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

Hans L. Bodlaender
Shlomo Moran
Manfred K. Warmuth

RUU-CS-88-33
November 1988

Rijksuniversiteit Utrecht

Vakgroep informatica

Padualaan 14 3584 CH Utrecht
Corr. adres: Postbus 80.089, 3508 TB Utrecht
Telefoon 030-531454
The Netherlands

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

Hans L. Bodlaender
Shlomo Moran
Manfred K. Warmuth

Technical Report RUU-CS-88-33
November 1988

Department of Computer Science
University of Utrecht
P.O. Box 80.089, 3508 TB Utrecht
The Netherlands

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

Hans L. Bodlaender *
Department of Computer Science, University of Utrecht,
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:

The *distributed bit complexity* of an asynchronous network of processors is a lower bound on the worst case bit complexity of computing any non-constant function of the inputs to the processors [MW]. This concept attempts to capture the amount of communication required for any "useful" computation on the network.

The aim of this kind of research is to characterize networks by their bit complexity. In [MW] Moran and Warmuth studied the bit complexity of a ring of n processors under the assumption 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. It was shown that for anonymous rings it takes $\Omega(n \log n)$ bits to compute any non-constant function. We would like to release the assumption that the processors are anonymous by allowing each of the processors to have a distinct identity (which becomes a second parameter to the program). If the set of possible identities grows double exponentially in n, then by a simple reduction to the anonymous case one can show that the lower bound holds as well [MW].

In this paper we show that the $\Omega(n \log n)$ bit lower bound for computing any non-constant function holds even if the set of possible identities is very small, that is, $n^{1+\varepsilon}$, 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 by Technion V.P.R. Fund - C. Wellner Research Fund. 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. The aim of this line of research is to establish lower bounds on the amount of resources (complexity) required to compute any non-constant function on a given network. The lower bound must hold for any algorithm that computes the non-constant function.

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]. In this paper we give results about the distributed bit complexity of a ring of n processors.

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" between the processors. Three cases are considered in the literature: In the first the processors are anonymous (i.e. they have no id's). In the second each processor has a distinct id. The id's 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.

In the case where the network has a leader then this leader can coordinate the computation. It can first instigate messages that collect the global information that is necessary to compute the non-constant function. Secondly, it will 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 used in practice. Previous research and this paper concentrate on the distributed complexity of the ring of n processors. It leads to an interesting case study as to how the processors may avoid symmetry.

The distributed bit complexity of a ring of n anonymous processors is $\Theta(n \log n)$ [MW]. This holds for arbitrary ring size and input size. However the lower bound proof of $\Omega(n \log n)$ in [MW] assumes that the processors are anonymous or that the processors have distinct identities but the set of possible identities is very large ($\Omega(n 2^{n^n})$, which is unrealistically large).

If the set of distinct 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 (use processor with identities in $\{1,2,\cdots,c+1\}$ as leaders). We show in this paper that the approach of giving processors distinct identities from a reasonably large set does not help to break the $\Omega(n \log n)$ lower bound.

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 ring of n processors, labeled with n distinct identities chosen from a set X of size at least $n^{1+\epsilon}$, for some $\epsilon > 0$. Then the worst case bit complexity of AL is $\Omega(n \log n)$.

Thus 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, we use "cut-and-paste" techniques to construct a computation of AL in which $\Omega(n \log n)$ bits are sent. cut-and-paste method was first used in [MW] for the anonymous case, where the processors have no id's. When the processors have id's then cutting and pasting is more involved, since in the final ring constructed all processors must have distinct id's from a small set of possible id's ($O(n^{1+\epsilon})$). 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 id's, and a computation of AL on this line requires $\Omega(n \log n)$ bits.

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

The main future challenge will be to determine the distributed bit complexity of other networks, such as the hypercube. Recently, the distributed bit complexity of the 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] and 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.

A *processor* consists of an input letter and an id. Let Σ be the input alphabet, which is allowed to be arbitrary large. Let X be a set of id's, and suppose that $|X| \geq n^{1+\varepsilon}$, for some constant $\varepsilon > 0$. Let $\sigma = \sigma_1 \cdots \sigma_n \in \Sigma^n$ be an input word and $x = (x_1, \cdots, x_n)$ be a sequence of n distinct identities taken from X. A *ring configuration* $R(\sigma, x)$ consists of processors $p_1, \cdots, p_n$, where p_i is connected by a link to $p_{i \pmod n + 1}$, (for $i = 1, \cdots, n$), and processor p_i has input σ_i and id x_i. A *line configuration* $L(\sigma, x)$ is defined similarly, except that there is no link between p_1 and p_n; informally, a line configuration can be viewed as a ring configuration in which there is an infinite delay on the link connecting p_1 and p_n. 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 $L(\tau\omega, xy)$.

Consider an execution of some distributed algorithm on a ring or a line configuration. The *history* of a link e in this execution, denoted by $h(e)$, is the sequence of messages delivered on e with their directions. Formally, $h(e) = (d_1 m_1 d_2 m_2 \cdots d_s m_s)$, 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 *length* of a history is the number of characters in it. Note that the length of a history of a link 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 $H(L) = (p_1,h_1,p_2,h_2,\cdots,h_{n-1},p_n)$, where the p_i's are the processors in L and h_i is the history of the link connecting p_i and p_{i+1}. *A segment of size m* of the history sequence $H(L)$ defined above is a sequence $(p_i,h_i,\cdots,h_{i+m-2},p_{i+m-1})$ $(i+m-1 \le n)$. The *length* of a segment of a history sequence is the sum of the lengths of the histories of its links.

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 history. Then $H_1 \cdot h \cdot H_2$ denotes the segment $(p_1,h_1,\cdots,h_{n-1},p_n,h,q_1,h'_1,\cdots,h'_{m-1},q_m)$. Finally, we say that a history sequence H of a line configuration L is *produced* by the algorithm AL if there is an execution of AL on L with history H.

A basic tool in our proof is converting executions of AL on ring configurations to executions on line configurations and vice-versa. Since 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, we may choose particular delay times for the proofs. The basic delay strategy [MW], here called *semi-synchronized execution*, is an execution in which internal computation at a processor takes no time 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 indefinitely.

Consider an execution of AL on $R(\sigma,x)$ which is (fully) synchronized (no link is blocked), and assume that this execution terminates in less than t time units. Without loss of generality, let $t = nk$ for some integer k. 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 $2t = 2nk$ 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 the $R(\sigma,x)$ 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 (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, to be denoted by $H(D)$, is the history sequence produced by to 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 a canonical execution of $D(\sigma,x)$, both $p_{n,k}$ and $p'_{1,1}$ output $f(x)$ and terminate. □

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 bounded from below by half of the history length of any segment of size n of H.

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 length of the history of a link is at most twice the number of bits delivered on it. □

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 the overall length of the words W_i is minimized. Then in the corresponding tree all internal nodes except possible one internal node of maximum level have degree r. 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 $\log_r v$. □

The above two lemmas imply the following.

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

In view of the above lemma, we assume the following for the rest of the paper.

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

In the below proofs we manipulate existing history sequences to create new ones. Two basic rules are used in our manipulations, which are presented below.

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 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. On a certain moment during this execution, the 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 history of the j'th link will be a prefix of h_j. □

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 $L = L_1L_2$ and $L' = L'_1L'_2$, where h is the 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 $\hat{L} = L_1L'_2$.

Proof: $\hat{E}$ is obtained by alternating the execution E, restricted to L_1, and E' restricted to L'_2. Suppose $h=(d_1m_1,\cdots,d_sm_s)$. 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 $\hat{L} = L_1L'_2$. □

One specific way in which Rule 2 above will be used is the *maximal shrinking*: Consider a history sequence $H = H_1{\cdot}h{\cdot}H_2{\cdot}h{\cdot}H_3$ that 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 histories are distinct (A similar technique was also used in [MW]). If a history segment H is produced by some execution of AL on the corresponding line of processors, then $E(H)$ denotes one such execution.

3. CONSTRUCTING FAMILIES OF CONTRADICTING COMPUTATIONS.

Let τ and ω be two input configurations s.t. $f(\tau) \neq f(\omega)$. We will construct two families of history sequences $F(\tau)$ and $F(\omega)$ that correspond to executions of AL on τ and ω, resp. The identities of the processors in the history sequences in $F(\tau)$ and $F(\omega)$ will belong to sets X_τ and X_ω, resp., where $X_\tau \cap X_\omega = \Phi$, $X_\tau \cup X_\omega = X$ and $0 \leq |X_\tau| - |X_\omega| \leq 1$. $F(\tau)$ and $F(\omega)$ will have similar properties. The description and construction of $F(\tau)$ is given below:

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

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

Property (b): H can be written as $H = LE \cdot h_L \cdot LI \cdot h_C \cdot RI \cdot h_R \cdot RE$, where LE, LI, RI and RE are segments of history sequences connected by links having histories h_L, h_C and h_R, satisfying the following:

Property (b1): There is a canonical line configuration $D = D(\tau, x)$, where $x \in X_\tau^n$, 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 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 id of a processor in $LE \cdot RE$ is also an id of a processor in $LI \cdot h_C \cdot RI$.

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 the set $F(\tau)$, we construct a list $F = (F_0, \cdots, F_i, \cdots)$, 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. (Note that $H(D)$ can be written as $LI \cdot h_C \cdot RI$, where h_C is the 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 histories H and H' iff they satisfy the following conditions:

(a) The identities appearing in H are distinct from those appearing in H'.

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

(c) There is a 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.

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

The *LEFT-JOIN* of H and H' is the history sequence $\hat{H} = L\hat{E} \cdot h \cdot LI_2' \cdot h_C' \cdot RI' \cdot h_R' \cdot RE'$, where $L\hat{E}$ 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 the 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 $F_0 = \Phi$ and F_{i+1} is obtained by applying one of the operations (i) - (iii) above to F_i. Then $\mathbf{F}$ is finite.

Proof: Recall the size of the canonical histories is $2t$. Define the *cost* $C(H)$ 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 cost $C(F)$ of F is defined as the sum of the costs of the history sequences in it. Since each id can occur in at most one history sequence in F_i, it follows that F_i contains at most $n^{1+\epsilon}$ history sequences. Hence $C(F_i)$ is bounded from above by $n^{1+\epsilon} \cdot 3^{2t}$. We now show that for all i, $C(F_{i+1}) \geq C(F_i) + 1$, thus proving the lemma.

If F_{i+1} is obtained from F_i by the operation *ADD*, then $C(F_{i+1}) = C(F_i) + C(H(D)) = C(F_i) + 1$. 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 $C(\hat{H}) \geq 3C(H')$ and $C(\hat{H}) \geq 3C(H)$, so $C(F_{i+1}) = C(F_i) + C(\hat{H}) - C(H) - C(H') \geq C(F_i) + 1$. □

Let $F = F(\tau)$ be a set F_N satisfying the above lemma. Some basic properties of history sequences in F for are given in the next lemma.

Lemma 3.2: For $H = LE \cdot h_L \cdot LI \cdot h_C \cdot RI \cdot h_R \cdot RE$ be in $F(\tau)$ the following properties hold:

(a) Both LE and RE are of size at most $\frac{1}{6}n$.

(b) Any segment of size $\leq n$ in $LI \cdot h_C \cdot RI$ contains at most $\frac{1}{12}n$ distinct histories.

(c) Any identity occurring in H does not occur in any other history sequence in F.

(d) There is an execution $E(H)$ of AL.

Proof:

(a) When a new history is constructed by a *LEFT–JOIN* operation, then the LE-segment of the new history is produced by maximally shrinking an LE-segment of size at most $\frac{1}{12}$ 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 $\frac{1}{12}$ 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 $\frac{1}{6}n$.

(b) This 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.

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

(d) This follows from the construction of $F(\tau)$ and the correctness of Rule 2. □

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 id set is large enough ($n^{1+\epsilon}$), then the same lower bound must hold.

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:

(ir1) Both LI and RI have size smaller than $\frac{1}{12}n$.

(ir2) Both LE and RE have size larger than $\frac{1}{12}n$.

(ir3) The size of RE is larger than $\frac{1}{12}n$ and the size of LI is smaller than $\frac{1}{12}n$.

(ir4) 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 is suffices to show that if both $F(\tau)$ and $F(\omega)$ contain a history sequence satisfying one of the properties (ir1)-(ir4) 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 (ir1), then no history sequence in $F(\omega)$ satisfies (ir1), 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 (ir1). 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 Lemma 3.2(a) 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 id's 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. □

In view of the above lemma, we may assume without loss of generality, that $F(\tau)$ does not contain a history sequence satisfying (ir1). In the sequel we denote $F(\tau)$ by F and show that no history in F satisfies any of the

conditions (ir2) to (ir4), 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 (ir2), 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 (ir2). 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 (ir2) the number of distinct histories in both LE and RE is at least $\frac{1}{12}n$, and by Lemma 3.2 (a) this number is 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 contain at least $\frac{1}{12}n$ distinct histories. By Lemma 2.3, the length 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 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 id's 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. 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 $\frac{1}{12}$ 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 $\frac{1}{12}$ distinct histories. Thus by Lemma 2.3 the length 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 (ir3) or (ir4), 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 (ir3) (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 analogue 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 history h_L and e be the link that has 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 (ir3), 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). Since each history sequence in F contains less than $2n$ distinct identities, there are at least $M = \frac{|X_\tau| - n + 1}{2n}$ 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 unfin-

ished. Note that $K = n^{\varepsilon'}$ for some $\varepsilon' > 0$.

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 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 length of the minimal length history in Q_i, and let j be such that $l_j = \max\{l_i : 1 \leq i \leq K\}$. 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 length $\Omega(n \log n)$. By Lemma 2.2, the length 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 complete the proof of the lemma, and hence the proof of Theorem 1. □□□

REFERENCES

[AAHK] K. Abrahamson, A. Adler, L. Higham and D. Kirkpatrick, "Randomized Function Evaluation on a Ring," Tech. Rep. 87-20, Dept. of Comp. Sc., Univ. of British Columbia, Vancouver, Canada, 1987.

[ASW] C. Attiya, M. Snir and M. K. Warmuth, "Computing on an anonymous ring," extented abstract in Proceedings PODC 1985, p. 161-173, 1985, to appear in JACM, October 1988.

[B1] H. L. Bodlaender, "A new lower bound technique for distributed extrema finding on rings of processors," Technical Report RUU-CS-87-11, University of Utrecht, August 87.

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

[BB] P. W. Beame and H.L. Bodlaender, Distributed Computing on Transitive Networks: The Torus, Techn. Rep. RUU-CS-88-31, Dept. of Comp. Sc., Univ. of Utrecht, 1988. To appear in proceedings STACS 89.

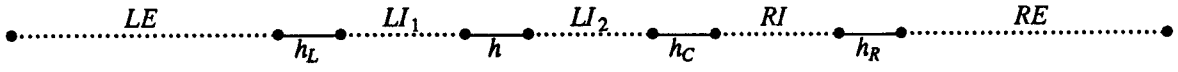
[DG] P. Duris and Z. Galil, "Two lower bounds in asynchronous distributed computation," proceedings FOCS 87, p. 326-330, 1987.

[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.

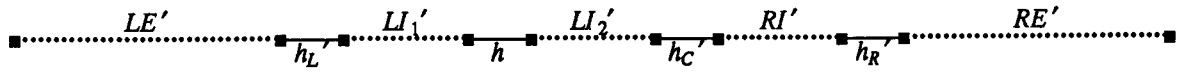
[MW] S. Moran and M. Warmuth, "Gap theorems for distributed computation," proceedings PODC, p. 131-140, 1986.

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

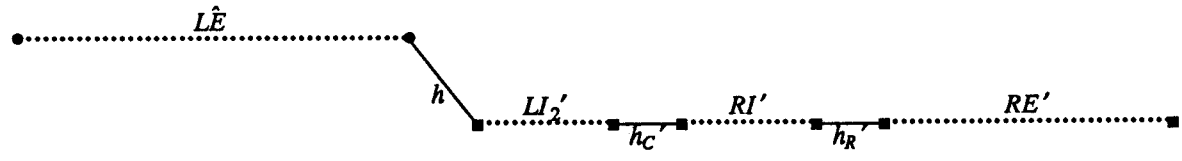
[R] K. Reidemeister, Einfuhrung in die Kombinatorische Topologie, Friedr. Vieweg & Sohn Akt. Ges., Braunschweig, 1932.



history line *H*

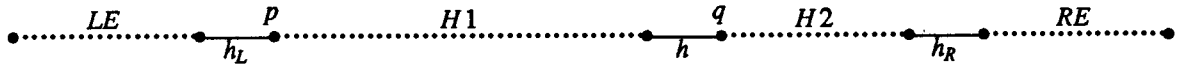


history line *H'*

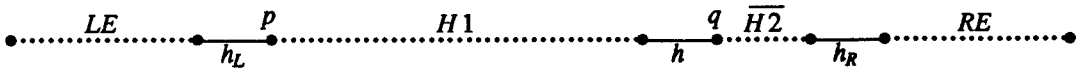


***LEFT JOIN*, $\hat{H}$, of H and H'**
($L\hat{E}$ is obtained by maximal shrinking of $LE \cdot h_L \cdot LI_1$.)

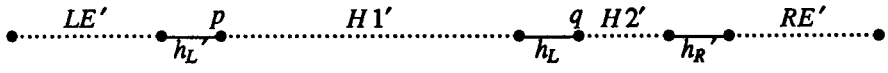
Figure 1: *LEFT JOIN*



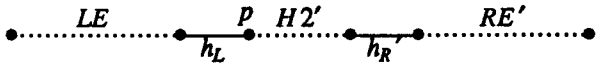
(a) history line H



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



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



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

Figure 2: The construction of Lemma 4.2.