ON THE COST OF SENSE OF DIRECTION 
IN DISTRIBUTED NETWORKS

by
Z. Harpaz and S. Zaks

Technical Report #378
October 1985

The work of this author was supported in part by a Technion grant No. 121-641.
ON THE COST OF SENSE OF DIRECTION IN DISTRIBUTED NETWORKS

Z. Harpaz and S. Zaks

Department of Computer Science
Technion, Haifa, Israel

"The work of this author was supported in part by a Technion grant No. J21-541."
ABSTRACT

Sense of direction in a distributed network is the amount of knowledge a processor has regarding its adjacent communication lines. We deal with the case where, in a complete network, the lines adjacent to each processor are colored with distinct colors, and study the communication complexity for various problems. For example, a minimum spanning tree is shown to require $\Omega(n^2)$ messages, where $n$ is the size of the network. Also studied is the connection between this and another labeling. We show that $\Omega(n^2)$ messages have to be transmitted in order to move from one labeling to another with a minimum number of changes in the labeling. The lower bounds apply also for synchronous networks.

1. INTRODUCTION

Many recent studies deal with distributed computations, where a network of processors has to solve a given problem by exchanging messages, and the total number of messages is usually regarded as the optimality criterion. Besides, the problem itself, what makes these computations difficult, is the amount of knowledge each processor has regarding the communication lines adjacent to itself. This notion is termed sense of direction in [8]. Even when the underlying graph is complete (an assumption made in many studies; see, for example, [LSP]), namely each processor is connected to all others, still the amount of additional knowledge about its adjacent edges (hidden in the labeling of the edges) affects the complexity of certain problems.

Two widely studied problems are finding a leader or a spanning tree (ST) and constructing a minimum-weight spanning tree (MST). In [KMZ1], it was shown that with no sense of direction (in this case, all the edges are indistinguishable) finding a ST might require $O(n \log n)$ messages (and can be done in $O(n \log n)$). With a global sense of direction (in this case, the processors are arranged in a ring, and the correct orientation of each edge is known), this can be done in $O(n)$ messages, as shown in [LMW] (see also [SSU]); actually, an $O(n)$ algorithm can be achieved with only partial sense of direction (for example, in the case where the processors are arranged in a ring and only a set of $O(\log n)$ of the edges adjacent to each processor are oriented), as studied in [ASZ]. In [KMZ2], it was shown that with no sense of direction
constructing a MST might require $\Omega(n^2)$ messages. In [SUZ] few intermediate cases (between the no sense of direction and the global sense of direction) are represented, and the complexity of finding a ST and a MST, of transferring between the models, and of achieving the appropriate labelings, are discussed.

We introduce another labeling and study the communication complexity of various problems. These include lower bounds of $\Omega(n^2)$ messages for constructing a MST and for transferring between this new labeling and another one (from [SUZ]). These transformations are the main result, and the proofs involve a partition of the complete graph into complete subgraphs of size four, using a result from [HCW], while the technique used is an extension of the idea presented in [KMZ2], where a simpler partition of the complete graph is used.

2. PRELIMINARIES

The model under investigation is a complete distributed network. Each processor has a distinct identity, a local memory, and communication lines connecting it to all other processors. We identify the network and the complete graph representing it. Therefore, processor and vertex will be used interchangeably, and so will communication line and edge. The processors all perform the same algorithm, that includes operations of sending a message to a neighbor, receiving a message from a neighbor and processing information in their (local) memory. We assume that the messages arrive, with no error, after a finite but otherwise arbitrary delay, and that any subset of the processors may start the algorithm. An execution of an algorithm consists of a sequence of events of sending and receiving messages. An algorithm may have more than one execution. An edge is used if it carried a message, and it is unused otherwise. The message complexity of an algorithm is the maximum number of messages the algorithm may send during any possible execution. The edge complexity of an algorithm is the maximum number of used edges upon the completion of any execution of the algorithm. All our results will deal with the edge complexity, and will clearly apply also to message complexity.

The numbers 1, 2, ..., $n$ are associated with the processors; these numbers are not known to the processors, and we use them in order to ease the discussion. With the
communication line \((i,j)\) connecting processor \(i\) and processor \(j\) we associate a label \(l(i,j)\), known only to processor \(i\). Denote \(N_i = \{1, 2, \ldots, n\} - \{i\}\) and \(N' = \{1, 2, \ldots, n-1\}\). The following labelings are defined:

1. A complement labeling: for every \(i\)
   \[ l(i,j) \mid j \in N_i \Rightarrow N' \]
   and for every \(i\) and \(j\)
   \[ l(i,j) + l(j,i) = n. \]
   (See example for \(n = 6\) in Figure 1.) Let \((v,w),(v,s) \in E\). We say that \((v,w)\) is the complement edge of \((v,s)\) if \(l(v,w) + l(v,s) = n\).

2. A matching labeling: for every \(i\)
   \[ l(i,j) \mid j \in N_i \Rightarrow N' \]
   and for every \(i\) and \(j\)
   \[ l(i,j) = l(j,i). \]
   (See example for \(n = 6\) in Figure 2.) Note that a matching labeling exists only for an even \(n\); an extension for the odd case is straightforward and is left to the reader.

Each such labeling represents different knowledge a processor has about the network. The matching labeling corresponds to coloring the edges of the network. The complement labeling is one of the labelings studied in [SUZ]. We are interested in constructing a MST given a matching labeling and in transformations between a matching labeling and a complement labeling with a minimum number of changes in the labels.

This amounts to changing exactly half of the labels when \(n\) is even, and almost half of the labels when \(n\) is odd.

\(K_n = (V,E)\) denotes the complete, undirected graph on \(|V| = n\) vertices. A vertex division \(<V_1,V_2,\ldots,V_n>\) of \(V\) is a partition of \(V\) into disjoint sets of four vertices each; namely \(V = V_1 \cup V_2 \cup \ldots \cup V_n\), where \(|V_i| = 4\) for all \(i\) and \(V_i \cap V_j = \emptyset\) for \(i \neq j\). The graph induced by a vertex division \(<V_1,V_2,\ldots,V_n>\) is \(G' = (V,E')\), where \(E' = \{(i,j) \mid \exists k: i,j \in V_k\}\). An edge division \(<E_1,E_2,\ldots,E_{n-1}>\) of \(E\) is a
Figure 1: A complement labeling.

Figure 2: A matching labeling.
partition of $E$ into disjoint sets $E_i$ such that each graph $G_i = (V, E_i)$ is induced by a vertex division.

**Theorem ([HCW]):** The complete graph $K_n$ has an edge division if and only if $n = 4 \pmod{12}$.

### 3. TRANSFORMING BETWEEN THE TWO LABELINGS

We prove in this section the $\Omega(n^2)$ lower bounds for transforming between the matching labeling and the complement labeling in a complete network with $n$ processors.

**Theorem 1:** The edge complexity of transforming a matching labeling to a complement labeling is at least $\Omega(n^2)$.

**Proof:** We show two different matching labelings such that the sets of labels that have to be changed in order to obtain corresponding complement labelings cannot be equal. Thus, every transformation algorithm must distinguish between the two labelings, and in order to do so we show that it must send messages on at least $\Omega(n^2)$ edges.

Let $n \equiv 8 \pmod{24}$. Partition $V$ into two disjoint sets $X = \{x_1, x_2, \ldots, x_m\}$ and $Y = \{y_1, y_2, \ldots, y_m\}$, where $m = \frac{n}{2}$. Note that $|X| = |Y| = 4 \pmod{12}$. Let $G_1 = (X, E_1)$, and $G_2 = (Y, E_2)$ be the subgraphs induced by $X$ and $Y$, respectively.

Let $<A_1, A_2, \ldots, A_{m-1}>$ be an edge division of $E_1$; $<B_1, B_2, \ldots, B_{m-1}>$ is obtained from $<A_1, A_2, \ldots, A_{m-1}>$ by replacing every $x_i$ by $y_i$. We first define two matching labelings.

**First matching labeling ($l_1$):**

Each $A_i$ and $B_i$ consists of disjoint 4-cliques. Let $x_a, x_b, x_c$ and $x_d$ be vertices of a 4-clique that belongs to $A_i$ ($B_i$). Since in a matching labeling $l(a, b) = 2$, for all $a$ and $b$, it is sufficient to define it for $a < b$. We define $l_1$ as follows:

$$l_1(x_a, x_b) = l_1(y_a, y_b) = l_1(x_c, x_d) = l_1(y_c, y_d) = 2i - 2,$$
$$l_1(x_b, x_d) = l_1(y_b, y_d) = l_1(x_a, x_c) = l_1(y_a, y_c) = 2i - 1.$$
for $1 \leq i < j \leq m$, 
\[ l_1(x_i, y_j) = l_1(x_j, y_i) = n - l_1(x_i, x_j), \]
and for every $1 \leq i \leq m$, 
\[ l_1(x_i, y_i) = n. \]

The labeling $l_1$ for $n = 32$ and $i = 3$, is depicted in Figure 3.

Second matching labeling ($l_2$):

Let $x_a, x_b, x_c, x_d, y_a, y_b, y_c$ and $y_d$ be as above. The second labeling $l_2$ is derived from $l_1$ by changing the labels of the edges induced by exactly one set \{ $x_a, x_b, x_c, x_d, y_a, y_b, y_c, y_d$ \} in the following way (denote $\bar{x}_a, \ldots, \bar{y}_d$ as the members of this set). Suppose $\bar{x}_a, \bar{x}_b, \bar{x}_c$ and $\bar{x}_d$ ($y_a, y_b, y_c$ and $y_d$) belong to $A_{x_i}(B_{i1})$. Define $l_2$ as follows:

\[ l_2(\bar{x}_a, \bar{x}_b) = l_2(\bar{y}_a, \bar{y}_b) = l_2(\bar{x}_c, \bar{x}_d) = l_2(\bar{y}_d, \bar{y}_c) = 3i_0 - 2, \]
\[ l_2(\bar{x}_a, \bar{y}_b) = l_2(y_a, y_b) = l_2(\bar{x}_c, \bar{y}_d) = l_2(y_c, y_d) = 3i_0 - 1, \]
\[ l_2(\bar{x}_a, \bar{x}_c) = l_2(\bar{y}_a, \bar{y}_c) = l_2(\bar{x}_b, \bar{x}_d) = l_2(y_b, y_d) = 3i_0, \]
\[ l_2(\bar{x}_a, \bar{y}_d) = l_2(\bar{y}_a, \bar{x}_d) = l_2(y_a, y_d) = l_2(\bar{x}_b, \bar{y}_c) = n - (3i_0 - 2), \]
\[ l_2(\bar{x}_a, \bar{y}_c) = l_2(\bar{y}_a, \bar{x}_c) = l_2(y_a, y_c) = l_2(\bar{x}_b, \bar{x}_d) = n - (3i_0 - 1), \]
\[ l_2(\bar{x}_a, \bar{y}_b) = l_2(y_a, y_b) = l_2(\bar{x}_c, \bar{y}_d) = l_2(y_c, y_d) = n - 3i_0. \]

and
\[ l_2(i, j) = l_1(i, j) \]

for all other cases. The $f$-clique changed in $l_2$ will be denoted by $G(l_2)$. Figure 4 shows the labels that were changed in $l_2$ for the $f$-clique of Figure 3 (namely $i_0 = 3$).

We now return to the discussion. Let $G$ be labeled with $l_1$. At each set \{ $x_a, x_b, x_c, x_d, y_a, y_b, y_c, y_d$ \} as above, the edges labeled $3i - 2$ and their complements constitute two edge-disjoint cycles, as depicted in Figure 5. The edges labeled $3i - 1, 3i$ and their complements, respectively, constitute similar cycles. In order to change $l_1$ into a complement labeling, the following actions should be taken on each of the above cycles: two vertices at distance two apart should exchange the labels (as known to them) of the edges adjacent to them of the cycle. For example, the labelings
Figure 3: The first labeling \( l_1 \).

Figure 4: The second labeling \( l_2 \).
Figure 5: The 4-cycles in $l_1$. on the left cycle of Figure 5 will be changed to one of the labelings shown in Figure 6. If $G$ is labeled with $l_2$, the edges labeled $3i_0-2$, $3i_0-1$, $3i_0$ and their complements, respectively, still constitute cycles as before, but the vertices in the corresponding cycles in $G(l_2)$ are ordered differently. Every choice of two vertices in $G(l_2)$ that will exchange their labels when $G$ is labeled with $l_1$, yields a faulty choice when $G$ is labeled with $l_2$. (A choice is faulty if it causes two adjacent vertices, on a certain cycle, to exchange their labels.) In this case, the exchange of the labels will not yield a complement labeling since the edge connecting these two vertices will have the same label at

![Diagram of 4-cycles in $l_1$]

Figure 6: The resulting complement labels of $l_1$. 

Technion - Computer Science Department - Technical Report CS0378 - 1985
its two endpoints, for example, if \( x_a, y_a, z, y_c \) where chosen for the labels \( 3i-2 \) and \( n-(3i-2) \), then with \( l_2, x_c, y_a \) are adjacent on a \( 3i_0-2, n-(3i_0-2) \) cycle: \[ l_2(x_c, y_a) = l_2(y_a, x_c) = n-(3i_0-2) \]

By checking all the possible transformations on \( G \) we can show that every choice of vertices for a transformation from the matching labeling \( l_1 \) to a complement labeling is a faulty choice if the transformation is applied on \( G \) with \( l_2 \).

Let \( A \) be an algorithm for transforming a matching labeling to a complement labeling, \( G' (G'') \) be a complete graph labeled with \( l_1 (l_2) \). Let \( e_1 \) be an execution of algorithm \( A \) that does not send any messages on the edges of \( G(l_2) \). There exists an execution \( e_2 \) of \( A \) on \( G'' \) which yields the same results as \( e_1 \). A processor cannot distinguish between the endpoints of its adjacent unused edges, therefore since all the edges of \( G(l_2) \) are unused in \( e_1 \) (and the set of labels at each processor is identical), the processors cannot determine whether \( G(l_2) \) is labeled with \( l_1 \) or with \( l_2 \). Thus applying \( A \) on \( G' \) and on \( G'' \) must yield the same results. But this contradicts the previous discussion (saying that no two such resulting labelings can be equal).

We conclude that in every transformation, messages have to be sent on at least one edge of every possible subgraph \( G(l_2) \). Since these \( G(l_2) \)'s are edge-disjoint, and there are \( \frac{m-1}{3} \times \frac{m}{4} = \mathcal{O}(n^2) \) of them, we obtain a lower bound of \( \Omega(n^2) \) edges used by any transformation algorithm. This completes the proof.

Q.E.D.

By starting with two complement labelings corresponding to the matching labelings \( l_1 \) and \( l_2 \), a similar proof is obtained for the reverse transformation:

**Theorem 2**: The edge complexity of transforming a complement labeling to a matching labeling is at least \( \Omega(n^2) \).

### 4. LOWER BOUNDS FOR THE MATCHING LABELING

Given a complete graph \( K_n \) we prove here lower bounds of \( \Omega(n^2) \) for constructing a minimum-weight spanning tree given a matching labeling, for verifying whether the given labeling is a matching labeling, and for obtaining such a labeling.
Theorem 3: The edge complexity of constructing a minimum-weight spanning tree, given a matching labeling, is at least $\Omega(n^2)$.

Proof: The proof is a modification of the one in [KMZ]. It is based on the fact that, although unused edges are distinguishable (due to their distinct labels), still in certain situations a processor cannot tell whether their other endpoints are exchanged. We show two different graphs, with different total weight of their minimum-weight spanning tree. Thus, in order to construct a MST every algorithm must distinguish between these graphs and in order to do so we show that it must send messages on at least $\Omega(n^2)$ edges.

Let $G$ be a complete graph $K_n$ labeled with $l_1$ (see section 3). The first graph $\hat{G}$ is constructed from $G$ by giving weights on its edges as follows: the edges in $E_1$ and $E_2$ are weighted 0, and all the other edges of $G$ are weighted 1. The second graph $\tilde{G}$ is constructed from $G$ by exchanging the labels and weights of four edges as follows: let $x_a, x_b, y_a, y_b$ be a cycle in $G$. The new labels and weights are (as depicted in Figure 7):

\begin{align*}
\bar{I}(x_a, x_b) &= l_1(x_a, y_b), \\
\bar{I}(x_a, y_b) &= l_1(x_b, x_a), \\
\bar{I}(y_a, y_b) &= l_1(y_a, x_b), \\
\bar{I}(y_a, x_b) &= l_1(y_b, y_a), \\
\bar{w}(x_a, x_b) &= \bar{w}(y_a, y_b) = 1, \\
\bar{w}(x_a, y_b) &= \bar{w}(y_a, x_b) = 0.
\end{align*}

Let $A$ be an algorithm for constructing a MST, and $e_1$ an execution of $A$ on $G$ that does not send any messages on the edges of the above cycle. The same execution is also possible on $\hat{G}$, since a processor cannot distinguish between the endpoints of its adjacent unused edges. But this is a contradiction since the weight of a MST of $\hat{G}$ is 1, while the weight of a MST of $\tilde{G}$ is 0.

Thus, in order to construct a MST every algorithm must send messages on at least one edge of every one of the above cycles. But there are $O(n^2)$ (see [KMZ]) such disjoint 4-cycles and so we obtain a lower bound of $\Omega(n^2)$.

Q.E.D.
Theorem 4: The edge complexity of determining whether a given labeling is a matching labeling is at least $\Omega(n^2)$.

Proof: For contradiction, assume that there exists an algorithm $A$ which determines whether a given labeling in a complete network is a matching labeling, without using $\Omega(n^2)$ edges. Let $v, s, t$ be three distinct vertices, where the edges $(v, s), (v, t)$ are unused in an execution of $A$ on a network which is labeled with a matching labeling. If we exchange the labels $l(v, s)$ and $l(v, t)$, and apply $A$ again, it might produce the same answer, since the endpoints of the edges are indistinguishable (and so a processor cannot determine whether the above exchange took place or haven't); but now the labeling is no longer a matching labeling, a contradiction.

Q.E.D.

In a similar way it can easily be shown that

Theorem 5: The edge complexity of constructing a matching labeling is at least $\Omega(n^2)$.

All our proofs are based on the fact that the same execution of an algorithm is executed on two different graphs, and the fact that the execution is synchronous will not change this argument. Therefore, all our lower bounds apply also for synchronous networks.

Acknowledgement: We thank Ornan-Gersel for proposing the matching labeling.
REFERENCES


