# Rapid Almost-Complete Broadcasting in Faulty Networks[*]

Rastislav Královič[1] and Richard Královič[1,2]

[1] Department of Computer Science, Comenius University,
Mlynská dolina, 84248 Bratislava, Slovakia.
[2] Department of Computer Science, ETH Zurich, Switzerland.

**Abstract.** This paper studies the problem of broadcasting in synchronous point-to-point networks, where one initiator owns a piece of information that has to be transmitted to all other vertices as fast as possible. The model of fractional dynamic faults with threshold is considered: in every step either a fixed number $T$, or a fraction $\alpha$, of sent messages can be lost depending on which quantity is larger.

As the main result we show that in complete graphs and hypercubes it is possible to inform all but a constant number of vertices, exhibiting only a logarithmic slowdown, i.e. in time $O(D \log n)$ where $D$ is the diameter of the network and $n$ is the number of vertices.

Moreover, for complete graphs under some additional conditions (sense of direction, or $\alpha < 0.55$) the remaining constant number of vertices can be informed in the same time, i.e. $O(\log n)$.

## 1 Introduction

Fault tolerance has been a crucial issue in the distributed computing since its beginnings [3,5,6,10,16,25]. Because a typical distributed system is designed to contain a large number of individual components, attention must be paid to the fact that, even if the failure probability of a single component is negligible, the probability that some components fail may be high. There are numerous ways how to cope with failures, using either probabilistic or deterministic approaches. In the probabilistic setting, it is supposed that a failure probability of each component follows some probability distribution [4,8,11,26,27]. Failures of individual components are usually assumed to be independent random events. The goal is to design algorithms and protocols that perform well with high probability if the failures follow the conjectured distribution.

The deterministic approach, which is pursued also in this paper, copes with failures in a different way. Instead of considering a failure probability distribution for each individual component, algorithms and protocols are designed to perform well in the worst case, under some a-priori constraints on the failure behavior. [1,2,7,12–14,19,20,22,24,28]. These constraints may take the form of considering only computations with a limited overall number of faults [1,19], limited number

---

of faults during any single computation step [7, 13, 14, 24, 28], or during any window of first $t$ steps [20], requiring that after some finite time there is a long enough fault-free computation [10, 15] etc. While the probabilistic model is analyzed with respect to the expected behavior, the deterministic models have been mostly analyzed for the worst case scenario.

We shall focus our attention on synchronous point-to-point distributed systems, i.e. systems in which the communication is performed by sending messages along links connecting pairs of vertices. Moreover, the vertices are synchronized by a common clock, and the delivery of every message takes exactly one time unit. This model has been widely considered [7, 8, 12–14, 20, 22, 24, 26–28] not only for its theoretical appeal, but for its practical relevance as well (e.g. many wireless networking standards, like IEEE 802.11, or GSM, operate in discrete time steps). We shall consider only one type of failures: *message loss*.

The oldest deterministic model of faults considered in this setting is the static model [1, 3], in which it is assumed that at most a fixed constant number $k$ of messages may be lost in every step, and moreover, the failures are always located on the same links. Later, other models have been considered, too, like the dynamic model [7, 13, 14, 24, 28] in which the $k$ failures may be located on arbitrary links in every step, linearly bounded faults [20], fractional faults [22], etc.

We continue in the analysis of the *fractional model with threshold* from [12]. Here, the number of messages lost in one time step is bounded by the maximum of a fixed threshold $T$ and a fixed fraction $\alpha$ of sent messages. This restriction implies that if, in a given step, fewer than $T$ messages are sent they may all be lost. On the other hand, if there are many messages sent, at least a fixed fraction $1 - \alpha$ of them is delivered. The threshold $T$ is always assumed to be one less than the edge connectivity, since this is the largest value under which the network stays connected. This model has been developed in order to avoid some unrealistic special cases of static and dynamic models (the number of faults is independent on the actual network traffic), as well as those of fractional model (if just one message is being sent, its delivery is always guaranteed).

The broadcasting problem is a crucial communication task in the study of distributed systems (e.g. [21]). One vertex, called initiator, has a piece of information that has to be distributed among all remaining vertices. The broadcasting has not only been used as a test-bed application for the study of the complexity of communication in various communication models, but has served as a building stone of many applications (e.g. [29]) as well.

We analyze the broadcasting in complete graphs and hypercubes. The broadcasting time in these graphs has been studied in the static [19], dynamic [13, 14, 24], and simple threshold [12][3] models, and the results are summarized in Table 1.

---

[3] If the number of messages sent in a given time step is less than the edge connectivity $c(G)$ in the simple threshold model, all of them may be lost. Otherwise at least one of them is delivered.

| Model | $K_n$, chordal sense of direction | $K_n$ unoriented | $Q_d$, $n = 2^d$ |
|---|---|---|---|
| static | $\Theta(1)$ | $\Theta(1)$ | $d + 1$ [19] |
| dynamic | $\Theta(1)$ | $\Theta(1)$ [24] | $d + 2$ [13] |
| fractional | $\Theta(\log n)$ | $\Theta(\log n)$ [22] | $O(d^3)$ [22] |
| simple threshold | $\Omega(n)$, $O(n^2)$ [12] | $\Omega(n^2)$, $O(n^3)$ [12] | $O(n^4 d^2)$ [12] |

**Table 1.** Known time complexities of the complete broadcasting in various models.

| Scenario | Almost complete broadcasting | Complete broadcasting |
|---|---|---|
| $K_n$, unoriented | $O(\log n)$ | $\Omega(\log n)$ [22], $O(n^3)$ [12] |
| $K_n$, chordal sense of direction | $O(\log n)$ | $\Omega(\log n)$ [22], $O(\log n)$ |
| $K_n$, $\alpha < 0.55$ | $O(\log n)$ | $\Omega(\log n)$ [22], $O(\log n)$ |
| $Q_d$ | $O(d^2)$ | $\Omega(d)$, $O(n^4 d^2)$ [12] |

**Table 2.** Results for the complete and almost complete broadcasting in the fractional model with threshold.

We address a natural relaxation of the broadcasting problem in which we allow a small constant number of vertices to stay uninformed in the end (a problem called *almost complete broadcasting*), and analyze the worst case time needed to solve the problem. Our main motivation to study almost complete broadcasts is the fact that in large faulty networks it is often vital to finish a communication task fast, even subject to some small error. In the probabilistic setting, this is modelled by allowing a failure probability that tends to zero with increasing network size: in the worst case the task is not successful but this worst case scenario has a small probability. Since in our deterministic setting we study the worst case, another model of allowed error must be chosen. If we look at the broadcast as an optimization problem where the task is to inform as many vertices as possible, it is natural to introduce a constant additive error by allowing a constant number of vertices to stay uninformed[4].

For complete graphs and hypercubes, we show that the problem can be solved in time $O(D \log n)$, where $D$ is the diameter of the graph and $n$ is the number of its nodes.

Moreover, we show that if the complete graph is equipped with the chordal sense of direction, complete broadcasting can be performed in time $O(\log n)$. This is asymptotically optimal since the broadcasting time in the fractional model is a lower bound for the fractional model with threshold. Similarly we show that the broadcasting can be completed in time $O(\log n)$ for values $\alpha < 0.55$. The overview of the results can be found in Table 1.

---

[4] so that the uninformed vertices comprise at most an $O(1/n)$ fraction of all vertices

## 2 Definitions

We consider a synchronous, point-to-point distributed system with a coordinated start-up. The system consists of a number of nodes and a number of communication links connecting some pairs of nodes. The system is modelled by an undirected graph, in which vertices correspond to nodes and edges correspond to communication links. In this respect, we shall use the terms "node" and "vertex" interchangeably. Sometimes we need to argue about outgoing and incoming links; in this cases we consider a directed graph obtained from the undirected one by replacing each edge by two opposite arcs.

At the beginning of the computation all nodes are active and start performing the given protocol. The computation consists of a number of steps: at the beginning of each step, messages sent during the previous step are delivered to their destinations, then each vertex performs some local computation, possibly sending some messages[5], and the next step begins.

The failure model we consider is the *fractional dynamic faults with threshold* from [12], which can be described as a game between the algorithm and an adversary: in a time step $t$ the algorithm sends $m_t$ messages and the adversary may destroy up to

$$F(m_t) = \max\{c(G) - 1, \lfloor \alpha \, m_t \rfloor\}$$

of them, where $c(G)$ is the edge connectivity of the graph and $\alpha$ is a known, fixed constant $0 < \alpha < 1$. There is no built-in mechanism of acknowledgements, so the sender node is not informed whether a particular message was delivered or destroyed.

We consider the problem of broadcasting, where an initiator has a piece of information to be transmitted to all remaining vertices. We call a broadcast *complete* if all vertices have the information after the termination of the algorithm. A broadcast is called *almost-complete* if there is a fixed constant $c$ (independent on the network size) such that after the termination there are at most $c$ uninformed vertices. Hence, to prove the existence of an almost-complete broadcasting algorithm for a family of graphs $\mathcal{G}$, one has to prove that there exists a constant $c$ such that for each $G \in \mathcal{G}$ the broadcasting algorithm informs all but $c$ vertices of $G$.

In all presented algorithms only the informed vertices send messages. Arcs (i.e. directed edges) leading from an informed vertex can be classified as being either active, passive or hyperactive during the computation:

**Definition 1.** *Let $e$ be an arc leading from an informed vertex. We call $e$ active if it leads to an uninformed vertex. We call an arc $e$ passive, if some message has been delivered via the opposite arc of $e$. Finally, we call an arc $e$ hyperactive if it leads to an informed vertex, and is not passive.*

If the arc $e$ is passive, the source vertex of $e$ is aware of the fact that the destination vertex of $e$ has already been informed. The main idea of our algorithms is to perform appropriate number of *simple rounds* defined as follows:

---

[5] i.e. a vertex may send different message to each of its neighbors in one step

**Definition 2.** *A* simple round *consists of two time steps. In the first step, every informed vertex sends a message along each of its incident arcs, excluding the passive ones.*[6] *In the second step, all vertices that have received a message send an acknowledgement (and mark the arc as passive). Vertices that receive acknowledgement mark the corresponding arc as passive.*

For the remainder of this paper, let $0 < \alpha < 1$ be a known fixed constant, and let us denote

$$X := \frac{1}{\alpha(1 - \alpha)}$$

The rest of the paper is organized as follows. In the next two sections we present algorithms for the almost-complete broadcasting on complete graphs and hypercubes, respectively, that run in time $O(D \log n)$. Then we show how to obtain broadcast in complete graphs equipped with chordal sense of direction, and for unoriented complete graphs for $\alpha < 0.55$, having the same time complexity.

Due to space restrictions some technical parts have been omitted from this paper, and can be found in the technical report [23].

## 3  Complete Graphs

In a complete graph $K_n$, all $n$ vertices have degree $n - 1$, and $n - 1$ is also the edge connectivity. Hence, in each step $t$ the adversary can destroy up to $\max\{n-2, \lfloor \alpha m_t \rfloor\}$ messages, where $m_t$ is the number of messages sent in the step $t$. In this section we present an algorithm that informs all but a constant number of vertices in logarithmic time. The idea of the algorithm is very straightforward – just repeat simple rounds sufficiently many times. However, the arguments given in the analysis of a simple round below hold only if there are enough informed vertices participating in the round. To satisfy this requirement two steps of a simple greedy algorithm are performed, during which each informed vertex just sends the message to all vertices. After two steps of this algorithm, the number of informed vertices is as shown in Lemma 1.

**Lemma 1.** *After two steps of the greedy algorithm, at least*

$$1 + \min\left\{\frac{n}{2}, (n-1)(1-\alpha)\right\}$$

*vertices are informed.*

After these two steps, the algorithm performs a logarithmic number of simple rounds. To show that logarithmic number of simple rounds is sufficient to inform all but one vertex we first provide a lower bound on the number of acknowledgements delivered in each round, and then we show that each delivered acknowledgement decreases a certain measure function.

---

[6] In this step, a message is sent via all active and hyperactive arcs. The former can inform new vertices, the latter exhibit only useless activity. However, the algorithm can not distinguish between active and hyperactive arcs.

**Theorem 1.** *Let $\varepsilon > 1$ be an arbitrary constant. For large enough $n$ it is possible to inform all but at most $X\varepsilon$ vertices in logarithmic time. Moreover, the number of remaining hyperactive arcs is at most $X(n-2)$.*

*Proof.* At the beginning, two steps of the greedy algorithm are executed. Then, a logarithmic number of simple rounds is performed. Now consider the situation at the beginning of the $i$-th round. Let $k_i$ be the number of uninformed vertices, and $h_i$ the number of hyperactive arcs. We claim that if $k_i > X\varepsilon$ or $h_i > X(n-2)$ then at least $[k_i(n-k_i) + h_i](1-\alpha)^2$ acknowledgements are delivered in this round. Since there are $k_i(n-k_i)+h_i$ messages sent in this round, in order to prove the claim it is sufficient to show that $\alpha(1-\alpha)[k_i(n-k_i)+h_i] \geq n-2$. Obviously, if $h_i > X(n-2)$ the inequality holds, so consider the case $k_i > X\varepsilon$. We prove that in this case $k_i(n-k_i) \geq X(n-2)$, i.e. $k_i^2 - nk_i + X(n-2) \leq 0$. Let $f(n) := 1/2\left(n - \sqrt{n^2 - 4X(n-2)}\right)$; the roots[7] of the equation $k_i^2 - nk_i + X(n-2) = 0$ are $f(n)$ and $n - f(n)$, so we want to show that $f(n) \leq k_i \leq n - f(n)$. Since $\lim_{n \mapsto \infty} f(n) = X$, we get that $k_i > X\varepsilon > f(n)$ holds for large enough $n$. Hence, the only remaining step is to show the inequality $k_i \leq n - f(n)$. From Lemma 1 it follows that $n - k_i > \min\{n/2, (n-1)(1-\alpha)\}$. Since $f(n) < n/2$, if $n - k_i > n/2$ it holds $k_i < n - f(n)$. So let us suppose that $n - k_i > (n-1)(1-\alpha)$, i.e. $k_i < 1 + \alpha(n-1)$. Let $n \geq \frac{\varepsilon + \alpha(1-\alpha)^2}{\alpha(1-\alpha)^2}$. Then it holds for large enough $n$ that

$$k_i < 1 + \alpha n - \alpha \leq n - \frac{\varepsilon}{\alpha(1-\alpha)} = n - \varepsilon X \leq n - f(n).$$

We have proved that if $k_i > X\varepsilon$ or $h_i > X(n-2)$ then at least

$$[k_i(n-k_i) + h_i](1-\alpha)^2$$

acknowledgements are delivered in round $i$.

To conclude the proof we show that after logarithmic number of iterations we get $k_i \leq X\varepsilon$ and $h_i \leq X(n-2)$. Let $M_i := 2(n-1)k_i + h_i$; then every delivered acknowledgement decreases $M_i$ by at least one: indeed, if the acknowledgement was delivered over a hyperactive arc, $h_i$ decreases by 1. If, on the other hand, the acknowledgement was delivered over an active arc, the number of uninformed vertices is decreased by at least one, and the number of hyperactive arcs is increased by at most $2n-3$ (new hyperactive arcs are between the newly informed vertex and any other vertex, with the exception of the arc that delivered the acknowledgement which is passive).

From Lemma 1 it follows that either $n - k_i > n/2$ or $n - k_i > (n-1)(1-\alpha)$. In the first case it follows that at least $(1-\alpha)^2[k_i(n-k_i)+h_i] > (1-\alpha)^2[k_in/2+h_i] \geq \frac{(1-\alpha)^2}{4}M_i$ acknowledgements are delivered. In the second case we get that at least $(1-\alpha)^2[k_i(n-k_i)+h_i] > (1-\alpha)^2[k_i(n-1)(1-\alpha)+h_i] \geq \frac{(1-\alpha)^3}{2}M_i$ acknowledgements are delivered. Let $c := \min\{\frac{(1-\alpha)^2}{4}, \frac{(1-\alpha)^3}{2}\}$, then obviously every iteration decreases the value of $M_i$ at least by factor $c$. Since

---

[7] Assume that $n$ is large enough such that $f(n)$ is real number.

the value of $M$ at the beginning of the algorithm is $M_1 = O(n^2)$, $\log_{1/c} M_1 = O(\log n)$ steps are sufficient to inform all but a constant number (at most $X\varepsilon$) of vertices and to ensure that the number of remaining hyperactive arcs is linear (at most $X(n-2)$).

## 4  Hypercubes

In this section we consider $d$-dimensional hypercubes. The hypercube $Q_d$ has $2^d$ vertices, and both diameter and edge connectivity are $d$. We present an algorithm that informs all but a constant number of vertices in time $O(d^2)$.

The general idea is the same as for complete graphs: first we perform two initialization steps to make sure there are enough informed vertices for the subsequent analysis to hold. Next, simple rounds are repeated for a sufficient number of times. The analysis, however, is more complicated in this case.

The next lemma covers the initialization steps. In the first step, the initiator sends a message to all its neighbors, and at least one of these messages is delivered. In the second step, the initiator sends a message to all its neighbors again; moreover, each of the vertices informed in the first step sends a message to all its neighbors except the initiator.

**Lemma 2.** *After the first two steps of the algorithm, at least $\frac{1-\alpha}{2}(2d-1)$ vertices are informed.*

For the rest of this section we suppose that there are at least $\frac{1-\alpha}{2}(2d-1)$ informed vertices. We show that after $O(d^2)$ simple rounds all but some constant number of vertices are informed, and there are only linearly many hyperactive arcs. At the end of this section, we shall be able to prove the following theorem.

**Theorem 2.** *Let $\varepsilon \in (0,1)$ be an arbitrary constant. For large enough $d$ it is possible to inform all but at most $X/(1-\varepsilon)$ vertices of $Q_d$ within $O(d^2)$ time steps. Moreover, the number of remaining hyperactive arcs is at most $X(d-1)$.*

In our analysis we need to assert that enough acknowledgements are delivered, given the number of informed vertices. To bound the number of sent messages, we rely heavily upon the following isoperimetric inequality due to Chung et. al. [9]:

*Claim.* [9] Let $S$ be a subset of vertices of $Q_d$. The size of the edge boundary of $S$, denoted as $\partial(S)$ is defined as the number of edges connecting $S$ to $Q_d \setminus S$. Let $\partial(k) = \min_{|S|=k} \partial(S)$, and let lg denote the logarithm of base 2. It holds that

$$\partial(k) \geq k(d - \lg k)$$

The first step in the analysis is to prove that if there are enough uninformed vertices, or enough hyperactive arcs at the beginning of a round $i$, then sufficiently many acknowledgements are delivered in this round:

**Lemma 3.** *Consider a d-dimensional hypercube with $k$ non-informed vertices and $h$ hyperactive arcs. Let $\varepsilon \in (0,1)$ be an arbitrary constant, and let $k > X/(1-\varepsilon)$ or $h > X(d-1)$. Then in the second step of a simple round at least $\beta(h + \partial(k))$ acknowledgements are delivered, where $\beta = (1-\alpha)^2$.*

*Sketch of the proof.* Let $S$ be the set of informed vertices. In the first step of the round, $h + \partial(S)$ messages are sent. Since the edge boundary of informed and uninformed vertices is the same, at least $h + \partial(k)$ messages are sent in the first step of the round. The idea of the proof is to show that $\alpha(h+\partial(k)) \geq d-1$, so in the first step at most $\alpha(h+\partial(k))$ messages are lost, and at least $(1-\alpha)(h+\partial(k))$ of them are delivered. Next we prove that $\alpha(1-\alpha)(h+\partial(k)) \geq d-1$, so in the second step at least $(1-\alpha)^2(h+\partial(k))$ messages are delivered. Since $1-\alpha < 1$, it is sufficient to prove that $\alpha(1-\alpha)(h+\partial(k)) \geq d-1$. If $h > X(d-1)$ then clearly $h + \partial(k) \geq X(d-1)$ and the statement holds. Hence, the main goal of the proof is to show that for $k > X/(1-\varepsilon)$, it holds $\partial(k) \geq X(d-1)$. To do so, the inequality $2^d - k \geq \frac{1-\alpha}{2}(2d-1)$, which is granted by Lemma 2, is used.

In the rest of the proof of Theorem 2 we show that $O(d^2)$ simple rounds are sufficient to inform almost all vertices. The analysis is divided into two parts. In the first part we prove that within $O(d^2)$ rounds at least $2^d/3$ vertices are informed. In the second part we show that another $O(d^2)$ rounds are sufficient to finish the algorithm.

**Lemma 4.** *After performing $O(d^2)$ simple rounds on $Q_d$ at least $2^d/3$ vertices are informed.*

*Sketch of the proof.* Let $l := 2^d - k$ be the number of informed vertices. From Lemma 3 it follows that at least $\beta\partial(k)$ acknowledgements are delivered in one simple round. Since the edge boundary of informed vertices is also the boundary of uninformed vertices, the number of delivered acknowledgements is at least $\beta\partial(l)$. Furthermore, every delivered acknowledgement adds one passive arc, so the number of passive arcs grows at least by $\beta\partial(l)$ each round, which we show to be at least a factor of $\left(1 + \frac{1}{\frac{d}{\beta \lg 3}}\right)$. Because the number of passive arcs cannot grow over $d2^d/3$ without informing at least $2^d/3$ vertices, we get the statement of the lemma.

**Lemma 5.** *Let $\varepsilon \in (0,1)$ be an arbitrary constant, and let $k_i \leq (2/3)2^d$ be the number of uninformed vertices and $h_i$ the number of hyperactive arcs of an $d$-dimensional hypercube at the beginning of round $i$. Then after $O(d^2)$ simple rounds there are at most $X/(1-\varepsilon)$ uninformed vertices and at most $X(d-1)$ hyperactive arcs.*

*Sketch of the proof.* Similarly to the proof of Theorem 1, let us consider the measure $M_i := 2dk_i + h_i$ which decreases with every acknowledgement delivered. We show that as long as the requirements of Lemma 3 hold, $M_i$ decreases in each round by a factor $\left(1 + \frac{\beta \lg(2/3)}{d}\right)$. Since $M_i \leq (5/3)d2^d$, we get the statement of the lemma.

Combining Lemma 2 with Lemma 4 and Lemma 5 completes the proof of Theorem 2.

## 5 Complete Broadcast in Complete Graphs

In Section 3 we have shown how to inform all but some constant number of vertices in a complete graph $K_n$ in time $O(\log n)$. A natural question is to ask if it is possible to inform also the remaining vertices in the same time complexity. In this section we partially answer this question. In particular, we show in the following subsection that if the graph is equipped with a chordal sense of direction, then the complete broadcasting can be performed in time $O(\log n)$. In the subsequent subsection, we show that if the constant $\alpha < 0.55$, complete broadcast can be performed in time $O(\log n)$ without the sense of direction, too.

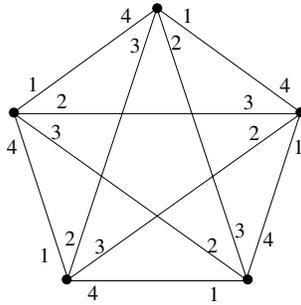### 5.1 Chordal Sense of Direction



**Fig. 1.** $K_5$ with the chordal sense of direction

Let us consider a complete graph with a fixed Hamiltonian cycle $\mathcal{C}$ (unknown to the vertices). We say that the complete graph has a chordal sense of direction if in every vertex the incident arcs are labeled by the clockwise distance on $\mathcal{C}$ (see Figure 1). The notion of a sense of direction has been defined formally for general graphs, and it has been known to significantly reduce the complexity of many distributed tasks (e.g. [17, 18]).

We show how to perform a complete broadcast on a complete graph with the sense of direction in time $O(\log n)$. The process consists of three steps. First, using Theorem 1, all but a constant number of vertices are informed. In the second phase the information is delivered to all but one vertex. In the last phase the remaining single vertex is informed.

The sense of direction is essential to our algorithm. Since there is a unique initiator of the broadcasting, all vertices can derive unique identifiers defined as their distance on $\mathcal{C}$ from the initiator. Furthermore, the sense of direction allows

each vertex to know the identifier of a destination vertex of any of its incident arcs.

**Lemma 6.** *It is possible to inform all vertices but one on complete graphs with chordal sense of direction in time $O(\log n)$. Furthermore, after finishing the algorithm vertex $0$ or vertex $1$ knows a constant number of candidates for the uninformed vertex.*

*Proof.* The outline of the algorithm is as follows: At first the algorithm from Theorem 1 is performed, which ensures that all but a constant number of vertices are informed. Afterwards a significant group of vertices negotiate a common set $U$ of *candidates* for uninformed vertices, such that all uninformed vertices are in $U$ and the size of $U$ is constant. The vertices then cooperate to inform all vertices in $U$ but one. As a side effect, the set $U$ will be known to vertex 0 or vertex 1, hence satisfying the second claim of the lemma. Now we present this algorithm in more detail:

**Phase 1** Run the algorithm from Theorem 1. This phase takes $O(\log n)$ time and ensures that there are at most $X\varepsilon$ uninformed vertices and at most $X(n-2)$ hyperactive arcs.

**Phase 2** Each vertex $v$ that has at most $3X(1+\varepsilon)$ non-passive (i.e. active or hyperactive) links leading to the set of vertices $U_v$ sends a message containing $U_v$ to vertices with number 0 and 1.

Now we show that at least one of these messages is delivered. It is easy to see that there are at least $2n/3$ vertices satisfying the above-mentioned condition, otherwise there would be more than $n/3$ vertices with at least $3X(1+\varepsilon)$ non-passive links, so there would be more than $nX(1+\varepsilon)$ active or hyperactive arcs. But since the number of uninformed vertices is at most $k \leq X\varepsilon \leq n/2$ for large $n$, there are $k(n-k) \leq X\varepsilon(n-X\varepsilon)$ active arcs. So the total number of active or hyperactive arcs is at most $X\varepsilon(n-X\varepsilon)+X(n-2) \leq Xn(1+\varepsilon)$, which is a contradiction.

The rest of the algorithm will be time-multiplexed into two parts. In even time steps, the case that the vertex 0 received a message in phase 2 is processed. In odd time steps, the case that the vertex 1 received a message is processed analogously. Hence, we can restrict to the first case in the rest of the algorithm description. As there are only two cases the asymptotic complexity of the algorithm is unaffected by the multiplexing.

**Phase 3** The vertex 0 received at least one message containing a set of possibly uninformed vertices. It is obvious that the set of uninformed vertices is a subset of every received message. Hence the set $U$ can be defined as the intersection of the received messages: Indeed, every uninformed vertex is in $U$ and the size of $U$ is at most $3X(1+\varepsilon) = O(1)$. The set $U$ is then distributed using the algorithm in Theorem 1 among at least $n - X\varepsilon$ vertices in time $O(\log n)$.

**Phase 4** There are at least $n - X\varepsilon$ vertices aware of the set $U$. In this phase they cooperate to inform all but one vertex in $U$, using an idea similar to Lemma 2 in [12]: every vertex aware of the set $U$ iterates through all pairs

$[i, j]$ $(i, j \in U)$ in lexicographical order; in each time step it sends the original message to both vertices $i$ and $j$. Since in each time step at least $2n - X\varepsilon$ messages are sent, at least one of them is delivered (for large enough $n$). As all vertices process the same pair $[i, j]$ in every time step, this ensures that a new vertex is informed whenever both $i$ and $j$ were uninformed. Hence, at the end of this phase all vertices but one are informed. The time complexity of this phase is $O(|U|^2) = O(1)$.

It is obvious that after finishing the Phase 4 the claim of the Lemma holds.

Finally, we show how to inform the last remaining vertex, thus proving the following theorem:

**Theorem 3.** *It is possible to perform broadcasting on complete graphs with chordal sense of direction in time $O(\log n)$.*

*Sketch of the proof.* Suppose that after performing the algorithm from Lemma 6 all vertices with the exception of some vertex $v$ are informed and vertex 0 knows a set $U$ of constant size containing candidates for $v$. The algorithm from Lemma 6 is used again to broadcast $U$ with two possible outcomes: either $v$ was informed during the broadcast, or all other vertices have the same set of candidates, which they try to inform one by one.

## 5.2 Without Sense of Direction

As a last result in this paper we show that it is possible to perform broadcasting on complete graphs in time $O(\log n)$ for small values of $\alpha$ (i.e. $\alpha \lesssim 0.55$) even without the sense of direction. The idea is to use the algorithm from Theorem 1 to inform all but constantly many vertices. Next, instead of repeating 2-step simple rounds, some $\log n$-step extended rounds are repeated, such that each extended round informs a yet uninformed vertex. During an extended round messages are sent for $O(\log n)$ steps in such a way that in every step the number of hyperactive arcs is decreased by some factor[8] unless a new vertex is informed.

**Theorem 4.** *Let $1 - \alpha - 2\alpha^2 + \alpha^3 > 0$. Then it is possible to perform broadcasting on complete graphs without sense of direction in time $O(\log n)$.*

*Proof.* The algorithm is described as Algorithm 1.

At first, the algorithm from Theorem 1 is performed, ensuring that there are at most $k \leq X\varepsilon$ uninformed vertices and at most $h \leq X(n-2)$ hyperactive arcs ($X$ and $\varepsilon$ have the same meaning as in Theorem 1). The purpose of one iteration of the loop on lines 3–14 is to inform at least one uninformed vertex. Taking $L_1 := X\varepsilon = O(1)$ ensures that all vertices will be informed.

The loop on lines 4–10 reduces the number of hyperactive arcs to zero unless a new vertex is informed. One iteration of this loop either informs a new vertex or reduces the number of hyperactive arcs from $h$ to $(1 - Y/2)h$, where $0 < Y < 1$ is

---

[8] in this part we need the assumption that $\alpha$ is small enough

**Algorithm 1** Complete graphs without sense of direction

---

1: perform almost-complete broadcast according to Theorem 1
2: let $k$ denote the number of uninformed vertices, let $h$ denote the number of hyper-
active arcs
3: **loop** $L_1$ times     // *Perform $L_1$ extended rounds*
4:     **loop** $L_2(n)$ times     // *In each iteration $h$ decreases by a constant factor*
5:         $E :=$ set of all currently active or hyperactive arcs; $P := \emptyset$
6:         **loop** $L_3$ times
7:             send the message via all arcs in $E \cup P$
8:             $P := P \cup \left\{ e \mid \begin{array}{l} \text{a message has been delivered in this step} \\ \text{via the opposite arc of } e \end{array} \right\}$
9:         **end loop**
10:     **end loop**
11:     **loop** $L_4(n)$ times     // *Inform new vertex and decrease a*
12:         perform one simple round
13:     **end loop**
14: **end loop**

The values of $L_1$, $L_2(n)$, $L_3$ and $L_4(n)$ are specified in the analysis of the algorithm, such that $L_1, L_3 = O(1)$ and $L_2(n), L_4(n) = O(\log n)$.

---

a constant (depending on $\alpha$) defined later. Hence the number of hyperactive arcs decreases exponentially with number of iterations of the loop and $\log_{1/(1-Y/2)} h$ iterations are sufficient to eliminate all hyperactive arcs. Since the condition $h \leq X(n-2)$ holds before every execution of the loop (this is provided either directly by Theorem 1 or by the loop on lines 11–13), we can define $L_2 := \log_{1/(1-Y/2)}(X(n-2)) = O(\log n)$.

Now we describe one iteration of the loop on lines 4–10. We distinguish two types of arcs that are hyperactive at the beginning of the considered iteration: An arc $e$ is a *single hyperactive arc* if and only it is hyperactive and the opposite arc of $e$ is passive at the beginning of the iteration. Otherwise (i.e. if both $e$ and the opposite arc of $e$ are hyperactive at the beginning of the iteration), $e$ is a *double hyperactive arc*.

Let $E$ be the set of all active or hyperactive arcs at the beginning of the iteration, and $P$ be the set of all arcs opposite to arcs through which some message has been delivered in the current iteration. Furthermore, let $k'$ be the number of uninformed vertices at the beginning of the current iteration, $h'$ be the number of hyperactive arcs at the beginning of the current iteration and $p = |P \setminus E|$ be number of arcs in $P$ that were passive at the beginning of the current iteration. It clearly holds that $|E| = k'(n-k') + h'$ and that $k'(n-k') + h' + p$ messages are sent on every execution of line 7. Since at least $n-1$ messages are lost (because we may assume that no new vertex is informed), at most $\alpha(k'(n-k') + h' + p)$ of them are lost, i.e. at least $(1-\alpha)(k'(n-k') + h' + p)$ are delivered.

Now assume by contradiction that the number of hyperactive arcs does not decrease below $(1-Y/2)h'$, and no new vertices are informed during the current iteration of the loop on lines 4–10. Consider any message delivered over an arc

$e$ which is a double hyperactive arc or an arc in $P \setminus E$; it is easy to see that the opposite arc of $e$ is passive after the delivery and that it was hyperactive at the beginning of the iteration. This fact yields that at most $(Y/2)h'$ messages are delivered over a double hyperactive arc or an arc in $P \setminus E$ on any execution of line 7.

Now we show a lower bound on the number of messages that pass over double hyperactive arcs or arcs in $P \setminus E$ or single hyperactive arcs whose opposite arcs are not in $P \setminus E$. Intuitively, every such message ensures some progress of the algorithm, since either an arc is made passive (in the first two cases) or a new arc is added to $P \setminus E$ (in the third case). As no messages passes over active arcs by our assumption, and at most $p$ messages pass over single hyperactive arcs whose opposite arcs are in $P \setminus E$, there are at least $(1-\alpha)(k'(n-k')+h'+p)-p$ messages satisfying one of these three cases. Using the inequalities $k'(n - k') \geq n - 1$ and $p \leq h'$ yields $(1 - \alpha)(k'(n - k') + h' + p) - p \geq (1 - \alpha)(n - 2) + (1 - 2\alpha)h'$. Because $h' \leq X(n - 2)$ which is equivalent to $(n - 2) \geq \alpha(1 - \alpha)h'$, we have $(1-\alpha)(k'(n-k')+h'+p)-p \geq (1-\alpha-2\alpha^2+\alpha^3)h'$. Defining $Y := 1-\alpha-2\alpha^2+\alpha^3$, which is positive and less than one by the assumption of the Lemma, we have shown that there are at least $Yh'$ messages satisfying one of the three cases.

However, at most $(Y/2)h'$ of them satisfies the first two cases, hence there are at least $(Y/2)h'$ arcs added to $P$ in every execution of line 8. So taking $L_3 := 2/Y + 1$ ensures that $P$ contains opposite arcs to all single hyperactive arcs at the beginning of the last iteration of the loop on lines 6–9. However, this is a contradiction with the fact that new arcs are added to $P$ at line 8.

We conclude the proof with the analysis of the loop on lines 11–13. In the first iteration of the loop a new vertex is informed, because there are no hyperactive arcs left after the loop on lines 4–10 finished (unless the new vertex has already been informed in that loop). Due to Theorem 1, next $O(\log n)$ iterations are sufficient to ensure that $h \leq X(n - 2)$, which is an invariant required by the loop on lines 4–10. Hence putting $L_4(n) := O(\log n)$ (according to Theorem 1) is sufficient to make the algorithm work correctly in time $L_1(L_2(n)L_3 + L_4(n)) = O(\log n)$.

## 6    Conclusions, Open Problems, and Further Research

We have studied the problem of almost complete broadcast under the model of fractional dynamic faults with threshold. We showed that both in complete graphs and in hypercubes, it is possible to inform all but constantly many vertices in time $O(D \log n)$ where $D$ is the diameter of the graph and $n$ is the number of vertices.

Moreover, we have proved that if the complete graph is equipped with the chordal sense of direction, or the parameter $\alpha < 0.55$, a complete broadcast can be performed in time $O(\log n)$.

This research leaves many open questions and directions for further research, from which we mention at least a few. One obvious question is to ask if it is possible to perform a complete broadcast in complete graphs also for large

values of $\alpha$ in polylogarithmic time. The difficulty of broadcast in the fractional dynamic model with threshold stems from the fact that, in order to inform the last few vertices, all informed vertices must cooperate very tightly. In general, the relationship between the almost complete and complete broadcast in various models is worth studying. We have also not considered non-constant values of $\alpha$. It would be interesting to extend our results to more general classes of graphs.

We finish by noting that there is a lack of any non-trivial lower bounds in the model of fractional faults with threshold.

# References

1. R. Ahlswede, L. Gargano, H. S. Haroutunian, and L. H. Khachatrian. Fault-tolerant minimum broadcast networks. *Networks*, 27, 1996.
2. A. Bagchi and S. L. Hakimi. Information dissemination in distributed systems with faulty units. *IEEE Transactions on Computers*, 43(6):698–710, 1994.
3. K. A. Berman and M. Hawrylycz. Telephone problems with failures. *SIAM Journal on Algebraic and Discrete Methods*, 7(1):13–17, 1986.
4. P. Berman, K. Diks, and A. Pelc. Reliable broadcasting in logarithmic time with Byzantine link failures. *Journal of Algorithms*, 22(2):199–211, 1997.
5. L. A. Bjork. Recovery scenario for a db/dc system. In *ACM'73: Proceedings of the annual conference*, pages 142–146, New York, NY, USA, 1973. ACM Press.
6. J.-M. Chang and N. F. Maxemchuk. Reliable broadcast protocols. *ACM Transactions on Computer Systems*, 2(3):251–273, 1984.
7. B. Chlebus, K. Diks, and A. Pelc. Broadcasting in synchronous networks with dynamic faults. *Networks*, 27, 1996.
8. B. S. Chlebus, K. Diks, and A. Pelc. Optimal broadcasting in faulty hypercubes. In *FTCS*, pages 266–273, 1991.
9. F. R. K. Chung, Z. Füredi, R. L. Graham, and P. Seymour. On induced subgraphs of the cube. *Journal of Combinatorial Theory Series A*, 49:180–187, 1988.
10. E. W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Commun. ACM*, 17(11):643–644, 1974.
11. K. Diks and A. Pelc. Almost safe gossiping in bounded degree networks. *SIAM Journal on Discrete Mathematics*, 5(3):338–344, 1992.
12. S. Dobrev, R. Královič, R. Královič, and N. Santoro. On fractional dynamic faults with threshold. In P. Flocchini and L. Gasieniec, editors, *SIROCCO*, volume 4056 of *Lecture Notes in Computer Science*, pages 197–211. Springer, 2006.
13. S. Dobrev and I. Vrťo. Optimal broadcasting in hypercubes with dynamic faults. *Information Processing Letters*, 71(2):81–85, 1999.
14. S. Dobrev and I. Vrťo. Dynamic faults have small effect on broadcasting in hypercubes. *Discrete Applied Mathematics*, 137(2):155–158, 2004.
15. S. Dolev. *Self-stabilization*. MIT Press, Cambridge, MA, USA, 2000.
16. M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32(2):374–382, 1985.
17. P. Flocchini, B. Mans, and N. Santoro. On the impact of sense of direction on message complexity. *Information Processing Letters*, 63(1):23–31, 1997.
18. P. Flocchini, B. Mans, and N. Santoro. Sense of direction in distributed computing. In *International Symposium on Distributed Computing*, pages 1–15, 1998.
19. P. Fraigniaud. Asymptotically optimal broadcasting and gossiping in faulty hypercube multicomputers. *IEEE Transactions on Computers*, 41(11):1410–1419, 1992.

20. L. Gasieniec and A. Pelc. Broadcasting with linearly bounded transmission faults. *Discrete Applied Mathematics*, 83(1–3):121–133, 1998.
21. S. Hedetniemi, S. Hedetniemi, and A. Liestman. A survey of broadcasting and gossiping in communication networks. *Networks*, 18:319–349, 1988.
22. R. Královič, R. Královič, and P. Ružička. Broadcasting with many faulty links. In J. F. Sibeyn, editor, *SIROCCO*, volume 17 of *Proceedings in Informatics*, pages 211–222. Carleton Scientific, 2003.
23. R. Královič and R. Královič. Rapid almost-complete broadcasting in faulty networks. *Technical report*, 2007, arXiv:cs.DC/0703122.
24. Z. Liptak and A. Nickelsen. Broadcasting in complete networks with dynamic edge faults. In F. Butelle, editor, *OPODIS*, Studia Informatica Universalis, pages 123–142. Suger, Saint-Denis, rue Catulienne, France, 2000.
25. M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.
26. A. Pelc. Broadcasting in complete networks with faulty nodes using unreliable calls. *Information Processing Letters*, 40(3):169–174, 1991.
27. A. Pelc and D. Peleg. Feasibility and complexity of broadcasting with random transmission failures. In *PODC '05: Proceedings of the twenty-fourth annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 334–341, New York, NY, USA, 2005. ACM Press.
28. N. Santoro and P. Widmayer. Distributed function evaluation in the presence of transmission faults. In *SIGAL '90: Proceedings of the International Symposium on Algorithms*, pages 358–367, London, UK, 1990. Springer-Verlag.
29. I. Stanoi, D. Agrawal, and A. E. Abbadi. Using broadcast primitives in replicated databases. In *ICDCS '98: Proceedings of the The 18th International Conference on Distributed Computing Systems*, pages 148–155, Washington, DC, USA, 1998. IEEE Computer Society.