Scheduling calls for multicasting in tree-networks

Johanne Cohen* Pierre Fraigniaud†‡ Margarida Mitjana§

Abstract

In this paper, we show that the multicast problem in trees can be expressed in term of arranging rows and columns of boolean matrices. Given a $p \times q$ matrix M with 0-1 entries, the *shadow* of M is defined as a boolean vector x of q entries such that $x_i = 0$ if and only if there is no 1-entry in the ith column of M, and $x_i = 1$ otherwise. (The shadow x can also be seen as the binary expression of the integer $x = \sum_{i=1}^{q} x_i 2^{q-i}$. Similarly, every row of M can be seen as the binary expression of an integer.) According to this formalism, the key for solving a multicast problem in trees is shown to be the following. Given a $p \times q$ matrix M with 0-1 entries, finding a matrix M^* such that:

1. M^* has at most one 1-entry per column;
2. every row r of M^* (viewed as the binary expression of an integer) is larger than the corresponding row r of M, $1 \leq r \leq p$; and
3. the shadow of M^* (viewed as an integer) is minimum.

We show that there is an $O(q(p+q))$ algorithm that finds M^* for any $p \times q$ boolean matrix M.

The application of this result is the following: Given a directed tree T whose arcs are oriented from the root toward the leaves, and a subset of nodes D, there exists a polynomial-time algorithm that computes an optimal multicast protocol from the root to all nodes of D. According to usual communications systems, the resulting communication protocol can be set on top of either 1-port or all-port send- and receive-instructions, with edge-disjoint long-distance calls. Such protocols can in turn find applications in the management of large multi-point applications (*e.g.*, video server, data-bases, etc.) in which a specific node broadcasts information to a large number of users connected by a tree-network (e.g., Core-Based Tree).

Keywords: information dissemination, multi-point applications, group communications, broadcasting, multicasting.

*Laboratoire de Recherche en Informatique, Bât. 490, Univ. Paris-Sud, 91405 Orsay cedex, France. Additional support by the DRET of the DGA.

†Laboratoire de Recherche en Informatique, Bât. 490, Univ. Paris-Sud, 91405 Orsay cedex, France. Additional support by the CNRS

‡Correspondence author. Email: `Pierre.Fraigniaud@lri.fr`.

§Dept. Matematica i Telematica, Universitat Politecnica de Catalunya, 08034 Barcelona, Spain.

1 Introduction

1.1 Motivations

Recent advances in telecommunication systems enhanced standard point-to-point communication protocols to multi-point protocols. These latter protocols are of particular interest for group applications. Those groups involve more than two users (some may even involve thousands of users) sharing a common application, as video-conferences, distributed data-bases, media-spaces, games, etc. Several protocols have been proposed to handle and to control a large group of users. We refer to [7, 21] for surveys on multi-point applications and protocols. Solutions differ according to the type of traffic that is induced by the shared application, and according to the quality of service required by the users. Multi-point architectures are often based on tree-networks [26], either a single tree connecting all the users (*e.g.*, Core-Based Tree [1]), or several trees (*e.g.*, PIM [6]). The traffic between the users is then routed along the edges of the tree(s).

One of the major communication problem related to multi-point applications consists to broadcast a message from one user to all the users of the application. This operation is called *broadcast* at the application level, though it is actually a *multicast* at the network level. The repetition of point-to-point connections between the source and the several destinations would significantly increase the traffic in the network, and it makes this solution not applicable in practice [7]. Thence, the source must require the help of other nodes to relay messages. A broadcast message will then reach the destinations after having been relayed by several intermediate nodes (each intermediate node may possibly get one copy of the message if it belongs to the group). In order to preserve the broadcast application from transmission errors, and to bound the interval between successive receptions of consecutive packets, the number of hops between the source and each destination must be as small as possible. The aim of this paper is to provide a polynomial algorithm which returns, for any tree T, and for any source $u \in V(T)$, a multicast protocol from u to an arbitrary subset of nodes of T that minimizes the number of hops.

We consider multicasting from the root to a set of destination nodes of a directed tree T whose arcs are oriented from the root toward the leaves. When the set of destination nodes is the set of all nodes, this problem correspond to the broadcast problem. We focus our work on oriented trees because, although a bidirectional channel can be reserved between members of a group to facilitate bidirectional exchanges, it happens frequently that the bandwidth reserved in each direction differs from each other as the application is often not symmetric. For instance, consider members connected to a video server: the main point is to insure a fast broadcast of the multi-media traffic *from* the server, and thus the bandwidth of the connections from or toward the server may differ of a few order of magnitude.

1.2 Models

We will consider both 1-port and all-port models. In the 1-port model, we assume that, at any given time, each node of the tree can *call* at most one other node of the tree. In the all-port model, a node can call many other nodes simultaneously. Moreover, according to modern communication facilities (e.g., circuit-switched, wormhole, WDM, or, in some sense, ATM), long-distance calls are allowed, in the sense that the receiver of a call is not necessarily a neighboring node of the initiator of the call, and a message crossing a node can cut-through the node if required. As a restriction though, we want the calls performed at the same time to not share any edge. This latter restriction

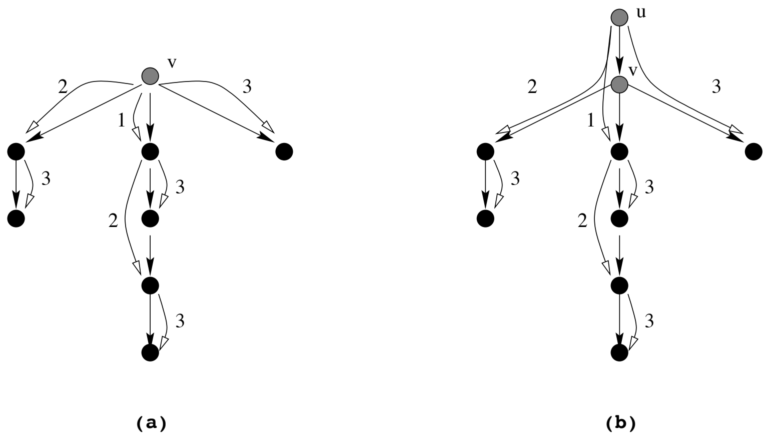

Figure 1: A broadcast in the 1-port model (a), and a multicast in the all-port model (b). Destination nodes are colored in black.

is set to avoid contention on the links. In particular, it means that, in the all-port model, a node x cannot initiate more than $\deg^+(x)$ calls, where $\deg^+(x)$ is the out-degree of node x. For instance, on Figure 1(b), the source node u cannot inform more than one other node at a time.

The set of all calls performed at the same time is called a *round*. For instance, on Figure 1(a), the first round is composed of one call, the second round is composed of two calls, and the third round is composed of four calls. We will express the cost of our broadcast protocols in terms of number of rounds. (That is we will be interested in minimizing the latency of the protocol rather than its throughput. Note that the pipeline technique may then be applied to our protocols in order to decrease the throughput for broadcasting long messages [10].) The aim of this paper is to show that there exist polynomial-time algorithms that computes the multicast time of any directed tree T under both 1-port and all-port edge-disjoint models. Comparing the two protocols on Figure 1(a) and (b) makes clear that these two constraints give rise to similar types of problems. Actually, it will be shown that the multicast problem can be solved by using a reduction to a problem on boolean matrices.

1.3 Previous works.

A huge literature has been devoted to group-communication problems under different hypotheses [7, 10, 13, 14, 21]. The related decision problems are often NP-complete for general networks [20, 25], and this gave rise to several approximation algorithms [2, 18, 23] and heuristics [11, 24]. Tree-networks deserved a specific interest in this context. Proskurowski [22] has shown that computing the broadcast time of a tree is polynomial in the 1-port model when only neighbor-to-neighbor calls are allowed. Still in this model, Slater, Cockayne and Hedetniemi [25] have derived a polynomial algorithm to find center nodes of undirected trees, that is nodes having minimal broadcast time among the nodes of the considered tree. Farley and Proskurowski [9] have studied the broadcasting problem in undirected trees when, at the beginning of the process, more than one node know the information to broadcast, whereas Harutuynuan and Labahn independently showed that, for any n, there exists an undirected tree-network whose broadcast time from any source is at most roughly $1.44\lceil\log_2 n\rceil$ [12, 19].

When long-distance calls are allowed, Cohen [4] has shown that there exists a polynomial-time algorithm to compute an optimal broadcast protocol in directed trees under the all-port edge-disjoint model. However, although this algorithm can be extended to the multicast problem in which the set of destinations is a subset of the nodes of the tree, it yields an inefficient protocol. In the 1-port edge-disjoint model, Farley [8] has shown that every *undirected* n-node network has a broadcast time of $\lceil \log_2 n \rceil$ (see also [17]). This result has been extended in [5] to the case in which the routes are chosen according to a shortest path routing function. However, the results of [5, 8] do not hold in directed networks: take as a counter example the digraph in which a node u has a unique outgoing arc to a node v which has in turn $n-2$ outgoing arcs to $n-2$ vertices $w_1, \ldots, w_{n-2}$, each connected by an outgoing arc to node u. Actually, broadcasting in a directed network gives rise to an NP-complete decision problem in the 1-port edge-disjoint model.

Note that some authors have also considered the vertex-disjoint constraint. In this context, the broadcasting problem was studied for specific architectures [15, 16], and approximation algorithms have been derived [18]. Actually, vertex-disjoint hypotheses also yield complex problems, and the broadcast problem is still open for trees (see [3] for a first attempt in this direction).

1.4 Our results.

First, we will show that the broadcast problem in directed trees under the 1-port edge-disjoint model gives rise to the following matrix problem (Lemma 2 in Section 2). Given a $p \times q$ matrix M with p rows, q columns, and 0-1 entries, the *shadow* of M is defined as a 1-dimensional boolean vector x of q entries such that $x_i = 0$ if and only if there is no 1-entry in the ith column of M, and $x_i = 1$ otherwise. According to this formalism, the key for solving a multicast problem in directed trees is shown to be the following.

Minimal contention-free matrix problem. Given a $p \times q$ matrix M with 0-1 entries, finding a matrix M^* such that[1]:

1. M^* has at most one 1-entry per column;
2. every row r of M^* (viewed as the binary expression of an integer) is larger than the corresponding row r of M, $1 \leq r \leq q$; and
3. the shadow of M^* (viewed as an integer) is minimum.

Such matrix M^* is called a *minimal contention-free version* of M. Note that the minimal contention-free version of a matrix is not necessarily unique, even up to a permutation of the rows. On the other hand, the shadow of a minimal contention-free version of a matrix is unique.

As an example, let us consider Figure 1(a). The corresponding matrix is

$$M = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

[1]Since the shadow can also be seen as the binary expression of an integer, and since, similarly, every row of M can be seen as the binary expression of an integer, the comparison of shadows and rows must be understood as comparing the corresponding integers.

as there are respectively 2, 4, and 1 nodes in the three branches (this correspondence will be formally established in Section 2). Since M has a single 1-entry per column, a minimal contention-free version of M is M itself, and the shadow is $7 = (111)_2$. Now, assume that the rightmost branch of the tree of Figure 1(a) contains three nodes instead of only one. Then the corresponding matrix is

$$M = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}, \tag{1}$$

and the reader can check that a minimal contention-free version of M is

$$M^* = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \tag{2}$$

M^* has a shadow equal to $14 = (1110)_2$. We will show that the matrix M^* determines a broadcast protocol from the root according to the 1-entries of the matrix. For instance, at round 1, v calls the second (middle) branch; at round 2, v calls the third (rightmost) branch; and, at round 3, v calls the first (leftmost) branch. At round 4, v is idle. We will show that there is an $O(q(p+q))$-time algorithm that computes a minimal contention-free version of M, for any $p \times q$ boolean matrix M (Theorem 1 in Section 3).

Using the previous result, we will show that multicasting from the root of an arbitrary directed tree under the several models considered before can be solved in polynomial time (Section 4).

2 Broadcast problems and contention-free boolean matrices

In this section, we consider the 1-port edge-disjoint model only. A broadcast protocol $\mathcal{B}$ can be described by the list of all calls performed by $\mathcal{B}$. The construction of our broadcast algorithms for trees is based on the so-called *shadow* of a broadcast protocol. Let $T = (V, E)$ be any oriented tree, and let $\mathcal{B}$ be a broadcast protocol in T performing in r rounds.

Definition. The *shadow* of $\mathcal{B}$ on an arc $e \in E$ is the r-dimensional vector $(x_1, \dots, x_r)$, $x_i \in \{0, 1\}$, such that $x_i = 1$ if and only if there is a call passing through e at round i. The *restriction* of $\mathcal{B}$ on a vertex $u \in V$ with d outgoing links $e_1, \dots, e_d$ is the $d \times r$ matrix with entries in $\{0, 1\}$ such that there is a 1 at entry i, j if and only if u gives a call through link e_i at round j of $\mathcal{B}$. The *shadow* of $\mathcal{B}$ on $u \in V$ is then the r-dimensional vector $(x_1, \dots, x_r)$ such that $x_i = 1$ if and only if there is a 1-entry in column i of the restriction of $\mathcal{B}$ on u, and 0 otherwise.

The shadow of a broadcast protocol $\mathcal{B}$ on an arc e (resp. on a vertex u) is denoted by $\text{shad}(\mathcal{B}, e)$ (resp. $\text{shad}(\mathcal{B}, u)$). As shadows can be seen as binary representations of integers, we denote by $\text{bin}(\mathcal{B}, e)$ (resp. $\text{bin}(\mathcal{B}, u)$) the integer whose binary representation is $\text{shad}(\mathcal{B}, e)$ (resp. $\text{shad}(\mathcal{B}, u)$). Let $\mathcal{B}$ be a broadcast protocol in T performing in r rounds. For any vertex u, and for any link e, we have $\text{bin}(\mathcal{B}, u) \le 2^r - 1$, and $\text{bin}(\mathcal{B}, e) \le 2^r - 1$. The previous inequalities suggest the following definition.

Definition. Let $T = (V, E)$ be any directed tree, and let $\mathcal{B}$ be a broadcast protocol from the root in T. Let $u \in V$, and $e \in E$. $\mathcal{B}$ is said *lexicographically optimal* in u (resp. in e) if $\text{bin}(\mathcal{B}, u) \le \text{bin}(\mathcal{B}', u)$ (resp. $\text{bin}(\mathcal{B}, e) \le \text{bin}(\mathcal{B}', e)$) for any broadcast protocol $\mathcal{B}'$ in T.

2.1 Broadcasting in a path

Let P_n be the path of n nodes, and let u be one extremity of the path. An optimal broadcast protocol $\mathcal{B}$ from u performs in $d = \lceil \log_2 n \rceil$ rounds as follows. Let us label the nodes consecutively from 0 to $n-1$, starting at u labeled 0. If $n = 2^d$ then u calls node $n/2$ at the first round, and we are let with two simultaneous broadcasts from the extremity of a path of length 2^{d-1}. The algorithm is then defined by induction. Note that, in the case $n = 2^d$, the source u needs to call at every round so that the broadcast can complete in $\lceil \log_2 n \rceil$ rounds. In the general case, let us decompose $n-1$ is base 2, that is $n - 1 = \sum_{i=0}^{d-1} x_i 2^i$. The $\lceil \log_2 n \rceil$-rounds algorithm $\mathcal{B}$ performs as follows. Node u gives a call at round j, $j = 1, \ldots, d$, if and only if $x_{d-j} = 1$. Moreover, if u does give a call at round j, then it calls node v_j labeled $n - 1 - \sum_{i=d-j}^{d-1} x_i 2^i$. Upon reception of a call from u at round j, node v_j starts a broadcast to the sub-path of P_n composed of nodes lying between node v_j and node v_k where $k = n - 2 - \sum_{i=d-j-1}^{d-1} x_i 2^i$. This sub-path is of size 2^{d-j}.

Lemma 1 *$\mathcal{B}$ is lexicographically optimal in u.*

Proof. When an internal node receives a call at round j, $j = 1, \ldots, d$, it can inform at most $2^{d-j} - 1$ other nodes during the $d - j$ remaining rounds. Thus, any broadcast algorithm $\mathcal{B}'$ from u satisfies $\sum_{i=1}^{d} \mathrm{shad}(\mathcal{B}', u)_i \, 2^{d-i} \geq n - 1$. Since, by definition, $\sum_{i=1}^{d} \mathrm{shad}(\mathcal{B}, u)_i \, 2^{d-i} = n - 1$, we get $\mathrm{bin}(\mathcal{B}, u) \leq \mathrm{bin}(\mathcal{B}', u)$. ■

2.2 Broadcasting in a star

Let T be a star of p branches rooted at u, and let n_i be the number of nodes of the ith branch, $i = 1, \ldots, p$. T has $n = \sum_{i=1}^{p} n_i + 1$ nodes in total. Assume w.l.g. that $n_1 \geq n_2 \geq \ldots \geq n_p$. We denote by v_i the neighbor of u in the ith branch, and $e_i = (u, v_i)$, $i = 1, \ldots, p$. Let $q = \lceil \log_2(n_1 + 1) \rceil$. A broadcast from u to T takes at least q rounds.

Let $\mathcal{B}_i$ be the lexicographically optimal broadcast protocol from u to the ith branch, $i = 1, \ldots, p$, as defined in Section 2.1. Let M be the $p \times q$ matrix whose ith row is $\mathrm{shad}(\mathcal{B}_i, e_i)$. As it is defined, M is a "merging" of shadows, but it cannot be directly recognized as the restriction of a 1-port broadcast protocol from u to T since there might be contentions between the several shadows. For instance, if T is a star of two branches of one node each, then $\mathrm{shad}(\mathcal{B}_1, e_1) = \mathrm{shad}(\mathcal{B}_2, e_2) = [1]$, and $M = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ is not a restriction in u of a broadcast protocol since u would then have to call two nodes simultaneously, which is in contradiction with the 1-port hypothesis. However, M can be transformed in

$$M^* = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

which is the restriction of the broadcast protocol from u in T which performs as follows: at the first round u calls the node of the first branch, and, at the second round, u calls the node of the second branch. A similar example has been considered before when matrix M of Equation 1 was transformed into the matrix M^* of Equation 2.

Lemma 2 *Let T be a star of p branches of length at most $2^q - 1$ nodes each. Let M be the $p \times q$ matrix whose p rows are the p shadows $\mathrm{shad}(\mathcal{B}_i, e_i)$ of p broadcast algorithms from u to the p branches of T. Assume that all $\mathcal{B}_i$'s are lexicographically optimal in u. Then any contention-free* *version M^* of M determines a broadcast protocol $\mathcal{B}$ from u. Moreover, if M^* is minimal, then $\mathcal{B}$ is lexicographically optimal in u, and conversely.*

Proof. Let M^* be a contention-free version of M. To show that M^* is the restriction of a broadcast protocol $\mathcal{B}$ from u, we give a broadcast protocol from u as a function of the structure of M^*. For every r, $1 \leq r \leq p$, the rth row of M^* is larger than the corresponding row in M. Therefore, consider a particular row L^* of M^*, and let L be the corresponding row in M. Assume both rows correspond to the rth branch of the star. If $L^* = L$ then L^* is indeed the shadow of a broadcast protocol in the rth branch. Thus assume that $L \neq L^*$, and let i be the leftmost bit position for which L and L^* differ. Note that, in this case, $L_i^* = 1$ and $L_i = 0$ because $L_i^* \geq L_i$. L^* defines a broadcast protocol in the rth branch of the star as follows. From round 1 to round $i-1$, do as in the original broadcast protocol L. At round i, u calls its neighbor v_r in the rth branch. During the remaining rounds, u does not call the rth branch anymore. However, v_r simulates the calls of u according to L. That is, if u calls w at round $j > i$ in L, then v_r calls node w at round j. Therefore, L^* is the shadow of a broadcast protocol in the rth branch of the star. M^* has at most one 1-entry per column, thus $\mathcal{B}$ satisfies the 1-port model. Also, if the shadow of M^* is minimum, then $\mathcal{B}$ is lexicographically optimal.

Let M^* be the restriction in u of a broadcast protocol $\mathcal{B}$ from u. Assume that $\mathcal{B}$ is lexicographically optimal in u. To satisfy the 1-port model, there is at most one 1-entry in each column of M^*. Moreover, from Lemma 1, since M^* allows to broadcast in each branch, the rth row of M^* is larger than the rth row of M. Indeed, each row of M is the shadow of a lexicographically optimal broadcast protocol from u. Finally, since $\mathcal{B}$ is lexicographically optimal in u, the shadow of M^* is minimum. Therefore, M^* is a minimal contention-free version of M. ■

According to the previous lemma, the key to find an optimal broadcast protocol in a star is to solve the minimal contention-free matrix problem as stated in Section 1.4. Actually, we will see in Section 4 that solving the minimal contention-free matrix problem is also the key to solve the broadcast and multicast problems in any arbitrary directed tree. Therefore, the next section is entirely devoted to solving the minimal contention-free matrix problem.

3 A polynomial algorithm for the minimal contention-free boolean matrix problem

Let M be a $p \times q$ boolean matrix. Our algorithm will transform M in a $p \times q^*$ minimal contention-free version of M denoted by M^*. The total number of columns of any minimal contention-free version of M is denoted by $\mathcal{C}(M)$. $\mathcal{C}(M)$ and M^* will be computed by a sequence of elementary operations of two types: *insertion* of a zero-column at position 0, and *shifting* of an existing zero-column from position $t-1$ to position t (columns are labeled from left to right). The shift operation has an important consequence on the 1-entries of the matrix. When a zero-column is shifted one position to the right, from position $t-1$ to position t, that is when the two columns $t-1$ and t are exchanged, the entries of the matrix are modified according to the following rule:

Rule 1. for every i, $1 \leq i \leq p$, if there is a 1-entry originally at position t of row i, then, after the exchange, all 1-entries of row i at position $> t$ are switched to 0.

This rule comes from the simple fact that, for any k, $2^{k+1} > \sum_{i=0}^{k} a_i 2^i$ for any $a_i \in \{0,1\}$, $i = 0, \ldots, k$. Therefore, any row modified according to rule 1 is larger than the original row, whatever are the entries of the row left to position t.

Using rule 1, our algorithm is formally described in Algorithm 1, and an example is provided on Figure 2. Informally, Algorithm 1 performs as follows. The q columns of M are considered from left to right. Problems occur when there are two or more 1-entries in the current column (Instruction 6). On Figure 2(a), this occurs at column 4 since there is a single 1-entry in each of the three leftmost columns of M. Algorithm 1 then tries to increase the number of zero-columns by shifting existing zero-columns from their current position to the left of the current column, and applying rule 1 (Instruction 13). Possibly, one zero-column is inserted at position 0 (Instruction 18). The goal is to obtain enough zero-columns on the left of the current column to spread out the contending 1's over these zero-columns. On Figure 2(a), there is no zero-column at that time of the algorithm, and thus a zero-column is inserted at position 0, as shown on Figure 2(b). Then the two first columns are exchanged. This exchange has a major consequence: according to rule 1, all 1-entries, but the leading 1, of the first row are switched to 0. This creates a new zero-column, and one of the two contending 1's of column 4 vanishes (see Figure 2(c)). The algorithm then considers position 5 (now the 6th column from the left). Four 1-entries are contending at position 5 of the matrix. The rightmost zero-column is then shifted to the right. It is worth to notice that it is always the rightmost zero-column not next to the current column that is considered. Choosing this column instead of any zero-column has a tremendous effect of the shadow of the resulting matrix. The effect of this shift in the example is to delete one contending 1-entry (see Figure 2(d)). The zero-column is then shifted once more to the right. Again, it deletes one contending 1-entry (see Figure 2(e)). Once there are enough zero-columns to solve all conflicts between 1-entries in the current column, the contending 1's are spread out over these columns. Note that if after all possible shifting, there is still not enough zero-columns to absorb the contenting 1's, then some zero-columns are inserted again (Instruction 23). In our example, there are one zero-column and two contending 1's, so there is no need to insert new zero-column (see Figure 2(e)). Now, the choice of the unique 1-entry of column 5 which is *not* moved to a zero-column matters. Algorithm 1 keeps in place the 1-entry which corresponds to the row with the minimum lexicographic order, starting from the current column (Instruction 25). In our example, it means that the 1-entry of row 5 will be let in place, while the 1-entry of row 4 will be moved to the zero-column. Indeed, from the current position, row 4 is 110 whereas row 5 is 100. After that, we are let with the matrix on Figure 2(f) in which the last 1-entry of row 4 has been switched to 0. The effect of the choice of the smallest row is to postpone other conflicts with this row as far as possible. In the example, it transforms the penultimate column into a zero-column. Therefore, the conflict appearing at position 7 can be easily solved. We claim that the resulting matrix is a minimal contention-free version of the original matrix. Its shadow is $(10111111)_2$.

Remark. Note that it is not difficult to approximate $\mathcal{C}(M)$ up to an additive factor of 1. Indeed, let M_0 be the matrix obtained from M by switching all 1-entries, but the leading 1-entry of each row, to zero. The reader can check that computing $\mathcal{C}(M_0)$ and M_0^* is easy. For instance, on the example of Figure 2, $\mathcal{C}(M_0) = 6$, and M_0^* is the identity matrix. Moreover, we have $\mathcal{C}(M_0) \leq \mathcal{C}(M) \leq \mathcal{C}(M_0)+1$. Indeed, eventually, we have to solve all contentions induced by leading 1's, that is $\mathcal{C}(M) \geq \mathcal{C}(M_0)$. Now, let M_1 be the $p\times(q+1)$ matrix obtained from M_0 by adding one zero-column at position $q+1$. All rows of M_1 are larger than the corresponding rows of M, therefore a minimal contention-free version of M_1 will give a contention-free version of M. Therefore, $\mathcal{C}(M) \leq \mathcal{C}(M_1) = \mathcal{C}(M_0) + 1$. Unfortunately, approximating $\mathcal{C}(M)$ up to an additive factor of 1 is not enough to provide a good approximation algorithm for the broadcast time of a tree. Indeed, we will see in Section 4 that one

Algorithm 1

```
For i:=1 to q do
/* We sparse the columns from column 1 to column q */
    C_i := current column;
    if C_i is a zero-column then
        Z := Z ∪ {C_i};
        /* Z currently denotes the set of zero-columns left to the current column */
    else
        if there is more than a single 1-entry in C_i then
            nb_1 := # 1's in C_i;
            W := set of consecutive zero-columns immediately to the left of C_i;
            not_yet_inserted := true;
            While nb_1 − 1 > |W| and (Z ≠ W or not_yet_inserted) do
            /* while there is still not enough zero-column immediately to the left of C_i,  */
            /* but still some zero-columns that can be pushed immediately to the left of C_i */
                Z' := Z \ W;
                c := rightmost zero-column in Z';
                Shift c one column to the right, and apply rule 1;
                Z := set of zero-columns left to C_i;
                W := set of consecutive zero-columns immediately to the left of C_i;
                nb_1 := # 1's in C_i;
                if nb_1 − 1 > |W| and W = Z and not_yet_inserted then
                /* One needs to insert a zero-column */
                    Insert a zero-column at position 0;
                    not_yet_inserted := false;
                    Z := set of zero-columns left to C_i;
                EndIf
            EndWhile
            /* Now, either there is enough zero-columns to solve all contentions, */
            /* or all zero-columns are immediately to the left of C_i */
            if nb_1 − 1 > |W| then insert nb_1 − |W| − 1 zero-columns left to C_i;
            /* The nb_1 1's will be spread out over the zero-columns of W */
            Truncate each row with a 1 in C_i in order to keep only entries to the right of C_i;
            ℓ := index of the row of minimum lexicographic order among the truncated rows;
            W' := nb_1 − 1 rightmost columns of W
            Spread out the nb_1 1's of C_i over W'; the 1-entry of row ℓ stays in C_i;
            Z := set of zero-columns left to the current column;
        EndIf
    EndIf
EndFor
```

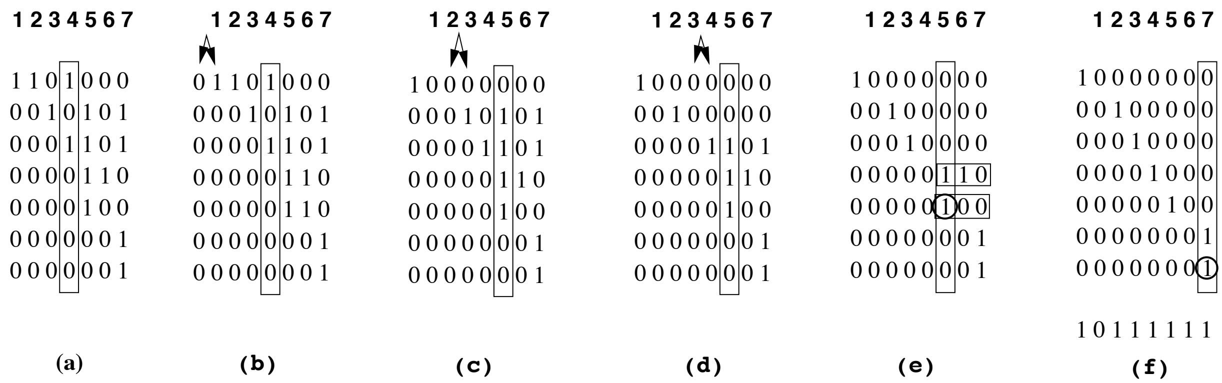

Figure 2: An example of the execution of Algorithm 1.

often need to solve the minimal contention-free matrix problem at all levels of the tree, and thus one would cumulate the error at each level.

Theorem 1 *Algorithm 1 is an $O(q(p+q))$-time algorithm that computes a minimal contention-free version of any $p \times q$ boolean matrix.*

Lemma 3 *Algorithm 1 is an $O(q(p+q))$-time algorithm.*

Proof. First, let us show that Algorithm 1 performs in $O(q(q+p))$ steps. The *for*-loop is executed q times, but the part "else" (Instruction 5) is not performed more than p times because there are p rows, and solving a contention between 1-entries creates at least one row whose all entries are 0 after the current position. Let i be an index of the *for*-loop for which there is a contention. From what was said before, there are at most p such indices. Let k_i be the number of contending 1-entries: $\sum_i k_i \leq 2(p-1)$. All instructions before the *while*-loop do not require more than $O(p+q)$ time units. The *while*-loop is executed at most $q\ k_i$ times because each execution of the loop corresponds to a right-shift of a zero-column, and one cannot move a zero-column to the right more than q times, this for each of the k_i 1-entries. Actually, one can slightly modify the algorithm so that there are no more than q right-shifts in total, for all conflicts. Indeed, when shifting the zero-columns to the right, one can jump columns that were already exchanged with a zero-column since rule 1 was already applied. Altogether, rule 1 cannot be applied more than q times. Application of rule 1 has a cost of $O(q)$ since at most one row is updated after a right-shift. All other instructions inside the *while*-loop have a cost of $O(p+q)$. Instruction 25 has a cost of $O(q\ k_i)$, same as Instruction 27. Therefore, in total, the complexity is $O(q(q+p)+\sum_i q\ k_i)$ that is $O(q(q+p))$. ■

The fact that Algorithm 1 computes a minimal contention-free version of any $p \times q$ boolean matrix M is based on the following lemmas.

Lemma 4 *If every rows A_i and B_i of two matrices A and B satisfy $A_i \leq B_i$, then* $\text{shad}(A^*) \leq \text{shad}(B^*)$.

Proof. Straightforward. ■

Notation. Given two matrices A and B of the same number of rows p, and of respectively q and q' columns, AB denotes the $p \times (q+q')$ matrix obtained by putting A and B next to each other.

Lemma 5 $\mathrm{shad}((AB)^*) \le \mathrm{shad}(\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}^*)$.

Proof. Let

$$\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}^* = \begin{pmatrix} X & 0 \\ Y & Z \end{pmatrix}.$$

For any row i, we have $X_i \ge A_i$. Also, for any row j, we have $Y_j Z_j \ge B_j$. $X' = X + Y$ has at most one 1-entry per column, that is $X'Z$ has at most one 1-entry per column. Moreover, $X'Z$ satisfies that, for any row i, $(X'Z)_i \ge (AB)_i$. Since $\mathrm{shad}(X'Z) = \mathrm{shad}(\begin{pmatrix} X & 0 \\ Y & Z \end{pmatrix})$, the lemma holds. ■

Note that the inequality in Lemma 5 can be strict. For instance

$$\mathrm{shad}(\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}^*) = 101$$

whereas

$$\mathrm{shad}(\begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}^*) = 111.$$

Lemma 6 *If* $(AB)^* = AB'$ *where* B' *has the same number of columns as* B*, then* $\mathrm{shad}((AB)^*) = \mathrm{shad}(\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}^*)$.

Proof. By lemma 5, we just have to show that $\mathrm{shad}((AB)^*) \ge \mathrm{shad}(\begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix}^*)$. Let

$$C = \begin{pmatrix} A & 0 \\ 0 & B \end{pmatrix} \text{ and } C' = \begin{pmatrix} A & 0 \\ 0 & B' \end{pmatrix}.$$

We have, for any row i, $C'_i \ge C_i$, and there is at most one 1-entry per column of C'. Since $\mathrm{shad}(C') = \mathrm{shad}((AB)^*)$, the lemma holds. ■

Lemma 7 *Let* X, X', Y, Y' *be 1-dimensional vectors, and let* A *and* A' *be multi-dimensional matrices. Let*

$$M = \begin{pmatrix} X' & 0 & 1 & X \\ Y' & 0 & 1 & Y \\ A' & 0 & 0 & A \end{pmatrix}, \quad M_X = \begin{pmatrix} X' & 0 & 1 & X \\ Y' & 1 & 0 & 0 \\ A' & 0 & 0 & A \end{pmatrix} \text{ and } M_Y = \begin{pmatrix} X' & 1 & 0 & 0 \\ Y' & 0 & 1 & Y \\ A' & 0 & 0 & A \end{pmatrix}$$

where there is at most one 1-entry per column in $\begin{pmatrix} X' \\ Y' \\ A' \end{pmatrix}$. *Then*

$$\mathrm{shad}(M^*) = \min(\mathrm{shad}(M_X^*), \mathrm{shad}(M_Y^*)).$$

Proof. From Lemma4, $\text{shad}(M^*) \leq \min(\text{shad}(M^*_X), \text{shad}(M^*_Y))$. The equality holds because at least one 1-entry in the block $\begin{pmatrix} X' & 0 & 1 \\ Y' & 0 & 1 \end{pmatrix}$ must be moved to the left. ■

Lemma 8 *With the same notations as in lemma 7, if* $Y \leq X$ *then* $\text{shad}(M^*_Y) \leq \text{shad}(M^*_X)$.

Proof. Assume for the purpose of contradiction that $\text{shad}(M^*_Y) > \text{shad}(M^*_X) = \text{shad}(M^*)$. We get

$$M^*_X = \begin{pmatrix} X' & 0 & 1 & X'' \\ Z & z & 0 & 0 \\ B' & b & 0 & B \end{pmatrix}$$

where $Zx \geq Y'1$, $(B'b0B)_i \geq (A'00A)_i$ for every row i of these matrices, and $X'' \geq X$. The first row of M^*_X is necessarily of the form $X'01X''$ because if this row is $\geq X'100$, then $\text{shad}(M^*_Y) \leq \text{shad}(M^*_X)$.

By Lemma 6, we get

$$\text{shad}(M^*_X) = \text{shad}(\begin{pmatrix} X' & 0 & 0 & 0 \\ 0 & 0 & 1 & X' \\ Y' & 1 & 0 & 0 \\ A' & 0 & 0 & A \end{pmatrix}^*) \tag{3}$$

On the other hand, by Lemma 5, $\text{shad}(M^*_Y) > \text{shad}(M^*_X)$ implies that

$$\text{shad}(M^*_X) < \text{shad}(\begin{pmatrix} X' & 1 & 0 & 0 \\ Y' & 0 & 0 & 0 \\ 0 & 0 & 1 & Y \\ A' & 0 & 0 & A \end{pmatrix}^*)$$

That is, by Lemma 4,

$$\text{shad}(M^*_X) < \text{shad}(\begin{pmatrix} X' & 1 & 0 & 0 \\ Y' & 0 & 0 & 0 \\ 0 & 0 & 1 & X \\ A' & 0 & 0 & A \end{pmatrix}^*) \tag{4}$$

Equations 3 and 4 give

$$\text{shad}(\begin{pmatrix} X' & 0 & 0 & 0 \\ 0 & 0 & 1 & X' \\ Y' & 1 & 0 & 0 \\ A' & 0 & 0 & A \end{pmatrix}^*) = \text{shad}(\begin{pmatrix} X' & 0 & 0 & 0 \\ 0 & 0 & 1 & X'' \\ Z & z & 0 & 0 \\ B' & b & 0 & B \end{pmatrix}) < \text{shad}(\begin{pmatrix} X' & 1 & 0 & 0 \\ Y' & 0 & 0 & 0 \\ 0 & 0 & 1 & X \\ A' & 0 & 0 & A \end{pmatrix}^*)$$

Assume $b = 0$ and $z = 0$. Thus $Z > Y'$, and therefore $B'_i > A'_i$ for at least one row of A' and B'. Let us consider the row i such that $B'_i > A'_i$ and such that the rightmost bit position for which there is a 1-entry in B'_i is minimum. Let k and k' be the rightmost bit positions for which there is a 1-entry in Z and B'_i respectively. If $k > k'$ then replacing the row $(B'000)_i$ by $(A'100)_i$ would decrease the shadow. If $k < k'$ then replacing the $Z000$ by $Y'100$ would also decrease the shadow. Therefore $z \neq 0$. However, $z = 1$ implies that the matrix

$$\begin{pmatrix} X' & 1 & 0 & 0 \\ Z & 0 & 0 & 0 \\ 0 & 0 & 1 & X'' \\ B' & 0 & 0 & B \end{pmatrix}$$

is a contention-free version of

$$\begin{pmatrix} X' & 1 & 0 & 0 \\ Y' & 0 & 0 & 0 \\ 0 & 0 & 1 & X \\ A' & 0 & 0 & A \end{pmatrix}$$

yielding a contradiction. Thus $b = 1$, but then the same argument as for $z = 1$ yields another contradiction. Therefore $\text{shad}(M_Y^*) \leq \text{shad}(M_X^*)$. ■

Lemma 9 *Let $M = AxByC$ where A is a matrix with at most one 1-entry per column, x is a zero-column, B is a matrix with exactly one 1-entry per column, y is a column with two contending 1-entries, and C is an unspecified boolean matrix. Let M' be the matrix resulting from M after an exchange between x and the first column of B. We have* $\text{shad}(M^*) = \text{shad}(M'^*)$.

Proof. We already know that $\text{shad}(M^*) \leq \text{shad}(M'^*)$. The proof of the other inequality is by induction on the number of columns q of B. Assume $q = 1$, that is

$$M = \begin{pmatrix} A_1 & 0 & 1 & 0 & C_1 \\ A_2 & 0 & 0 & 1 & C_2 \\ A_3 & 0 & 0 & 1 & C_3 \\ A_4 & 0 & 0 & 0 & C_4 \end{pmatrix}.$$

Let

$$M^* = \begin{pmatrix} A'_1 & a_1 & a_2 & a_3 & C'_1 \\ A'_2 & b_1 & b_2 & b_3 & C'_2 \\ A'_3 & c_1 & c_2 & c_3 & C'_3 \\ A'_4 & d_1 & d_2 & d_3 & C'_4 \end{pmatrix}$$

with $A'_1 a_1 a_2 a_3 \geq A_1 010$, $A'_2 b_1 b_2 b_3 \geq A_2 001$, and $A'_3 c_1 c_2 c_3 \geq A_3 001$. If $a_1 a_2 a_3 \geq 100$, then, by Lemma 4, $\text{shad}(M^*) = \text{shad}(M'^*)$. If $010 \leq a_1 a_2 a_3 < 100$, then we can assume w.l.g. that $b_1 b_2 b_3 \geq 100$. Actually, we can assume that

$$M^* = \begin{pmatrix} A'_1 & 0 & 1 & 0 & C'_1 \\ A'_2 & 1 & 0 & 0 & 0 \\ A'_3 & 0 & 0 & 1 & C'_3 \\ A'_4 & d_1 & d_2 & d_3 & C'_4 \end{pmatrix}.$$

Let

$$M'' = \begin{pmatrix} A'_1 & 1 & 0 & 0 & 0 \\ A'_2 & 0 & 1 & 0 & 0 \\ A'_3 & 0 & 0 & 1 & C'_3 \\ A'_4 & d_1 & d_2 & d_3 & C'_4 \end{pmatrix}.$$

We have $\text{shad}(M''^*) \leq \text{shad}(M^*)$. Now,

$$M' = \begin{pmatrix} A'_1 & 1 & 0 & 0 & 0 \\ A'_2 & 0 & 0 & 1 & C'_2 \\ A'_3 & 0 & 0 & 1 & C'_3 \\ A'_4 & d_1 & d_2 & d_3 & C'_4 \end{pmatrix},$$

and we get $\text{shad}(M'^*) \leq \text{shad}(M''^*) \leq \text{shad}(M^*)$, that is the lemma holds for $q = 1$.

Assume the lemma holds for every q, $1 \leq q < q_0$, and let us show that it holds for q_0. A 1-entry in $AxBy$ must be moved to the left. For any move of a 1-entry in A, one can find a move of a 1-entry in B that preserves the shadow. Therefore, one can assume that it is a 1-entry in B that is moved to the left. Moreover, we can assume that this 1-entry, denoted by $\mathbf{1}$, is moved in xB.

- If $\mathbf{1}$ is moved in B at least one column to the right of the first column of B, then one can apply the induction hypothesis, that is exchanging the first column of B with x, and then putting back $\mathbf{1}$ to its original position, without changing the shadow.
- If $\mathbf{1}$ is moved to the first column of B, then we apply Lemma 7, and then putting back $\mathbf{1}$ to its original position. The result of these operations is just as exchanging x with the first column of B. The shadow is preserved.
- If $\mathbf{1}$ is moved in x, then we can exchange $\mathbf{1}$ with the 1-entry on the first column of B, and then put back $\mathbf{1}$ to its original position, without changing the shadow.

Thus the result hold for q_o too. ■

We have now enough material to prove Theorem 1.

Proof of Theorem 1. Algorithm 1 constructs a finite sequence of matrices $M_0 = M, M_1, \ldots, M_k$, such that M_i is obtained from M_{i-1} either by shifting a zero-column to the right, or by distributing 1-entries over zero-columns. Lemmas 8 and 9 (generalized to an arbitrary number of contending 1-entries) insure that $\text{shad}(M_i^*) = \text{shad}(M_{i-1}^*)$, that is $\text{shad}(M^*) = \text{shad}(M_k^*)$. Since M_k is a contention-free version of M, we get that $\text{shad}(M^*) = \text{shad}(M_k)$. ■

4 Application to the broadcasting problem in tree-networks

4.1 All-port model

As an example of application of Theorem 1 to the multicast problem in trees, let us consider the following problem. We are given a directed tree whose arcs are oriented from the root u toward the leaves, and a set D of nodes of the tree. We want to compute the minimum number of rounds that are required to multicast an information from u to all nodes in D. We are considering the all-port edge-disjoint communication model. In this context, Cohen [4] has shown that there exists a polynomial-time algorithm that computes an optimal broadcast protocol from u to all nodes of T. To directly extend this algorithm to the multicast problem, we would make use of intermediate nodes that are not destination nodes, and this is not desirable in general. Combining Theorem 1 and the protocol in [4] allows to overcome that problem.

Corollary 1 *There exists a polynomial-time algorithm that computes an optimal multicast protocol from any source u to any destination set D in any directed tree under the all-port edge-disjoint model, and such that only the source and the destination nodes participate to the protocol.*

Sketch of the proof. The algorithm in [4] proceeds bottom-up from the leaves to the source. Each node x has a list of calls stating when and to whom x gives a call in its subtree, and when and by who x is informed. This list is constructed from the lists of all the children of x in the tree.

When the multicast problem is considered, the algorithm fails in the following case: assume a node $x \in D$ has one of its children y not in D, and that y has k children $z_1, \ldots, z_k$ in D. The algorithm in [4] requires the help of $y \notin D$. If we do not want y to be involved in the protocol, then x can be required to successively call z_1, z_2 up to z_k. More importantly, x cannot give a call simultaneously in the subtrees of the z_i's, whereas y is able to do so in the all-port model. Therefore, giving the set of calls of y, one must schedule these calls so that x can simulate the behavior of y. One can represent the set of calls from y to the subtrees of the z_i's by a matrix M such that $M_{i,j} = 1$ if and only if y gives a call to the subtree of z_i at round j. Theorem 1 gives a polynomial-time algorithm to schedule optimally these calls. Note that since this procedure must be applied at all the levels of the tree, one does not only need to compute a contention-free version of M with the minimum number of columns (i.e., number of rounds), but one also need to minimize the shadow. ■

4.2 1-port model

We are currently working on an extension of Theorem 1 to make use of this result in the 1-port model. Again, the idea is to construct the protocol bottom-up from the leaves to the root. To make clear why Theorem 1 needs to be slightly adapted, let us consider the simple case of a *fork*, that is a particular type of directed tree in which the root u has a single child v which is the root of a star of p branches. Let X_i be the shadow on v of an optimal broadcasting algorithm applied to the ith branch, $i = 1, \ldots, p$, and let M be the $p \times q$ array whose ith row is X_i.

A non necessarily optimal broadcast protocol in the 1-port edge-disjoint model consists in two phases: first u informs v, then v informs the p branches according to a minimal contention-free version of M. This protocol may be suboptimal because it can be more efficient to have both u and v informing the p branches (in the 1-port edge-disjoint model, u and v can call two distinct branches simultaneously). So the question is when to inform v? Before v is informed, u only can inform the branches, and there is a contention in M when there is more than a single 1-entry on a column. After v has been informed, there is a contention in M when there is more than *two* 1-entries in a column. For instance, consider the following fork: u is connected to v, and v has two branches, composed of two nodes w_1, w_2, and four nodes w'_1, w'_2, w'_3, w'_4, respectively. One can broadcast from u in three rounds in this fork under the 1-port edge-disjoint model: (1) u calls w'_1, (2) u calls w_1, and w'_1 calls w'_3, and (3) u calls v, w'_1 calls w'_2, w'_3 calls w'_4, and w_1 calls w_2. If u calls v before the third round, then one more round in required.

The adaptation of Algorithm 1 to this situation is very specific to the multicast problem. This is why we do not provide more details about it in this paper.

References

[1] T. Ballardie, P. Francis, and J. Crowcroft, *Core based tree (CBT): an architecture for scalable inter-domain multicast routing*, in proceedings of SIGCOMM, ACM press, 1993, pp. 85–95.

[2] A. Bar-Noy, S. Guha, J. Naor, and B. Schieber, *Multicasting in heterogeneous networks*, in ACM Symposium on Theory of Computing (STOC), ACM press, 1998.

[3] B. Birchler, A. Esfahanian, and E. Torng, *Information dissemination in restricted routing networks*, in proceedings of the International Symposium on Combinatorics and Applications, 1996, pp. 33–44.

[4] J. Cohen, *Broadcasting, multicasting and gossiping in trees under the all-port line model*, in 10th ACM Symposium on Parallel Algorithms and Architectures (SPAA), ACM press, 1998, pp. 164–171.

[5] J. Cohen, P. Fraigniaud, J.-C. Konig, and A. Raspaud, *Optimized broadcasting and multicasting protocols in cut-through routed networks*, IEEE Transaction on Parallel and Distributed Systems, (to appear). (See also IPPS '97.).

[6] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C.-G. Liu, and L. Wei, *An architecture for wide-area multicast routing*, in SIGCOMM, ACM press, 1994, pp. 126–135.

[7] C. Diot, W. Dabbous, and J. Crowcroft, *Multipoint communication: a survey of protocols, functions, and mechanisms*, IEEE journal on selected areas in communications, 15 (1997), pp. 277–290.

[8] A. Farley, *Minimum-time line broadcast networks*, Networks, 10 (1980), pp. 59–70.

[9] A. Farley and A. Proskurowski, *Broadcasting in trees with multiple originators*, SIAM Journal Alg. Disc. Meth., 2 (1981), pp. 381–386.

[10] P. Fraigniaud and E. Lazard, *Methods and Problems of Communication in Usual Networks*, Discrete Applied Mathematics, 53 (1994), pp. 79–133.

[11] P. Fraigniaud and S. Vial, *Approximation algorithms for broadcasting and gossiping*, Journal of Parallel and Distributed Computing, 43 (1997), pp. 47–55.

[12] H. Harutyunyan. Doctoral Thesis (in Russian), Armenia.

[13] S. Hedetniemi, S. Hedetniemi, and A. Liestman, *A survey of gossiping and broadcasting in communication networks*, Networks, 18 (1986), pp. 319–349.

[14] J. Hromković, R. Klasing, B. Monien, and R. Peine, *Dissemination of information in interconnection networks (broadcasting and gossiping)*, in Combinatorial Network Theory, D.-Z. Du and D. F. Hsu, eds., Kluwer Academic, 1995, pp. 125–212.

[15] J. Hromković, R. Klasing, and E. Stohr, *Dissemination of information in vertex-disjoint paths mode*, Computer and Artificial Intelligence, 15 (1996), pp. 295–318.

[16] J. Hromković, R. Klasing, E. Stohr, and W. H., *Gossiping in vertex-disjoint paths mode in d-dimensional grids and planar graphs*, Information and Computation, 123 (1995), pp. 17–28.

[17] J. Hromković, R. Klasing, W. Unger, and H. Wagener, *Optimal algorithms for broadcast and gossip in the edge-disjoint path modes*, Information and Computation, 133 (1997), pp. 1–33.

[18] G. Kortsarz and D. Peleg, *Approximation algorithms for minimum time broadcast*, in Proceedings of the Israel Symposium on Theory of Computer Science, no. 601 in Lecture Notes in Computer Science, Springer-Verlag, 1992.

[19] R. Labahn, *Extremal broadcasting problems*, Discrete Applied Mathematics, 23 (1989), pp. 139–155.

[20] M. Middendorf, *Minimum broadcast time is NP-complete for 3-regular planar graphs and deadline 2*, Information Processing Letters, 46 (1993), pp. 281–287.

[21] W. Mostafa and M. Singhal, *A taxonomy of multicast protocols for internet applications*, Computer communications, 20 (1998), pp. 1448–1457.

[22] A. Proskurowski, *Minimum broadcast trees*, IEEE Transactions on Computers, c-30 (1981), pp. 363–366.

[23] R. Ravi, *Rapid rumor ramification: approximating the minimum broadcast time*, in 35-th Symposium on Foundations of Computer Science, IEEE Computer Society Press, 1994, pp. 202–213.

[24] P. Scheuermann and G. Wu, *Heuristic algorithms for broadcasting in point-to-point computer networks*, IEEE Transactions on Computers, C-33 (1984), pp. 804–811.

[25] P. Slater, E. Cockayne, and S. Hedetniemi, *Information dissemination in trees*, SIAM Journal on Computing, 10 (1981), pp. 692–701.

[26] P. Winter, *Steiner problem in networks: a survey*, IEEE Networks, 17 (1987), pp. 129–167.