart Braem, Ingrid Moerman, Chris Blondia, and Piet Demeester. A survey on wireless body area networks. *Wirel. Netw.*, 17(1):1–18, January 2011.
- [24] Jian Li and Petre Stoica. Mimo radar with colocated antennas. *IEEE Signal Processing Magazine*, 24(5):106–114, 2007.
- [25] Samaneh Movassaghi, Mehran Abolhasan, Justin Lipman, David Smith, and Abbas Jamalipour. Wireless body area networks: A survey. *IEEE Communications Surveys Tutorials*, 16(3):1658–1686, January 2014.
- [26] Jun-ichi Naganawa, Karma Wangchuk, Minseok Kim, Takahiro Aoyagi, and Jun-ichi Takada. Simulation-based scenario-specific channel modeling for wban cooperative transmission schemes. *IEEE Journal of Biomedical and Health Informatics*, 19:559–570, May 2015.