The Distributed Bit Complexity of the Ring: From the Anonymous to the Non-anonymous Case

(final version, October 1991)

Hans L. Bodlaender *
Department of Computer Science, Utrecht University,
3508 TA Utrecht, the Netherlands.

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

Manfred K. Warmuth ***
Department of Computer and Information Sciences,
University of California,
Santa Cruz, CA 95064.

ABSTRACT:

In [MW] it was shown that computing any non-constant function on a ring of n processors requires $\Omega(n\log n)$ bits and this bound is tight. The model of [MW] assumed that all the processors in the ring are identical (anonymous [ASW]), i.e. all processors run the same program and the only parameter to the program is the input of the processor. In a relaxed model the anonymity is broken by providing each processor with a distinct identity which becomes a second parameter to the program that is executed by all processors. If the set of possible identities grows double exponentially in n, then by a reduction to the anonymous case one can show that the lower bound holds as well [MW].

In this paper we show that the lower bound of $\Omega(n\log n)$ bits for computing any non-constant function holds even if the set of possible identities is "very small", that is - of size $n^{1+\epsilon}$, for any positive ϵ.

* Part of this work was done while this author was at the Laboratory for Computer Science at MIT being supported by a grant from the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

** This research was supported in part by Technion V.P.R. Funds - Wellner Research Fund, and by the Foundation for Research in Electronics, Computers and Communications, administrated by the Israel Academy of Sciences and Humanities. Part of this work was done while this author was visiting the University of California at Santa Cruz being supported by ONR grant N00014-86-K-0454.

*** This author gratefully acknowledges the support of ONR grant N00014-86-K-0454. Part of this work was done while this author was visiting the Technion, Israel Institute of Technology, supported by the Wolberg fund.

1. INTRODUCTION.

Consider a distributed network of processors that communicate asynchronously along the links of the network. Each processor receives an external input. The processors compute a function of the total input configuration of the network and then terminate. Our goal is to establish lower bounds on the amount of resources (complexity) required to compute any non-constant function on a given network.

As a complexity measure of an algorithm we use the worst case message and bit complexity over all possible input configurations, all possible choices (if any) for the identities of the processors and all possible delay times of the communication links of the network. The minimum worst case message and bit complexity for computing any non-constant function on the network is called the (asynchronous) *distributed message* and *bit complexity*, respectively, of the network [MW]. This paper provides results about the distributed bit complexity of a ring of processors with distinct identifiers.

In general we seek to investigate how the distributed bit complexity depends on the network. Intuitively, this complexity increases with the amount of "symmetry" given in the network which is determined by the two following factors.

a) The topology of the network: To compute the function value, each processor needs to get some global information about the input configuration. Topological symmetry in the network makes it hard to gather global information without causing a large number of messages/bits to be sent.

b) The degree of anonymity of the processors. Three cases are considered in the literature: In the first, the processors are anonymous (i.e. they have no identities). In the second, each processor has a distinct identity. The identities can be used to break symmetry in the network and cut down message traffic. In a third case, the network has one distinguished processor (the *leader*) who can coordinate the computation.

If the network has a leader, then this leader can first instigate messages that collect the global information that is necessary to compute the non-constant function. Then, it can broadcast the function value to the network. The bit complexity of such an algorithm can be linear in the size (number of edges + number of vertices) of the network. Note that in some networks the topology of the network distinguishes a small number of processors. For example in the star network the central processor is naturally the leader and in a chain the two end processors can coordinate the computation. The definition of distributed complexity is mainly useful for highly symmetric networks, such as the ring, the hypercube or the torus. Each of these networks is used in practice.

The distributed complexity of a network gives a theoretical limit of the capabilities of the network. It can be used to guide the choice of networks for given applications. Previous research and this paper concentrate on the distributed complexity of the ring. The ring provides an interesting case study for investigating how processors may co-operate to break symmetry.

The distributed bit complexity of a ring of n anonymous processors is $\Theta(n\log n)$ [MW]. The proof of $\Omega(n\log n)$ bits lower bound in [MW] holds if either the processors are anonymous or have distinct identities chosen from a very large set (of size $\Omega(n2^{n^n})$).

If the set of possible identities is very small, i.e. in $\{1,2,\cdots,n+c\}$, for some constant c, then it is easy to compute non-constant functions in $O(n)$ bits (by using processors with identities in $\{1,2,\cdots,c+1\}$ as leaders). We show in this paper that if the processors have distinct identities chosen from an only moderately large set, then the $\Omega(n\log n)$ lower bound still holds.

Theorem 1. Let f be any non-constant function on Σ^n for some arbitrary alphabet Σ, and let *AL* be any asynchronous algorithm that computes f on rings of n processors, labeled with n distinct identities chosen from a set X of size at least $n^{1+\varepsilon}$, for some $\varepsilon > 0$. Then the worst case bit complexity of *AL* is $\Omega(n\log n)$.

It seems that in highly symmetric networks such as the ring the only way to compute non-constant functions in $O(n)$ bits is essentially to pre-elect a leader. However, electing a leader costs $\Omega(n\log n)$ messages even if the set of possible identities is only of size cn, for any constant $c>1$ [PKR, B1].

The proof of Theorem 1 is constructive: given an algorithm *AL* that computes a non-constant function on a ring of n processors, "cut-and-paste" techniques are used to construct a computation of *AL* in which $\Omega(n\log n)$ bits are sent. Similar techniques were first used in [MW] for the anonymous case, where the processors have no identities. When the processors have identities then cutting and pasting is more involved, because the final ring must be constructed from processors with distinct identities chosen from a set of size $O(n^{1+\varepsilon})$. The key idea is to iteratively apply cutting and pasting to many "lines" of processors in parallel. A case analysis shows that in the final set of lines there must be a line that can be embedded in a ring of n processors with distinct identities, and a computation of *AL* on this line requires $\Omega(n\log n)$ bits.

The lower bound of Theorem 1 does not depend on the size of the input alphabet, and it holds uniformly for all ring sizes. In contrast, it has been shown that the distributed *message* complexity of the anonymous ring *does* depend on the size of the input alphabet as well as on the properties of the ring size. Specifically, large input alphabets, as well as small non-divisors of the ring size, can reduce the distributed message complexity of the anonymous ring [B2, DG, MW].

A challenge will be to determine the distributed bit complexity of other networks, such as the hypercube. Recently, the distributed bit complexity of the anonymous torus was shown to be $\Theta(n)$ [BB]. The proofs for the case when the processors have identities are expected to be more involved (see this paper) than in the anonymous case. Some techniques developed in this paper for rings are likely to be applied to other networks. Also, it is an interesting open problem to determine whether the lower bound of Theorem 1 can be extended to the case where the set of distinct identities is of size cn, for some constant c.

Probabilistic ways to break symmetry are studied in [AAHK], where the "probabilistic" distributed bit complexity of anonymous rings is shown to be $\Theta(n\sqrt{\log n})$. It would be interesting to know whether this bound remains if the processors have distinct identities in a reasonably large range.

2. DEFINITIONS AND BASIC RESULTS.

Let Σ be the input alphabet of arbitrary large size. Let X be a set of identities, and suppose that $|X| \geq n^{1+\varepsilon}$, for some constant $\varepsilon > 0$. A *processor* consists of an input letter and an identity. Let $\sigma = \sigma_1 \cdots \sigma_m \in \Sigma^m$ be a word of some length m, and let $x = (x_1, \cdots, x_m)$ be a sequence of m (not necessarily distinct) identities taken from X. A *ring configuration* $R(\sigma,x)$ consists of processors $p_1, \cdots, p_m$, where p_i is connected by a link to $p_{i(mod\ m)+1}$, (for $i = 1, \cdots, m$), and processor p_i has input σ_i and identity x_i. All the processors in the ring run the same algorithm. A processor may wake up spontaneously or by receiving a message from a neighbor. While running the algorithm, the processor may send messages to its left and right neighbors (our result applies also to the case where all the processors agree on "left" and "right"). All messages arrive in finite, but unpredictable and unbounded time. A *line configuration* $L(\sigma,x)$ is defined similarly to a ring configuration, except that there is no link between p_1 and p_m. An execution of algorithm AL on a line configuration is an initial part of an execution on the corresponding ring configuration, in which there is a temporary block on the link connecting p_1 and p_m, and all other messages are delivered. The *size* of a ring configuration (or a line configuration) is the number of processors in the configuration. We use "$\cdot$" to denote the concatenation of two sequences. Thus, for line configurations $L_1 = L(\tau,x)$ and $L_2 = L(\omega,y)$, $L_1 \cdot L_2$ denotes the line configuration $L(\tau\omega,xy)$.

Let f be a function from Σ^n to $\{0,1\}$. An algorithm *AL computes f* if for all $\sigma \in \Sigma^n$, for all sequences x of n distinct identities, and for all executions of *AL* on ring configuration $R(\sigma,x)$, every processor eventually outputs $f(\sigma)$ and terminates. Note that since all the processors run the same program and the function value does not depend on the identities, only functions that are invariant under cyclic shifts of the input words can be computed by a ring of processors.

Consider an execution of some distributed algorithm on a ring or a line configuration. The *link-history* of a link e in this execution, denoted by $h(e)$, consists of the messages delivered on e with their directions. Formally, $h(e)$ is the word $d_1 m_1 d_2 m_2 \cdots d_s m_s$, where d_i is either R (for right) or L (for left), and $m_i \in \{0,1\}^+$ is the i^{th} message delivered on e, in direction d_i. A message is considered delivered when it is accepted and in case of a tie, messages from the left are delivered before message from the right. The *cost* of a link-history is the number of characters in it. Note that the cost of a link-history is at most twice the number of bits delivered on the link. In [MW] a similar notion of history was defined for processors instead of links.

A *history sequence* of a line configuration $L(\sigma,x)$ is a sequence $H = H(L) = (p_1,h_1, ,p_2,h_2, \cdots ,h_{n-1},p_n)$, where the p_i's are the processors in L and the h_i's are link-histories. If there is an execution of an algorithm *AL* on a line configuration L in which the link-history of the link connecting p_i and p_{i+1} is h_i, then the history sequence H is *produced* by the algorithm *AL*, and $E(H)$ denotes an arbitrary execution of *AL* that produces H. *A segment of size m* of the history sequence $H(L)$ is a sequence $(p_i,h_i, \cdots ,h_{i+m-2},p_{i+m-1})$ $(i+m-1 \le n)$. The *cost* of a segment of a history sequence is the sum of the costs of all its link-histories.

Let $H_1 = (p_1,h_1, \cdots ,h_{n-1},p_n)$ and $H_2 = (q_1,h'_1, \cdots ,h'_{m-1},q_m)$ be two segments of history sequences, and let h be a link-history. Then $H_1 \cdot h \cdot H_2$ denotes the segment $(p_1,h_1, \cdots ,h_{n-1},p_n,h,q_1,h'_1, \cdots ,h'_{m-1},q_m)$.

A basic tool in our proof is converting executions of *AL* on ring configurations to executions on line configurations and vice-versa. If *AL* computes f then the output value of any execution of *AL* on a ring configuration $R(\sigma,x)$ must be $f(\sigma)$ for all possible delay times of the asynchronous links. Therefore we may choose particular delay times for the proofs. The basic delay strategy [MW], here called *semi-synchronized execution*, is an execution in which all processors wake up at time 0, internal computation at a processor is instantaneous and links are either *blocked* (very large delay) or are *synchronized* (it takes exactly one time unit to traverse the link); each unblocked link may become blocked at any time, and once it becomes blocked it remains so until no more messages are sent along the unblocked links (this must eventually happen if the message complexity of AL is finite). In our proof we construct particular semi-synchronized computations, in which some processor outputs an incorrect function value before the block is released. After releasing the block, the incorrect output cannot be withdrawn.

Consider an execution of AL on $R(\sigma,x)$ which is (fully) synchronized (no link is blocked). If any such execution takes more than $n\log n$ time steps then it requires at least $n\log n$ messages, since in each step at least one message is sent. Thus Theorem 1 holds trivially in this case. Therefore, we may assume for the rest of the paper the following:

Assumption T: All fully synchronized executions of AL require fewer than $n\log n$ time steps.

Let $k = \lceil \log n \rceil$, and let $t = kn$. By Assumption T, all executions of AL terminates in fewer than t time units. This value of t is fixed throughout the paper. As in [MW], we associate the *canonical line configuration* $D(\sigma,x) = L(\sigma^{2k}, x^{2k})$ with the ring configuration $R(\sigma,x)$. The $2t = 2kn$ processors in $D(\sigma,x)$ are denoted by $p_{1,1}, p_{2,1}, \cdots, p_{n,1}, p_{1,2}, \cdots, p_{n,k}, p'_{1,1}, \cdots, p'_{n,k}$. Note that, by definition, processors $p_{i,j}$ and $p'_{i,j}$ have identity x_i and input σ_i. Informally, $D(\sigma,x)$ consists of $2k$ copies of $R(\sigma,x)$ that were cut at the link $p_n{-}p_1$ and then concatenated to one line of $2kn$ processors. Thus, processors $p_{i,j}$ and $p'_{i,j}$ in $D(\sigma,x)$ correspond to the processor p_i in the j^{th} and $(k+j)^{th}$ copies of $R(\sigma,x)$.

Let $D = D(\sigma,x)$ be a canonical line configuration. A *canonical execution* of AL on D is a semi-synchronized execution in which for $i = 1, \cdots, t$, the i^{th} leftmost link and the i^{th} rightmost links of D are blocked at time i (recall that, by the definition of line configuration, the link connecting $p_{1,1}$ and $p'_{n,k}$ is blocked at time 0). Finally, the *canonical history sequence* of D, denoted by $H(D)$, is the history sequence produced by the canonical execution of AL on D. The relation between a synchronized execution on a ring configuration $R(\sigma,x)$ and a canonical execution on the corresponding line configuration $D(\sigma,x)$ is given by the following:

Lemma 2.1: [MW]. In the canonical execution of $D(\sigma,x)$, both $p_{n,k}$ and $p'_{1,1}$ output $f(x)$ and terminate. $\square$

The processors $p_{n,k}$ and $p'_{1,1}$ will be called the *center processors* of $D(\sigma,x)$.

Lemma 2.2 [MW]: Let $D = D(\sigma,x)$ be the canonical line configuration of $R = R(\sigma,x)$, and let $H = H(D)$. The bit complexity of the synchronized execution of AL on R is at least half of the cost of the history sequence of any segment of H of size n.

Proof: Any n consecutive histories in H are prefixes of the histories of the corresponding edges in a synchronized execution on the ring configuration $R(\sigma,x)$. The result now follows from the observation that the cost of a link-history is at most twice the number of bits delivered on that link. □

Lemma 2.3 [MW]: Let $W_1,...,W_k$ be k distinct words over an alphabet of size $r > 1$. Then there is a word W_i s.t. $|W_i| \geq \log_r(k/2)$ (for $1 \leq i \leq k$) and $|W_1|+|W_2|+\cdots+|W_k| > (k/2)\log_r(k/2)$.

Proof: Represent the W_i with an r-ary tree, s.t. each W_i corresponds to a path from the root to an internal node or a leaf of the tree. In the tree each leaf is responsible for some W_i. Assume that the total length of the words W_i is minimized. Then all the leaves are in the last two levels of the tree. Assume further that the W_i's are chosen so that the leaves in the maximal level are as far to the left as possible. Then in the corresponding tree all but at most one of the internal nodes have degree at least 2, and hence at least half of the nodes are leaves. The lemma is implied by the fact that the average height of the leaves in an r-ary tree with v leaves is at least $\log_r v$. □

Lemmas 2.2 and 2.3 above imply the following.

Lemma 2.4: Let $R = R(\sigma,x)$ be a ring configuration, let $D = D(\sigma,x)$, and let AL be a given algorithm. If there is a sequence of n consecutive link-histories in $H(D)$ that contains at least $\frac{1}{12}n$ distinct histories, then AL requires $\Omega(n\log n)$ bits in the worst case. □

In view of Lemma 2.4, we assume the following for the rest of the paper.

Assumption Q: There is no canonical history sequence resulting from AL in which n consecutive histories contain more than $\frac{1}{12}n$ distinct histories.

In the proofs below we manipulate existing history sequences to create new ones. The following two basic rules are used in our manipulations.

Rule 1: Let $H = (q_1,h_1,\cdots,q_i,h_i,q_{i+1},\cdots,h_{m-1},q_m)$ be a history sequence that is produced by an execution E on a line $L = L(\sigma,x)$, and let h'_i be any fixed prefix of h_i. Then there are histories $h'_1,\cdots,h'_{i-1}$, $h'_{i+1},\cdots,h'_{m-1}$, where h'_j is a prefix of h_j, such that the history sequence $H' = (q_1,h'_1,\cdots q_i,h'_i,q_{i+1},\cdots,h'_1,q_m)$ is produced by some execution E' of AL on L.

Proof: Consider the execution E of AL on L. At a certain moment during this execution, the link-history of the i'th link will be h'_i. Then stop this execution by blocking all links. Clearly, for each j, $1 \le j \le m$, the link-history of the j'th link will be a prefix of h_j. □

One specific case where Rule 1 is applicable is when h'_i and h_i are link-histories occurring in the same canonical history sequence, and the corresponding links connect copies of the same processors.

Rule 2: Let $H = H_1 \cdot h \cdot H_2$ and $H' = H'_1 \cdot h \cdot H'_2$ be two history sequences, corresponding to executions E and E', respectively, of AL on line configurations $L = L_1 L_2$ and $L' = L'_1 L'_2$, where h is the link-history of the links connecting L_1 with L_2 and L'_1 with L'_2. Then the history sequence $\hat{H} = H_1 \cdot h \cdot H'_2$ is produced by an execution $\hat{E}$ of AL on the line configuration $\hat{L} = L_1 L'_2$.

Proof: $\hat{E}$ is obtained by merging the executions E, restricted to L_1, and E' restricted to L'_2, as follows. Suppose h=$(d_1 m_1 \cdots d_s m_s)$. Let e be the link between L_1 and L'_2. Then, for i from 1 to s: if d_i=R, then do a part of execution E on L_1, until message m_i is sent on link e; else do a part of execution E' on L'_2, until message m_i is sent on link e. The resulting execution $\hat{E}$ of AL produces the history sequence $H_1 \cdot h \cdot H'_2$ on $\hat{L} = L_1 L'_2$. □

One specific way in which Rule 2 will be used is called *maximal shrinking*: If a history sequence $H = H_1 \cdot h \cdot H_2 \cdot h \cdot H_3$ corresponds to an execution of AL, then the history sequence $H' = H_1 \cdot h \cdot H_3$ also corresponds to some execution of AL. By repeated applications of this operation, any segment of a history sequence can be *shrunk* to a segment in which all the link-histories are distinct (a similar technique was also used in [MW]).

3. CONSTRUCTING FAMILIES OF CONTRADICTING COMPUTATIONS.

Let τ and ω be two input configurations such that $f(\tau) \neq f(\omega)$, and let X be the set of all possible identities, where $|X| \ge n^{1+\varepsilon}$. We split X into two disjoint sets, X_τ and X_ω, whose sizes differ by at most one. We then construct two families of history sequences $F(\tau)$ and $F(\omega)$ that correspond to executions of AL on τ and ω, respectively. The identities of the processors in the history sequences in $F(\tau)$ and $F(\omega)$ will belong to sets X_τ and X_ω, respectively, and each identity will occur in at most one history sequence. $F(\tau)$ and $F(\omega)$ will have similar properties. Informally, in the construction of $F(\tau)$ [$F(\omega)$] we attempt to construct history sequences of small size, in which some processors output $f(\tau)$ [$f(\omega)$ respectively]. If such small history sequences are constructed in both $F(\tau)$ and $F(\omega)$, then they can be embedded into the ring, leading to an execution of AL in which two processors output different function values. This contradicts the correctness of AL. Otherwise, we show that the bit complexity of AL is $\Omega(n\log n)$. The rest of this section is outlined as follows: We first give a number of properties satisfied by each history sequence H of $F(\tau)$. Each sequence H is given as a decomposition into smaller sequences and the decomposition is used in an iterative construction for F_τ maintaining the given properties for the history sequences. After showing that the iterative construction terminates after finitely many steps, a number of additional properties of the elements of $F(\tau)$ are proven which are crucial for the proof of Theorem 1 in the next section.

Each history sequence H in $F(\tau)$ will satisfy the following properties:

Property (a): There is an actual execution of AL, $E(H)$, that produces H.

Property (b): H is given as a decomposition: $H = LE \cdot h_L \cdot LI \cdot h_C \cdot RI \cdot h_R \cdot RE$, where LE, LI, RI and RE are segments of history sequences connected by the links with histories h_L, h_C and h_R. The parts LE and RE might be empty in which case the connecting link histories h_L and h_R, respectively, are empty as well. The decomposition of H satisfies the following:

Property (b1): There is a canonical line configuration $D = D(\tau,x)$, where x is a sequence of n distinct identities from X_τ, such that $LI \cdot h_C \cdot RI$ is a segment of the canonical history sequence $H(D)$. Moreover, the rightmost processor of LI and the leftmost processor of RI (which are connected by a link with link-history h_C) are the center processors of D, and hence they output $f(\tau)$.

Property (b2): All the identities of the processors in $LE \cdot RE$ are distinct from each other and no identity of a processor in $LE \cdot RE$ is also an identity of a processor in $LI \cdot h_C \cdot RI$. If two processors in $LI \cdot h_C \cdot RI$ are distance kn apart for some integer k, then they have the same identity and otherwise they have different identities.

Property (c): Any identity occurring in H does not occur in any other history sequence in $F(\tau)$.

The segments LE and RE of H, as well as the histories h_L and h_R, may be empty. The segment $LI \cdot h_C \cdot RI$ will be called the *inner part* of H.

To construct a set $F(\tau)$, we construct a sequence $\mathbf{F} = (F_0, \cdots, F_i, \cdots)$, where each F_i is a set of history sequences. F_0 is the empty set, and F_{i+1} is obtained by applying one of the operations (i) - (iii) below to F_i. Eventually we get a set F_N to which none of these operations is applicable; this F_N is $F(\tau)$.

(i) *ADD*: Assume that there are n identities in X_τ that do not appear in any history sequence in F_i, and let $x=(x_1, \cdots, x_n)$ be a sequence of such identities. Let $D = D(\tau,x)$ be the canonical line configuration of $R(\tau,x)$.

F_{i+1} is obtained by adding $H(D)$, the canonical history sequence of D, to F_i. $H(D)$ will be written as the concatenation $LI \cdot h_C \cdot RI$, where h_C is the link-history of the link connecting the center processors of $D(\tau, x)$.

(ii) *LEFT-JOIN* (see Figure 1): This operation replaces two history sequences H and H' in F_i by their *LEFT-JOIN*, which is the history sequence $\hat{H}$ defined below; the resulting set is F_{i+1}. The property of $\hat{H}$ which we need for our proof is that its inner part is shorter than the inner parts of both of H and H'. The definition of this operation follows:

Let $H = LE \cdot h_L \cdot LI \cdot h_C \cdot RI \cdot h_R \cdot RE$ and $H' = LE' \cdot h_L' \cdot LI' \cdot h_C' \cdot RI' \cdot h_R' \cdot RE'$ be in F_i. The operation *LEFT−JOIN* can be applied to the history sequences H and H' iff they satisfy the following conditions:

(a) The size of the inner part of H is at least as large as the size of the inner part of H'.

(b) There is a link-history h such that $LI = LI_1 \cdot h \cdot LI_2$ and $LI' = LI_1' \cdot h \cdot LI_2'$, where the size of LI_1 is at most n.

(c) LE is of size at most $\frac{1}{12}n$.

The *LEFT-JOIN* of H and H' is the history sequence $\hat{H} = \hat{LE} \cdot h \cdot LI_2' \cdot h_C' \cdot RI' \cdot h_R' \cdot RE'$, where $\hat{LE}$ is obtained by performing a maximal shrinking on the segment $LE \cdot h_L \cdot LI_1$ (Note that all the identities in $LE \cdot h_L \cdot LI_1$ are distinct and different from those of $LI_2' h_C' RI' h_R' RE'$.)

(iii) *RIGHT-JOIN*: This operation is defined similarly to *LEFT-JOIN*.

For a family $F(\tau)$ to exist, we need the following lemma.

Lemma 3.1: Let $\mathbf{F} = (F_0, \cdots, F_i, \cdots)$ be a sequence of sets of history sequences such that $F_0 = \emptyset$ and F_{i+1} is obtained by applying one of the operations (i) - (iii) above to F_i. Then $\mathbf{F}$ is finite.

Proof: Let t be the constant defined in Section 2, after Assumption T. Then the size of every canonical history sequence is $2t$. Define the *value* $V(H)$ of a history sequence $H = LE \cdot h_L \cdot LI \cdot h_C \cdot RI \cdot h_R \cdot RE$ to be 3^{2t-s}, where s is the size of $LI \cdot h_C \cdot RI$. The value $V(F)$ of F is defined as the sum of the values of the history sequences in it. Since each identity can occur in at most one history sequence in F_i, it follows that F_i contains fewer than $n^{1+\varepsilon}$ history sequences. Hence $V(F_i)$ is bounded from above by $n^{1+\varepsilon} \cdot 3^{2t}$. We now show that for all i, $V(F_{i+1}) \geq V(F_i) + 1$, thus proving the lemma.

If F_{i+1} is obtained from F_i by the operation *ADD*, then $V(F_{i+1}) = V(F_i) + V(H(D)) = V(F_i) + 1$. Suppose F_{i+1} is obtained from F_i by a *LEFT-JOIN* operation (The case of a *RIGHT-JOIN* operation is similar). Now note that the size of the inner part of $\hat{H}$, the segment $LI_2' \cdot h_C' \cdot RI'$, is smaller than the size of the inner part of H' and thus also smaller than the size of the inner part of H. This implies that $V(\hat{H}) \geq 3V(H')$ and $V(\hat{H}) \geq 3V(H)$, so $V(F_{i+1}) = V(F_i) + V(\hat{H}) - V(H) - V(H') \geq V(F_i) + 1$. □

Let $F(\tau)$ be the last set of any maximal sequence of history sequences as described in the statement of Lemma 3.1. Note that $F(\tau)$ is not defined uniquely. In the next lemma we show that besides Properties (a), (b) and (c) above, the following additional properties hold for any history sequence H in $F(\tau)$.

Property (d): Both *LE* and *RE* are of size at most $\frac{1}{6}n$, and all the link-histories in *LE* [*RE* respectively] are distinct.

Property (e): Any segment of size $\leq n$ in $LI \cdot h_C \cdot RI$ contains at most $\frac{1}{12}n$ distinct histories, and all the identities in it are distinct.

All properties are proven by induction on the construction of $f(\tau)$, i.e. if they hold before an operation is applied then they also hold afterwards.

Lemma 3.2: Every sequence H in $F(\tau)$ satisfies properties (a) - (e).

Proof:

Property (a): Whenever a new history is added to $F(\tau)$ by the operation *ADD* then Property (a) follows from the definition of canonical history sequence. The operations $LEFT-JOIN$ and $RIGHT-JOIN$ preserve Property (a) by the correctness of Rule 2.

Property (b1) and Property (b2): Again, *ADD* trivially maintains these properties and from the conditions of the *JOIN* operations it follows that these operations also maintain Property (b1) and Property (b2).

Property (c): This follows directly from the construction of $F(\tau)$.

Property (d): When a new history sequence is constructed by a $LEFT-JOIN$ operation, the *LE*-segment of it is produced by maximally shrinking an *LE*-segment of size at most $\frac{1}{12}n$ and part of an $LI-$segment of size at most n. The latter part is contained in a canonical history sequence. Therefore by Assumption Q it contains at most $\frac{1}{12}n$ distinct histories. We conclude that after maximal shrinking the new *LE*-segment has size at most $\frac{1}{6}n$. A similar argument shows that the *RE*-segments of histories produced by a $RIGHT-JOIN$ are at most of size $\frac{1}{6}n$. The distinctness of the link-histories in *LE* and in *RE* is implied by the properties of maximal shrinking.

Property (e): The first part follows from Assumption Q and the fact that there exists a canonical line configuration that contains $LI \cdot h_c \cdot RI$ as a segment. The second follows from property (b1) in the definition of $F(\tau)$. □

4. PROOF OF THEOREM 1.

Let F be $F(\sigma)$ for $\sigma \in \{\tau,\omega\}$. We call a history sequence of F *finished* if a certain condition (defined below) holds. We then show in the Main Lemma that the existence of such *finished* history sequences in both $F(\tau)$ and $F(\omega)$ implies the $\Omega(n\log n)$ lower bound. Finally we prove by a counting argument that if either $F(\tau)$ or $F(\omega)$ does not contain a finished history sequence and the size of the identity set is $\Omega(n^{1+\varepsilon})$, then the $\Omega(n\log n)$ lower bound hold as well.

A history sequence $H = LE{\cdot}h_L{\cdot}LI{\cdot}h_C{\cdot}RI{\cdot}h_R{\cdot}RE$ in F is called *left unfinished (right unfinished)* if LI (RI) is of size at least $\frac{1}{12}n$ and LE (RE) is of size at most $\frac{1}{12}n$. A history sequence is called *finished* iff it is neither left unfinished nor right unfinished.

Main Lemma: If both $F(\tau)$ and $F(\omega)$ contain a finished history sequence then the message complexity of AL is $\Omega(n\log n)$.

Proof: If $H = LE{\cdot}h_L{\cdot}LI{\cdot}h_C{\cdot}RI{\cdot}h_R{\cdot}RE$ is finished then at least one of the following must hold:

(A) Both LI and RI have size smaller than $\frac{1}{12}n$.

(B) Both LE and RE have size larger than $\frac{1}{12}n$.

(C) The size of RE is larger than $\frac{1}{12}n$ and the size of LI is smaller than $\frac{1}{12}n$.

(D) The size of RI is smaller than $\frac{1}{12}n$ and the size of LE is larger than $\frac{1}{12}n$.

To prove the lemma it suffices to show that if both $F(\tau)$ and $F(\omega)$ contain a history sequence satisfying one of the properties (A)-(D) above, then the bit complexity of AL is $\Omega(n\log n)$. This is done in the next 3 lemmas.

Lemma 4.1: If a history sequence H in $F(\tau)$ satisfies (A), then no history sequence in $F(\omega)$ satisfies (A), and vice versa.

Proof: Assume that the lemma is false. Then there is a history sequence $H_\tau = LE{\cdot}h_L{\cdot}LI{\cdot}h_C{\cdot}RI{\cdot}h_R{\cdot}LE$ in $F(\tau)$ that satisfies (A). So the size of the segment $LI{\cdot}h_C{\cdot}RI$ is at most $\frac{1}{6}n$ and all identities in this segment (and thus in all of H_τ) are distinct. By Property (d) the size of LE and of RE is at most $\frac{1}{6}n$. Clearly the total size of H_τ is at most $\frac{1}{2}n$. By Property (b1), in the execution $E(H_\tau)$ the center processors output $f(\tau)$. Similarly, there is a history sequence H_ω in $F(\omega)$ that has size at most $\frac{1}{2}n$ and in the execution $E(H_\omega)$ the center processors output $f(\omega)$.

Observe that all identities occurring in H_τ and H_ω are distinct. Thus it is possible to embed the line configurations of these history sequences into a ring configuration $R = R(\sigma,x)$ of size n: Concatenate the processors of H_τ and H_ω and close the line to a ring of size n by adding the needed number of processors. Now repeat both executions $E(H_\tau)$ and $E(H_\omega)$ on the corresponding segments of R (block all links not contained in the two segments). We get an execution of AL on a ring of size n with distinct identities taken from X in which some processors output $f(\tau)$ and another output $f(\omega)$. This is a contradiction. $\square$

In view of the above lemma, we may assume without loss of generality, that $F(\tau)$ does not contain a history sequence satisfying (A). In the sequel we denote $F(\tau)$ by F and show that no history sequence in F satisfies any of the conditions (B) to (D), unless the lower bound holds. Clearly, this completes the proof of the Main Lemma.

Lemma 4.2: If a history sequence H in F satisfies (B), then the bit complexity of AL is $\Omega(n\log n)$.

Proof: Let $H = LE{\cdot}h_L{\cdot}LI{\cdot}h_C{\cdot}RI{\cdot}h_R{\cdot}RE$ be a history sequence satisfying (B). Assume first that $LI{\cdot}h_C{\cdot}RI$ is of size at most n. Then all the identities in H are distinct; moreover, by (B) and by Property (d), the number of distinct histories in both LE and RE is at least $\frac{1}{12}n$ and at most $\frac{1}{6}n$. By Assumption Q, the number of distinct histories in $LI{\cdot}h_C{\cdot}RI$ is at most $\frac{1}{12}n$. Thus, by applying maximal shrinking to $LI{\cdot}h_C{\cdot}RI$ we get a history sequence $\hat{H}$ whose size is at most $\frac{5}{12}n$, and which contains at least $\frac{1}{12}n$ distinct histories. By Lemma 2.3, the cost of $\hat{H}$ is $\Omega(n\log n)$. Since $\hat{H}$ is of size smaller than n, it can be embedded in a ring configuration R of size n. Thus, we have an execution of AL on R whose bit complexity is $\Omega(n\log n)$.

We are left with the case where the size of the inner part of H is larger than n. In this case we use a construction depicted in Figure 2.

Let $p = p_{i,j}$ be the leftmost processor in LI, and let q be the rightmost processor in H which has the same identity as p (note that q is either $p_{i,j'}$ or $p'_{i,j'}$ for some j'). Let h be the link-history of the link to the left of q. Then H can be written as $LE{\cdot}h_L{\cdot}H1{\cdot}h{\cdot}H2{\cdot}h_R{\cdot}RE$ (see Figure 2(a)). Observe that all identities in LE and $H2{\cdot}h_R{\cdot}RE$ are distinct. Let $\overline{H2}$ be the segment produced by maximal shrinking from $H2$. Replace $H2$ by $\overline{H2}$ in H to get the history sequence $\overline{H} = LE{\cdot}h_L{\cdot}H1{\cdot}h{\cdot}\overline{H2}{\cdot}h_R{\cdot}RE$ (see Figure 2(b)). By the definition of q, the segment $H2$ is of size at most n. Thus by Assumption Q it contains at most $\frac{1}{12}n$ distinct histories and hence $\overline{H2}$ is of size at most $\frac{1}{12}n$.

By Property (b1) of the history sequences in $F(\tau)$ there is a canonical line configuration D, that contains $h_L{\cdot}LI{\cdot}h_C{\cdot}RI$ as a consecutive subsequence, and in which p and q have the same identity and input. This implies that h_L is a prefix of h or vice versa, so assume that h_L is a prefix of h. We use this to produce a history sequence of size less than n that contains LE. Then we use the fact that LE has at least $\frac{1}{12}n$ distinct histories to derive the lower bound (In the symmetric case h is a prefix of h_L and one can construct a history sequence of size less than n that contains RE). By using Rule 1 on h and h_L we get a history sequence $H' = LE' \cdot h_L' \cdot H1' \cdot h_L \cdot H2' \cdot h_R' \cdot RE'$ (see Figure 2(c)) that is produced by some execution of AL. By using Rule 2 on H and H' (with $h = h_L$) we get a history sequence $\hat{H} = LE \cdot h_L \cdot H2' \cdot h_R' \cdot RE'$ (see Figure 2(d)), that is produced by another execution of AL. In $\hat{H}$ all the identities are distinct. The sizes of LE and RE' are at most $\frac{1}{6}n$, and the size of $H2'$ is at most $\frac{1}{12}n$ and thus the total size of $\hat{H}$ sums to at most $\frac{5}{12}n$. As above we embed $\hat{H}$ in a ring of size n and run the execution $E(\hat{H})$ on the segment $\hat{H}$ of the ring. $\hat{H}$ contains the segment LE which has at least $\frac{1}{12}n$ distinct histories. Thus by Lemma 2.3 the cost of LE is $\Omega(n \log n)$ and this completes the proof of the lemma. □

Lemma 4.3: If a history sequence H in F satisfies (C) or (D), then the bit complexity of AL is $\Omega(n \log n)$.

Proof: Let $H = LE \cdot h_L \cdot LI \cdot h_C \cdot RI \cdot h_R \cdot RE$ be a history sequence satisfying (C) (the other case is similar). The proof follows the same outline of the proof of Lemma 4.2, and is only sketched:

Assume first that $LI \cdot h_C \cdot RI$ is of size at most n. In this case the proof is identical to the proof of the analogous case in Lemma 4.2. Assume now that the size of the inner part of H is larger than n. Let p, q, and h be as in Lemma 4.2. Then H can be written as $LE \cdot h_L \cdot H1 \cdot h \cdot H2 \cdot h_R \cdot RE$, as in the proof of Lemma 4.2. However, in order to make the same technique work here, we *must* have that h is a prefix of h_L and not vice versa (as only the size of RE and not the size of LE is known to be at least $\frac{1}{12}n$). Let e_L be the link that has link-history h_L and e be the link that has link-history h. By the definition of canonical history sequences, the fact that h is a prefix of h_L holds if e_L is closer to the left center processor than e is to the right center processor, since this means that in the corresponding canonical execution e was blocked before e_L.

By (C), the size of LI is smaller than $\frac{1}{12}n$, and hence e_L is at distance at most $\frac{1}{12}n$ from the left center. Since there are at least n processors between e_L and e, e is distance at least $\frac{11}{12}n$ from the right center, which implies that h is a prefix of h_L. This completes the sketch of the proof of Lemma 4.3 and proof of the Main Lemma. □□

The Main Lemma aboves implies that if both $F(\tau)$ and $F(\omega)$ contain a finished history sequence, then Theorem 1 holds. Thus, in order to complete the proof of Theorem 1, it is suffices to prove the following:

Lemma 4.4: If $F(\tau)$ or $F(\omega)$ contains only unfinished history sequences, then the bit complexity of *AL* is $\Omega(n\log n)$.

Proof: We prove the lemma for $F = F(\tau)$. First observe that there are at most $n-1$ identities in X_τ that do not occur in any history sequence *H* in *F* (otherwise Operation (i) is applicable to *F*, in contrast with the definition *F*). Also, each history sequence in *F* contains at most $4n/3$ distinct identities: at most *n* in *LI* and *RI* and at most $n/3$ in *LE* and *RE*. Hence, there are at least $M = \frac{3(|X_\tau|-n+1)}{4n}$ distinct history sequences in *F*. At least half of the history sequences of *F* are either right or left unfinished. Without loss of generality, assume that there are at least $K = \frac{1}{2}M$ distinct history sequences in *F* that are left unfinished. Note that $\log(K) = \Omega(\log n)$.

Let $H_1, \cdots, H_K$ be the history sequences in *F* which are left unfinished. Then the left inner part of each H_i is of size at least $\frac{1}{12}n$. Let Q_i be the set of the link-histories of the first $\frac{1}{12}n$ links of the inner part of H_i. By the definition of *F*, the operation *LEFT-JOIN* can be applied to no pair of these H_i's, and hence the sets Q_i are disjoint. Let l_i be the minimal cost of a link-history in Q_i, and let *j* be such that $l_j = \max\{l_i : 1 \le i \le K\}$. By Lemma 2.3, $l_j = \Omega(\log K) = \Omega(\log n)$, which means that the inner part of Q_j contains a history segment of size $\frac{1}{12}n$ and of cost $\Omega(n\log n)$. By Lemma 2.2, the cost of this segment is a lower bound on the bit complexity of the synchronized computation of the ring *R* that corresponds to the inner part of H_j. This completes the proof of the lemma, and hence the proof of Theorem 1. □□□

Acknowledgements: We would like to thank the referees for their many helpful remarks.

REFERENCES

[AAHK] K. Abrahamson, A. Adler, L. Higham and D. Kirkpatrick, "Randomized Function Evaluation on a Ring," Distributed Computing Vol. 3, No. 3, pp. 107-117 (1989).

[ASW] C. Attiya, M. Snir and M. K. Warmuth, "Computing on an anonymous ring," JACM Vol. 35, No. 4, pp. 845-875 (1988).

[B1] H. L. Bodlaender, "A New Lower Bound Technique for Distributed Leader Finding and Other Problems on Rings of Processors," Technical Report RUU-CS-88-18, University of Utrecht, April 87. To appear in Theoretical Computer Science.

[B2] H. L. Bodlaender, unpublished note.

[BB] P. W. Beame and H.L. Bodlaender, "Distributed Computing on Transitive Networks: The Torus", proc. of the 6th STACS, pp. 294-303 (1989).

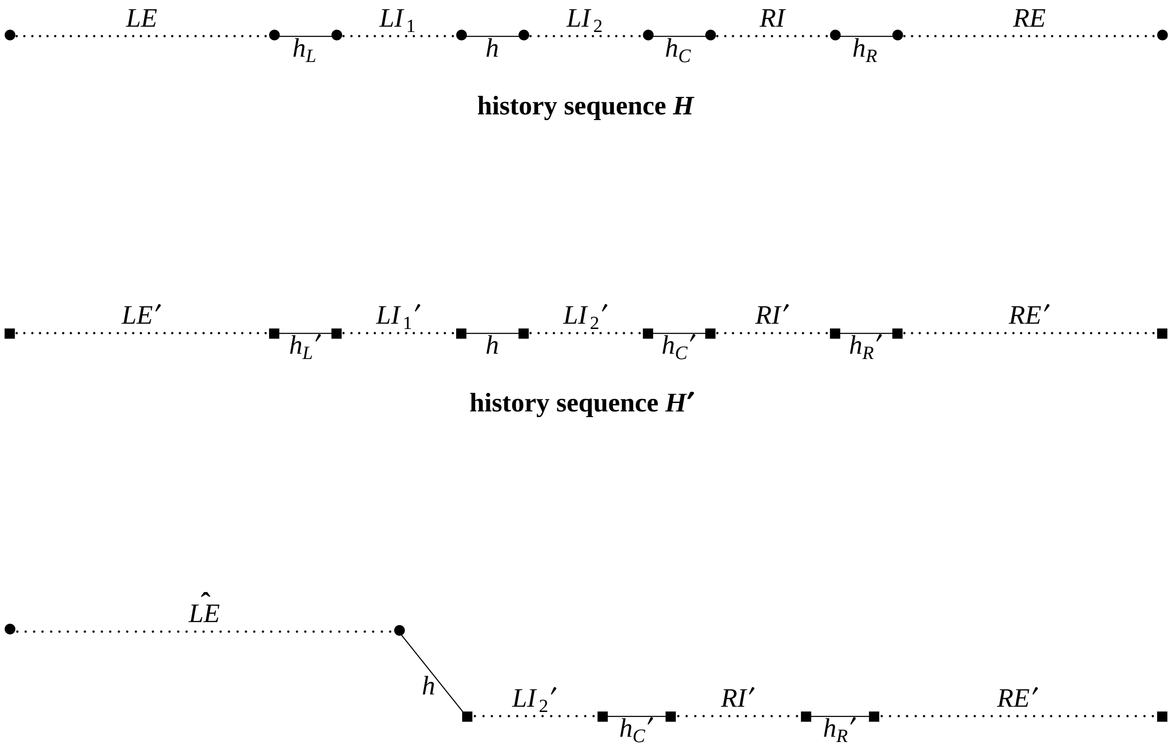
[DG] P. Duris and Z. Galil, "Two lower bounds in asynchronous distributed computation," proc. of the 28th FOCS, pp. 326-330 (1987).

[MZ] Y. Mansour and S. Zaks, "On the bit complexity of distributed computations in a ring with a leader," Information and Computation, Vol. 75, No. 2, 1987, pp. 162-177.

[MW] S. Moran and M. Warmuth, "Gap theorems for distributed computation," proceedings of the 5th PODC, pp. 131-140 (1986).

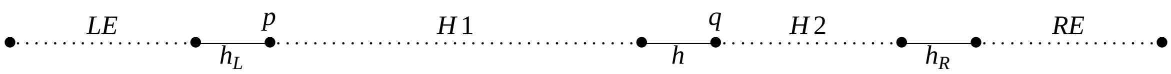
[PKR] J. Pachl, E. Korach and D. Rotem, "Lower bounds for distributed maximum-finding algorithms," JACM Vol. 31, pp. 905-918, 1984.

[R] K. Reidemeister, Einfuhrung in die Kombinatorische Topologie, Friedr. Vieweg & Sohn Akt. Ges., Braunschweig, 1932.

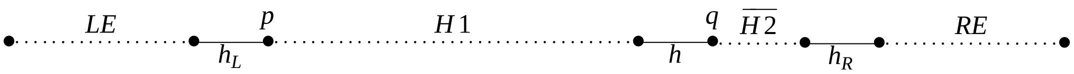


LEFT JOIN*, $\hat{H}$, of *H* and *H′
($\hat{LE}$ is obtained by maximal shrinking of $LE{\cdot}h_L{\cdot}LI_1$.)

Figure 1: *LEFT JOIN*

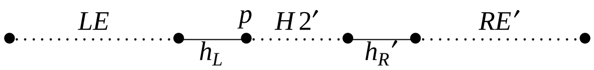


(a) history sequence H



(b) history sequence $\overline{H}$, obtained by maximal shrinking of $H2$

(c) history sequence H', obtained by applying Rule 1 on $\overline{H}$



(d) history sequence $\overline{H}$, obtained by applying Rule 2 to H and H'

Figure 2: The construction of Lemma 4.2.