LOWER BOUNDS FOR DISTRIBUTED ELECTION ALGORITHMS IN CIRCULAR CONFIGURATIONS OF PROCESSES

J. Pachl*, E. Korach+ and D. Rotem*

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

Research Report CS-81-33

November 1981

*Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada

+Department of Combinatorics and Optimization
University of Waterloo
Waterloo, Ontario, Canada

LOWER BOUNDS FOR DISTRIBUTED ELECTION ALGORITHMS IN CIRCULAR CONFIGURATIONS OF PROCESSES

J. Pachl, E. Korach and D. Rotem

The election problem is the problem of selecting a single process ("leader") in a homogeneous configuration of asynchronous processes with no central controller. An election algorithm is one that solves the problem. We estimate the number of messages required by election algorithms in message-based unidirectional rings of processes. We give a simple combinatorial formulation of the problem, and show that $n \ln n + O(n)$ is a lower bound for the average number of messages sent in rings of size n. The bound is optimal, because it matches a previously known upper bound.

1. Overview.

As the feasibility of building large networks of autonomous processors increases, so does the interest in decentralized algorithms suited for large configurations of concurrent processes. An elemental problem, whose solution is likely to be used as a building block in more complex distributed algorithms, is that of choosing a single node in a network. Since the choice must be made by the nodes themselves, not by any central authority, we refer to the problem as _the election problem_, and call its solution _an election algorithm_.

The election problem is to be solved by executing an identical algorithm in each participating node. Of course, if all the nodes are exactly identical then one cannot guarantee absence of deadlock; thus we assume that the nodes differ in one parameter: Each node is assigned a unique label, of which it alone is aware when the algorithm starts executing.

Gallager [4] proposed a solution to the election problem in a general configuration of processes. Le Lann [9] investigated the important special case of a circular configuration. His solution was modified and improved in several published papers [2] [6] as well as in several still sitting on referees' desks. Thus today we know several election algorithms for both unidirectional and bidirectional rings, and also their efficiency as measured by the execution time and the total number of messages.

The only paper known to us which is concerned with lower bounds is [1], where Burns shows that the worst case number of messages needed by any election algorithm in bidirectional rings of size n is at least $(n \log_2(n/2))/8$, and at least $(n \log_2 n)/4$ when n is a power of 2. Since obviously every bidirectional ring can be used as a unidirectional one, the bound is also valid for unidirectional rings.

In this paper we concentrate on unidirectional rings, which are apparently easier to understand, and show that the average number of messages needed by any election algorithm in unidirectional rings of size n is at least

$$n \ln n + O(n) \cong 0.693\, n \log_2 n + O(n).$$

The bound is the best possible (modulo O(n)), because it is actually achieved by the Chang and Roberts algorithm [2]. This raises an interesting question concerning the difference between unidirectional and bidirectional algorithms. Apparently, no bidirectional algorithm with average behaviour better than $n \ln n + O(n)$ is known. The question whether the capability of sending messages in both directions can be effectively utilized to decrease the (average or worst case) number of messages remains open.

The paper is organized as follows: In section 2 we describe the election problem in some detail. In section 3 we enumerate all election algorithms by means of certain sequences of labels; that will enable us to use a simple counting argument to prove our main result in section 4. In the last section we speculate how our method could be used to derive sharp lower bounds on the worst case number of messages.

2. The election problem.

In our main problem, a number of identical nodes (=processes) are connected by one-way (simplex) communication channels to form a circular configuration so that each node transmits (sends) into exactly one channel, and receives from exactly one channel. We refer to such configuration as a unidirectional ring. The nodes communicate only by messages sent along the channels. Messages can be arbitrarily long, but sending and receiving in each node are atomic actions which cannot overlap in time.

There is no central node. All nodes execute the same algorithm; they differ from one another only in that each has a unique (integer) label. Initially each node knows only its own label; it knows neither the values of other nodes' labels nor the number of nodes in the ring. We say that the ring with n nodes is labeled $s = (s_1 \cdots s_n)$ if the output port of the node labeled s_i is connected by a communication channel to the input port of the node with label s_{i+1} for $i = 1,\ldots,n-1$, and to the input of the node with label s_1 for $i = n$. Thus if a ring is labeled by s, it is also labeled by any cyclic permutation of s.

The execution is asynchronous, in the sense that the nodes can only synchronize explicitly through messages (not implicitly by a real-time clock). However, messages are delivered with arbitrary (finite) delays. Thus while every transmitted message is eventually received, there is no a priori bound on its transmission delay, and the delays are mutually independent. In particular, there is no bound on the number of messages that can be simultaneously in transit on any one channel. For the sake of simplicity, and with no loss of generality, we assume that all nodes start execution simultaneously.

The <u>election problem</u> is to choose a single node in the ring, the so called <u>leader</u> of the network. An <u>election algorithm</u> is an algorithm which, when executed in every node in the ring, will always (ie. for every ring size, for every label assignment and for every choice of transmission delays) result in electing a unique leader. We consider only <u>deterministic</u> algorithms; that is, the behaviour of each node is uniquely determined by the node's label and by the messages received in the node. We shall use a slightly nonstandard termination criterion: the election terminates when the leader has been notified that every node in the ring knows the leader's label. This has the technically useful consequence that there must be a chain of consecutive messages spanning the full circle (ending in the elected node), as we will see later. Other termination criteria have been used, but they are all essentially equivalent as far as the total number of messages is concerned. They differ by at most 2n messages (in a ring of size n), which is a low order term in our results.

At this point we wish to avoid the potentially most obscure aspect of the election problem, the indeterminism caused by variable transmission delays. We adopt two additional assumptions, thereby making execution in each ring deterministic. The assumptions will allow us to proceed directly to the mathematical heart of the matter. In Appendix we return to the question of indeterminism, and show that the assumptions (2.1) and (2.2) do not limit the generality of our results.

Thus we assume, in the rest of the paper (except Appendix), that

(2.1) In each channel, the messages are received in the same order as sent.

(2.2) The nodes can receive messages only by executing the blocking receive command ([10], p.481).

The blocking receive command means "wait until the next message arrives at the input port" (if the next message has already arrived then the waiting time is zero). Thus a node can only receive a message if it commits itself to suspend all its activity until a message arrives. An important consequence of (2.1) and (2.2) is that the execution of every election algorithm in every labeled ring is deterministic (ie. independent of the transmission delays). Indeed, in the unidirectional ring every node receives all messages from a single source, (2.1) guarantees that they are received in a deterministic (delay independent) order, and the blocking receive primitive (along with the absence of a real-time clock) ensures that only the order (not actual arrival times) of received messages matters.

3. Full information algorithms.

In order to be able to find lower bounds (over all election algorithms), we want to enumerate all election algorithms in a simple manner which would allow us to estimate the number of messages used by any algorithm in any ring configuration. To this end, we define a class of algorithms (which we call the full information algorithms), and show that the algorithms in the class are as good as all other election algorithms.

The idea of our simplification is this: To specify an algorithm, one has to say *when* to send (or, at a particular moment in time, *whether* to send) and *what* to send. We eliminate the latter decision by considering the algorithms in which every node sends everything it knows (hence the name "full information" algorithms); these algorithms only have to specify whether to send or not.

A _full information algorithm_ is an election algorithm in which the content of each message is a finite sequence of labels such that

(1) the content of a message is a sequence (s_1) of length 1 iff the message sender has label s_1 and it has previously received no message, and

(2) the content of a message is a sequence $(s_1\ s_2\ \cdots\ s_k)$ of length $k > 1$ iff the message sender has label s_k and the content of the last message previously received in the node is $(s_1\ s_2\ \cdots\ s_{k-1})$.

This means that every message contains all the information about ring labels that is potentially available to the sending node at the time when the message is sent. (The last received message always contains more information than the previous ones, in view of (2.1).)

Our first result says that, at least as far as the number of messages is concerned, every election algorithm can be replaced by a full information algorithm.

3.1. Theorem. For every election algorithm A' there is a full information algorithm A which uses the same number of messages as A' on any ring.

Proof is straightforward: The content of every message sent by a node executing algorithm A' is a function of the local label and of the content of the messages received previously in the node. Hence the content of every message is a function of a sequence of labels. Instead of sending the value (result) of this function (as A' does), the algorithm A sends its arguments; the message recipient then computes the function value every time it is needed. Thus A is constructed from A' by (i) replacing every "send" operation by send $(s_1 \cdots s_k\ s_{k+1})$ where $(s_1 \cdots s_k)$ is the content of the last previously received message and s_{k+1} is the local label, and (ii) replacing every "receive" operation by a subroutine which receives a message of the form $(s_1 \cdots s_k)$ and computes the value which the corresponding message in the execution of A' would contain.

[]

For full information algorithms, we shall estimate the number of messages by counting certain sequences of labels. The concatenation of two sequences s and t of integers is denoted st. When s and t are two sequences, we say that t is a *subsequence of* s if s = rtu for some sequences r and u; we say that t is a *prefix of* s if s = tu for some u. Two sequences of integers are said to be *disjoint* if no integer belongs to both. When s is a sequence, we denote by len(s) its length, and by C(s) the set of cyclic permutations of s. Clearly $|C(s)| = \text{len}(s)$.

Let Z be the set of integers. We denote by D the set of all finite nonempty sequences of <u>distinct</u> integers:

$$D = \{ (s_1 \cdots s_k) \mid k \geq 1,\ s_i \in Z \text{ for } 1 \leq i \leq k, \text{ and } s_i \neq s_j \text{ for } i \neq j \}.$$

For $s \in D$ and $E \subseteq D$, we denote by $N(s,E)$ the number of subsequences of s which belong to E . A subset E of D is called <u>exhaustive</u> if it has these two properties:

<u>Prefix property</u>: if $tu \in E$ and $\text{len}(t) \geq 1$ then $t \in E$.

<u>Cyclic permutation property</u>: if $s \in D$ then $C(s) \cap E \neq 0$. Note that, in view of the latter property, every sequence of length 1 is in E .

Since the assumptions (2.1) and (2.2) make the execution of every election algorithm on every unidirectional ring deterministic, we get the following useful fact:

<u>3.2. Lemma</u>. Let t be a subsequence of $s \in D$. A message with content t is transmitted when a full information algorithm A executes on a ring labeled t if and only if a message with content t is transmitted when A is executed on a ring labeled s.

[]

In the next section we shall use the following theorem to get a bound on the average number of messages.

<u>3.3. Theorem</u>. For every election algorithm A there exists an exhaustive set $E \subseteq D$ such that A requires at least $N(s,E)$

messages when executed on a ring labeled s.

Proof. In view of 3.1 we can assume that A is a full information algorithm. Define E as follows:

$E = \{ t \mid t \in D$ and a message with content t will be transmitted when A executes on a ring labeled t $\}$.

First we show that E is exhaustive. The prefix property follows from Lemma 3.2 and from the definition of a full information algorithm. To prove the cyclic permutation property, take any sequence $s = (s_1 \ldots s_k) \in D$. Assume, without loss of generality, that the algorithm elects the node labeled s_1 to be the leader of the ring labeled s. By our termination criterion, the election is not over until the node labeled s_1 has been notified that all the other nodes know s_1 to be the leader's label. Hence at least one message with content ts, where $\text{len}(t) \geq 0$, must be sent (by the node labeled s_k) before the election is over. Obviously $\text{len}(ts) \geq k$. The prefix of ts of length k is a cyclic permutation of s which belongs to E, hence $C(s) \cap E \neq 0$.

To show that at least N(s,E) messages will be sent in the ring labeled s, it is enough to prove that for each subsequence t of s such that $t \in E$, at least one message with content t will be sent. The latter claim follows from the definition of E and from Lemma 3.2.

[]

It can be shown that, conversely, every effectively computable exhaustive subset E of D corresponds in this way to an election algorithm. This fact will not be needed in the sequel, and will not be proved here.

We illustrate Theorem 3.3 with an example:

Example (Chang and Roberts [2]). In each node execute this program:

```
integer max, m

begin
  max := local.label
  Send( local.label )
  Receive( m )

  while  m ≠ max  do
    if  m > max  then
      max := m
      Send( m )
    end if

    Receive( m )
  end while

  if  max = local.label  then
    Send( local.label )
    Receive( m )
  end if

  stop   { leader's label is max }
end
```

In this algorithm, the content of every message is a single integer. Receive(m) is a blocking receive command; it stands for "wait until next message arrives, and then copy its content into the variable m". The algorithm elects the node with the largest label to be the leader. The corresponding exhaustive set is

$$E = \{ (s_1\ s_2\ \cdots\ s_k) \mid s_1 = \max_{1 \le j \le k} s_j \}$$

It should be observed that the description of E is, at least in this case, considerably simpler than the programming language description of the algorithm. We intend to pursue the idea of describing distributed algorithms by the information content of their messages in more detail elsewhere.

4. Lower bound for the average number of messages.

We define A(I) as

$$A(I) = \min_{E} \frac{1}{n!} \sum_{s \in P(I)} N(s,E).$$

where E is an exhaustive set, and P(I) is the set of all permutations of I.

It follows from Theorem 3.3 that an election algorithm with a corresponding exhaustive set E will send at least $\frac{1}{n!} \sum_{s \in P(I)} N(s,E)$ messages on the average on rings labelled by permutations in P(I). Hence A(I) is a lower bound on the average number of messages sent by any election algorithm on rings labelled by P(I). Furthermore, as shown in the next lemma, A(I) depends only on the cardinality of the set I.

4.1. Lemma. If $|I_1| = |I_2|$ then $A(I_1) = A(I_2)$.

Proof: Since $|I_1| = |I_2|$ there is a 1-1 mapping $f: Z \to Z$ such that $f(I_1) = I_2$. It is easy to verify directly from the definition that a set $E \subseteq D$ is exhaustive if and only if

$f(E)$ is exhaustive. The lemma follows from the fact that for $s \in P(I_1)$ we have $f(s) \in P(I_2)$ and $N(s,E) = N(f(s), f(E))$.

[]

Lemma 4.1 allows us to define $A(n) = A(I)$ when $n = |I|$. By the previous arguments in this section it follows that $A(n)$ is a lower bound for the average number of messages sent by any election algorithm in unidirectional rings of size n.

4.2. Theorem. For $n \geq 1$ we have

$$A(n) \geq (n+1)H_n - n \qquad \text{where } H_n = \sum_{i=1}^{n} \frac{1}{i}.$$

Proof: For a fixed n we consider the set $I = \{1,2,\dots,n\}$. For a given exhaustive set E, $s \in P(I)$ and $1 \leq k \leq n$ let

$$s(k)=\{t \mid t \text{ is a subsequence of } s \text{ with len}(t)=k \text{ and } t \quad E\}.$$

It follows from this definition that

$$N(s,E) = \sum_{k=1}^{n} |s(k)| \text{ and } \sum_{s \in P(I)} N(s,E) = \sum_{k=1}^{n} \sum_{s \in P(I)} |s(k)|$$

There is a total of $(n+1-k)$ subsequences t of length k in each permutation $s \in P(I)$. By the cyclic permutation property of E, it follows that at least one out of the k cyclic permutations of t must be in E, hence

$$\sum_{s \in P(I)} |s(k)| \geq \frac{n!\,(n+1-k)}{k}$$

Therefore for every exhaustive set E we have

$$\frac{1}{n!} \sum_{s \in P(I)} N(s,E) \geq \sum_{k=1}^{n} \frac{(n+1-k)}{k} = (n+1)H_n - n.$$

The theorem follows by the definition of A(n).

[]

By the well known approximation to H_n,

$$H_n = \ln n + \gamma + 0(n^{-1})$$

where $\gamma \cong 0.577$, we have

$$A(n) \geq n \ln n + 0(n) \cong .693\, n \log_2 n + 0(n).$$

The Chang and Roberts algorithm (described in section 3) requires $n \ln n + 0(n)$ messages on average (see [2], p. 282), which shows that $A(n) \leq n \ln n + 0(n)$. We conclude that

$$A(n) = n \ln n + 0(n).$$

5. A recursive formula for the worst case number of messages.

Obviously, the average case bound in the previous section is also a lower bound for the worst case number of messages. In this section we suggest a method that could possibly lead to a sharper worst case bound. The method also further illustrates the calculus of exhaustive sets. Since we have not been able to actually find a sharper bound, we only present a brief outline and omit the proofs of our formulas.

For each finite set I of integers, define

$$W(I) = \min_{E} \max_{s \in P(I)} N(s,E)$$

where the minimum is taken over all exhaustive sets $E \subseteq D$.

As in Lemma 4.1, one can show that W(I) depends only on the cardinality of I, and we can write $W(n) = W(I)$ when $n = |I|$. From Theorem 3.3 it follows that W(n) is a lower bound on the worst case number of messages sent by any election algorithm in unidirectional rings of size n.

In analogy with our result in section 4, we would like to find a constant K such that

$$W(n) = K\, n \log_2 n + O(n)$$

The algorithm constructed recently by Dolev et al [3] proves that

$$W(n) \leq 2\, n \log_2 n + O(n)$$

Since clearly $W(n) \geq A(n)$, from 4.2 we get

$$W(n) \geq n \ln n + O(n). \qquad (5.1)$$

By combining k sets of cardinality n we are able to prove the recursive inequality

$$W(kn) \geq k\, W(n) + n\, W(k) - kn. \qquad (5.2)$$

Since W(1) = 1, from (5.2) we get, for any fixed $k \geq 1$,

$$W(n) \geq \frac{(W(k)-k)\ n \log_2 n}{k \log_2 k} + 0(n) \qquad (5.3)$$

Formula (5.3) has an interesting consequence: We can obtain a bound of the form

$$W(n) \geq K\ n \log_2 n + 0(n)$$

by estimating W(n) for a <u>single</u> value of n. For example, one can show that W(2)=3, which together with (5.3) gives

$$W(n) \geq \frac{1}{2}\ n \log_2 n + 0(n)$$

We can also prove that $W(k) \geq 3k-4$ for every k. This along with (5.3) yields

$$W(n) \geq \frac{(2k-4)\ n \log_2 n}{k \log_2 k} + 0(n) \qquad (5.4)$$

From (5.4) we get the best result when k=5, namely

$$W(n) \geq \frac{6}{5 \log_2 5}\ n \log_2 n + 0(n) \qquad (5.5)$$

However, $\frac{6}{5 \log_2 5} = 0.51\ldots$ is well below $\ln 2 = 0.69\ldots$ and thus (5.5) does not improve upon (5.1).

<u>Appendix</u>: Eliminating indeterminism.

The assumptions (2.1) and (2.2) in section 2 allowed us to ignore the indeterminism introduced to the system by indeterminate transmission delays. In this appendix we will argue that our results stay true even when the assumptions are relaxed.

Consider a unidirectional ring in which messages can arrive out of order (contrary to (2.1)) and processes can test their input ports (or rather the associated buffers) for emptiness (contrary to (2.2)). Theorem 3.1 and its proof remain valid in this more general setting.

For a given election algorithm, the total number of messages sent in the ring now depends on two factors: label assignment and the choice of transmission delays. There does not seem to be any canonical probability distribution on delay choices; we define the average number of messages as in [2] and [7]: Given a set I of labels, for each permutation of I take the worst possible choice of delays (ie. the one resulting in the largest number of messages) and then average over all permutations. The "worst possible choice of delays" appears to be fairly typical, and therefore the average number of messages defined in this way is a realistic performace measure.

Now we can get a lower bound on the average number of messages in rings of size n, as follows: For each election algorithm and for each labeled ring of size n <u>select</u> one way to execute the algorithm on the ring, count the number of messages (in the selected execution history), and then average over all label permutations. Suppose that we select one execution history for each election algorithm and each ring, and that we make the selections for distinct rings mutually consistent, so that 3.2 is true. We can then proceed as in section 3 to prove a modified version of Theorem 3.3: For every election algorithm there exists an exhaustive subset E of D such that, for a certain choice of delays, the algorithm requires at least N(s,E) messages when executed on a ring labeled s. It follows that, with the average number of messages defined as above, the results in sections 4 and 5 remain true.

The mutual consistency of the selected execution histories for different rings can be achieved by constructing the selected histories recursively. First we define the selections on linear segments, starting with segments of length 1 and proceeding recursively to longer ones. When the selections are made for all segments, we select histories for rings (to agree with those on segments).

Although we have described the construction informally, it can be easily formalized: Following Greif [5] and Lamport [8], we take an <u>execution history</u> to be a partially ordered set of <u>events</u>, where each event is either the transmission or the reception of a message. Each event has its <u>location</u> (the node where it is executed) and a distributed algorithm is specified (locally) by defining all possible sequences of events in a single location (this is, essentially, a denotational version of Burns' operational model). It is straightforward to define which execution histories are allowed by a given algorithm, and which pairs of execution histories are consistent. The recursive selection construction can then be carried out formally.

References.

[1] J. E. Burns, A formal model for message passing systems, Tech. Report No. 91 (September 1980), Indiana University Computer Science Department.

[2] E. Chang and R. Roberts, An improved algorithm for decentralized extrema-finding in circular configurations of processes, Comm. A. C. M. 22 (1979), 281-283.

[3] D. Dolev, M. Klawe and M. Rodeh, a paper in preparation, (private communication May 1981).

[4] R. G. Gallager, Finding a leader in a network with O(E)+O(NlogN) messages, M. I. T. Internal Memorandum.

[5] I. Greif, Semantics of communicating parallel processes, MAC TR-154, M. I. T. (1975).

[6] D. S. Hirschberg and J. B. Sinclair, Decentralized extrema-finding in circular configurations of processors, Comm. A. C. M. 23 (1980), 627-628.

[7] E. Korach, D. Rotem and N. Santoro, A probabilistic algorithm for decentralized extrema-finding in a circular configuration of processors, Univ. of Waterloo Res. Report CS-81-19 (May 1981).

[8] L. Lamport, Time, clocks, and the ordering of events in a distributed system, Comm. A. C. M. 21 (1978), 558-565.

[9] G. Le Lann, Distributed systems -- towards a formal approach, Information Processing 77 (ed. B. Gilchrist), pp. 155-160, North-Holland 1977.

[10] A.S. Tanenbaum, Computer Networks, Prentice-Hall 1981.