Nordic Journal of Computing 1(1994), 332–345.

SPARSE NETWORKS SUPPORTING EFFICIENT RELIABLE BROADCASTING

BOGDAN S. CHLEBUS[*] KRZYSZTOF DIKS[*,†]
Instytut Informatyki, Uniwersytet Warszawski
ul. Banacha 2, 02-097 Warszawa, Poland

ANDRZEJ PELC[‡]
Département d'Informatique, Université du Québec à Hull
C.P. 1250, succ."B", Hull, Québec J8X 3X7, Canada

Abstract. Broadcasting concerns transmitting information from a node of a communication network to all other nodes. We consider this problem assuming that links and nodes of the network fail independently with given probabilities $p < 1$ and $q < 1$, respectively. For a positive constant ε, broadcasting in an n-node network is said to be ε-safe, if source information is transmitted to all fault-free nodes with probability at least $1 - n^{-\varepsilon}$. For any $p < 1$, $q < 1$ and $\varepsilon > 0$ we show a class of n-node networks with maximum degree $O(\log n)$ and ε-safe broadcasting algorithms for such networks working in logarithmic time.

1. Introduction

Broadcasting concerns transmitting information from a node of a communication network to all other nodes. It is closely related to gossiping where each node of a network holds a piece of information and all nodes need to learn the total information. Messages may be directly transmitted to adjacent nodes only, and every node may communicate with at most one neighbor in a unit of time.

The following are two important parameters of a broadcasting or gossiping algorithm: the total time used and the total number of two-party transmissions ("phone calls"). Many papers have been devoted to the study of algorithms optimizing one or both of these parameters. An extensive bibliography can be found in [10].

Recently a lot of attention has been devoted to broadcasting and gossiping in the presence of faulty links [2–8]. Two alternative assumptions about faults are usually made: either an upper bound k on the total number of faults is supposed [2, 7, 8] or it is assumed that links fail independently with fixed probability p [3–6]. If an upper bound is imposed and the worst case

* partly supported by EC Cooperative Action IC-1000 (project ALTEC: Algorithms for Future Technologies).
† partly supported by NSERC International Fellowship.
‡ partly supported by NSERC grant OGP0008136

Received August 1993. Accepted July 1994.

is considered, the maximum number of faults that can be tolerated must be smaller than the connectivity of the network. Thus, for large networks, the stochastic approach seems to be more realistic.

In the presence of faults two ways of constructing a broadcasting algorithm are possible. One way is non-adaptive, that is, all calls have to be predetermined by specifying in advance which pairs of nodes communicate in a given time unit, without the possibility of modifying the sequence of calls depending on which calls succeeded and which failed. Mostly this approach has been studied in literature [2, 3, 6, 7, 8]. (In [8] it was called static). Another way of broadcasting in the presence of faults is adaptive, that is, every node can decide which node it should call in a given time unit, depending on the outcome of previous calls. However, in making this decision, a node can only take advantage of the information currently available to it, that is, no existence of a central monitor supervising the execution of the scheme is assumed. Adaptive algorithms were studied in [4, 5].

If random faults are assumed, we cannot expect to perform broadcasting with absolute certainty and thus we look for highly reliable algorithms. Let ε be a positive constant. A broadcasting algorithm working for an n-node network is called ε-safe if the probability of broadcasting information throughout the network is at least $1 - n^{-\varepsilon}$.

Efficient ε-safe broadcasting algorithms working under assumption of random link failures and fault-free nodes were studied in [3–5, 11]. Bienstock [3] constructed n-node networks with $O(n \log n)$ links for which a non-adaptive ε-safe broadcasting algorithm could be shown to work in logarithmic time. His construction, however, is quite involved.

In this paper we study ε-safe broadcasting algorithms working under a more general assumption: both links and nodes fail independently with given probabilities $p < 1$ and $q < 1$, respectively. Under this scenario the aim of the algorithm is to transmit information to all fault-free nodes. For any p, $q < 1$ and $\varepsilon > 0$ we construct simple n-node networks with maximum degree $O(\log n)$; for those networks we show a non-adaptive ε-safe broadcasting algorithm working in logarithmic time. The algorithm uses $O(n \log n)$ calls. Thus, using a simpler construction we get the same performance as in [3] under a more general fault model. Although we consider only permanent faults, our non-adaptive algorithm works also for other types of failures, such as fail-stop faults. We also construct an adaptive ε-safe broadcasting algorithm working in worst case logarithmic time and using an expected linear number of calls. Finally, in case of fault-free nodes ($q = 0$), we construct an adaptive ε-safe broadcasting algorithm working in worst case logarithmic time and using a linear number of calls in the worst case. All these characteristics are of minimal possible order of magnitude.

The paper is organized as follows: in section 2 we give a precise description of the communication model used in this paper, in section 3 we construct the family of sparse networks supporting our broadcasting algorithms, in section 4 we describe the algorithms, and in section 5 their reliability and efficiency are analyzed. Section 6 contains conclusions.

We use the following notation. For any random event E, $\overline{E}$ denotes it complement. For a set X, $|X|$ denotes its size. For any positive number x, we write $\log x$ instead of $\log_2 x$.

2. The Model

The communication network is represented as a simple undirected graph whose vertices are nodes of the network and edges are communication links. Information to be broadcasted is initially stored in a node called the source. It will be referred to as source information. Links fail with fixed probability $p < 1$ and nodes other than the source fail with fixed probability $q < 1$. All failures are stochastically independent and the fault status of all components is permanent, that is, it does not change during the execution of the algorithm. The source is assumed fault-free.

We consider only synchronous algorithms. A basic step of a broadcasting algorithm is an attempt made by a node v to communicate with its neighbor w. Such an attempt takes a unit of time and we say in this case that v calls w. In our algorithms a node v can call at most one neighbor or be called by at most one neighbor in a unit of time, these two possibilities being exclusive. A call from v to w is successful if v, w and the joining link are fault-free. During such a call, the node which already has source information, transmits it to the other node and some control messages can also be exchanged between v and w. When a call from a fault-free node v to w does not succeed, v becomes aware of it but it does not know the reason of failure (faulty link, faulty destination node or both). In this case no information is transmitted. Faulty nodes do not make calls: if a call from a faulty node is scheduled by an algorithm, it is not executed.

We consider two types of broadcasting algorithms: *non-adaptive*, in which the sequence of calls made by every node is given in advance, and *adaptive*, in which each fault-free node can decide which node to call in a given time unit using information currently available to it.

We say that a broadcasting algorithm is successful if upon its completion all fault-free nodes get the source information. Let ε be a positive constant. A broadcasting algorithm working for an n-node network is called ε-safe, if it succeeds with probability at least $1 - n^{-\varepsilon}$. Two complexity measures of a broadcasting algorithm are considered in this paper: the number of time units used by the algorithm and the total number of calls (both successful and not) made during its execution. For non-adaptive algorithms these parameters are fixed in advance, while for adaptive algorithms there are two natural ways of measuring complexity: counting worst case or expected value of running time and of the number of calls.

3. Construction of Networks

In this section we describe n-node networks with maximum degree $O(\log n)$ for which efficient ε-safe broadcasting algorithms will be presented later.

Let $c \geq 2$ be a positive integer defined later. For each $n \geq 2c$ we define an n-node network $G_n(c)$. Let $d = c\lfloor \log n \rfloor$ and $\lfloor s = n/d \rfloor$. For clarity of presentation we assume that d divides n and $s = 2^{h+1} - 1$, for some $h \geq 0$. Partition the set of all nodes into groups $S_1, \ldots, S_s$, each of size d. In every group S_i, $1 \leq i \leq s$, enumerate consecutive nodes from 0 to $d-1$. For any $i = 1, \ldots, s$ and $j = 0, \ldots, d-1$, assign label (i,j) to the j-th node in the i-th group. We assume (1,0) to be the source of broadcasting. We will later indicate easy modifications of our algorithms allowing to drop this assumption. Arithmetic operations on the second integers forming labels are performed modulo d. Arrange all groups S_i into complete binary tree T with $h+1$ levels enumerated $0, 1, \ldots, h$, starting from the level containing the root. The group S_1 is the root of T. For every $1 \leq i \leq \lfloor s/2 \rfloor, S_{2i}$ is the left child of S_i and S_{2i+1} is the right child of S_i in the tree T. For every $1 < i \leq s$, the group $S_{\lfloor i/2 \rfloor}$ is the parent of the group S_i. If S_i is a parent or a child of S_j we say that these groups are adjacent in T.

The set of edges of $G_n(c)$ is defined as follows. If groups S_i and S_j are adjacent in T, there is an edge in $G_n(c)$ between every node from S_i and every node from S_j. There are no other edges in $G_n(c)$. Notice that $G_n(c)$ has the following properties:

- for every $1 \leq i \leq s$, $|S_i| \in O(\log n)$;
- $G_n(c)$ has maximum degree $O(\log n)$;
- the height h of the tree T is less than $\log n$.

4. Broadcasting Algorithms

In this section we construct non-adaptive and adaptive ε-safe broadcasting algorithms working for graph $G_n(c)$ defined in section 3. We first describe three procedures used in these algorithms.

1. Procedure Multicall (S_i, S_j, k)

The aim of this procedure is communication between nodes of group S_i and nodes of group S_j. S_j is a child of S_i in the tree T. The procedure uses one time unit.

procedure Multicall (S_i, S_j, k);
begin
    for all $0 \leq r < d$ **in parallel do**
        (i,r) calls $(j,\ r+k)$
end;

2. Procedure One_To_All ((i,r), S_j)

The aim of the procedure is communication between a node of group S_i and all nodes of group S_j. Groups S_i and S_j are adjacent in the tree T. The procedure uses d time units.

procedure One_To_All ((i,r), S_j);
begin
  for $k := 0$ **to** $d-1$ **do**
    (i,r) calls (j,k)
end;

3. Procedure Adaptive Multicalls (S_i, S_j)

This procedure is adaptive. For groups S_i and S_j adjacent in the tree T, nodes from S_i call consecutive nodes from S_j. A fault-free node u from S_i is called active if u does not have yet the source information; as soon as it gets it, it stops being active. Calls are made only by active nodes. The procedure uses d time units.

procedure Adaptive Multicalls (S_i, S_j);
begin
  for $k := 0$ **to** $d-1$ **do**
    for all $0 \leq r < d$ **in parallel do**
      if (i,r) is active **then**
        (i,r) calls (j, $k+r$)
end;

We are now ready to describe the main broadcasting algorithms.

The Non-adaptive Broadcasting Algorithm (NBA)

The algorithm consists of 3 identical stages. The aim of the first stage is to disseminate source information originally stored in node (1,0) (the source) belonging to group S_1 (the root of T) down the tree T in such a way that at least one fault-free node in each group gets the information with high probability. Nodes which get information in the first stage are called leaders of their respective groups. Every group may have many leaders. In stages 2 and 3 leaders transmit information to other fault-free nodes in their group. In order to do that a leader of group S_i transmits source information to nodes of an adjacent group S_j in stage 2 and subsequently these nodes transmit source information to other nodes of group S_i in stage 3.

```
Algorithm NBA;
begin
for stage := 1 to 3 do
    for step := 0 to d − 1 do
    begin
        for each S_i on an even level in T, less than h do
        begin
            MultiCall (S_i, S_2i, step);
            MultiCall (S_i, S_2i+1, step);
        end;
        for each S_i on an odd level in T, less than h do
        begin
            MultiCall (S_i, S_2i, step);
            MultiCall (S_i, S_2i+1, step);
        end
    end
end;
```

Since the algorithm NBA contains 3 stages, each consisting of d steps taking 4 time units each, it works in time $O(\log n)$. Clearly every node is involved in at most one call in a unit of time.

The Adaptive Broadcasting Algorithm (ABA)

The idea of the adaptive algorithm is fairly similar to the above. However, in the present case we need to avoid making too many calls on average, since NBA used $\Theta(n \log n)$ calls and our present goal is the expected number of $O(n)$ calls. As before, the algorithm consists of 3 stages. This time they are not identical but their role in the broadcasting process is similar as in the non-adaptive case.

A node u in group S_i is called a left sender (right sender) if $1 \le i \le \lfloor s/2 \rfloor$, and u has source information but it has not yet transmitted it to any node from S_{2i} (S_{2i+1}). Notice that at the beginning only node (1,0) (the source) is a left and right sender.

```
Stage 1
begin
    for step := 0 to d − 1 do
    begin
        for each S_i on an even level in T do
        begin
            if (i,r) is a left sender in S_i then
                (i,r) calls (2i, r + step);
            if (i,r) is a right sender in S_i then
                (i,r) calls (2i + 1, r + step)
        end;
```

```
        for each S_i on an odd level in T do
        begin
            if (i,r) is a left sender in S_i then
                (i,r) calls (2i, r + step);
            if (i,r) is a right sender in S_i then
                (i,r) calls (2i + 1, r + step)
        end
    end
end;
```

Stage 1 of ABA takes $4d$ time units. Every group S_i can have at most one leader upon completion of this stage. When a node u becomes the leader of S_i (that is, it has obtained the source information from the leader of $S_{\lfloor i/2 \rfloor}$) and $2i \le s (2i+1 \le s)$ then u becomes a left sender (right sender). If S_i is the left child (right child) of $S_{\lfloor i/2 \rfloor}$ then the leader of $S_{\lfloor i/2 \rfloor}$ stops being a left sender (right sender) at this point. A left sender (right sender) from S_i calls different nodes from S_{2i} (S_{2i+1}).

In the second stage the leader of every group S_i, $1 < i \le s$, calls all nodes from $S_{\lfloor i/2 \rfloor}$. The leader of S_1 calls all nodes from S_2.

```
Stage 2
begin
    for each leader (i,r) such that
        S_i is on an even level in T and it is
        the left child of its parent do
            One_To_All ((i,r), S_⌊i/2⌋);
    for each leader (i,r) such that
        S_i is on an even level in T and it is
        the right child of its parent do
            One_To_All ((i,r), S_⌊i/2⌋);
    for each leader (i,r) such that
        S_i is on an odd level in T and it is
        the left child of its parent do
            One_To_All ((i,r), S_⌊i/2⌋);
    for each leader (i,r) such that
        S_i is on an odd level in T and it is
        the right child of its parent do
            One_To_All ((i,r), S_⌊i/2⌋);
    One_to_All ((1,0), S_2)
end;
```

Stage 2 uses $5d$ time units.

In stage 3 those nodes from group S_i, $1 < i \le s$, which do not have yet source information, call nodes from $S_{\lfloor i/2 \rfloor}$ in order to obtain this information transmitted there in stage 2 by the leader of S_i. Nodes from S_1 call nodes from S_2.

Stage 3
begin
Adaptive Multicalls (S_1, S_2);
for each S_i such that
$1 < i \leq s$, S_i is on an even level in T and it is
the left child of its parent **do**
Adaptive Multicalls (S_i, $S_{\lfloor i/2 \rfloor}$);
for each S_i such that
$1 < i \leq s$, S_i is on an even level in T and it is
the right child of its parent **do**
Adaptive Multicalls (S_i, $S_{\lfloor i/2 \rfloor}$);
for each S_i such that
$1 < i \leq s$, S_i is on an odd level in T and it is
the left child of its parent **do**
Adaptive Multicalls (S_i, $S_{\lfloor i/2 \rfloor}$);
for each S_i such that
$1 < i \leq s$, S_i is on an odd level in T and it is
the right child of its parent **do**
Adaptive Multicalls (S_i, $S_{\lfloor i/2 \rfloor}$)
end;

Stage 3 uses less than $5d$ time units. Hence the entire algorithm ABA works in (worst case) logarithmic time. Clearly every node is involved in at most one call in a unit of time.

Call Saving Adaptive Broadcasting Algorithm (ABA)*

Our last algorithm is an adaptive broadcasting algorithm working in worst case logarithmic time and using a linear number of calls in worst case. However, it will be proved ε-safe only under the additional assumption that all nodes are fault-free (i.e. $q = 0$). The algorithm ABA* works in two stages. Stage 1 is exactly the same as in ABA. Upon its completion every group S_i has at most one leader: a node knowing the source information. Let (i, r_i) be the leader in S_i. In Stage 2 every leader (i, r_i) tries to transmit source information to all nodes in its group. This is done using intermediary nodes from group S_j, where $j = \lfloor i/2 \rfloor$ for $i \geq 2$ and $j = 2$ for $i = 1$. The leader (i, r_i) tries to inform consecutive nodes $(i, r_i + k)$, for $k = 1, \ldots, d-1$. The total number of transmission attempts made by a leader cannot exceed cd. A leader attempts to inform node $(i, s+1)$ only after having informed node (i,s). Transmission attempts are executed using consecutive intermediaries from group S_j. Every attempt consists of three consecutive calls:

- ∘ between the leader and the current intermediary,
- ∘ between the intermediary and the target node,
- ∘ between the leader and the intermediary.

The aim of the last call is to inform the leader if the second call has been successful, i.e. if the link between the intermediary and the target node is fault-free. If both links used in an attempt are fault-free, the target node has been informed and the leader starts attempts to inform the next node from its group; otherwise it tries to inform the same node using consecutive intermediaries. Broadcasting source information to nodes in group S_i is executed using procedure Group Broadcast (S_i, S_j).

procedure Group Broadcast (S_i, S_j);
begin
    $t := 0$; $current := r_i + 1$
    while $(t < cd)$ **and** $(current \neq r_i)$ **do**
    begin
        (i, r_i) calls $(j, r_i + t)$
        $(j, r_i + t)$ calls $(i, current)$
        $(j, r_i + t)$ calls (i, r_i)
        if all calls were successful
        then $current := current + 1$
        $t := t + 1$
    end
end

Stage 2 of the algorithm can be now formally written as follows

Stage 2
begin
    Group Broadcast (S_1, S_2)
    for all S_{2i} on odd levels in T **in parallel do**
        Group Broadcast (S_{2i}, S_i)
    for all S_{2i+1} on odd levels in T **in parallel do**
        Group Broadcast (S_{2i+1}, S_i)
    for all S_{2i} on even positive levels in T **in parallel do**
        Group Broadcast (S_{2i}, S_i)
    for all S_{2i+1} on even positive levels in T **in parallel do**
        Group Broadcast (S_{2i+1}, S_i)
end

Stage 2 works in worst case time $O(d)$ and uses $O(n)$ calls in worst case. Since complexity of Stage 1 is the same, the entire algorithm ABA* works in worst case logarithmic time and uses a linear number of calls in the worst case. Clearly every node is involved in at most one call in a unit of time.

Note that the algorithms can be easily adapted to work with any binary tree formed with groups of nodes, it was merely convenient and efficient to assume a complete tree. Hence, one can take any group S_i to be the root and any node (i, j) to be the source. Such a modification at most doubles the height of the tree and the running time of the algorithms.

5. Reliability and Complexity of Broadcasting Algorithms

In this section we estimate the probability that the broadcasting algorithms described in section 4 are successful. We also discuss their complexity. The first result is:

THEOREM 1. *Let $p < 1$ be the link failure probability and $q < 1$ be the node failure probability. For every $\varepsilon > 0$ there exist integers c, $n_0 > 0$ such that for every $n \geq n_0$, each of the algorithms NBA and ABA working for the network $G_n(c)$ is ε-safe.*

PROOF. We give the proof only for algorithm NBA. The adaptive case is similar. Let

$$c = max\left(\left\lceil \frac{-4(1+\varepsilon]}{\log(1-(1-p)^2(1-q))} \right\rceil, \left\lceil \frac{8(1+2\varepsilon)}{(1-p)(1-q)\log \mathrm{e}} \right\rceil\right)$$

and

$$n_0 = max\left(min\left\{n : \frac{n}{c\lfloor \log n \rfloor} \geq 2\right\}, min\{n : n^{\varepsilon} \geq 2\}\right).$$

Let E denote the event that NBA is successful. Consider the following events:

- E_1 upon completion of the first stage at least one node in every group S_i obtains source information (every group has a leader).
- E_2 between every pair of nodes in the same group there exists a path of length 2 whose both links and the intermediate node are fault-free.

First notice that $E_1 \cap E_2 \subset E$. Indeed, in view of E_1, every group has a leader. In the second stage a leader u of group S_i transmits source information to all its fault-free neighbors, provided that the joining links are fault-free. In the third stage these neighbors transmit information to every fault-free node v in S_i, provided that respective joining links are fault-free. By E_2 there is a path of length 2 between u and v without faulty components and consequently v obtains source information upon completion of the third phase.

We will show that $\Pr(\overline{E_1}) \leq n^{-2\varepsilon}$ and $\Pr(\overline{E_2}) \leq n^{-2\varepsilon}$, thus $\Pr(\overline{E}) \leq n^{-\varepsilon}$, for sufficiently large n. The event $\overline{E_1}$ implies that during the first stage of NBA source information has not been passed along some branch of the tree T (that is, some group of this branch does not have a leader). Fix such a branch $B = (S_{i_0}, S_{i_1}, \ldots, S_{i_h})$, where $S_{i_0} = S_1$, and estimate the probability of the event P that information has not been passed along this branch. Every fault-free node from group S_{i_j} calls different nodes from group $S_{i_{j+1}}$ in d consecutive steps. These attempts are independent and they have success probability $r_1 = (1-p)(1-q)$ (both the destination node and the joining link must be fault-free). Upon a successful call from a leader of S_{i_j}, some node of $S_{i_{j+1}}$ becomes a leader and information can be passed further along branch B. Hence $\Pr(P)$ does not exceed the probability of at most h successes in d Bernoulli trials with success probability r_1.

Since $h < \lfloor \log n \rfloor$, $\Pr(P)$ does not exceed the probability of at most $\lfloor \log n \rfloor$ successes in a series of d trials with success probability r_1. Consider such a series of trials and let X be the number of successes. By Chernoff bound (cf. [1, 9]) we get $\Pr(X \le (1-\lambda) r_1 d) \le e^{-\lambda^2 r_1 d/2}$, for any $0 < \lambda < 1$. Since $c > 1/r_1$, we have

$$0 < \lambda = \frac{r_1 c - 1}{r_1 c} < 1$$

and

$$(1-\lambda) r_1 d = \frac{1}{r_1 c} \cdot r_1 c \lfloor \log n \rfloor = \lfloor \log n \rfloor,$$

hence

$$\Pr(P) \le \Pr(X \le \lfloor \log n \rfloor) \le \mathrm{e}^{-\lambda^2 r_1 c \lfloor \log n \rfloor /2}.$$

Since there are less than n branches in the tree T, we get (for $n \ge 2$)

$$\begin{aligned} \Pr(\overline{E_1}) &\le n \Pr(P) \le n\ \mathrm{e}^{-\lambda^2 r_1 c \log n/4} \\ &= n \cdot n^{-\lambda^2 r_1 c \log \mathrm{e}/4} = n^{1-(r_1 c - 2 + \frac{1}{r_1 c}) \log \mathrm{e}/4} \\ &\le n^{1 - r_1 c \log \mathrm{e}/8}, \\ \text{because } c &\ge \left\lceil \frac{8(1+2\varepsilon)}{r_1 \log \mathrm{e}} \right\rceil \ge \lceil 4/r_1 \rceil \ \text{ implies} \end{aligned}$$

$$r_1 c - 2 + \frac{1}{r_1 c} \ge \frac{r_1 c}{2}.$$

Since $r_1\, c\, \log \mathrm{e}/8 \ge 1 + 2\varepsilon$, we finally get

$$\Pr(\overline{E_1}) \le n^{-2\varepsilon}.$$

Next, we estimate $\Pr(\overline{E_2})$. Every group contains at least d nodes. In view of $n/d \ge 2$ there are at least two groups. Between every pair of nodes in a group there exist at least d disjoint paths of length 2. The probability that in a single path u-w-v the intermediate node or one of the links are faulty is

$r_2 = 1-(1-p)^2(1-q)$. Consider two fault-free nodes u, v in a group and fix d disjoint paths of length 2 between them. Since the events that these paths contain a faulty component are independent, the probability that each of them does, is r_2^d. Since there are less than n^2 pairs of nodes in the network, we get

$$\Pr(\overline{E_2}) \leq n^2 r_2^d \leq n^2 r_2^{c \log n/2}, \text{ for } n \geq 2$$

and since $c \log r_2 \leq -4(1+\varepsilon)$, we obtain

$$\Pr(\overline{E_2}) \leq n^2 \cdot n^{c \log r_2/2} \leq n^2 \cdot n^{-(2+2\varepsilon)} = n^{-2\varepsilon}.$$

Since $n^\varepsilon \geq 2$ for $n \geq n_0$, this implies

$$\Pr(\overline{E}) \leq \Pr(\overline{E_1}) + \Pr(\overline{E_2}) \leq 2n^{-2\varepsilon} \leq n^{-\varepsilon},$$

which concludes the proof. □

In section 4 we noticed that both algorithms NBA and ABA work in (worst case) logarithmic time. This order clearly cannot be decreased even without faults. It follows that the number of calls used by NBA is $O(n \log n)$ and the worst case number of calls used by ABA is also $O(n \log n)$. It is easy to see that in both cases order $n \log n$ is exact. Moreover it can be proved (cf. [5]) that every non-adaptive broadcasting algorithm using $o(n \log n)$ calls is successful with probability converging to 0, so NBA is asymptotically optimal among ε-safe algorithms, with respect to the number of calls. On the other hand, in case of ABA, the average number of calls is linear. Indeed, during the first two stages only leaders of groups make calls, and since there are $O(n/\log n)$ leaders, the number of calls in these phases is $O(n)$. In stage 3 every node u which does not yet have source information calls nodes from a group adjacent to its own group S_i until it finds a node previously informed by the leader of S_i. If this leader appeared in stage 1, the expected number of calls made by u in stage 3 is $\lceil 1/((1-p)^2(1-q)) \rceil$, otherwise u makes d calls. Hence the expected number of calls made by u in stage 3 is at most

$$\lceil 1/((1-p)^2(1-q)) \rceil + c \lfloor \log n \rfloor \cdot n^{-\varepsilon} \in O(1)$$

and consequently the total expected number of calls is linear.

Theorem 1 and the above remarks imply the following Corollary.

Corollary. *Let $p < 1$ be the link failure probability and $q < 1$ the node failure probability. There exists a family of n-node networks with maximum degree $O(\log n)$ which support a non-adaptive ε-safe broadcasting algorithm working in logarithmic time, as well as an adaptive ε-safe broadcasting algorithm working in (worst case) logarithmic time and using an average linear number of calls.*

Our next theorem concerns the reliability of algorithm ABA* in case when nodes are fault-free.

THEOREM 2. *Let $p < 1$ be the link failure probability and assume that nodes are fault-free ($q = 0$). For every $\varepsilon > 0$ there exist integers c, $n_0 > 0$ such that for every $n \geq n_0$, the algorithm ABA* working for the network $G_n(c)$ is ε-safe.*

PROOF. The proof of the theorem is an immediate consequence of the following lemma. □

LEMMA. *Assume that after Stage 1 of algorithm ABA* there is a leader in every group S_i. Then, after Stage 2 of ABA* all nodes of the network know source information, with probability at least $1 - n^{-2\varepsilon}$.*

PROOF. Let $c = \lceil \frac{8(1+2\varepsilon)}{(1-p)^2 \log \mathrm{e}} \rceil$ and $n_0 = min \left\{ n : \frac{n}{c \lfloor \log n \rfloor} \geq 2 \right\}$. Consider $\lfloor \log n \rfloor$ consecutive nodes in S_i. Let E be the event that not all of these nodes are informed after a total of $d = c \lfloor \log n \rfloor$ attempts. Since in d consecutive attempts all intermediaries are distinct, $\Pr(E)$ does not exceed the probability of at most $\lfloor \log n \rfloor$ successes in a series of d Bernoulli trials with success probability $r = (1-p)^2$. All nodes in S_i can be divided into c sets of size $\lfloor \log n \rfloor$. Hence an argument similar to that in the proof of theorem 1 shows that the probability of informing all nodes in all groups is at least $1 - n \Pr(E)$, which is at least $1 - n^{-2\varepsilon}$ if $n \geq n_0$ for c and n_0 as above. □

COROLLARY 1. *Let $p < 1$ be link failure probability and assume that all nodes are fault-free. There exists a family of n-node networks with maximum degree $O(\log n)$ which support an adaptive ε-safe broadcasting algorithm working in worst case logarithmic time and using a linear number of calls in the worst case.*

6. Conclusions

We presented three broadcasting algorithms working correctly with high probability in the presence of random faults in n-nodes networks. Two of them tolerate both link and node failures: the non-adaptive algorithm NBA works in logarithmic time and uses $O(n \log n)$ calls, while the adaptive algorithm ABA works in worst case logarithmic time and uses O(n) calls on average. In case when only links are subject to failures and all nodes are fault-free we presented an adaptive algorithm ABA* working in worst case logarithmic time and using $O(n)$ calls in the worst case. It seems difficult to obtain a similar performance in case of faulty links and nodes. In this general case, the difficulty is to decide when an informed node should give up attempts to inform a target node: too many unsuccessful attempts may be a waste because the target node may be faulty and should be given up, too few attempts risk to give up a fault-free node that must be informed. In case of fault-free nodes there is no need to make this decision: attempts are made until the target node is informed or until all available trials are exhausted. As we proved, logarithmically many trials are then enough to inform all nodes in the group, with high probability, thus yielding a worst case linear number of calls in the entire algorithm. In the general case, however, the following problem remains open.

PROBLEM. Assume that $p < 1$ is the link failure probability and $q < 1$ is the node failure probability. Does there exist an ε-safe adaptive algorithm working in worst case logarithmic time and using a linear number of calls in the worst case?

References

[1] ANGLUIN, D., VALIANT L. G. Fast probabilistic algorithms for Hamiltonian circuits and matchings. *J. Comput. System Sci. 18* (1979), 155–193.

[2] BERMAN, K. A., HAWRYLYCZ, M. Telephone problems with failures. *SIAM J. Alg. Disc. Meth. 7* (1986), 13–17.

[3] BIENSTOCK, D. Broadcasting with random faults. *Disc. Appl. Math. 20* (1988), 1–7.

[4] CHLEBUS, B. S., DIKS, K., PELC A. Optimal broadcasting in faulty hypercubes. *Proc. 21st Int. Symp. on Fault-Tolerant Computing*, Montreal, Canada (1991), 266–273.

[5] DIKS, K., PELC, A. Reliable gossip schemes with random link failures. *Proc. 28th Ann. Allerton Conf. on Comm. Control and Comp.* (1990), 978–987.

[6] DIKS, K., PELC A. Almost safe gossiping in bounded degree networks. *SIAM J. Disc. Math. 5* (1992), 338–344.

[7] GARGANO, L. Tighter time bounds on fault tolerant broadcasting and gossiping. *Networks, 22* (1992), 469–486.

[8] HADDAD, R. W., ROY, S., SCHAFFER, A. A. On gossiping with faulty telephone lines, *SIAM J. Alg. Disc. Meth. 8* (1987), 439–445.

[9] HAGERUP, T., RUB C. A guided tour of Chernoff bounds. *Inf. Proc. Letters 33* (1989/90), 305–308.

[10] HEDETNIEMI, S.M., HEDETNIEMI, S. T., LIESTMAN, A. L. A survey of gossiping and broadcasting in communication networks. *Networks 18* (1988), 319–349.

[11] SCHEINERMAN, E. R., WIERMAN, J. C. Optimal and near-optimal broadcast in random graphs. *Disc. Appl. Math. 25* (1989), 289–297.