Message Complexity Versus Space Complexity in Fault Tolerant Broadcast Protocols

Shlomo Moran
Department of Computer Science, the Technion, Haifa 32000, Israel

Let N be a network of asynchronous processors, viewed as vertices, communicating by sending messages over unreliable unidirectional edges, and let r be a specified vertex in N. We consider the problem of constructing efficient and reliable protocols to broadcast messages from r to all other vertices of N: Suppose that for some vertex v in N, at least k edges must be deleted in order to disconnect v from r; we say that a protocol PR is *reliable* for v if whenever less than k edges fail to deliver a message broadcasted from r by PR, this message will eventually reach v. A protocol is *faithful* if it is reliable for all vertices of the network. A general lower bound on the message complexity of faithful protocols, which is at most linear in the network size, is given. It is also shown that this bound can always be achieved by protocols which use the local memories of the vertices to record messages. On the other hand, it is shown that for certain networks, all faithful protocols that use no memory for local computations have a message complexity which is exponential in the network size. A characterization of networks that have faithful protocols with optimal message *and* space complexities is also given.

1. INTRODUCTION

We consider an asynchronous network consisting of vertices (representing processors) which communicate by sending messages along unreliable edges. The network has no shared memory, so that the only way for processors to communicate is by sending messages. The time required for a message to go along an edge is unpredictable and unbounded; moreover, an edge may fail to deliver a message, either by destroying it or by changing its content (e.g., as a result of channel noise). It is also assumed that edges may fail and recover at any time, and that the network do not have a mechanism that detects such failures.

A protocol (or a distributed algorithm) over such a network is an algorithm which is executed at each individual vertex in order to accomplish some common task, like

Part of this work was done while the author was at IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.

NETWORKS, Vol. 19 (1989) 505–519

CCC 0028-3045/89/050505-15$04.00

choosing a leader, reaching an agreement etc. In the last few years numerous protocols were designed and investigated for various classes of networks (e.g. [1,4–12,16,17]).

The quality of a given protocol in a given network can be evaluated by the following properties:

1. Reliability: The way the protocol fulfills its task in the presence of faulty edges, which may destroy or change the content of messages.
2. Communication cost: The *message complexity* of the protocol in the network.
3. Computational overhead: The *space* or *time complexity* of the protocol, which is the amount of space or time of the vertices in the network which is consumed by the protocol.

While properties 1 and 2 above were studied for many protocols (e.g., the list of references above), only a little work on the relation between all the three properties above has been done. In this paper we study this relation for *broadcast* protocols, which are possibly the simplest protocols that require the cooperation of the whole network. Those protocols distinguish a single vertex, r, in the network, and their task is to enable r to deliver messages it receives from an outside source (e.g., an upper layer algorithm) to all other vertices in the network. Broadcast protocols play an essential roll in many distributed algorithms. For instance, in most of the Byzantine general protocols, in each step every vertex broadcasts its current state to all other vertices in the network. We shall restrict the discussion to networks with unidirectional edges, in which each edge can deliver messages only in one specified direction (see [1,6,7,11,17]). The question whether these results can be generalized to bidirectional networks seems to remain an interesting open problem. It should be noted, however, that our results are applicable to bidirectional networks in which a failure of an edge to deliver messages in one direction does not imply a similar failure in the other direction.

The message complexity and the space (or time) complexity of a broadcast protocol in a given network will be defined as a function of the number of messages broadcasted by the protocol. More specifically, the message complexity of a broadcast protocol is the maximal possible number of times a single message broadcasted from r is sent along edges of the network, and the space (time) complexity of such protocol is the amount of space (time) it consumes as a function of the number of messages it broadcasts. Formal definitions of these concepts will be given in the next section. In this paper we shall use only space complexity as a measure of the computation cost of a given protocol; however, generalizations of the results to similar results concerning time complexity are, in general, straightforward.

Broadcast protocols with optimal message and space complexities are obtained by using the edges of a directed spanning tree rooted at r to broadcast messages—the message complexity of such a protocol is $n - 1$ for networks with n vertices, and the space complexity at each individual vertex is a constant, independent of the number of messages it broadcasts (or the number of vertices in the network). These protocols are highly unreliable, since for each vertex v, one failure of an edge to deliver a message may prevent v from receiving this message. In [10] the following *multitree* approach is suggested in order to increase the reliability of the protocol: Assume that G contains k edge-disjoint spanning trees rooted at r. Then by broadcasting messages simultaneously along the edges of each of these trees, we are guaranteed that if less

then k edges fail to deliver a message, this message will eventually reach every vertex in the network. Moreover, it is not hard to see that this protocol has the minimal possible message and space complexities required for achieving this reliability. However, the *maximum* number of edge-disjoint spanning trees rooted at r equals the *minimum* number of edges whose deletion disconnect some vertex in G from r [2,13]. In other words: The maximum reliability that can be achieved by using such as protocol is determined by the "lowest common level" of the network. In this paper we deal with protocols that achieve, in a sense, the maximum possible reliability for each individual vertex in the network. We show that, unlike the cases discussed above, the existence of efficient protocols of this kind may depend heavily on the topology of the given network:

Let v be a vertex that may be disconnected from r by deleting k edges of the network, but not by deleting $k - 1$ edges. Then if k or more edges may fail to deliver a message broadcasted by r, no protocol can guarantee that this message will ever reach v. A protocol is *reliable* for v iff whenever less than k edges fail to deliver such a message, this message will eventually reach v. A protocol is *faithful* iff it is reliable for all the vertices in the network.

The property of being faithful appears to be a natural and interesting extension of reliability properties of protocols studied in [10]. The goal of this paper is to study the design and the cost, in term of computational overhead, of such protocols. In particular, we derive a general lower bound on the message complexity of faithful protocols, and then we show that this lower bound can always be achieved by simple protocols that can be designed in time that is polynomial in the size of the network. However, these protocols have the disadvantage that their space complexity is linear in the number of messages. Next we consider protocols with a constant space complexity, and show that in certain networks, if such protocols are faithful, then their message complexity must be larger by an exponential factor than the above mentioned lower bound. We conclude by showing that those networks which admit faithful broadcast protocols with optimal message *and* space complexities must satisfy a certain graph theoretic property, which is a generalization of Edmonds' branching theorem [2,13,15,18–20].

2. PRELIMINARY RESULTS

2.1. Graph Theoretic Preliminaries

We give below some definitions from graph theory, which are used in the sequel. The reader is referred to any standard text book in graph theory (e.g., [3]) for a more detailed exposition of these definitions.

A directed graph (digraph) $G = (V,E)$ consists of a set of vertices V and a set of edges E, where each edge e in E is associated with an ordered pair of vertices (u,v). We say that e is directed *from u to v*, and denote it by $u \xrightarrow{e} v$; we also say that e is *leaving u* and *entering v*, and that u is the *tail* and v is the *head* of e. The number of edges entering (leaving) a vertex u is denoted by $d_{in}(u)$ ($d_{out}(u)$). For a subset F of E, $G - F$ denotes the graph $G' = (V,E - F)$.

A directed path in G is a sequence $p = (u_0,e_1,u_1, \ldots ,e_k,u_k)$, where e_i is an edge

directed from u_{i-1} to u_i; $V(p)$ ($E(p)$) denotes the set of vertices (edges) occurring in the path p. For $s,t \in V$, an (s,t) *path* in G is a directed path from s to t. If p_1 is an (s,v) path and p_2 is a (v,t) path, then $p_1 \cdot p_2$ is the (s,t) path resulted by the concatenation of p_1 and p_2. A digraph $T = (V,E)$ is a *directed tree rooted at* r iff $r \in V$ and T contains a unique (r,v) path for each vertex v in $V - \{r\}$.

A set C of edges is an (s,t) *cut* iff there is a set $S_C \subseteq V$ such that $s \in S_C$, $t \in V - S_C$ and C is the set of all the edges in G which are directed from a vertex in S_C to a vertex in $V - S_C$. Note that every set of vertices S_C that separates s from t uniquely defines an (s,t) cut C, and vice-versa. C is a *minimum* (s,t) *cut* if it is an (s,t) cut of minimum possible cardinality; this cardinality is denoted by $\delta_G(s,t)$. An edge e is an (s,t) *bottleneck* in G if $\delta_{G-\{e\}}(s,t) = \delta_G(s,t) - 1$. Thus, e is an (s,t) bottleneck iff there is a minimum (s,t) cut that contains e.

We shall use, without mentioning it, the following version of Menger's theorem.

Theorem [14]. Let $G = (V,E)$ be a digraph. Then for each pair (s,t) in V there is a set of $\delta_G(s,t)$ edge-disjoint (s,t) paths.

2.2. Broadcast Protocols and Faithful Broadcast Protocols

A *broadcast network*, or simply a *network*, is defined as an ordered pair $N = (G,r)$, where $G = (V,E)$ is a digraph and $r \in V$ is a specified vertex in V. Each vertex represents a processor, and the processors communicate by sending messages along the edges of G, in their specified directions. A broadcast protocol, PR, in a network $N = (G,r)$, is a protocol which is intended to deliver messages from r to all other vertices of G. Such a protocol consists of $|V|$ programs, one for each vertex v in the network; $PR(v)$ denotes the program executed by vertex v. It is convenient to view $PR(v)$ as a nonterminating program that may include operations performed by v in its local memory, as well as the operations *RECEIVE(e)* and *SEND(F)*. These last two operations use a specified buffer which may contain a single message received or sent by v, and are defined below:

RECEIVE(e) is an operation whose argument e is an edge entering v; this operation then fetches a message that was sent along e, stores it in the specified buffer, and then deletes it from e. If there are no messages available in e this operation does nothing. *SEND(F)* is an operation whose argument F is a subset of the edges leaving v. This operation sends the message stored in the specified buffer of v along each edge in F, and then deletes it from the buffer. Again, if the buffer is empty then this operation does nothing.

A broadcast protocol is initiated by having r send a message it receives from an outside source along some of the edges leaving it. For simplicity, we assume that all messages broadcasted by the protocol are distinct.

PR is a *routing protocol* iff each $PR(v)$ contains only *SEND* and *RECEIVE* operations. Routing protocols are of special interest, since their executions require no computational overhead. In fact, the "standard" protocols based on spanning trees rooted at r, as well as similar protocols for networks with bidirectional edges given in [10], are routing protocols. It is implicit in the definition that routing protocols cannot modify the messages broadcasted by them. In the conclusions we shall point out some possible effects of removing this restriction, thus allowing such protocols to append to the messages they broadcast information that can be used by the protocol.

Let *PR* be a broadcast protocol on $N = (G,r)$, and let $v \in V - \{r\}$. The *reliability of PR for v* is the maximal number k satisfying the property (R) below:

(R) Whenever less than k edges fail to deliver a message broadcasted by r using *PR*, this message will eventually reach v.

It follows easily from the definitions that the reliability of *PR* for v is at most $\delta_G(r,v)$. *PR* is *reliable* for v if its reliability for v is equal to $\delta_G(r,v)$.

Definition 2.1. A broadcast protocol is *faithful* iff it is reliable for all $v \in V - \{r\}$.

Let *PR* be a broadcast protocol over a network $N = (G,r)$. The *message complexity* of *PR* is the maximal number of times a single message broadcasted from r may be received by vertices in N (by using the *RECEIVE* operation). The *space complexity* of *PR* at a vertex v in N is the space complexity of the program $PR(v)$, defined as a function f_v of the number of distinct messages received by v. Thus, a function f_v is a space complexity of $PR(v)$ if during the reception of at most n distinct messages, $PR(v)$ uses at most $f_v(n)$ space units; for this case, it is assumed that each message occupies one space unit. If for each $v \in V$, f_v is the space complexity of $PR(v)$, then the function f defined by $f(n) = \max_v\{f_v(n)\}$ is a space complexity of *PR*. In other words—the space complexity of *PR* is the least upper bound of the space complexities of the $PR(v)$'s.

The following definition and lemma provide a simple lower bound on the message complexity of faithful protocols.

Definition 2.2. Let $N = (G,r)$ be a network. Then the *total reliability* of N, denoted by $TR(N)$, is the sum:

$$TR(N) = \sum_{v \in V - \{r\}} \delta_G(r,v).$$

Lemma 2.1. Let *PR* be a faithful protocol in N. Then the message complexity of *PR* is at least $TR(N)$.

Sketch of Proof. Assume first that N is acyclic, and consider an execution of *PR* on a single message m, such that no edge fails during this execution. We claim that for each $v \in V - \{r\}$, v must receive m along at least $\delta_G(r,v)$ distinct edges entering v:

Let $\delta_G(r,v) = k$, and assume that during the execution of *PR*, v receives m along $k' < k$ edges entering it. Since N is acyclic, the tails of these k' edges cannot receive any message originated by v, and hence cannot distinguish this execution from a similar one in which all the messages sent along these edges are destroyed. In this latter execution v will never receive m—contradicting the assumption that *PR* is faithful, and hence reliable for v.

If N is not acyclic, a similar argument applies for executions in which arbitrarily long delays are imposed on all messages whose route close a cycle in N. The details are left to the reader.

It follows that the number of edges along which m must be sent is at least

$$\sum_{v \in V - \{r\}} \delta_G(r,v) = TR(N)$$

which implies the lemma. ∎

3. BOTTLENECKS, CRITICAL NETWORKS, AND MESSAGE-OPTIMAL PROTOCOLS

Let $N = (G,r)$ be a given network. Recall that an edge e in G is an (r,t) bottleneck iff $\delta_{G-\{e\}}(r,t) < \delta_G(r,t)$. We say that e is a *bottleneck* (in N) if it is an (r,t) bottleneck for some $t \in V - \{r\}$. In this section we first provide a simple characterization of bottlenecks, and later use this characterization to prove the existence of message-optimal faithful protocols, and also to construct efficiently such protocols.

Lemma 3.1. Let $\delta_G(r,t) = k$. An edge e is an (r,t) bottleneck iff in every set of k edge-disjoint (r,t) paths, one of the paths contains e.

Proof. By Menger theorem [14].

Lemma 3.2. Let $N = (G,r)$ be a network, and let the edge $u \xrightarrow{e} v$ be a bottleneck in N. Then e is an (r,v) bottleneck.

Proof. By definition, e is an (r,t) bottleneck for some $t \in V - \{r\}$. If $t = v$ then we are done, so assume that $t \neq v$.

Let $\delta_G(r,v) = l$. By Lemma 3.1, it is suffices to show that in every set $\{p_1,\ldots,p_l\}$ of l edge-dijoint (r,v) paths one of the p_i's contains e. We shall assume that there is such a set in which e is not used by any of the p_i's, and derive a contradiction.

Let $\delta_G(r,t) = k$ and let $C = \{e_1,\ldots,e_k\}$ be a minimum (r,t) cut containing $e = e_k$. For $i = 1,\ldots,k$ let $e_i = u_i \rightarrow v_i$, where $u_i \in S_C$ and $v_i \in V - S_C$ ($v_k = v$). Since C is also an (r,v) cut, each (r,v) path p_i must contain at least one edge of C. By the assumption, none of the p_i's contains $e = e_k$, and hence $k > l$. Without loss of generality, let e_i be the last edge of C occurring in p_i for $i = 1,\ldots,l$. Then each $p_i = p_{i1} \cdot p_{i2}$, where p_{i1} is an (r,v_i) path, whose last edge is e_i, and p_{i2} is a (v_i,v) path, with $V(p_{i2}) \subseteq V - S_C$ (if $v_i = v$ then p_{i2} is the empty path.) Similiarly, there are k edge-disjoint (r,t) paths $\{q_1,\ldots,q_k\}$, where each q_i contains exactly one edge of C. Let $q_i = q_{i1} \cdot q_{i2}$, where q_{i1} is an (r,v_i) path whose last edge is e_i and q_{i2} is an (v_i,t) path. In particular, q_{k1} is an (r,v) path and $V(q_{k1}) \subseteq S_C \cup \{v\}$.

The proof is now completed by the observation that the set of paths $\{q_{i1} \cdot p_{i2}\colon i = 1,\ldots,l\} \cup \{q_{k1}\}$ is a set of $l + 1$ edge-disjoint (r,v) paths. Thus we have that $l + 1 \leq \delta_G(r,v) = l$, which is the desired contradiction. ∎

Definition 3.1. A network $N = (G,r)$ is *critical* if every edge e in G is a bottleneck in N.

The main positive result of this paper will follow from the next theorem:

Theorem 3.3. Let $G = (V,E)$ be a digraph and let $r \in V$ be such that $N = (G,r)$ is critical. Then for every $v \in V - \{r\}$ it holds that $d_{in}(v) = \delta_G(r,v)$. In particular $|E| = TR(G,r)$.

Proof. Clearly, for all v it holds that $d_{in}(v) \geq \delta_G(r,v)$. Assume, for contradiction, that for some v we have that $d_{in}(v) > \delta_G(r,v)$. Then, clearly, at least one of the edges entering v is not an (r,v) bottleneck. Let e be such an edge; then by Lemma 3.2, e is not an (r,t) bottleneck for any $t \in V$, which contradicts the assumption that G is critical for r. ∎

Corollary 3.4. Given $G = (V,E)$ and r, there is an $O(|E| \cdot TR(G,r))$ algorithm to construct a subgraph $G' = (V,E')$ of G such that (G',r) is critical and $TR((G,r)) = TR((G',r))$.

Proof. (An outline): G' is constructed by repeatedly deleting from E edges $u \xrightarrow{e} v$ which are not (r,v) bottlenecks. By Lemma 3.2 the deletion of such edges does not decrease $TR(G,r)$. The task is accomplished by solving $|V| - 1$ maximum flow problems, in the following way:

For each vertex v, in its turn, compute a maximum $(0 - 1)$ flow from r to v, and delete all the edges entering v whose flow function is 0 in this flow. Clearly, those edges are not (r,v) bottlenecks, and hence, By Lemma 3.2, are not (r,t) bottlenecks for any vertex t. Note that after the deletion of those edges we must have that $d_{in}(v) = \delta_G(r,v)$, as a maximum (0–1) flow from r to v consists of $\delta_G(r,v)$ edge-disjoint (r,v) paths. It follows that the graph G' produced after the procedure is repeated for all vertices in $V - \{r\}$ has exactly $TR(G,r)$ edges, and $\delta_{G'}(r,v) = \delta_G(r,v)$ for all $v \in V - \{r\}$.

The computation of a maximum $(0 - 1)$ (r,v) flow can be done by using one of the algorithms for maximum flow which are based on the construction of augmenting paths [3], in $O(\delta_G(r,v) \cdot |E|)$ time, since it requires the construction of $\delta_G(r,v)$ augmenting paths, each in $O(|E|)$ time. The total time required by the algorithm is, therefore

$$O\left(\sum_{v \in V - \{r\}} \delta_G(r,v) \cdot |E|\right) = O(TR(G,r) \cdot |E|)$$

as claimed. ∎

Theorem 3.5. For any network $N = (G,r)$ there is a faithful protocol of optimal message complexity and linear space complexity. Moreover, such a protocol can be designed in $O(|E| \cdot TR(G,r))$ time.

Proof. Let $G' = (V,E')$ be a subgraph of G critical for r. We define a protocol PR which uses the edges in E' to broadcast messages originated by r:

Let $v \in V - \{r\}$ be given. Let $\{e_1, \ldots, e_k\}$ be the edges in E' entering v and let F_v be the set of edges in E' leaving v (note that by Theorem 3.3 we must have that $k = \delta_G(r,v)$). The protocol $PR(v)$ at vertex v is given below (the *RECEIVE* and *SEND* operations in it were defined in Section 2.2). This protocol uses a list *OLD* to record the messages received by v.

```
repeat
  for i = 1 to k do
    begin
      RECEIVE(e_i);
        if the message received is not
        in OLD then
          begin
            add the message to OLD;
            SEND(F_v)
          end (of the if statement);
```

end (of the *for* statement)
for ever

The protocol at vertex r, $PR(r)$, will send each message broadcasted from r on all edges in E' leaving r.

Clearly, every message broadcasted from r will be sent at most one time along each edge in E', and hence the message complexity of PR is bounded by $|E'| = TR(G,r)$. Also, if for some $v \in V$, less than $\delta_G(r,v)$ edges fail to deliver a message broadcasted by r, then G' contains an (r,v) path p which contains none of these faulty edges. Hence that message will be forwarded along the edges of p with no interruptions, and will eventually reach v. This means that PR is reliable for each $v \in V - \{r\}$, and hence is faithful.

The complexity of designing PR is the complexity of constructing G', which by Corollary 3.4 is $O(|E| \cdot TR(G,r))$. ■

4. NETWORKS THAT HAVE NO EFFICIENT AND FAITHFUL ROUTING PROTOCOLS

In this section we prove that in some cases, using the local memory of the nodes in the network is essential for achieving message efficient faithful protocols; in particular, it is shown that for certain networks, every faithful routing protocol must have an exponential message complexity. We start with the following observation:

Let PR be a routing protocol over a network $N = (G,r)$. Then, without loss of generality, we may assume that for each vertex v in V, $PR(v)$ is of the form:

repeat
  $RECEIVE(e_1)$
  $SEND(F_1)$
  :
  .
  $RECEIVE(e_k)$
  $SEND(F_k)$
for ever

where $e_1, \ldots, e_k$ are distinct edges entering v and $F_1, \ldots, F_k$ are subsets of the edges leaving v (some of which are possibly empty). In particular, we may assume that for each edge $u \rightsquigarrow v$, the operation $RECEIVE(e)$ appears at most one time in $PR(v)$: If it appears more than once, then we restrict our discussion to those executions of PR in which all the messages that arrive on e are fetched by the first occurrence of $RECEIVE(e)$ in $PR(v)$, which makes this protocol behave exactly like one in which $RECEIVE(e)$ appears exactly once.

It follows that each $PR(v)$ may be identified by a binary relation $\{(e_1,f_1), \ldots, (e_k,f_k)\}$, where $e_1, \ldots, e_k$ are (not necessarily distinct) edges entering v, $f_1, \ldots, f_k$ are edges leaving v, and every message received by $PR(v)$ via e_i is sent by $PR(v)$ along f_i. Hence, a routing protocol PR can be identified by:

1) The binary relation R which is the union of the binary relations related to $PR(v)$ over all $v \in V$, and by

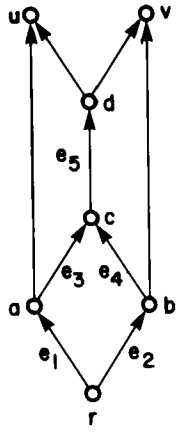

FIG. 1. The graph B.

2) the edges leaving r along which $PR(r)$ sends the messages it receives for broadcasting.

Lemma 4.1. Let PR be a faithful routing protocol in a network $N = (G,r)$, let R be the corresponding binary relation and let $u \overset{e}{\rightarrow} v$ be an (r,t) bottleneck in G. Then

(i) There is an edge f entering u such that $(f,e) \in R$.

(ii) If $t \neq v$, then there is an edge g leaving v such that $(e,g) \in R$.

Proof. We prove only (i), as the proof of (ii) is similar. By the definition of R, if (i) is false then no message received by $PR(u)$ is forwarded via e. Let C be a minimum (s,t) cut containing e. Then if all the edges in C except e fail, no message broadcasted from r will reach t—contradicting the assumption that PR is faithful, and hence is reliable for t. ∎

Let B be the graph in Figure 1. For this graph we can prove the following:

Lemma 4.2. Let PR be a faithful routing protocol in the network $N = (B,r)$. Then a message m received by $PR(c)$ on e_3 or on e_4 must be forwarded by it along e_5.

Proof. It is easily verified that the following holds in B:

1) e_3 is an (r,v) bottleneck.

2) e_4 is an (r,u) bottleneck.

3) e_5 is the only edge leaving c.

The lemma now follows by applying Lemma 4.1 (ii) to e_3 and to e_4. ∎

Corollary 4.3. Whenever no edge fails to deliver a message m broadcasted by a protocol satisfying Lemma 4.2, m is forwarded twice along e_5.

Proof. Since both e_1 and e_2 are (s,c) bottlenecks, every message m broadcasted by r must be sent along both e_1 and e_2. Since e_3 $[e_4]$ is an (s,v) bottleneck $[(s,u)$ bottleneck], Lemma 4.1 (i) implies that $a[b]$ forwards m along $e_3[e_4]$. Thus, m will reach c twice: once along e_3 and once along e_4. Therefore, by Lemma 4.2, m will be forwarded twice along e_5, as claimed.

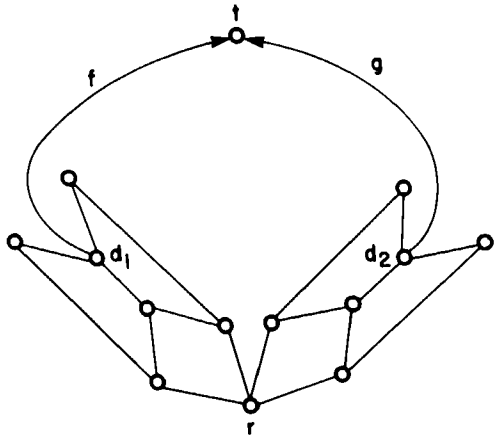

FIG. 2. The graph M.

Note. It is easily observed that Lemma 4.2 and Corollary 4.3 are valid for any faithful protocol PR in (B,r) and for any message m, provided $PR(c)$ cannot decide, upon receiving m on e_4, whether it had already received it on e_3 (or vice versa). We claim that if $PR(c)$ *can* decide for *every* message m received on e_4 whether it had already received it on e_3, then the (bitwise) space complexity of $PR(c)$ is at least linear (in the number of messages):

Suppose that n messages $m_1, \ldots, m_n$ are broadcasted by PR. By the faithfulness of PR, they all must be forwarded via e_3 and e_4. suppose that a certain subset of them is destroyed while sent through e_3, and that all the undestroyed messages arrive c on e_3 before any of them arrive on e_4. Then, under the assumption above, $PR(c)$ must be able to decide for every message m_i whether or not it received it on e_3 ($i = 1, \ldots, n$). The only information $PR(c)$ may use for this task is the content of c's local memory (which includes the *state* of $PR(c)$, i.e.—the value of its program counter).

There are 2^n possible subsets of $\{m_1, \ldots, m_n\}$, and no two distinct subsets can be recorded by the same memory content. Thus, the number of distinct memory contents that may be used by $PR(c)$ during the broadcasting of n messages is at least 2^n, and hence at least one of them must require n bits. We conclude that if PR is a faithful protocol with sub-linear (bitwise) space complexity, than Lemma 4.2 and Corollary 4.3 still hold for certain messages m in certain executions of PR.

Let M be a graph composed of two copies of B having a common root r, a new vertex t and two new edges $d_1 \xrightarrow{f} t$ and $d_2 \xrightarrow{g} t$ (see Fig. 2).

Lemma 4.4. Let PR be a faithful routing protocol in the network $N = (M,r)$. If no edge fails during the broadcasting of a message m in N using PR, then m is forwarded twice along f (g).

Proof. By applying Corollary 4.3 to each copy of B in M, we get that a message broadcasted by r will reach d_1 (d_2) twice. The lemma now follow by Lemma 4.1 (i), the fact that f and g are (s,t) bottlenecks, and the fact that $d_{in}(d_1) = d_{in}(d_2) = 1$. ■

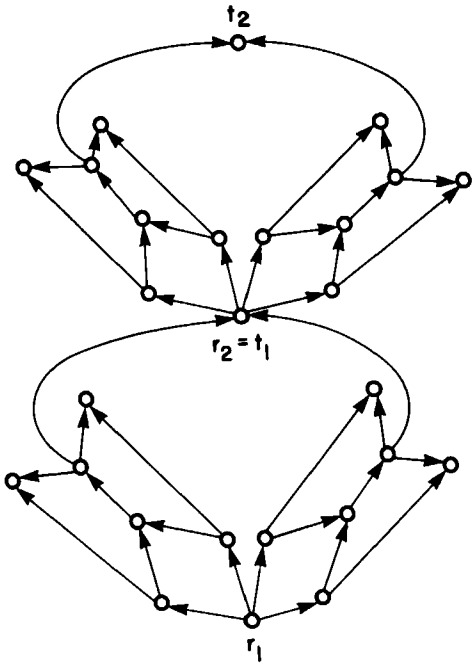

FIG. 3. The graph G_2.

We are now ready to prove the main negative result of this paper.

Theorem 4.5. For each $n > 0$ there exists a network $N_n = (G_n,r)$ with $TR(N_n) = 20n$, such that the message complexity of any faithful routing protocol for N_n is larger than 2^n.

Proof. The network N_n is defined as follows: $N_n = (G_n,r)$, where G_n is composed of n copies of M in the following way: Assume the n copies are $M_1, \ldots, M_n$; for a vertex (edge) x in M, let x_i denote the corresponding vertex (edge) in M_i, and let $r_1 \equiv r$. G_n is constructed by identifying t_{i-1} with r_i, $i = 2, \ldots, n$. G_2 is depicted in Figure 3. It is not hard to verify the following observations on G_n:

1) For $i = 2, \ldots, n$, $\delta_{G_n}(r,r_i) = 2$. Hence, for each vertex x_i in the ith copy of M in G_n, it holds that $\delta_{G_n}(r,x_i) = \delta_M(r,x)$.

2) By 1) and the fact that M is critical for r, G_n is also critical for r. Hence $TR(N_n) = TR((G_n,r)) = |E(G_n)| = |E(M)|n = 20n$.

3) Let PR be a faithful routing protocol on N_n. Then, by applying Lemma 4.1 to each of the edges entering r_i, every message received by $PR(r_i)$ on both edges entering it is sent by $PR(r_i)$ along the four edges leaving it.

4) Under the assumptions of 3) above, Lemma 4.4 implies that every message sent by r_i along the four edges leaving it is received by $t_i = r_{i+1}$ *twice* along each of the edges entering it.

Assuming that no edge fails during the execution of the protocol, and applying 3) and 4) above, an easy induction shows that every message broadcasted by PR is sent 2^i times on each edge entering t_i. The Theorem follows by substituting $i = n$. ∎

Note. The note following Corollary 4.3 indicates that Theorem 4.5 might hold for any faithful protocol PR with sub-linear space complexity (though a formal proof of this could be rather involved).

5. A GRAPH-THEORETIC CHARACTERIZATION

In this section we show that the networks that have message-optimal faithful routing protocols are characterized by a property that can be viewed as a generalization of Edmonds' branching theorem [2].

Definition 5.1. Let $N = (G,r)$ be a broadcast network. An *optimal tree cover* for N is a family $\mathbf{T} = \{T_1, \ldots, T_s\}$ of edge-disjoint directed trees rooted at r satisfying:

(a) Each T_i is a subgraph of G.

(b) For each $v \in V - \{r\}$ it holds that

$$\delta_G(r,v) = |\{i: 1 \leq i \leq s, v \in V(T_i)\}|.$$

Example. Let $N = (G,r)$ be such that for each $v \in V - \{r\}$ it holds that $\delta_G(r,v) = k$. Then by Edmonds' branching theorem G contains k edge-disjoint spanning trees rooted at r, which constitute an optimal tree cover for N.

Theorem 5.1. A network $N = (G,r)$ has a message optimal faithful routing protocol iff it has an optimal tree cover.

Proof. (i) If: Assume that N has an optimal tree cover, and let $\mathbf{T} = \{T_1, \ldots, T_s\}$ be this cover. Let PR be a routing protocol that propagates messages from r along the edges of each of the T_i's in $\mathbf{T}$ simultaneously. We claim that PR is faithful:

Consider a vertex $v \in V - \{r\}$ such that $\delta_G(r,v) = k$, and assume that less than k edges fail to deliver a message broadcasted by PR. Then since there are k edge-disjoint trees rooted at r in $\mathbf{T}$ which contain v, one of these trees contains no edge that failed. Hence, the message propagated by PR along the edges of this tree will eventually reach v. Thus, PR is reliable for all $v \in V - \{r\}$, and hence is faithful.

To see that PR is also message optimal, we must show that its message complexity is bounded by $TR(G,r)$. The message complexity of PR is clearly bounded by the number of edges in $\cup_{i=1}^{i=s} T_i$, since PR send each message along each edge at most once. Thus we have that the message complexity of PR is bounded by

$$\begin{aligned}
\sum_{i=1}^{s} |E(T_i)| &= \sum_{i=1}^{s} |V(T_i) - \{r\}| \qquad \text{[since } T_i \text{ is a tree]}, \\
&= \sum_{i=1}^{s} |\{v: v \in V(T_i) - \{r\}\}| \\
&= \sum_{v \in V-\{r\}} |\{i: v \in T_i\}| \qquad \text{[changing order of summation]} \\
&= \sum_{v \in V-\{r\}} \delta_G(r,v) \qquad \text{[by definition of optimal tree cover]} \\
&= TR(G,r),
\end{aligned}$$

as claimed.

(ii) Only if: Assume that N has a message optimal faithful routing protocol PR. We define an optimal tree cover for N associated with PR as follows:

Let R be the binary relation identified with PR, defined in Section 5: namely, R is the set of all pairs (e,f) where e is an edge entering some vertex v, f is an edge leaving v, and $PR(v)$ sends along f every message it receives on e. Let $\{e_1, \ldots, e_s\}$ be the edges leaving r; note that each e_i is a bottleneck, and hence must be used by PR. We use the relation R to construct an optimal tree cover $\mathbf{T} = \{T_1, \ldots, T_s\}$ as described below:

For $i = 1, \ldots, s$ let $E_i \subseteq E$ be the minimal set satisfying:

(i) $e_i \in E_i$.

(ii) if $e \in E_i$ and $(e,f) \in R$, then $f \in E_i$.

Let $T_i = G(E_i)$. We claim that the family $\mathbf{T} = \{T_1, \ldots, T_s\}$ is an optimal tree cover for N. We must show that the T_i's are edge-disjoint trees rooted at r that satisfy the condition that for each $v \in V - \{r\}$ it holds that

$$\delta_G(r,v) = |\{i: 1 \leq i \leq s, v \in V(T_i)\}|. \qquad (*)$$

The following definition is used in the proof.

Definition 5.2. For $1 \leq i \leq s$, and for $e \in E_i$, $D_i(e)$ is an integer defined by: $D_i(e_i) = 1$, and for $e \neq e_i$, $D_i(e)$ is the minimal integer k such that there is an edge e' with $D_i(e') = k - 1$ and $(e',e) \in R$.
Similarly, for $v \in V(T_i)$, $D_i(v)$ is the minimal k for which there is an edge $u \overset{e}{\to} v$ in E_i with $D_i(e) = k$ (intuitively, $D_i(v)$ is the length of the shortest (r,v) path in T_i).

The proof proceeds now through a sequence of claims.

(1) T_i contains an (r,v) path for each $v \in V(T_i)$. The proof of this claim is an easy induction on $D_i(v)$.

(2) Let $E' = \cup_{i=1}^{s} E_i$. Then, by definition, E' is the the set of all edges that deliver messages broadcasted by PR.

Since PR is faithful we have, by Lemma 2.1 that $|E'| > TR(G,r)$, and in fact that for each $v \in V - \{r\}$ the number of edges in E' entering v is at least $\delta_G(r,v)$. Since the message complexity of PR is $TR(G,r)$ we have that $|E'| \leq TR(G,r)$. Putting the last two facts together we get that:

(2a) $|E'| = TR(G,r)$, and hence

(2b) for each $v \in V - \{r\}$, the number of edges in E' entering v is $\delta_G(r,v)$, and E' contains no edges entering r.

(2c) Each message broadcasted by PR while no edge fails is sent exactly once along each edge in E'.

(3) If $e \neq \bar{e}$, and (e,f), $(\bar{e},\bar{f})$ are in R, then $f \neq \bar{f}$: Otherwise, e and $\bar{e}$ must enter the same vertex, say v, and each message broadcasted by PR when no edge fails will be received by v along both e and $\bar{e}$; hence, this message will be sent twice along $f = \bar{f}$, which contradicts (2c) above.

(4) For $1 \leq i < j \leq s$, $E_i \cap E_j = \Phi$: If this is false, then there is an edge $e_0 \in E_i \cap E_j$ for which $k = D_i(e_0)$ is minimized. First note that $e_0 \notin \{e_i,e_j\}$, since if, for example, $e_i \in E_j$ then PR may send the same message twice along e_i, in contradiction to (2c). This implies that $k > 1$ and that there are $e \in E_i$ and $\bar{e} \in E_j$ such that $D_i(e)$

$= k - 1$, and both (e,e_0) and $(\bar{e},e_0)$ are in R. By the minimality of k we have that $e \neq \bar{e}$, and hence, by (3) above, $e_0 \neq e_0$, a contradiction.

(5) By (4) above, if an edge e_i fails to deliver a message broadcasted by PR, this message will not be sent along any edge in E_i $(1 \leq i \leq s)$.

(6) For each $v \in V - \{r\}$, the number of distinct T_i's that contain v is at least $\delta_G(r,v)$: Assume, for contradiction, that $\delta_G(r,v) = k$ and that v occurs in only $k - l$ T_i's, say $T_1, \ldots, T_{k-l}$ $(l \geq 1)$. If the edges $e_1, \ldots, e_{k-l}$ fail to deliver a message broadcasted from r, then by (5) above this message will not be sent along any of the edges in $\cup_{i=1}^{k-l} E_i$, and hence will never reach v—contradicting the assumption that PR is reliable for v.

We now conclude the proof of the theorem: By (2b) we have that for each $v \in V - \{r\}$ the number of edges in E' entering v is $\delta_G(r,v)$. By (6) the number of distinct T_i's that contain v is *at least* $\delta_G(r,v)$. It follows that

(a) the number of distinct T_i's that contain v equals $\delta_G(r,v)$, and

(b) for each $v \in V(T_i) - \{r\}$, E_i contains exactly one edge entering v, and E_i contains no edge entering r. (a) implies that (*) hold. (b) and the fact that, by (1), T_i contains an (r,v) path for each $v \in V(T_i)$, implies that each T_i is a tree rooted at r (see [3]). By (4), the T_i's are edge-disjoint. This completes the proof of the theorem. ■

An interesting corollary to Theorem 4.5 is that there are broadcast-networks which do not have an optimal tree cover. In fact, the network (B,r) in Figure 1 is the smallest example of such a network. It has been recently shown by Martin Tompa that deciding whether a given network has an optimal tree cover is NP-complete [20].

6. CONCLUSIONS AND FURTHER RESEARCH

In this paper we have studied broadcast protocols in an unreliable environment. These protocols achieve, in a sense, the maximum possible reliability for each individual vertex in the network. It was shown that there is a trade-off between the communication cost and the computational overhead of such protocols. In particular, we have shown that in certain networks, decreasing the space complexity of the protocol by at most linear factor may increase its message complexity by exponential factor.

A possible way to overcome this difficulty is by allowing the protocol to use the messages it broadcasts to deliver information that can be used to improve its performance, (e.g., by appending to each message information concerning the edges/vertices that had already passed it). A possible drawback of such a technique is, beside the increase of the messages' length, that in the case that faulty edges deliver erroneous messages, such errors may affect the behavior of the protocol in a way that is hard to predict.

As mentioned in the introduction, broadcast protocols are the simplest possible protocols that require the cooperation of the full network. It is interesting whether similar results can be obtained for more complex protocols, like protocols for choosing a leader and reaching a consensus [1,4].

It is also interesting whether similar results to those obtained in this paper can be proved for networks with bidirectional edges. The results in [10,22] could be useful for this kind of research.

References

[1] D. Dolev, M. Klawe, and M. Rodeh, An $O(n \log n)$ unidirectional distributed algorithm for extrema finding in a circle. *J. of Algorithms* **3** (1982) 245–260.

[2] J. Edmonds, *Edge Disjoint Branchings, in Combinatorial Algorithms*. Algorithmic Press Inc. (1973) 91–96.

[3] S. Even, *Graph Algorithms*. Computer Science Press (1979).

[4] M. J. Fischer, N. A. Lynch, and M. S. Paterson, Impossibility of distributed consensus with one faulty process. *Proc. ACM Sym. Principles of Database Systems* (1983) 1–7.

[5] G. Fredrickson and N. Lynch, The impact of synchronous communication on the problem of electing a leader in a ring. 16th Ann. ACM Symp. on Theory of Computing, Washington D.C. (1984) 493–503.

[6] E. Gafni and Y. Afek, Election and traversal in unidirectional networks. 3rd Ann. ACM Symp. on Principles of Distributed Computing, Vancouver, B.C., Canada, August (1984) 190–198.

[7] E. Gafni and W. Korfhage, Distributed election in unidirectional Eulerian networks. Proceedings Twenty-Second Annual Allerton Conference on Communication, Control, and Computing, Allerton, IL, October 3–5 (1984).

[8] R. G. Gallager, P. M. Humblet, and P. M. Spira, A distributed algorithm for minimum-weight spanning trees. *ACM Transact. Programming Languages Syst.* **5** (1983).

[9] D. S. Hirshberg and J. B. Sinclair, Decentralized extrema-finding in circular configurations of processes. *Commun. ACM* **23** (1980) 627–628.

[10] A. Itai and M. Rodeh, The multi-tree approach to reliability in distributed networks. 25th Symposium on the Foundations of Computer Science (Oct. 1984).

[11] E. Korach, S. Kutten, and S. Moran, A modular technique for the design of efficient leader finding algorithms. Proceedings of the 4th symposium on Principles of Distributed Computing, Minaki, Canada, August (1985).

[12] E. Korach, S. Moran, and S. Zaks, Tight lower and upper bounds for some distributed algorithms for a complete network of processors. 3rd Ann. ACM Symp. on Principles of Distributed Computing, Vancouver, B.C., Canada, August (1984) 199–207.

[13] L. Lovász, On two minimax theorems in graph theory. *J. Combinatorial Theory* (Ser. B) **21**(2) (1976) 96–103.

[14] K. Menger, Zur Allgemeinen Kurventheorie. *Fund. Math.* **10** (1927) 96–115.

[15] S. Moran, An improvement of an algorithm for construction of edge disjoint branching. TR 341, Dept. of Computer Science, the Technion (1984).

[16] J. Pachl, E. Korach, and D. Rotem, Lower bounds for distributed maximum-finding algorithms. *J. ACM* **31(4)** (1984) 905–918.

[17] G. L. Peterson, An $O(n \log n)$ unidirectional algorithm for the circular extrema problem. *Trans. Programming Languages Syst.* **4** (1982) 758–762.

[18] Y. Shiloach, Edge disjoint branchings in directed multigraphs. *Inf. Proc. Lett.* **8** (1979) 24–27.

[19] R. E. Tarjan, A good algorithm for edge disjoint branchings. *Inf. Proc. Lett.* **3** (1975) 51–53.

[20] M. Tompa, private communication (1986).

[21] P. Tong and E. L. Lawler, A faster algorithm for finding edge-disjoint branchings. *Inf. Proc. Lett.* **17** (1983) 73–76.

[22] A. Zehavi and A. Itai, Three tree connectivity TR 406 CS dept Technion (March 1986).

Received April, 1986
Accepted February, 1988