JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING **46,** 80–90 (1997)
ARTICLE NO. PC971389

Optimal Distributed Algorithms in Unlabeled Tori and Chordal Rings[1,2]

Bernard Mans[3]

Department of Computing, School of Mathematics, Physics, Computing, and Electronics, Macquarie University, Sydney, New South Wales 2109, Australia

We study the message complexity of distributed algorithms in tori and chordal Rings when the communication links are unlabeled, which implies that the processors do not have "sense of direction." We introduce the paradigm of *handrail* which allows messages to travel with a consistent direction. We give a distributed algorithm which confirms the conjecture that the leader election problem for unlabeled tori of N processors can be solved using $\Theta(N)$ messages instead of $O(N \log N)$. Using the same *handrail* paradigm, we solve the election problem using $\Theta(N)$ messages in unlabeled chordal rings with one chord (of length approximately $\sqrt{N}$). This solves the long-standing open problem of the minimal number of unlabeled chords required to decrease the $O(N \log N)$. message complexity. For each topology, we give an algorithm to compute the sense of direction in $\Theta(N)$ messages (improving the $O(N \log N)$ previous results). This proves the more fundamental result that any global distributed algorithm for these labeled topologies can be used with a similar asymptotic complexity in the respective unlabeled class. © 1997 Academic Press

1. INTRODUCTION

One of the main themes of investigation in distributed computation concerns the identification of the factors which are significant for the computability and the communication complexity of problems. Recently, it has become clear that an important role is played by structural information; that is, a priori knowledge available to the entities about the structure of the system.

In particular, *sense of direction* (denoted as $\mathcal{SD}$) has been identified as being significant for computability and communication complexity [8]. Informally, $\mathcal{SD}$ refers to the capability of a processor to distinguish between adjacent communication links, according to some globally consistent scheme [20]. More formally, it represents the ability of the processors to adequately combine the relationship between the labeling, the topological structure, and the local view of the system that an entity has [7].

In this paper (extended from [14]), we address the message complexity of distributed algorithms in tori and in chordal rings when the communication links are unlabeled, which implies that processors do not have a globally consistent labeling of the communication links. They have no "sense of direction" but have only a topological awareness.

We first study the impact of structural information limited to topological awareness on the *leader election* problem (the problem of moving the system from an initial situation where the nodes are in the same computational state to a final situation where exactly one node is in a distinguished computational state, called *leader*, and all the others are in the same state, called *defeated*). The election process may be started independently by any subset of the processors. It is assumed that every processor P_i has a distinct identity id_i chosen from some infinite totally ordered set ID, and is only aware of its own identity (in particular, it does not know those of its neighbors).

Leader election is a widely used solution for distributed algorithms requiring one process to act as a coordinator, initiator, or sequencer, or otherwise to perform some special role. The election problem occurs, for instance, in token-passing when the token is lost or the owner has failed; in this a case, the remaining processors elect a leader to issue a new token. Several other problems encountered in distributed systems can be solved by election; for example; *crash recovery* (a new server should be found to continue the service when the previous server has crashed), *mutual exclusion* (where values for election can be defined as the last time the process entered the critical section), and *group server* (where the choice of a server for an incoming request is made through an election between all the available servers managing a replicated resource).

The lack of sense of direction is known to increase the message complexity of the election problem. For instance, in arbitrary networks, the election problem requires $\Theta(e + N \log N)$ messages [9], instead of $\Theta(N \log N)$ [15]. In the complete graph, the complexity increases from $\Theta(N)$ [13] to $\Theta(N \log N)$ messages [11]. In the torus, an $O(N)$ messages algorithm with sense of direction has been given [18]. It was conjectured that the problem can be solved with

[1]This research was supported by the Macquarie University Research Grant "Structural Information in Interconnection Networks."

[2]Some of the results of this work were presented at the 3rd International Colloquium on Structural Information and Communication Complexity (SIROCCO '96), Certosa di Pontignano, Siena, Italy, June 6–8, 1996.

[3]E-mail: bmans@mpce.mq.edu.au.

linear complexity when sense of direction is not available. For chordal rings, the main open problem is to determine the minimal sets of links that must be added to the ring (which requires $\Theta(N \log N)$ messages) in order to achieve a linear election algorithm. Along the years, a succession of papers [2, 10, 17] finally proved that a unique chord was sufficient [22], but without sense of direction no result better than $O(N \log N)$ messages was known.

In Section 2, we give a distributed algorithm which confirms the conjecture suggested in [18] that the leader election problem for unlabeled tori of N processors can be solved using $\Theta(N)$ messages instead of $O(N \log N)$. In Section 3, we solve the election problem using a $\Theta(N)$ message complexity for the unlabeled chordal ring with one chord (of length approximately $\sqrt{N}$). Finally, in Section 4, we give a distributed algorithm to compute the sense of direction in $\Theta(N)$ messages for each topology, improving the $O(N \log N)$ previous results, and Section 5 concludes by showing the relationship between solutions for labeled and unlabeled networks.

2. DISTRIBUTED ALGORITHMS FOR UNLABELED TORI

The labeled square torus of size N is a two-dimensional (wrap-around) array of $\sqrt{N} \times \sqrt{N}$ processors. Each processor is labeled $P_{i,j}$ with $0 \leq i, j \leq (\sqrt{N} - 1)$ and is connected by a bidirectional link to four processors; $P_{i,\, j+1 \bmod \sqrt{N}}$, $P_{i,\, j-1 \bmod \sqrt{N}}$, $P_{i+1 \bmod \sqrt{N},\, j}$, and $P_{i-1 \bmod \sqrt{N},\, j}$, through links labeled *east*, *west*, *north*, and *south*, respectively.

The same construction can extend to rectangular tori and higher dimensions. In the following, unless specified, we will denote as a torus any $(\sqrt{N} \times \sqrt{N})$-processor topology with the previous definition.

2.1. Sense of Direction in Tori

Before proceeding with the unlabeled torus, we provide an intuitive description of the fundamental properties used as sense of direction in labeled tori.

For example, consider a portion of a torus depicted in Fig. 1, the communication topology is a 2-dimensional torus where the edge labels {*north*, *south*, *east*, *west*} are assigned in the natural globally consistent way. This labeling is a sense of direction [7]. Consider for instance the three paths, starting from node O, whose associated labels are c_1 = [*north*, *east*, *south*, *north*], c_2 = [*east*, *north*, *west*, *east*], and c_3 = [*east*, *east*]. Using the rules of the globally consistent labeling, it is trivial to deduce that the two paths corresponding to c_1 and c_2 will end in the same node *NE*, while the one corresponding to c_3 will end in a different node *EE*. With sense of direction, there is a relationship between labeling and capability of distinguishing among paths. Intuitively, the labeling is a sense of direction if it is possible to understand, from the labels associated to the edges, whether different paths from any given node x end in the same node or in different nodes.

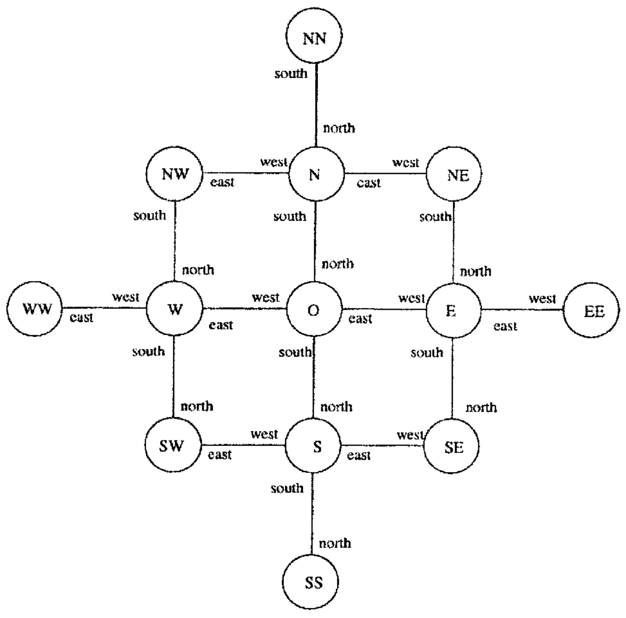

FIG. 1. Sense of direction in a portion of a torus.

2.2. Electing a Leader in a Torus with $\mathcal{SD}$

We briefly recall the main features of Peterson's algorithm [18] for square bidirectional tori. In its preliminary definition, all processors are assumed to be initially active and then process in phases. The number of active processes at each phase is reduced by a constant factor (but more than half since we must avoid an $O(N \log N)$ message complexity).

The basic goal of each active processor at the ith phase is to mark off the boundary of a square distance d on a side ($d = \alpha^i$ for some constant α). This is done by sending a "looking" message at distance d to the *north* then d to the *east*, d *south*, and finally d *west* to the original node (see Fig. 2). Marking its boundaries, an active processor can *see* or *be seen* by another active processor. When the "looking" message encounters the boundary of a processor on the same phase, the message either continues its way after becoming a "SawSmaller" (if the value of the encountered processor is smaller), or becomes a "SeenbySmaller" and continues along the boundary of the encountered processor with a larger value. The active processor is promoted to the next phase $i+1$ iff either (a) it receives both

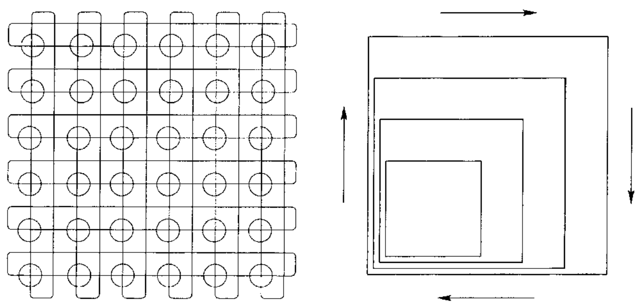

FIG. 2. Leader election steps (right) in a torus (left).

a "SawSmaller" and a "SeenbySmaller" message, or (b) it did not see an active processor. The efficiency of the algorithm is based on the choice of the size of the search area, which must be sufficiently large to cross another square.

The algorithm will terminate when a processor reaches phase $d = \alpha^i \geq \sqrt{N}$; at this time there are at most $(2-\alpha^2)^{-1}$ active processors surviving. To elect the leader among the remaining processors, a constant number of wrap-around phases (sending a probe message in one dimension) are needed. Each costs $O(\sqrt{N})$ messages. The optimal constant is $\alpha = 1.1795$.

Also, it is conjectured that the algorithm can easily extend to higher dimensions. However, if the torus is not square but rectangular with length l and width w ($l \leq w$), then the algorithm can be adapted to use $\Theta(n + l \log l/w)$ messages.

2.3. Electing a Leader in a Torus without $\mathcal{SD}$

When links are unlabeled, Peterson [18] suggested that the same algorithm can be used as is since "the algorithm only needs to mark off a square; the orientation of the square is irrelevant." It is only required to pass a message in a straight line or make the "appropriate" turn. To confirm the conjecture we introduce the detailed modifications. We first provide a message with the ability to be forwarded in the same direction (*east/west*, *north/south*) and then introduce the notion of *handrail*, partial structural information, which allows messages to turn consistently (clockwise or counterclockwise).

DEFINITION 2.1. (a) A *district labeling* of x is a function which maps the sequences of labels associated to the paths from x to any node y at a distance of at most 2 to the local name $\beta_x(y)$ used by x to refer to y.

(b) A district labeling associated to β_x for x is *consistent* iff $\forall y, z$ at a distance of at most 2 from x, $\beta_x(y) = \beta_x(z)$ implies that $y = z$.

LEMMA 2.1. *The algorithm presented in Fig. 3 computes a consistent district labeling for every node using exactly* $16N$ *messages in unlabeled square tori of N nodes with* $\sqrt{N} > 4$.

Proof. An arbitrary subset of processors can spontaneously start the execution of the algorithm; this is modeled by the reception of a WAKEUP message.

The goal of the algorithm is to acquire the identity and the position of each processor at distance 2. Each processor sends its identity to each of its neighbors, which forwards it to its three remaining neighbors. Overall, each processor has received 4 messages for its immediate neighbors and 4×3 for nodes at distances 2, that is, $16N$ messages overall. Note that this algorithm, suggested in [18], is similar to the prelabeling used in [22] for computing $\mathcal{SD}$ in $O(N \log N)$, and was independently discovered in [16].

For the sake of the explanation, and without loss of generality, we applied the algorithm in the portion of the torus depicted in Fig. 1, although the edge labeling does not contain any particular information in this case (it is only a local naming). When the algorithm terminates, the node O knows:

```
procedure district labelling (MyId)
begin
Mystatus := Asleep; $count$ = 0
$\forall$ link $r$, $H1_r := \emptyset$, $H2_r := \emptyset$

If Mystatus = Asleep:
(0) Upon receipt of WAKEUP or any other message on any link
  Mystatus := Awake
  $\forall$ link $k$, send ONE(MyId) on arc labelled $k$
  if not (WAKEUP) process message as awake
end WAKEUP

If Mystatus = Awake:

(1) Upon receipt of ONE($Id1$) on link $r$
    $H1_r := Id1$
    $Id2 := Id1$ ; $Id1 := MyId$
    $\forall$ link $k \neq r$ send TWO($Id1, Id2$) on link labelled $k$
    $count = count + 1$
end ONE

(2) Upon receipt of TWO($Id1, Id2$) on link $r$
    $H2_r := H2_r \cup \{Id2\}$
    $count = count + 1$
    if $count$ = 16 then Terminate
end TWO
end district labelling
```

FIG. 3. District labeling algorithm in an unlabeled torus.

- that N, E, S, W are its immediate neighbors,
- which link must be used to reach them, respectively *north*, *east*, *south*, *west*,
- for each link, which node at distance 2 can be reached, $H2_{north} = \{NW, NN, NE\}$, $H2_{east} = \{NE, EE, SE\}$, $H2_{south} = \{SE, SS, SW\}$, $H2_{west} = \{SW, WW, NW\}$, respectively,
- (if $\sqrt{N} > 4$), that two sets $H2_i$ and $H2_j$ the intersection of which is empty are in the same dimension (*north/south* or *east/west*), perpendicular if not.

To sum it up, the corresponding Fig. 1 is deduced by O (as shown in Fig. 4). ■

The restricted cases of electing a leader in unlabeled tori in $O(N)$ messages when $\sqrt{N} \leq 4$ are simple to design and are not presented here for brevity. It is worth noting that the consistent district labeling algorithm does not need to be started simultaneously at all nodes; it can be initiated by any subset of processors (each sleeping processor will be awakened by the first incoming message).

By construction of the consistent district labeling in Lemma 2.1, a message can be forwarded easily in the same dimension ("straightforward"): the message received on link i is sent through the link j which has an empty intersection set, $H2_i \cap H2_j = \emptyset$. By contradiction a "turn" can easily be made too ($H2_i \cap H2_j \neq \emptyset$), but the exact direction is unknown.

With the next lemma, we show that if the message carries specific information, a *handrail*, it is possible to make the appropriate turn. The processors do not have a sense of direction but the message forwarded has a globally consistent orientation (thanks to a *handrail*). Indeed, since the orientation of the square is irrelevant for the correctness of the algorithm,

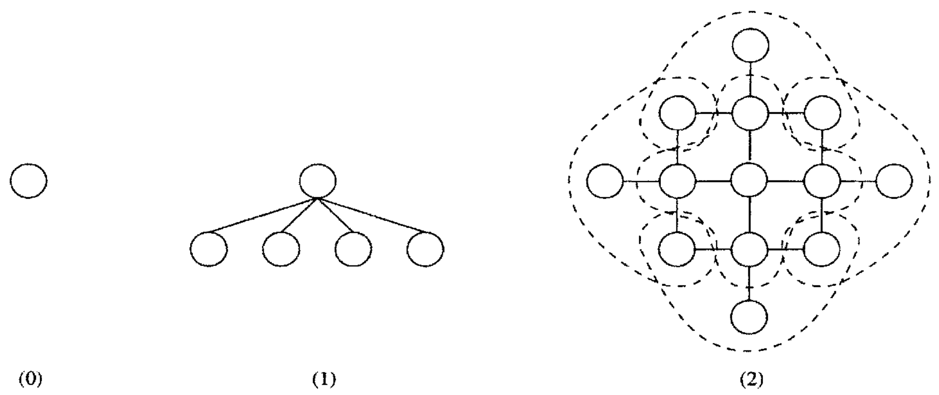

FIG. 4. View of a processor at distances 0 to 2 in an unlabeled torus.

the choice of the first direction and the first "turn" does not matter for the correctness, but the choice must be consistently applied. Moreover, it can be done by the initiator of the message.

LEMMA 2.2. *Using the* handrail *algorithm presented in Fig. 6, a message can travel along the boundaries of a square of a given size d with globally consistent turns in unlabeled tori.*

Proof. The algorithm is presented in Fig. 6. Before a message is sent, a processor, say X_1 in Fig. 5, chooses arbitrarily two perpendicular communication links locally labeled r and k and labels them accordingly, without loss of generality, say *east* and *south*, respectively. Enough information should be provided for the message to travel in the "world according to X_1."

The message contains its own *handrail*. The *handrail* is the immediate neighbor according to the other arbitrarily chosen perpendicular direction (Handrail := $H1_k$). The processor can then initiate the message on one of these two links (say *east*) to be forwarded at distance d in the same dimension (*east/west*). In our example, the message heading to the *east* contains the name of the processor at distance 1 at *south* (X_1 sets the handrail's value to Y_1 in the message).

Upon receipt of the message, the neighbour X_2 deduces from the handrail's value Y_1 that, according to X_1, its *south* is the link heading to Y_2: i.e., the perpendicular link which can lead to the processor Y_1 in 2 hops, $((1)k,\ H2_r \cap H2_k =$ Handrail). It updates the *handrail* value to Y_2 (its own relative *south* now) and forwards the updated message. As it can easily be guessed at this stage, the set of nodes used as a handrail is

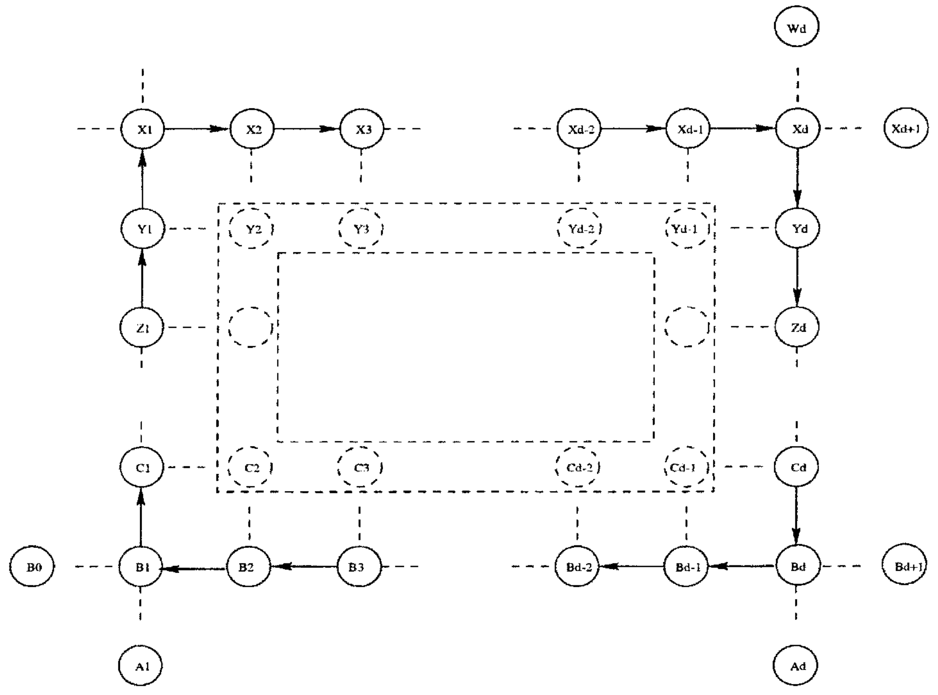

FIG. 5. Handrail in the unlabeled torus.

the set of immediate "inside" neighbours of the nodes of the visited path (as shown with the dashed area inside the traveling path in Fig. 5).

The message is forwarded with the same principle until reaching the processor at distance d, say X_d. To turn, the message uses the handrail as a pivot. X_d helps the message to find out which way to turn. Using the idea described above, X_d detects the relative *south* and forwards the message to this direction after updating the *handrail* value to X_{d-1} as the relative node at *west* (the new inside face). The straightforward process is repeated d times to reach a node, say B_d, where the same turn-and-forward process is repeated to B_1 and finally to X_1. ■

The following Theorem can be immediately deduced.

THEOREM 2.1. *The algorithm elects a* leader *using* $O(N)$ *messages in an unlabeled two-dimensional torus of* $\sqrt{N} \times \sqrt{N}$ *nodes.*

Proof. For brevity we only present the modifications to the original algorithm regarding the modifications due to the handrail algorithm.

By construction of the consistent district labeling in Lemma 2.1 as a preprocessing phase, all processors are initially active and therefore can process in phases.

Each processor chooses arbitrarily two perpendicular communication links and labels them accordingly, and sends a "looking" message (instead of a token) around the square boundary using the *handrail* algorithm (Fig. 6) as presented in Lemma 2.2. If the message becomes a "SeenbySmaller" and must continue along the boundary of a processor with the larger value, it will update its *handrail* value to the one stored locally in the visited node and which corresponds to the processor with the larger value. (Each processor stores the handrail of the processor's boundary it belongs to.)

After $\sqrt{N}$ phases, the "wrap around" phases are executed as in [18] (they required only straightforward communications). Therefore, the complexity of the algorithm remains unchanged, which proves the theorem. ■

COROLLARY 2.1. *The size of the messages in the Election algorithm in an unlabeled torus requires exactly* log *m extra bits, where m is the largest value in ID.*

Proof. Immediate. Only the handrail's value, a processor identity, is added in the message. ■

2.4. Generalization of the Handrail

Using the same *handrail* paradigm, one should be able to solve other distributed problems in unlabeled tori with the same asymptotic complexity as in the labeled case. Indeed, the *handrail* provides more than just making it possible to turn always "clockwise" or "counterclockwise." We have seen that, at each processor, the message has the choice either to keep on in the same direction or to turn consistently. By elimination of choices, the message can turn in the opposite direction, if desired, by choosing the remaining link and can update the handrail accordingly. Moreover, the message will be able to repeat this as many times as necessary and still being able to know its relative position from its initiator: the message must contain the number of hops in each direction (this can be compressed to a shortest path notation by elimination of combination or application of modulo, e.g., as presented in [6]).

```
procedure Handrail algorithm
$H1_r, H2_r$ are the sets of nodes accessible by link $r$ (at distance 1 and 2 respectively).
begin
If Mystatus = Initiator:

(0) Choose arbitrarily $k, r$ s.t. $H2_r \cap H2_k \neq \emptyset$ /* perpendicular edges */
    Handrail := $H1_k$
    send TOKEN(MyId,Handrail,d,0) on link $r$
    Upon receipt of TOKEN(MyId,Handrail,ToGo,SoFar) on link $k$
        Terminate

If Mystatus $\neq$ Initiator:

(1) Upon receipt of TOKEN($Id1$,Handrail,ToGo,SoFar) on link $r$
    SoFar := SoFar + 1
    find $k$, $k \neq r$ s.t. $H2_r \cap H2_k$ = Handrail
    if SoFar $\neq$ ToGo then /* must be straight forwarded */
      Handrail := $H1_k$
      find $s$ s.t. $H2_r \cap H2_s = \emptyset$ /* the same dimension */
      send TOKEN(MyId,Handrail,ToGo,SoFar) on link $s$
    else
      Handrail := $H1_r$
      send TOKEN(MyId,Handrail,ToGo,0) on link $k$ /* turn */
    endif
end TOKEN
end Handrail algorithm
```

FIG. 6. Consistent square travel with the handrail algorithm.

Extending the result to d-dimensional unlabeled tori or grids ($d > 2$) is based on the same consistent district labeling algorithm. To acquire the identity of each processor at distance 2, the algorithm costs 8 d messages. Again, the sets of nodes accessible by a link which have no intersection are in the same dimension, perpendicular otherwise. The handrail is chosen according to d arbitrarily chosen perpendicular links and corresponds to as many node identities (at most $d - 1$), which are necessary to turn consistently: one processor identity is sufficient to do consistent turns in a 2-dimensional subtorus, two are necessary for a 3-dimensional subtorus, etc.

3. UNLABELED CHORDAL RINGS

A common technique to improve reliability of ring networks is to introduce link redundancy; that is, to have each node connected to two or more additional nodes in the network. With alternate paths between nodes, the network can sustain several node and link failures. The overall topological structure of these *redundant rings* is always highly regular [1]; in particular, the set of ring edges (*regular*) and additional edges (*bypass*) form a *circulant graph.* Because of an uncoordinated literature, numerous terms for this topology (e.g., *chordal ring* or *distributed loop computer network*) are commonly used. A detailed survey of these topologies is presented in [3]. For sake of simplicity, we will use the term *chordal ring* in this paper.

A chordal ring $C_N\langle d_1, d_2, \ldots, d_k\rangle$ of size N and k-chord structure $\langle d_1, d_2, \ldots, d_k\rangle$, with $d_1 = 1$, is a ring R_N of N processors $\{p_0, p_1, \ldots, p_{N-1}\}$, where each processor is also directly connected to the processors at distance d_i and $N - d_i$ by additional incident chords. The link connecting two nodes is labeled by the distance which separates these two nodes on the ring, i.e., following the order of the nodes on the ring: the node p_i is connected to the node $p_{i+d_j \bmod N}$ through its link labeled d_j (as shown in Fig. 7(a)). In particular, if a link between two processors p and q, is labeled by distance d at processor p, this link is labeled by $N - d$ at the other incident processor q, where N is the number of processors. Note that both *rings* and *complete graphs* are chordal rings, denoted $C_N\langle 1\rangle$ and $C_N\langle 1, 2, 3, \ldots, \lfloor N/2\rfloor\rangle$, respectively. This link labeling, known as *distance* or *chordal* labeling, is a sense of direction [7]. Some instances of the chordal ring are reminiscent of the torus, as shown in Fig. 7. (It is worth pointing out that some designs for *redundant tori* and *redundant hypercubes* are also chordal rings, e.g. [4].)

3.1. Electing a Leader in a Chordal Ring with $\mathcal{SD}$

For this class of network topology, the fundamental problem of leader election has been extensively studied. The main problem is to determine the minimal sets of links that must be added to the ring in order to achieve a linear election algorithm. Over the years, the minimal number of necessary chords has decreased from $O(\log N)$ [2] to $O(\log\log N)$ [10] and to $O(\log\log\log N)$ [17]. The ultimate case is the chordal ring $C_N\langle 1, n\rangle$, where n is approximately $\sqrt{N}$. Using Peterson's algorithm, Fabri and Tel presented a $\Theta(N)$ solution for this case [22].

Fabri and Tel's Algorithm. We briefly recall the main features of the algorithm. Each processor has four communication links labeled $\{1, N-1, n, N-n\}$, conveniently denoted as $\{+1, -1, +n, -n\}$ in the following. All the processors are assumed to be initially active and then to process in phases. The basic goal of each active process on the ith phase is to mark off the boundary of a d-square ($d = \alpha^i$ for some constant α). This is done by sending a "looking" message at distance d through links $+1$, then d through links $+n$, d through links -1, and finally d through links $-n$ to the original node. Marking its boundaries, an active processor can *see* or *be seen* by another active processor.

The promotion rules applied and the modifications of the message boundary are as in Peterson's algorithm. The algorithm will terminate when a processor will reach phase $d = \alpha^i \geq \sqrt{N}$, at this time there are at most $(2 - \alpha^2)^{-1}$ active processors surviving. In this case, instead of "wrap-around" phases, an election is initiated along the ring (on link labeled $+1$ and -1) to distinguish the leader among the constant number of survivors. This termination phase costs $O(N)$ messages.

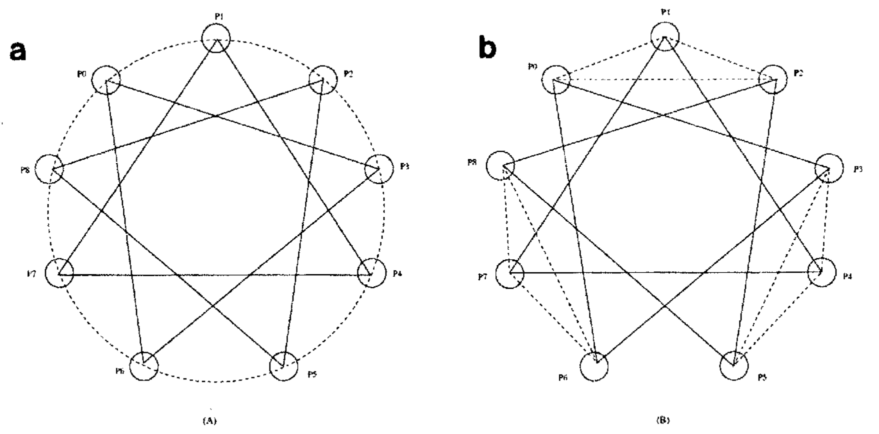

FIG. 7. (a) Chordal ring $C_9\langle 1, 3\rangle$ and (b) torus.

3.2. Electing a Leader in a Chordal Ring without $\mathcal{SD}$

Again, the previous observation that the orientation of the square is irrelevant holds. In Tel's algorithm, if marking the boundary is initiated through $+1$ edges, it terminated through $-n$ chords. Obviously, marking the boundary by reversing the path will have no impact on the correctness of the algorithm. Similarly, a sequence of turns $[+n, +1, -n, -1]$ or any permutation which respects the alternate dimensional sequence is still valid. The algorithm is only based on the ability of the search area to be sufficiently large to cross another boundary. Using the same *handrail* paradigm, we introduce the necessary modifications.

LEMMA 3.1. *The algorithm presented in Fig.* 3 *computes a consistent district labeling for every node using exactly* $16N$ *messages in an unlabeled chordal ring* $C_N\langle 1, n\rangle$, $(n \approx \sqrt{N}$ *and* $n > 3)$.

Proof. Same as in Lemma 2.1 using the algorithm presented in Fig. 3. Each processor sends its identity to each of its neighbors, which forwards it to its three remaining neighbours. The case $n \leq 3$ avoids consistency, but obvious solutions for this case can be designed to obtain linear complexity. ∎

For the sake of the explanation we applied the algorithm in a system depicted in Fig. 8, using the same notation as in Fig. 1. Again, when the algorithm terminates, the node O knows that N, E, S, W are its immediate neighbors; which link must be used to reach them, respectively *north*, *east*, *south*, *west*; for each link, which node at distance 2 can be reached, $H2_{north} = \{NW, NN, NE\}$, $H2_{east} = \{NE, EE, SE\}$, $H2_{south} = \{SE, SS, SW\}$, $H2_{west} = \{SW, WW, NW\}$ respectively; that two sets $H2_i$ and $H2_j$ the intersection of which is empty are in the same dimension (*north/south* or *east/west*), perpendicular if not.

In this case, however, the problem is different from the torus where both dimensions are of the same length and play a symmetric role in the algorithm. In the present setting, a node does not know which of *north/south* and *east/west* corresponds to the ring labeling $+1/-1$. This means that after this preprocessing phase a node cannot distinguish between a ring edge and a bypass chord.

LEMMA 3.2. *Using the* handrail *algorithm, a message can travel along the boundaries of a square of a given size d* $(d \leq \sqrt{N})$ *with "globally consistent turns" in an unlabeled chordal ring* $C_N\langle 1, n\rangle$, $(n \approx \sqrt{N})$.

Proof. Similar to Lemma 2.2 using the algorithm presented in Fig. 6. First, the consistent district labeling is computed as shown in Lemma 3.1. Second, a processor chooses arbitrarily two perpendicular communication links and labels them accordingly, without loss of generality say *east* and *south*, as shown in Fig. 8. The processor then initiates the message on one of these two links (say *east*) to be forwarded to distance d in the same dimension (*east/west*). The message heading to the *east* contains the name of the processor at distance 1 at *south* (O sets the handrail's value to S in the message).

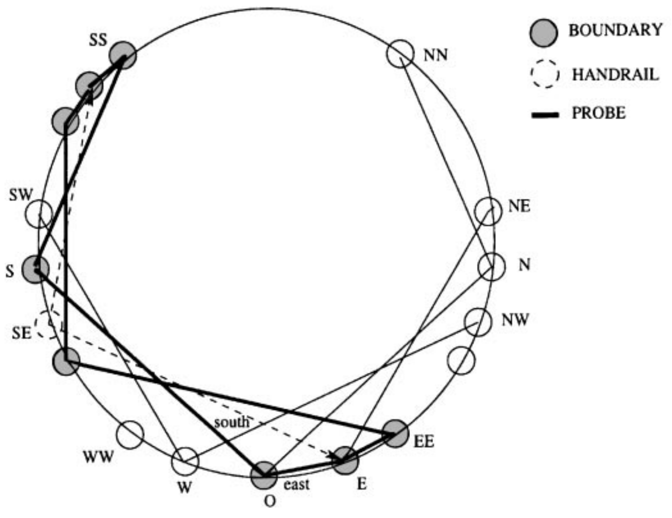

FIG. 8. Handrail in the unlabeled chordal ring $C_N\langle 1, n\rangle$.

Upon receipt of the *east* message, the immediate neighbor E of the node O deduces from the handrail's value S that its *south*, according to O, is the link heading to *SE* (the perpendicular link which can lead to the processor S in 2 hops). It updates the *handrail* value to its own relative *south* (*SE*) and forwards the updated message.

The same process is repeated until the message returns to its original node. As shown in Fig. 8 for an execution with $d = 2$, *SE* will be used repeatedly as handrail since it is the centre of a 3×3 grid). ∎

The Election Theorem follows.

THEOREM 3.1. *Using the handrail, the algorithm elects a* leader *using* $\Theta(N)$ *messages in unlabeled chordal rings* $C_N\langle 1, n\rangle$, $(n \approx \sqrt{N})$.

Proof. By construction of the consistent district labeling (Lemma 3.1) as a preprocessing phase, all processors are initially active and therefore can process in phases.

Each processor chooses arbitrarily two perpendicular communication links and proceeds as presented in Lemma 3.2. Each processor initiates a "looking" message. If the message becomes a "SeenbySmaller" and must continue along the boundary of a processor with the larger value, it updates its *handrail* value to the one stored locally in the visited node and which corresponds to the processor with the larger value. Every node stores the respective handrail's value of the boundary it belongs to.

After $\sqrt{N}$ phases, only a constant number of processors remain active. An election termination phase on the ring between them can be trivially achieved by sending a message in each perpendicular dimension (since the node cannot distinguish between a ring link and a bypass chord). The node receiving its own messages back declares itself as the leader. This technique only doubles the number of termination messages compared to the original algorithm. Each phase, including the termination, costs at most $O(N)$ messages, which proves the theorem. ∎

COROLLARY 3.1. *During the termination phase of the* Election *algorithm for an unlabeled chordal ring $C_N\langle 1, n\rangle$ ($n \approx \sqrt{N}$), the leader distinguishes between the chordal links and the ring links iff n and N are not co-prime.*

Proof. During the termination phase, the message coming back after N hops is the one which travels on the ring and allows the processor to distinguish between the $-1/+1$ ring links and $-n/+n$ chords. Both messages used N hops iff n and N are co-prime (i.e., $\gcd(n, N) = 1$). In this case, the processor cannot arbitrarily choose a link to be labeled $+1$ since each local link can be used to build a ring. Indeed, it is then possible to build two isomorphic chordal rings where one has a chord of length n^{-1} (modulo N), with n approximately $\sqrt{N}$ (e.g., $C_{23}\langle 1, 5\rangle$ is isomorphic to $C_{23}\langle 1, 9\rangle$). ■

Improvement of the Termination. If n and N are not co-prime, topological observations allow an improvement of the termination phase. First observe that in any phase i ($\alpha^i < \sqrt{N}$), because a message makes at most $\sqrt{N}$ hops in a direction before turning, a message initiated through a given link, say *east*, will always come back through the arbitrarily chosen perpendicular link (i.e., the *handrail*), say *south*.

Second, note that when $N \bmod n = 0$ (i.e., $n = \sqrt{N}$) the chords provides a wrap-around of size $\sqrt{N}$. At the last iteration, $d = \sqrt{N}$, two cases may occur:

- The message comes back after only $\sqrt{N}$ hops through the link *west* (opposite of *east*). The initiator immediately deduces that *east* is a chord link. It then distinguishes between the $-1/+1$ ring links (*south/north*) and the $-n/+n$ chords (*west/east*).
- The message comes back after only $3\sqrt{N}$ hops through link *east* (used for sending). The processor immediately deduces that *east* is a ring link. It then distinguish between the $-1/+1$ ring links (*west/east*) and the $-n/+n$ chords (*south/north*).

In both cases, the termination phase can proceed as in the original algorithm using only the ring.

COROLLARY 3.2. *The size of the messages in the Election algorithm in the unlabeled chordal ring requires exactly* log m *extra bits, where the largest value in ID is m.*

Proof. Immediate (as in Corollary 2.1). Only the handrail's value is added in the message. ■

4. COMPUTING SENSE OF DIRECTION

The *handrail* can be used as a generic tool to solve many other distributed problems in unlabeled topology, but its use may remain a tedious task for the designer of the distributed algorithm. Other alternatives may be considered.

Since sense of direction is known to improve the communication complexity of distributed algorithms, computation of $\mathcal{SD}$ as a preprocessing phase in unlabeled topology has been studied [21, 22]. So far, results have not been encouraging: any algorithm computing $\mathcal{SD}$ exchanges at least $\Omega(e - 1/2\ N)$ messages in a network with N nodes and e edges. This result is not attractive for dense topologies, $\Omega(N^2)$ for cliques and $\Omega(N \log N)$ for the hypercube; even if algorithms matching the lower bounds have been proposed. The interest is more relevant for topologies with a linear number of edges such as tori; however, the best algorithm known so far requires $O(N \log N)$ [22]. Here, we present a solution to build $\mathcal{SD}$ using $\Theta(N)$ messages for each topology. This proves that solutions for any problem in an unlabeled topology will immediately be deduced from the corresponding solution in the labeled topology without asymptotic overcost.

THEOREM 4.1. *A leader can compute sense of direction using exactly $N + \sqrt{N}$ messages in unlabeled two-dimensional tori with $\sqrt{N} \times \sqrt{N}$ nodes.*

Proof. The algorithm is executed in two concurrent phases, as shown in Fig. 9. First the leader arbitrarily chooses two perpendicular communication links and labels them accordingly, without loss of generality say *east* and *south*. The leader then initiates a message on each of these two links to be forwarded in the same dimension (*east/west* and *north/south*) as shown in the algorithm presented in Theorem 2.1. Both messages build their own *handrails* (according to the other arbitrarily chosen direction): the message heading to the *east* contains the name of the processor at distance 1 at *south*, the message heading to the *south* contains the processor at distance 1 at *east*.

Upon receipt of the *east* message, the immediate neighbor of the leader labels its incoming link as *west* and the other link on same dimension as *east*. With the handrail's name in the message, it deduces where is its *south* (the perpendicular link which can lead to the processor mentioned in two hops) and initiates a message to the *south* to be forwarded in the same dimension in order to take care of the *north/south* labeling in this column. It can then resume its *east/west* phase by changing the label of the handrail to its own *south* node and can forward the updated leader message. The second phase (started concurrently upon receipt of the first phase message) corresponds to the *north/south* labeling. The *north/south* message is just forwarded and does not generate any message in the *east/west* dimension. However, using the same technique as above, the message must contain the corresponding handrail's name (the processor at distance 1 at *east*) to distinguish *east* and *west* locally (and update accordingly). Each message will be stopped upon receipt by its initiator.

The algorithm will terminate in $O(N)$ time and will use exactly $\sqrt{N}$ messages for the first phase and N for the second, which proves the theorem. ■

COROLLARY 4.1. *A distributed algorithm computes sense of direction using $\Theta(N)$ messages in unlabeled two-dimensional tori with $\sqrt{N} \times \sqrt{N}$ nodes.*

```
procedure Torus SD algorithm
$H1_r, H2_r$ are the sets of nodes accessible by link $r$ (at distance 1 and 2 respectively).
begin
If Mystatus = Leader:

(0) Choose arbitrarily $k, r$ s.t. $H2_r \cap H2_k \neq \emptyset$ /* perpendicular edges */
    $east := r$; $south := k$
    choose $r'$ s.t. $H2_r \cap H2_{r'} = \emptyset$ ; $west := r'$
    choose $k'$ s.t. $H2_k \cap H2_{k'} = \emptyset$ ; $north := k'$
    Handrail := $H1_k$
    send ONE(Handrail) on link $r$
    Handrail := $H1_r$
    send TWO(Handrail) on link $k$
    Upon receipt of ONE(Handrail) and TWO(Handrail)
        Terminate

If Mystatus $\neq$ Leader:

(1) Upon receipt of ONE(Handrail) on link $r'$
    find $k$, $k \neq r'$ s.t. $H2_{r'} \cap H2_k$ = Handrail
    $west := r'$; $south := k$
    choose $r$ s.t. $H2_r \cap H2_{r'} = \emptyset$ ; $east := r$
    choose $k'$ s.t. $H2_k \cap H2_{k'} = \emptyset$ ; $north := k'$
    Handrail := $H1_k$
    send ONE(Handrail) on link $r$
    Handrail := $H1_r$
    send TWO(Handrail) on link $k$
    Upon receipt of TWO(Handrail)
        Terminate

(2) Upon receipt of TWO(Handrail) on link $k'$
    find $r$, $k' \neq r$ s.t. $H2_r \cap H2_{k'}$ = Handrail
    $east := r$; $north := k'$
    choose $r'$ s.t. $H2_r \cap H2_{r'} = \emptyset$ ; $west := r'$
    choose $k$ s.t. $H2_k \cap H2_{k'} = \emptyset$ ; $south := k$
    Handrail := $H1_r$
    send TWO(Handrail) on link $k$
    Terminate

end Torus SD algorithm
```

FIG. 9. Leader constructing sense of direction in a torus.

Proof. Immediate from Theorem 2.1 and Theorem 4.1. ∎

Remarks. Processor Naming. Note that the algorithm can name the processor in the commonly used way $(\{P_{i,j} : 0 \leq i, j \leq (\sqrt{N} - 1)\})$ without overcost: the leader names itself $P_{0,0}$ and initiates the naming of each $\{P_{0,j} : 0 \leq j \leq (\sqrt{N} - 1)\}$ during the first step (*east*), which will initiate the naming of their respective $\{P_{i,j} : 0 \leq i \leq (\sqrt{N} - 1)\}$ during the second step (*south*).

Grid. Without wraparound, in a regular d-dimensional array of processors, the algorithm presented in Theorem 4.1 is still valid. For each phase, the processor must initiate a message in both directions of the dimension which will be stopped when encountering the boundary.

Higher Dimensions. The algorithm can be designed in a simple manner by extending the handrail as shown in Section 2.

THEOREM 4.2. *A leader can compute sense of direction using at most $N + 2\sqrt{N}$ messages in unlabeled chordal ring $C_N\langle 1, n\rangle$, $(n \approx \sqrt{N})$.*

Proof. The algorithm is shown in Fig. 10. When n and N are not co-prime, the algorithm is executed in two phases. After completion of the election algorithm, by Corollary 3.1, the leader knows which pair of links is associated to $-1/+1$. It sends a probe message on one of these two links (labeled arbitrarily $+1$). The message is forwarded n times in the same dimension to be received by an immediate neighbor of the leader which sends back to the leader directly. Upon receipt, the leader can assign the label $+n$ to the link from which it receives the message, and ends this phase.

The rest of the algorithm can be achieved with a method similar to the algorithm used in Theorem 4.1 (providing concurrency and achieving an optimal time complexity, $O(\sqrt{N})$). However, for brevity, we give a simpler method based on the ring-based structure (requiring $O(N)$ time, though).

The leader initiates the second phase by sending a token through link $+1$, the message contains a *handrail* set to the identity of its neighbor accessible through link $+n$. The message will be forwarded along the same dimension (the ring) while the *handrail* will be updated accordingly. Upon receipt, each processor will be able to label consistently each of its links (knowing that -1 is the receiving link and $+n$ is the

```
procedure Chordal SD algorithm
$H1_r, H2_r$ are the sets of nodes accessible by link $r$ (at distance 1 and 2 respectively).
begin
If Mystatus = Leader or NewLeader /* do not know $\{+n\}$ */:
(1) choose $r$ s.t. $r \in \{-1,+1\}$ if known
    send PROBE(MyId,n) on link $r$
    Upon receipt of PROBE(MyId,n) on link $k$
        Mystatus := Initiator
        label(+1) := $r$; label($+n$) := $k$
        start(3)

If Mystatus $\neq$ Leader and Mystatus $\neq$ NewLeader:
(2) Upon receipt of PROBE(Id,d) on link $r'$
    if $d = 0$
      then if $\exists k$, s.t. $H1_k$ = Id
            then send PROBE(Id,n) on link $k$
            else Mystatus := NewLeader; start(1) with $r$ s.t. $H2_{r'} \cap H2_r \neq \emptyset$
      else
            choose $r$ s.t. $H2_r \cap H2_{r'} = \emptyset$
            send PROBE(Id,(d-1)) on link $r$

If Mystatus = Initiator: /* knows $\{+1\}$ and $\{+1\}$ */
(3) label(+1) = $r$; label($+n$) = $k$
    choose $r'$ s.t. $H2_r \cap H2_{r'} = \emptyset$ ; label($-1$) := $r'$
    choose $k'$ s.t. $H2_k \cap H2_{k'} = \emptyset$ ; label($-n$) := $k'$
    Handrail := $H1_k$
    send TOKEN(Handrail) on link $r$
    Upon receipt of TOKEN(Handrail)
        Terminate

If Mystatus $\neq$ Initiator:
(4) Upon receipt of TOKEN(Handrail) on link $r'$
    find $k$, $k \neq r'$ s.t. $H2_{r'} \cap H2_k$ = Handrail
    label($-1$) := $r'$; label($+n$) := $k$
    choose $r$ s.t. $H2_r \cap H2_{r'} = \emptyset$ ; label(+1) := $r$
    choose $k'$ s.t. $H2_k \cap H2_{k'} = \emptyset$ ; label($-n$) := $k'$
    Handrail := $H1_k$
    send TOKEN(Handrail) on link $r$
    Terminate

end Chordal SD algorithm
```

FIG. 10. Leader constructing sense of direction in a chordal ring.

corresponding *handrail*). The second phase will terminate when the neighbor of the leader (through link -1) receives the token. Overall, the algorithm requires exactly $N + \sqrt{N}$ messages.

When n and N are not co-prime, the algorithm may need to repeat the first phase twice. Indeed, the leader does not know which pair of links corresponds to $-1/+1$. It must choose arbitrarily and proceed as above. If after n hops the node receiving the message does not match a neighbor of the leader, this node can undoubtedly detect that its perpendicular pair of links corresponds to the dimension $-1/+1$. This node can then process the two phases as the leader in the previous case. This modification adds only $\sqrt{N}$ messages to the $(N + \sqrt{N})$ messages required above, which gives a $N + 2\sqrt{N}$ in the worst case, proving the theorem. ■

COROLLARY 4.2. *A distributed algorithm computes sense of direction using* $\Theta(N)$ *messages in unlabeled chordal rings* $C_N\langle 1, n\rangle$, $(n \approx \sqrt{N})$.

Proof. Immediate from Theorem 3.1 and Theorem 4.2.

5. CONCLUDING REMARKS AND OPEN PROBLEMS

This study shows how a limited number of edges, usually considered as a drawback, can be used efficiently to obtain an optimal communication complexity. We built partial structural information, which is proved sufficient to allow the design of linear Election algorithms, and built sense of direction without asymptotic overcost. Both algorithms use the same fundamental ideas to achieve efficiency, indicating that the technique is both powerful and general. Since the solution is based on a preprocessing phase which will work for any network with constant degree (i.e., linear number of edges), it can be used for such other topologies, e.g. [19]. The question is still open for topologies such as the hypercube where the degree is logarithmic. In this case such a method would cost $O(N \log N)$ preliminary messages and thus is not suitable. The best known complexity is $O(n \log \log n)$ messages [5], compared to $O(n)$ in the labeled case [6].

Finally, we showed that extending the result to d-dimensional unlabeled tori or grids ($d > 2$) was a simple matter. For brevity, we did not investigate the case of an unlabeled chordal ring with more than one chord. The problem does not appear as simple (even if one of the chords is known to have length $\sqrt{N}$).

ACKNOWLEDGMENTS

We thank B. Litow for his comments on this paper, and the anonymous referees for pointing out the relevance of a recent paper [16].

REFERENCES

1. Arden, B. W., and Lee, H. Analysis of chordal ring. *IEEE Trans. Comput.* **C-30,** 4 (Apr. 1981), 291–295.
2. Attiya, H., van Leeuwen, J., Santoro, N., and Zaks, S. Efficient elections in chordal ring networks. *Algorithmica* **4** (1989), 437–446.
3. Bermond, J.-C., Comellas, F., and Hsu, D. F. Distributed loop computer networks: A survey. *J. Parallel Distrib. Comput.* **24** (1995), 2–10.
4. Bruck, J., Cypher, R., and Ho, C.-T. Fault-tolerant meshes and hypercubes with minimal numbers of spares. *IEEE Trans. Comput.* **42,** 9 (Sept. 1993), 1089–1104.
5. Dobrev, S., and Ruzicka, P. Linear broadcasting and $n \log \log n$ election in unoriented hypercubes. *Proc. of the 4th International Colloquium on Structural Information and Communication Complexity, Sirocco'97.* Ascona, Switzerland, 1997. [To appear]
6. Flocchini, P., and Mans, B. Optimal elections in labeled hypercubes. *J. Parallel Distrib. Comput.* **33,** 1 (1996), 76–83.
7. Flocchini, P., Mans, B., and Santoro, N. Sense of direction: Formal definition and properties. *Proc. of 1st Colloquium on Structural Information and Communication Complexity.* Ottawa, Canada, 1994, pp. 9–34.
8. Flocchini, P., Mans, B., and Santoro, N. On the impact of sense of direction on message complexity. *Inform. Process. Lett.* **63,** 1 (1997), 23–31.
9. Gallager, R. G., Humblet, P. A., and Spira, P. M. A distributed algorithm for minimum spanning tree. *ACM Trans. Programming Languages Systems* **5,** 1 (1983), 66–77.
10. Kalamboukis, T. Z., and Mantzaris, S. L. Towards optimal distributed election on chordal rings. *Inform. Process. Lett.* **38** (1991), 265–270.
11. Korach, E., Moran, S., and Zaks, S. Optimal lower bounds for some distributed algorithms for a complete network of processors. *Theoret. Comput. Sci.* **64** (1989), 125–132.
12. Litow, B., and Mans, B. On isomorphic chordal rings. *Proc. of the Seventh Australasian Workshop on Combinatorial Algorithms (AWOCA'96).* Magnetic Island, 1996, pp. 108–111. [Univ. of Sydney, BDCS-TR-508]
13. Loui, M. C., Matsushita, T. A., and West, D. B. Election in complete networks with a sense of direction. *Inform. Process. Lett.* **22** (1986), 185–187. [See also *Inform. Process. Lett.* **28** (1988), p. 327.]
14. Mans, B. Optimal distributed algorithms in unlabeled tori and chordal rings. *Proc. of the 3rd International Colloquium on Structural Information and Communication Complexity, Sirocco'96.* Siena, June 1996, pp. 17–31.
15. Mans, B., and Santoro, N. On the impact of sense of direction in arbitrary networks. *Proc. of 14th International Conference on Distributed Computing Systems.* Poznan, 1994, pp. 258–265.
16. Olariu, S., Stojmenovic, I., and Zomaya, A. On the dynamic initialization of parallel computers. *Proc. of the 8th IEEE International Parallel Processing Symposium.* Geneva, 1997. [To appear]
17. Pan, Y. A near-optimal multi-stage distributed algorithm for finding leaders in clustered chordal rings. *Inform. Sci.* **76,** 1–2 (1994), 131–140.
18. Peterson, G. L. Efficient algorithms for elections in meshes and complete networks. Technical Report TR-140, Department of Computer Science, Univ. of Rochester, Rochester, NY 14627, 1985. [Has been replaced by Improved algorithms for elections in meshes and complete networks, Georgia Institute of Tech., Dec. 1986]
19. Preparata, F. P., and Vuillemin, J. The cube-connected cycles: A versatile network for parallel computations. *Comm. ACM* **25** (1981), 479–482.
20. Santoro, N. Sense of direction, topological awareness and communication complexity. *SIGACT NEWS* **2,** 16 (Summer 1984), 50–56.
21. Tel, G. Network orientation. *Int. J. Foundations Comput. Sci.* **5,** 1 (1994).
22. Tel, G. Sense of direction in processor networks. In Bartošek, M., Staudek, J., and Wiedermann, J. (Eds.). *Proc. of SOFSEM'95, Theory and Practice of Informatics,* Lecture Notes in Computer Science, Vol. 1012. Springer-Verlag, Berlin/New York, 1995, pp. 50–82.

Received August 27, 1996; revised September 22, 1997; accepted September 23, 1997

BERNARD MANS received his Magistère d'Informatique from Université René Descartes, Paris 5, France in 1989. He received his D.E.A. and his Ph.D. in Computer Science from Université Pierre et Marie Curie, Paris 6, France, respectively in 1989 and 1992. From 1992 to 1994, he was a post-doctoral fellow visiting Carleton University, Ottawa, Canada (partly with the support from I.N.R.I.A., France). From 1995 to 1997, he was Lecturer at the Department of Computer Science, at James Cook University, Australia. Since 1997, he is at the department of Computing at Macquarie University, Sydney. His research interests are parallel and distributed algorithms and complexity, fault-tolerance, parallel and distributed data structures, and graph theory.