Lower Bounds for Distributed Maximum-Finding Algorithms

J. PACHL, E. KORACH, AND D. ROTEM

University of Waterloo, Waterloo, Ontario, Canada

Abstract. This paper establishes several lower bounds of the form $\Omega(n \log n)$ for the number of messages needed to find the maximum label in a circular configuration of n labeled processes with no central controller.

Categories and Subject Descriptors: D.4.1 [**Operating Systems**]: Process Management—*synchronization*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*sequencing and scheduling*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Distributed algorithms, message complexity, communication rings, election algorithms, lower bounds

1. *Introduction*

As the feasibility of building large networks of autonomous processors increases, so does the interest in distributed algorithms suited for large configurations of parallel processes. An elemental problem, whose solution is likely to be used as a building block in more complex algorithms, is that of finding the maximum of a distributed set of integers.

Several recent papers [1–4, 6, 7, 9–12] investigate the following version of the problem. A number of asynchronous processes are connected by communication channels to form a ring in which each process can send messages to its immediate neighbor in the clockwise direction (the configuration is called a *unidirectional ring*). Every process has a unique (integer) label, which is initially known only to the process itself; no process knows the size of the ring. The configuration has no central controller and no real-time clock, and the processes can communicate only by messages sent through the communication channels. The messages are subjected to variable and independent delays, but every message is eventually delivered. With these assumptions, the aforementioned papers propose distributed algorithms to find the maximum label in the ring.

A preliminary version of this paper was presented at the 14th Annual ACM Symposium on Theory of Computing. (Pachl, J., Korach, E., and Rotem, D. A technique for proving lower bounds for distributed maximum-finding algorithms. In *Proceedings of the 14th Annual ACM Symposium on the Theory of Computing* (San Francisco, May 5–7), ACM, New York, 1982, pp. 378–382.)

Authors' present addresses: J. Pachl and D. Rotem, Department of Computer Science, University of Waterloo, Waterloo, Ontario, N2L 3G1, Canada; E. Korach, IBM Scientific Research Center, Haifa, Israel

The performance measure used to evaluate the algorithms is the total number of messages sent when all processes begin execution simultaneously. In this paper we prove a lower bound for the number of messages sent by any maximum-finding algorithm in unidirectional rings, and also lower bounds for the number of messages when the processes can send messages in both directions along the ring (the so-called *bidirectional ring*) or when the ring size is known in advance. All our bounds are of the form $\Omega(n \log n)$, where n is the size of the ring. The first lower bound of this form is due to Burns [1]; in Section 6 we explain how our results relate to his.

We estimate the worst case and the average number of messages. Both measures are computed in the papers previously mentioned, as well as in [5], for particular algorithms. Since we are concerned with lower bounds over all algorithms, we have to define the two quantities more precisely. The following definition agrees with the previous usage.

When a maximum-finding algorithm is executed in a ring of labeled processes, the total number of transmitted messages depends on two factors: communication delays and the assignment of labels. We say that the ring *is labeled by the sequence* $(s_1\ s_2 \cdots s_n)$ if there are n processes in the ring, their labels are $s_1, s_2, \ldots, s_n$, and the communication channels connect s_n with s_1, s_1 with $s_2, \ldots$, and s_{n-1} with s_n (if the ring is unidirectional, the channels lead from s_n to s_1, from s_1 to s_2, etc.). We also say that the ring *is labeled by the set* $\{s_1, s_2, \ldots, s_n\}$.

Let $I = \{s_1, s_2, \ldots, s_n\}$ be a set of integers (labels), and let A be a maximum-finding algorithm. To get the *worst case* number of messages sent by A in the rings labeled by I, we first find, for each permutation s of I, the communication delays that result in the largest number of messages being sent in the ring labeled by s; then we take the maximum over all permutations s of I. To get the *average* number of messages sent by A in the rings labeled by I, we again begin by finding the worst communication delays for each permutation s of I; then we average the message counts over all permutations s of I (assuming that all the $n!$ permutations are equally probable). Thus we average over all label assignments, but not over communication delays (there does not seem to be a canonical probability distribution on the delays).

Throughout the paper we assume that (i) all processes in the ring begin execution simultaneously, (ii) the communication channels are first-in, first-out (FIFO) queues, and (iii) maximum-finding algorithms are message driven. The next three paragraphs explain what the assumptions mean and why they do not limit the generality of our results.

We assume that all processes in the ring begin execution simultaneously, because that leads to the largest number of messages. (In general terms, the problem is to guarantee progress of the distributed computation in at least one location without effecting the progress in too many locations; this is easier when the computation begins in fewer locations. The task of *breaking symmetry*, in the terminology of [7], is most difficult when the configuration is completely symmetrical.) Moreover, since this paper is about lower bounds (i.e., minima taken over all algorithms), the assumption can only make our results more general: We estimate from below the minimum over the class of all the algorithms that work correctly when all processes begin simultaneously and, therefore, also from below the minimum over the (smaller) class of all the algorithms that work correctly without the assumption.

We assume that the communication channels are FIFO queues; that is, in each channel the messages are delivered in the same order as sent. Again, since we are interested in lower bounds, the assumption does not limit our results.

The last assumption, that the algorithms are message driven, is perhaps the most contentious of the three. In Section 2 it, together with assumption (ii), will allow us to eliminate nondeterminism. An algorithm is message driven if it processes each message before receiving the next one. On the implementation level, this means that the algorithm can be implemented using the *blocking receive* primitive [13, p. 481]; that is, the process can only receive the next message if it suspends its execution until the message arrives. Thus the process cannot receive conditionally (the conditional receive operation is "receive the next message if there is one, otherwise continue computation"). It should be noted that the maximum-finding algorithms in the papers quoted above are all message driven. Moreover, we wish to argue that neither the worst case nor the average number of messages, as defined above, would decrease if we allowed algorithms that were not message driven.

Our argument is that for every general algorithm A (which can execute the conditional-receive operation) there is a message-driven algorithm B such that, for every execution of B in a labeled ring, there is, in the same ring, an execution of A using at least as many messages as the execution of B. We assume that A sends finitely many messages (for A that sends infinitely many messages, any B will do). The algorithm B mimics A except for the conditional-receive operation, which B cannot use and which it simulates as follows: When A executes conditional-receive and eventually sends a message even if it receives none, B simply omits the operation. When A executes conditional-receive and does not send any message before it receives one, B executes blocking-receive and postpones all computation until after the next message arrives. (We note in passing that the correspondence from A to B is not effectively computable, because the problem of whether A is going to send a message before receiving one is undecidable; see the discussion of quiescent instantaneous descriptions in [1].) Now, for any execution of B, the transmission delays for A can be arranged so that no message is received by the conditional-receive operations in A omitted in B. This concludes our argument; we have shown that assumption (iii) does not limit the generality of our results.

The paper is organized as follows: In Section 2 we describe maximum-finding algorithms in unidirectional rings in terms of sequences of labels. The description leads to a simple combinatorial problem, which is solved in Section 3; the solution gives the exact lower bound for the average number of messages in unidirectional rings. In Sections 4 and 5 we derive recursive inequalities for the worst case and the average number of messages in bidirectional rings and in rings of known size, and solve the inequalities to get lower bounds of the form $\Omega(n \log n)$.

We write log for logarithm to the base 2, and ln for logarithm to the base e; H_n is the nth harmonic number [8, p. 73].

2. *Traces and Exhaustive Sets*

In this section and the next we deal with maximum-finding algorithms in unidirectional rings. In these two sections we use the following *termination criterion* for our algorithms: A maximum-finding algorithm is one that sends finitely many messages and then claims, in *at least one process* in the configuration, the value of the maximum label.

First we formalize the intuitive idea of the information content of a message. When an algorithm executes in a *unidirectional* ring of processes, we associate a (finite) sequence of labels with every message. The sequence is called the *trace* of

the message; it is defined recursively as follows: The trace is a sequence (s_1) of length 1 iff the message sender has label s_1 and has previously received no message. The trace is a sequence $(s_1 \cdots s_k)$ of length $k > 1$ iff the sender has label s_k and the last message previously received in the node has trace $(s_1 \cdots s_{k-1})$.

A message with trace $(s_1 \cdots s_k)$ potentially carries the information that k consecutive nodes in the ring are labeled by $s_1, \cdots, s_k$, with s_k being the sender's label. At the same time, the message can contain no information about the labels outside of the segment labeled by $(s_1 \cdots s_k)$. In this sense the content of the message is an encoded form of the trace.

We describe every algorithm executing in unidirectional rings by the set of the traces of the messages that can be sent by the algorithm. Such a general description omits many implementation details, but it is explicit enough to allow us to count messages.

In the sequel, 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, and that t is a *suffix* of s if $s = ut$ for some u. Two sequences of integers are *disjoint* if no integer belongs to both. When s is a sequence, we denote by $\text{len}(s)$ its length, and by $C(s)$ the set of cyclic permutations of s. Clearly $|C(s)| = \text{len}(s)$ whenever the elements of s are pairwise distinct.

Let Z be the set of integers. We denote by D the set of all finite nonempty sequences of *distinct* 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 $s \in D$, $E \subseteq D$ and $k \geq 1$, we denote by $N(s, E)$ the cardinality of the set

$$\{t \in E \mid t \text{ is a prefix of some } r \in C(s)\},$$

and by $N_k(s, E)$ the cardinality of

$$\{t \in E \mid t \text{ is a prefix of some } r \in C(s) \text{ and } \text{len}(t) = k\}.$$

Plainly $N_k(s, E) = 0$ for $k > \text{len}(s)$. A set $E \subseteq D$ is called *exhaustive* if it has these two properties:

Prefix property. If $tu \in E$ and $\text{len}(t) \geq 1$, then $t \in E$.
Cyclic permutation property. If $s \in D$, then $C(s) \cap E \neq \emptyset$.

Note that, in view of the latter property, every sequence of length 1 is in E.

Theorem 2.2 characterizes the set of the traces of the messages transmitted by a maximum-finding algorithm. More precisely, the theorem deals with those traces that belong to D, that is, with those message chains that do not wrap all the way around the ring. In Theorem 2.2 it is essential that the configuration be a *unidirectional* ring and that the algorithms in question be message driven.

The behavior of a message-driven process depends only on the values and order of incoming messages (not on their arrival times). Moreover, in a unidirectional ring every process receives messages from a single source and, since we assume that communication channels function as FIFO queues, the messages are received in a unique order (independent of communication delays). It follows that, for a given labeled unidirectional ring, all combinations of communication delays result in the same messages being sent. This proves the following preliminary result.

LEMMA 2.1. *Let $s, t, u \in D$ be such that both s and t contain u as a subsequence; let A be a maximum-finding algorithm. If A can be executed in the ring labeled by*

s so that a message with trace u is sent, then every execution of A in the ring labeled by t has a message with trace u.

The following theorem relates maximum-finding algorithms to exhaustive sets of sequences.

THEOREM 2.2. *For every maximum-finding algorithm A, there exists an exhaustive set $E(A) \subseteq D$ such that A transmits at least $N(s, E(A))$ messages when executed in the unidirectional ring labeled by s.*

PROOF. Define $E(A)$ to be the set of those $s \in D$ for which a message with trace s is sent when A is executed in the ring labeled by s.

First we show that $E(A)$ is exhaustive. The prefix property follows from the definition of trace and from Lemma 2.1. To prove the cyclic permutation property, let $s = (s_1 \cdots s_k)$ be any sequence in D and consider the ring labeled by s. At least one process must receive a message whose trace has length at least k (otherwise no process could ascertain the value of the maximum label); the trace has a prefix of length k, and the prefix is a cyclic permutation of s. Hence $E \cap C(s) \neq \emptyset$.

To show that at least $N(s, E(A))$ messages are sent in the ring labeled by s, it is enough to prove that at least one message with trace t is sent whenever $t \in E(A)$ is a prefix of a cyclic permutation of s. But this follows from the definition of $E(A)$ and from Lemma 2.1. □

It can be shown that, conversely, for every effectively computable exhaustive set $E \subseteq D$ there is a maximum-finding algorithm A such that $E = E(A)$ and A transmits exactly $N(s, E(A))$ messages in the ring labeled by s, for each $s \in D$. Since this fact is not needed in the present paper, it is not proved here.

Example 2.3. The set

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

is exhaustive. In fact, it is the exhaustive set corresponding to the Chang and Roberts algorithm [2]. It is proved in [2] that the number of messages sent by the algorithm in rings of size n is nH_n on average and $n(n+1)/2$ in the worst case. □

Example 2.4. For every sequence $s = (s_1\ s_2 \cdots s_k) \in D$ with $\text{len}(s) > 2$ define

$$I(s) = \{i \mid 2 \le i \le k-1,\ s_i > s_{i-1} \text{ and } s_i > s_{i+1}\}$$

and

$$\bigwedge s = (s_\alpha\ s_\beta \cdots s_\omega),$$

where $I(s) = \{\alpha, \beta, \ldots, \omega\}$ and $\alpha < \beta < \cdots < \omega$.

Thus $\bigwedge s$ is the sequence of local maxima in s (excluding the first and the last element of s). For every $s \in D$, define $\text{last}(s) \in D$ recursively by

(i) if $\text{len}(s) \le 2$, then $\text{last}(s) = s$;
(ii) if $\text{len}(s) > 2$, then $\text{last}(s) = \text{last}(\bigwedge s)$.

Thus $\text{last}(s)$ is a sequence of length 0, 1, or 2. Define

$$E = \{s \in D \mid \text{len}(\text{last}(t)) \neq 0 \text{ for every nonempty prefix } t \text{ of } s\}.$$

The set E is exhaustive. It corresponds to the basic variant of the maximum-finding algorithm described in [3] and [12]. It can be shown that $N(s, E) \le 2k\lfloor \log k \rfloor$ when $\text{len}(s) = k$. □

3. *A Lower Bound for Unidirectional Algorithms*

When I is a finite nonempty set of integers, let $\mathrm{Perm}(I)$ be the set of permutations of I; and when A is a (unidirectional) maximum-finding algorithm, let $\mathrm{ave}_A(I)$ denote the average number of messages transmitted by A in rings labeled by the sequences $s \in \mathrm{Perm}(I)$. Similarly, let $\mathrm{worst}_A(I)$ be the worst case number of messages. The following lemma is an immediate corollary of Theorem 2.2.

LEMMA 3.1. *If I has n elements, then*

(a) $ave_A(I) \geq 1/n! \sum_{s \in Perm(I)} N(s, E(A))$;
(b) $worst_A(I) \geq max_{s \in Perm(I)} N(s, E(A))$.

THEOREM 3.2. *For every unidirectional maximum-finding algorithm A and for every I with n elements, we have*

$$ave_A(I) \geq nH_n = n \sum_{k=1}^{n} \frac{1}{k}.$$

PROOF. We can rewrite (a) of Lemma 3.1 as

$$\begin{aligned} \mathrm{ave}_A(I) &\geq \frac{1}{n!} \sum_{s \in \mathrm{Perm}(I)} \sum_{k=1}^{n} N_k(s, E(A)), \\ &= \frac{1}{n!} \sum_{k=1}^{n} \sum_{s \in \mathrm{Perm}(I)} N_k(s, E(A)). \end{aligned}$$

For fixed k and $s \in \mathrm{Perm}(I)$ there are n prefixes t of cyclic permutations of s such that $\mathrm{len}(t) = k$. Since there are $n!$ permutations in $\mathrm{Perm}(I)$, there are $n!\ n$ instances of such prefixes t (for a fixed k); they can be gathered in groups of k, so that each group consists of all cyclic permutations of one sequence. By the cyclic permutation property, the set $E(A)$ intersects each such group. Hence

$$\sum_{s \in \mathrm{Perm}(I)} N_k(s, E(A)) \geq \frac{n!\ n}{k}.$$

It follows that

$$\mathrm{ave}_A(I) \geq nH_n. \qquad \square$$

THEOREM 3.3. *If I has n elements, then*

$$\min_A ave_A(I) = nH_n$$

where the minimum is taken over all maximum-finding algorithms A.

PROOF. By Theorem 3.2 we have

$$\min_A \mathrm{ave}_A(I) \geq nH_n;$$

by [2] there is an algorithm A such that

$$\mathrm{ave}_A(I) = nH_n. \qquad \square$$

COROLLARY 3.4. *If I has n elements, then*

$$0.69n\log n + O(n) \leq \min_A worst_A(I) \leq 1.36n\log n + O(n)$$

where the minimum is taken over all maximum-finding algorithms A.

PROOF. By [8, p. 74] we have $H_n = \ln n + O(1)$; hence

$$\min_A \text{worst}_A(I) \geq \min_A \text{ave}_A(I) \geq nH_n \geq 0.69n\log n + O(n).$$

The second inequality is proved in [3]. □

4. *Lower Bounds for Bidirectional Algorithms*

In the previous section we computed the exact lower bound for the average number of messages transmitted by maximum-finding algorithms in unidirectional rings. Now we turn to circular configurations in which processes can pass messages in both directions (the so-called *bidirectional rings*). Since in the bidirectional ring a process does not receive all its messages from a single source, the execution is influenced by transmission delays in an essential way. That is why bidirectional algorithms are more difficult to understand than unidirectional ones.

In the sequel it will be technically convenient to use a slightly different termination criterion. Namely, in this section a maximum-finding algorithm is one that claims in *every* process the value of the maximum label. The two problems (the maximum-finding problem in the previous two sections and the one here) are equivalent modulo n messages (where n is the size of the ring) in the following sense: If at least one process knows the maximum label, then the knowledge can be spread to all other nodes at the cost of sending n additional messages. Since all our bounds in this section are of the form $cn \log n + O(n)$, the new termination criterion does not change the results.

We shall again use the concept of the *trace* of a message. Informally, a message has the trace $(s_1 \cdots s_k)$ if it carries the information (possibly encoded) that k consecutive nodes in the ring are labeled $s_1, \ldots, s_k$, and if it contains no information about other labels. The concept can be defined formally in a manner similar to the definition for unidirectional rings in Section 2, but the informal definition is sufficient for our purposes. A simple but useful observation is that if a message with the trace $r \in D$ is sent (by a particular algorithm) in a *ring* labeled by $s = (s_1 \cdots s_k)$ and if r is a subsequence of s, then a message with the trace r can be also sent (by the same algorithm) in the linear *segment* labeled by s (i.e., the ring labeled by s in which the bidirectional channel between s_1 and s_k has been cut).

We again denote by $\text{ave}_A(I)$ and $\text{worst}_A(I)$ the average and the worst case number of messages used by the algorithm A in the rings labeled by the sequences $s \in \text{Perm}(I)$. To estimate $\text{ave}_A(I)$ and $\text{worst}_A(I)$, that is, the number of messages *sent* in *rings*, we first estimate the number of messages *received* in linear *segments*. Every execution in a labeled segment can be simulated in the corresponding ring (labeled by the same sequence); therefore, the number of messages received in a segment is a lower bound for the number of messages received (and hence also for the number of messages sent) in the ring.

We denote by $\text{ave}^*_A(I)$ and $\text{worst}^*_A(I)$ the average and the worst case number of messages received when the algorithm A is executed in the segments labeled by the sequences $s \in \text{Perm}(I)$. We define

$$\text{ave}^*_b(I) = \min_A \text{ave}^*_A(I)$$

and

$$\text{worst}^*_b(I) = \min_A \text{worst}^*_A(I),$$

where the minimum is taken over all (bidirectional) maximum-finding algorithms A. We prove recursive inequalities for $\text{ave}_b^*(I)$ and $\text{worst}_b^*(I)$; that is why we deal with these quantities instead of their unstarred counterparts. (A recursive construction is also used by Burns [1] in his proof of a worst case lower bound.)

LEMMA 4.1. *If I and I' are two sets of the same cardinality, then $\text{ave}_b^*(I) = \text{ave}_b^*(I')$ and $\text{worst}_b^*(I) = \text{worst}_b^*(I')$.*

PROOF. Both equalities follow from this simple observation: Since I and I' have the same number of elements, there is an order-preserving one-to-one function g from I onto I', and g can be extended to a one-to-one function from the set of integers onto itself. Now for every algorithm A there exists an algorithm B such that A needs the same number of messages in the ring labeled by $(g(s_0) \cdots g(s_n))$ as B in the ring labeled by $(s_0 \cdots s_k)$. Namely, B mimics A, using the "code" $g(s_j)$ for every label s_j. Hence $\text{ave}_B^*(I) = \text{ave}_A^*(I')$ and $\text{worst}_B^*(I) = \text{worst}_A^*(I')$, and the result follows. □

THEOREM 4.2. *If I and H are two disjoint sets having at least k elements each, then*

(a) $\text{worst}_b^*(I \cup H) \geq \text{worst}_b^*(I) + \text{worst}_b^*(H) + k/2$;
(b) $\text{ave}_b^*(I \cup H) \geq \text{ave}_b^*(I) + \text{ave}_b^*(H) + k/4$.

PROOF. Let A be a maximum-finding algorithm. We consider two arbitrary sequences $s = (s_1 \cdots s_i) \in \text{Perm}(I)$ and $r = (r_1 \cdots r_h) \in \text{Perm}(H)$, and denote by s_x and r_y their midpoints. That is, $x = i/2$ if i is even and $x = (i + 1)/2$ if i is odd, and $y = h/2$ if h is even and $y = (h + 1)/2$ if h is odd.

We can start the execution of A in the ring labeled sr by sending and receiving first as many messages as possible within the segments labeled s and r (such an execution takes place when the transmission delays on the channels between the two segments are very long). In fact, we can start with *any* execution of A in the segment labeled by s and *any* execution in the segment labeled by r, and extend them to an execution in the ring. We are going to show that every such execution that is *complete* (i.e., one in which every process claims the value of the maximum label) contains sufficiently many message receptions in addition to those executed within the two segments.

In every complete execution on the ring labeled sr, the process labeled s_x must receive a message whose trace contains at least one label in r; similarly, the process labeled r_y must receive a message whose trace contains a label in s. Among all such messages, select one with the shortest trace; call the trace t. Then t is a subsequence of either sr or rs.

Now we are ready to prove inequality (a). First we find $s \in \text{Perm}(I)$ such that an execution of A in the segment labeled by s involves $\text{worst}_A^*(I)$ message receptions, and similarly $r \in \text{Perm}(H)$ with $\text{worst}_A^*(H)$ receptions. Then we select a message (with trace t) as in the previous paragraph. Assume that t is a subsequence of sr (the case in which t is a subsequence of rs is treated symmetrically). We construct an execution of A in the segment labeled by sr that involves at least $\text{worst}_A^*(I) + \text{worst}_A^*(H) + k/2$ message receptions. Namely, we execute $\text{worst}_A^*(I)$ message receptions in one segment, $\text{worst}_A^*(H)$ in the other, and make sure that the message with the trace t is received. Assume that the latter is received by the process labeled by s_x (the case in which the receiver is labeled by r_y is treated symmetrically); then trace t contains at least one label from the sequence r. Since there are $\lfloor (i - 1)/2 \rfloor$ nodes between the node labeled by s_x and the nearest node in the segment labeled

by r, it follows that at least $i/2$ additional messages (including the one with trace t) are received. Since $i \geq k$, we have proved that

$$\text{worst}_A^*(I \cup H) \geq \text{worst}_A^*(I) + \text{worst}_A^*(H) + \frac{k}{2}.$$

The inequality (a) follows.

To prove (b), let i and h be the cardinalities of I and H. By the assumption, $i \geq k$ and $h \geq k$. The average $\text{ave}_A^*(I \cup H)$ may be expressed as follows: For every partition of $I \cup H$ into two sets I' and H' whose cardinalities are i and h, find the average of the message counts over the segments labeled by sr and rs, $s \in \text{Perm}(I')$, $r \in \text{Perm}(H')$; then compute the average over all such partitions (every partition is equally probable). We have seen that either in the segment labeled by sr or in the one labeled by rs the algorithm receives at least $k/2$ additional messages. Thus on average (which involves both sr and rs) we get at least $k/4$ additional messages. Since, by Lemma 4.1, $\text{ave}_b^*(I) = \text{ave}_b^*(I')$ and $\text{ave}_b^*(H) = \text{ave}_b^*(H')$, we obtain

$$\text{ave}_A^*(I \cup H) \geq \text{ave}_b^*(I) + \text{ave}_b^*(H) + \frac{k}{4}$$

and (b) follows. □

In the proof of the next theorem we need the following lemma.

LEMMA 4.3. *Let $c \geq 0$ be a real constant and g a real function defined on $\{1, 2, \ldots\}$ such that $g(n) \geq 0$ for all n. If*

$$g(n) \geq 2g\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + c\left\lfloor \frac{n}{2} \right\rfloor$$

for $n \geq 2$, then

$$g(n) \geq \tfrac{1}{2}c((n+2)\lfloor \log n \rfloor + 4 - 2^{\lfloor \log n \rfloor + 2})$$

for all $n \geq 1$, and therefore

$$g(n) \geq \tfrac{1}{2}cn \log n + O(n).$$

PROOF. We use the equality

$$\left\lfloor \log \left\lfloor \frac{k}{2} \right\rfloor \right\rfloor = \lfloor \log k \rfloor - 1.$$

The proof proceeds by induction on n:

Basis. For $n = 1$, the right-hand side is 0, and $g(1) \geq 0$ by the assumption.

Inductive step. Assume that the inequality holds for $1 \leq n < k$, where $k \geq 2$. Then

$$\begin{aligned} g(k) &\geq 2g\left(\left\lfloor \frac{k}{2} \right\rfloor\right) + c\left\lfloor \frac{k}{2} \right\rfloor \\ &\geq c\left(\left(\left\lfloor \frac{k}{2} \right\rfloor + 2\right)\left\lfloor \log \left\lfloor \frac{k}{2} \right\rfloor \right\rfloor + 4 - 2^{\lfloor \log \lfloor k/2 \rfloor \rfloor + 2}\right) \\ &\quad + c\left\lfloor \frac{k}{2} \right\rfloor \\ &\geq \tfrac{1}{2}c((k+2)\lfloor \log k \rfloor + 4 - 2^{\lfloor \log k \rfloor + 2}), \end{aligned}$$

which shows that the inequality holds for $n = k$. □

THEOREM 4.4. *If I has n elements then*

(a) $worst^*_b(I) \geq \frac{1}{4} n \log n + O(n)$;
(b) $ave^*_b(I) \geq \frac{1}{8} n \log n + O(n)$.

PROOF. Use Theorem 4.2 and Lemma 4.3. □

Theorem 4.4 gives lower bounds for the number of messages received in segments, and therefore also for the number of messages sent in rings. The result (a) is essentially due to Burns [1].

5. *Algorithms that Know the Ring Size*

All our results so far have been derived under the assumption that the size of the ring is not known when the algorithm starts execution. Now we briefly consider another version of the problem. We assume that each process knows, when the execution starts, not only its own label but also the size of the ring (i.e., the number of the processes in the ring, but not the range of their labels).

We know no $\Omega(n \log n)$ lower bound for the *average* number of messages in the rings of known size. Moreover, the forthcoming lower bounds for the *worst case* number of messages are weaker than those in Corollary 3.4 and Theorem 4.4, in the following sense: Corollary 3.4 and Theorem 4.4 say that for every algorithm A and *for every set* I of n labels there exists a permutation s of I such that at least $\Omega(n \log n)$ messages are sent by A in the ring labeled by s. For the rings of known size n, we can only prove that *there exist* (infinitely many) sequences s of length n for which at least $\Omega(n \log n)$ messages are sent in the ring labeled by s.

In fact, the stronger version (the one in Corollary 3.4 and Theorem 4.4) is not valid for the rings of known size. Indeed, there is an algorithm that works correctly in every ring of size n and requires only n messages to find the maximum in any ring labeled by a permutation of $(1\ 2 \cdots n)$.

We again start with unidirectional algorithms. When $E \subseteq D$ and $s \in D$, we denote by $N^*(s, E)$ the cardinality of the set

$$\{t \in E \mid t \text{ is a subsequence of } s\}.$$

The obvious modification of the proof of Theorem 2.2 establishes the following result.

THEOREM 5.1. *For every algorithm that finds the maximum label in every unidirectional ring of size $n \geq 2$ there exists a set $E = E(A) \subseteq D$ such that*

(i) *E has the prefix property;*
(ii) *if $s \in D$ and $len(s) = n$, then $C(s) \cap E \neq \emptyset$;*
(iii) *if $s \in D$ and $len(s) \leq n$, then A transmits at least $N^*(s, E)$ messages when executed in the segment labeled by s.*

To estimate $N^*(s, E(A))$, we proceed as follows: For every $E \subseteq D$ and $k = 1, 2, \ldots$, we define $W^*_E(k)$ to be the largest number w for which there exist infinitely many pairwise disjoint sequences $s \in D$ such that $\text{len}(s) = k$ and $w = N^*(s, E)$. We show that if E has the properties (i) and (ii) in Theorem 5.1, then $W^*_E(k)$ satisfies a recursive inequality, which implies

$$W^*_E(n) = \Omega(n \log n).$$

LEMMA 5.2. *Let n, k, and p be positive integers such that n is divisible by k and $2k \leq n$. Let $E \subseteq D$ have the properties (i) and (ii) in Theorem 5.1. If $F \subseteq D$ is an*

infinite set of pairwise disjoint sequences such that len(s) = k for all s ∈ F, then D contains infinitely many pairwise disjoint sequences of the form $s^{(1)}\, s^{(2)} \cdots s^{(p)}$, *where* $s^{(i)} \in F$ *for* $i = 1, 2, \ldots, p$ *and each string* $s^{(j-1)}s^{(j)}$, $j = 2, 3, \ldots, p$, *has at least one suffix* $t \in E$ *with* $len(t) > k$.

PROOF. For $p = 1$ the statement is trivially true. Assume that it is true for some $p \geq 1$. Thus there are infinitely many pairwise disjoint sequences of the form $s^{(1)}s^{(2)} \cdots s^{(p)}$ that satisfy the conclusion of Lemma 5.2; we take all their first components $s^{(1)}$ and concatenate them in groups of n/k to form sequences of length n. To each such sequence of length n, we apply (ii) and (i) of Theorem 5.1, and get $s^{(0)}$, $s^{(1)} \in F$ such that for some $s^{(2)} \cdots s^{(p)}$ the sequence $s^{(1)}s^{(2)} \cdots s^{(p)}$ satisfies the conclusion of Lemma 5.2 and moreover $s^{(0)}s^{(1)}$ has at least one suffix $t \in E$ with $\text{len}(t) > k$. But that shows that the statement holds with p replaced by $p + 1$; hence it holds for all p. □

LEMMA 5.3. *Let q, k, and n be positive integers such that* $q \geq 2$, *n is divisible by k and* $qk \leq n$. *If* $E \subseteq D$ *has the properties (i) and (ii) in Theorem* 5.1, *then*

$$W_E^*(qk) \geq qW_E^*(k) + 2qk - 4k.$$

PROOF. By the definition of $W_E^*(k)$, there is an infinite set $F \subseteq D$ such that every two sequences in F are disjoint, and $\text{len}(s) = k$ and $N^*(s, E) = W_E^*(k)$ for all $s \in F$. Apply Lemma 5.2 with $p = n/k$ to get infinitely many disjoint sequences $s^{(1)}\, s^{(2)} \cdots s^{(p)}$ of length n, satisfying the conclusion of Lemma 5.2. By (i) and (ii) in Theorem 5.1, for each such sequence there is j, $1 \leq j \leq p$, such that

$$s^{(j)}s^{(j+1)} \cdots s^{(p)}s^{(1)} \cdots s^{(j-1)}$$

has at least one suffix $t \in E$ with $\text{len}(t) > (p - 1)k$. Now let s be the prefix of

$$s^{(j)}s^{(j+1)} \cdots s^{(p)}s^{(1)} \cdots s^{(j-1)}$$

of length qk. To get a lower bound for $N^*(s, E)$, we sum $N^*(s^{(i)}, E)$ for $i = j, j + 1, \ldots, j + q - 1 \pmod p$ and get $qW_E^*(k)$; we add

$$N^*(s^{(i)}s^{(i+1)}, E) - N^*(s^{(i)}, E) - N^*(s^{(i+1)}, E)$$

for $i = j, j + 1, \ldots, j + q - 2 \pmod p$, which gives at least $(q - 2)k$ by Lemma 5.2; and we add the number of the prefixes of t that have not been counted so far, which gives at least $(q - 2)k$. The grand total is $qW^*(k) + (2q - 4)k$. □

LEMMA 5.4. *For every positive integer q there is a function* $f_q(n)$ *such that*

$$f_q(n) = \frac{2q - 4}{q \log q} n \log n + O(n),$$

and $W_E^*(n) \geq f_q(n)$ *whenever n is a power of q,* $n \geq 2$, *and E satisfies the properties (i) and (ii) in Theorem* 5.1.

PROOF. Let $f_q(n)$ be the solution of the recursive system

$$f_q(1) = 1,$$
$$f_q(qk) = qf_q(k) + (2q - 4)k.$$

(These equations define $f_q(n)$ only when n is a power of q; the other values are irrelevant.) Then

$$f_q(n) = \frac{2q - 4}{q \log q} n \log n + O(n).$$

In view of Lemma 5.3, it only remains to be shown that $W_E^*(1) \geq 1$. However, it is easy to see that at most $n-1$ sequences of length 1 lie outside of E. Indeed, if there were n distinct numbers $s_1, \cdots, s_n$ such that $(s_i) \notin E$ for each i, then (ii) and (i) in Theorem 5.1, applied to $s = (s_1 \cdots s_n)$, would lead to contradiction. □

The expression $(2q-4)/q\log q$ attains its maximum (over positive integers q) for $q = 5$. The approximate value of $6/(5\log 5)$ is 0.51. Theorem 5.5 follows directly from Theorem 5.1 and Lemma 5.4 for $q = 5$.

THEOREM 5.5. *If n is a power of 5, then any algorithm that finds maximum in each unidirectional ring of size n sends at least*

$$0.51n \log n + O(n)$$

messages in infinitely many rings of size n.

Next we turn to bidirectional algorithms. We again prove a lower bound $\Omega(n\log n)$ for the worst case number of messages, but with a smaller constant. The pattern of proof is similar to that in Section 4. We again take a maximum-finding algorithm to be one that claims the value of the maximum label in every process, and we use the term trace in the same meaning as in Section 4.

For every algorithm A we denote by $R_A^*(k)$ the largest number w for which there exist infinitely many pairwise disjoint sequences $s \in D$ such that $\text{len}(s) = k$ and at least w messages are *received* during some execution of A in the *segment* labeled by s.

LEMMA 5.6. *Let k and n be positive integers such that n is divisible by k and $2k \leq n$. If A is a maximum-finding algorithm for the rings of size n, then*

$$R_A^*(2k) \geq 2R_A^*(k) + \frac{k}{2}.$$

PROOF. By the definition of $R_A^*(k)$, there is an infinite set $F \subseteq D$ such that every two sequences in F are disjoint, $\text{len}(s) = k$ for every $s \in F$, and for every $s \in F$ the algorithm A can be executed in the segment labeled by s so that at least $R_A^*(k)$ messages are received. We concatenate the sequences in F in groups of n/k to form infinitely many pairwise disjoint sequences of length n.

Let r be one such sequence of length n. In the ring labeled by r, we execute A so that at least $R_A^*(k)$ messages are received within each segment labeled by $s \in F$. Now consider, for each subsequence s of r, $s \in F$, the midpoint s_x of s. (The rest of the proof is similar to the proof of (a) of Theorem 4.2.) If our contemplated execution of A (in the ring labeled by r) is complete, then the process labeled by s_x eventually receives a message whose trace contains a label not in s (otherwise, the process could not claim to know the maximum label). Among all such traces (of the messages received by the midpoint processes and containing at least one label from another segment), select the shortest one and call it t. From the minimality of t it follows that t is a subsequence of ss', for some $s, s' \in F$. Assume, without loss of generality, that t is the trace of a message received by the process labeled by the midpoint s_x of s; hence t contains at least one label in s'. The length of the segment labeled by ss' is $2k$, and A can be executed in the segment so that at least $2R_A^*(k) + k/2$ messages are received—namely, $R_A^*(k)$ messages within the segment labeled by s, the same number within the one labeled by s', and $k/2$ messages that form a chain from s' to s_x. Since there are infinitely many sequences r, there are infinitely many sequences ss'. The proof is complete. □

Since Theorem 5.5 follows from Lemma 5.3, so the next theorem follows from Lemma 5.6.

THEOREM 5.7. *If n is a power of 2, then every maximum-finding algorithm for the (bidirectional) rings of size n sends at least*

$$\tfrac{1}{4}n\log n + O(n)$$

messages in infinitely many rings of size n.

6. *Concluding Remarks*

We have established lower bounds of the form $\Omega(n\log n)$ for the average and worst case number of messages sent by distributed maximum-finding algorithms in unidirectional and bidirectional rings of unknown size, and for the worst case number of messages in the rings of known size. The problem of finding lower bounds for the average number of messages and for the rings of known size was raised in [3]. The bound (a) in our Theorem 4.4 is due to Burns [1], but he proved that there *exists* a sequence of length n satisfying the bound, whereas our proof shows that *every* set of size n can be permuted to yield such a sequence (see the remarks at the beginning of Section 5).

Since upper bounds of the form $O(n\log n)$ are known [1–6, 9, 10, 12], our lower bounds are the best possible bounds up to constant factors. Nevertheless, it is of interest to determine the values of the constant factors. We now summarize the best estimates known to date. In this summary we assume that bidirectional rings are *oriented*. That is, although messages in the ring can be sent in both directions, one direction is agreed upon by all processes. The assumption is not used in our lower bounds, but it is essential for upper bounds: Every unidirectional algorithm works also in oriented bidirectional rings. (However, $O(n\log n)$ messages suffice even for nonoriented bidirectional rings. The most recent algorithm [10] transmits $1.89n\log n + O(n)$ messages in the worst case.)

In unidirectional rings of unknown size, the average number of $cn\log n + O(n)$ messages, where $c = 1/\log e = 0.69\cdots$, is both sufficient and necessary, by [2] and Theorem 3.2. In bidirectional rings the same number is obviously sufficient, and the average of $\frac{1}{8}n\log n + O(n)$ messages is necessary by Theorem 4.4.

In terms of the worst case number of messages in rings of unknown size, $1.36n\log n + O(n)$ messages are sufficient in unidirectional (and hence also in bidirectional) rings [3]. By Theorems 3.2 and 4.4, $0.69n\log n + O(n)$ messages are necessary in unidirectional rings, and $\frac{1}{4}n\log n + O(n)$ in bidirectional ones.

For bidirectional rings the gaps are wider, confirming the fact that the nondeterminism inherent in bidirectional algorithms makes them more difficult to understand.

For rings of known size, we have no upper bounds (i.e., algorithms) better than those for rings of unknown size, and no nonlinear lower bounds for the average number of messages. In Theorems 5.5 and 5.7 we establish lower bounds for the worst case number of messages in rings of known size, $0.51n\log n + O(n)$ for unidirectional rings and $\frac{1}{4}n\log n + O(n)$ for bidirectional ones. However, these functions are lower bounds in a weaker sense than those in Sections 3 and 4: They are only proved for certain values of n, and we are only able to assert the *existence* of label sequences with bad behavior.

It should be noted that both our lower bounds for the *average* number of messages (Theorem 3.2 and (b) of Theorem 4.4) hold more generally, for the *average expected* number of messages sent by probabilistic maximum-finding

algorithms. (In a probabilistic algorithm, each process can proceed depending on the value of a random variable; to prevent implicit communication between processes, we require that the random variables used by different processes be independent. One probabilistic maximum-finding algorithm is described and analyzed in [9].) It is straightforward to extend the lower bound in (b) of Theorem 4.4 to probabilistic algorithms; the proof of the recursive inequality in (b) of Theorem 4.2 still works with "average expected" in place of "average." To extend the lower bound in Theorem 3.2, one needs a generalization of the concept of an exhaustive set; the idea will be developed elsewhere.

It remains an open problem whether the knowledge of the ring size or the capability of sending messages in both directions along the ring can be used to decrease the (average or worst case) number of messages.

REFERENCES

1. BURNS, J. E. A formal model for message passing systems. Tech. Rep. No. 91, Computer Science Dept., Indiana Univ., Bloomington, Ind., 1980.
2. CHANG, E., AND ROBERTS R. An improved algorithm for decentralized extrema-finding in circular configurations of processes. *Commun ACM 22*, 5 (May 1979), 281–283.
3. DOLEV, D., KLAWE, M., AND RODEH, M. An $O(n \log n)$ unidirectional distributed algorithm for extrema finding in a circle. *J Algorithms 3*, 3 (Sept. 1982), 245–260.
4. FRANKLIN, W. R. On an improved algorithm for decentralized extrema finding in circular configurations of processors. *Commun ACM 25*, 5 (May 1982), 336–337.
5. GALLAGER, R. G. Finding a leader in a network with $O(E) + O(N \log N)$ messages. M. I. T. Internal Memorandum, Massachusetts Institute of Technology, Cambridge, Mass.
6. HIRSCHBERG, D. S., AND SINCLAIR, J. B. Decentralized extrema-finding in circular configurations of processors. *Commun ACM 23*, 11 (Nov. 1980), 627–628.
7. ITAI, A., AND RODEH, M. Symmetry breaking in distributive networks. In *Proceedings of the 22nd Annual Symposium on Foundations of Computer Science*. (Nashville, Tenn., Oct. 28–30). IEEE, New York, 1981, pp 150–158.
8. KNUTH, D. E. *The Art of Computer Programming* Vol. 1, *Fundamental Algorithms* 2nd ed. Addison-Wesley, Reading, Mass., 1973
9. KORACH, E., ROTEM, D., AND SANTORO, N. A probabilistic algorithm for decentralized extrema-finding in a circular configuration of processors. Res. Rep. CS-81-19, Dept. of Computer Science, Univ. of Waterloo, Waterloo, Ontario, Canada, 1981.
10. KORACH, E., ROTEM, D., AND SANTORO, N. Distributed election in a circle without a global sense of orientation. *Int J. Comput Math* To be published.
11. LE LANN, G. Distributed systems—Towards a formal approach. In *Information Processing 77*, B. Gilchrist, Ed. Elsevier North-Holland, New York, 1977, pp. 155–160.
12. PETERSON, G. L. An $O(n \log n)$ unidirectional algorithm for the circular extrema problem. *ACM Trans Program Lang Syst 4*, 4 (Oct. 1982), 758–762.
13. TANENBAUM, A. S. *Computer Networks* Prentice-Hall, New York, 1981.

RECEIVED MARCH 1983; REVISED MARCH 1984; ACCEPTED APRIL 1984