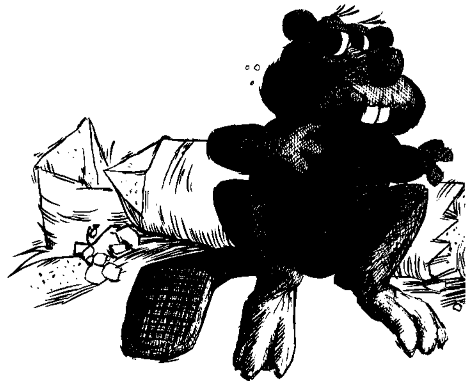
UNIVERSITY OF WATERLOO COMPUTER SCIENCE DEPARTMENT
UNIVERSITY OF WATERLOO COMPUTER SCIENCE DEPARTMENT
UNIVERSITY OF WATERLOO COMPUTER SCIENCE DEPARTMENT
UNIVERSITY OF WATERLOO COMPUTER SCIENCE DEPARTMENT



Average Complexity of a Distributed Orientation Algorithm

Violet Syrotiuk
Jan Pachl

Research Report
CS-87-23

March, 1987

Average Complexity of a Distributed Orientation Algorithm*

Violet Syrotiuk
Jan Pachl

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

ABSTRACT

We describe and analyze an asynchronous distributed algorithm for finding a consistent orientation in a ring of anonymous processes. In rings of N processes, the algorithm sends $O(N^{3/2})$ messages on average. Attiya, Snir and Warmuth proved that every such algorithm requires sending $\Omega(N^2)$ messages in the worst case.

* This work was supported by the Natural Sciences and Engineering Research Council of Canada.

Average Complexity of a Distributed Orientation Algorithm

Violet Syrotiuk
Jan Pachl

Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
N2L 3G1

1. The ring orientation problem

In this paper we consider the following *orientation problem,* which was introduced and studied by Attiya, Snir and Warmuth [1]: A number of processes are arranged in a ring configuration, in which each process is connected by communication channels to its two neighbors. The processes communicate by messages sent through the channels. Every process can discriminate between the two channels visible to it; thus in every process one channel may be called *left* and the other *right*. However, these local orientations need not be globally consistent. The goal is to reach an agreement, among all the processes, on a consistent orientation of the ring.

The processes are indistinguishable from each other, and all execute the same program; in the terminology of [1], the ring is *anonymous*. We also assume that the configuration is asynchronous. That is, the execution in different processes is not synchronized, and there is no bound on the message delivery time (although every message is eventually delivered).

By Theorem 4.4 in [1], any asynchronous algorithm to solve the orientation problem requires $\Omega(N^2)$ messages to be sent in the worst case. On the other hand, as is shown in [1], there is an algorithm that executes correctly in every *synchronous* ring, and needs only $O(N \log N)$ messages in the worst case. In this paper we describe an asynchronous algorithm and show that it sends $O(N^{3/2})$ messages on average.

2. An orientation algorithm

By Theorem 3.1 in [1], there is no algorithm that would solve the orientation problem in all rings without some information about the number N of processes in the ring. Thus from now on we assume that N is known to every process when the execution begins.

In each process, one channel is named *left* and the other *right*. Two processes in the ring have the same orientation if their *right* channels point in the same direction (either both clockwise or both counterclockwise).

Of the two possible ring orientations, the algorithm described below selects the one that is shared by the majority of processes. If N is even and there is no majority agreement, then the algorithm signals that there is no majority and does not select an orientation. This incompleteness is inherent in the problem; indeed, by Theorem 3.5 in [1] there is no orientation algorithm that would work correctly for all rings of N processes when N is even.

As has become customary in the study of "symmetry-breaking" distributed algorithms, we measure the complexity of the algorithm by the number of messages that are sent, under the assumption that all processes begin their execution simultaneously. For the sake of simplicity we make the same assumption in our description of the algorithm.

The algorithm terminates with an indication, in at least one process, that the orientation in the process agrees with the majority orientation ("The majority agrees.") or that there is no majority. (It would be straightforward to add another phase to the algorithm, to send the result to every process, at the cost of additional N messages.)

In our description of the algorithm, the processes execute operations *send* and *receive* to send and receive messages. The statement *send(dir, msg),* where $dir = left$ or $right$, sends the message *msg* to the channel specified by *dir*. The statement *from* := *receive(msg)* copies the next message received from either channel to the variable *msg* and returns either $from = left$ or $from = right$ to indicate the source of the message.

Every message *msg* in the algorithm has two fields: $msg = [vote, distance]$. Initially every process sends a message to its right channel. Each of these initial messages begins a chain of messages forwarded from process to process in one direction around the ring. The *vote* field records the number of favorable and unfavorable local orientations along the chain. The chain is terminated when the number of unfavorable local orientations exceeds the number of favorable ones. The *distance* field measures the length of the chain; it allows a process to detect that a received message is part of a chain that spans the whole ring.

When all processes begin their execution simultaneously, every process executes the following program:

```
constant   N
variables  vote, distance, from

begin
    send( right, [1, 1] )
    repeat
        begin
            from := receive( [vote, distance] )
            if distance = N then goto DONE
            if from = left then
                send( right, [vote+1, distance+1] )
            if from = right and vote > 0 then
                send( left, [vote-1, distance+1] )
        end

    DONE :
    if vote > 0   then claim "The majority agrees."
    if vote = 0   then claim "No majority."
end
```

3. The correctness of the algorithm

Both the proof of correctness in this section and the analysis in the next are based on combinatorial results about sequences of ones and zeros.

When s is a sequence of ones and zeros, denote by $\#_1 s$ and $\#_0 s$ the number of ones and the number of zeros in s, respectively. In [2], a nonempty sequence s of ones and zeros is called *dominating* if $\#_1 t > \#_0 t$ for every nonempty prefix t of s. Say that s is *weakly dominating* if $\#_1 t \geq \#_0 t$ for every prefix t of s.

that order, to the processes around the ring in the clockwise direction. For $1 \leq i \leq N$, define s_i to be 1 if the right channel in the process labeled i points clockwise, and 0 if the right channel in the process points counterclockwise. Define s to be the sequence $s_1 \, s_2 \, \cdots \, s_N$ of ones and zeros.

Since the right channel points clockwise in at least $N/2$ processes, we have $\#_1 s \geq \#_0 s$. By 3.1, at least one cyclic permutation $s_k \, s_{k+1} \, \cdots \, s_{k-1}$ of s is weakly dominating. Therefore the chain of messages starting at the process labeled k reaches all the way around the ring. (Since $s_k = 1$, the chain moves clockwise.) Hence the process labeled k receives a message $[vote, N]$. (The value $vote$ in the message is the difference between the number of ones and the number of zeros in s.)

[]

4. The message complexity of the algorithm

In this section we count the number of messages sent when the algorithm is executed in rings of N processes. Recall that we assume that all processes begin their execution simultaneously.

If all local orientations in the ring agree then every process starts a message chain of length N; thus N^2 messages are sent. In fact, by the lower bound cited above (Theorem 4.4 in [1]), *every* algorithm that solves the orientation problem sends $\Omega(N^2)$ messages for some choice of local orientations.

In the rest of the paper we study the *average* number of messages in rings of N processes. The average is computed over all choices of local orientations. Each such choice is called a *ring configuration* in the sequel. There are 2^N ring configurations; each can be represented by a sequence of ones and zeros as in the proof of 3.2. In the proof of Theorem 4.4 below we exploit a correspondence between clockwise chains of messages sent by the algorithm and weakly dominating sequences of ones and zeros.

The following result is proved in section 2.4 of [3].

4.1. Lemma. Let p and q be two integers, $p \geq q \geq 0$. Among the $\binom{p+q}{q}$ sequences s such that $\#_1 s = p$ and $\#_0 s = q$, exactly $\binom{p+q}{q} \frac{p+1-q}{p+1}$ sequences are weakly dominating.

[]

For $N>0$, let $L(N)$ be the set of all sequences of ones and zeros of length N, and let Pref(N) be the set of all nonempty prefixes of cyclic permutations of the sequence $1\ 2\ \cdots\ N$. When $s = s_1 s_2 \cdots s_N \in L(N)$ and $\alpha = \alpha_1 \alpha_2 \cdots \alpha_k \in \mathrm{Pref}(N)$, define

$$s[\alpha] = s_{\alpha_1} s_{\alpha_2} \cdots s_{\alpha_k}\ .$$

For $s \in L(N)$, denote by $D(s)$ the cardinality of the set

$$\{\, \alpha \in \mathrm{Pref}(N) \mid s[\alpha] \text{ is weakly dominating} \,\}\ .$$

The identity below will be useful in the proof of 4.3. It may be found in [4] as equation (1.109).

4.2. Lemma.

$$\sum_{k=0}^{n} 2^{-2k} \binom{2k}{k} = \frac{2n+1}{2^{2n}} \binom{2n}{n}$$

Proof. This may be proved by induction.

4.3. Lemma. For any integer $N>0$,

$$\sum_{s \in L(N)} D(s) = 2^N \left[2\sqrt{2/\pi}\ N^{3/2} + O(N) \right].$$

Proof. When $s \in L(N)$ and $p+q \leq N$, denote by $D_{p,q}(s)$ the cardinality of the set

$\{ \alpha \in \operatorname{Pref}(N) \mid \#_1 s[\alpha] = p,\ \#_0 s[\alpha] = q,\ \text{and}\ s[\alpha]\ \text{is weakly dominating} \}.$

For fixed p and q, $p+q \le N$, there are $\binom{p+q}{q}$ sequences t (of length $p+q$) such that $\#_1 t = p$ and $\#_0 t = q$. For each such t there are $N 2^{N-(p+q)}$ pairs (s,α) such that $s \in L(N)$, $\alpha \in \operatorname{Pref}(N)$, and $t = s[\alpha]$. Therefore, by 4.1,

$$\sum_{s \in L(N)} D_{p,q}(s) = N 2^{N-(p+q)} \binom{p+q}{q} \frac{p+1-q}{p+1}$$

when $p \ge q$. Obviously $D_{p,q}(s) = 0$ when $p < q$. Substituting $k = p+q$, we obtain

$$\begin{aligned}
\sum_{s \in L(N)} D(s) &= \sum_{k=0}^{N} \sum_{q=0}^{k} D_{k-q,q}(s) \\
&= \sum_{k=0}^{N} \sum_{q=0}^{\lfloor k/2 \rfloor} N 2^{N-k} \binom{k}{q} \frac{k+1-2q}{k+1-q} \\
&= N 2^N \sum_{k=0}^{N} 2^{-k} \sum_{q=0}^{\lfloor k/2 \rfloor} \binom{k}{q} \frac{k+1-2q}{k+1-q} \\
&= N 2^N \sum_{k=0}^{N} 2^{-k} \sum_{q=0}^{\lfloor k/2 \rfloor} \binom{k}{q} \left(1 - \frac{q}{k+1-q}\right) \\
&= N 2^N \sum_{k=0}^{N} 2^{-k} \left[\sum_{q=0}^{\lfloor k/2 \rfloor} \binom{k}{q} - \sum_{q=0}^{\lfloor k/2 \rfloor} \frac{k!}{q!\,(k-q)!} \frac{q}{k+1-q} \right] \\
&= N 2^N \sum_{k=0}^{N} 2^{-k} \left[\sum_{q=0}^{\lfloor k/2 \rfloor} \binom{k}{q} - \sum_{q=0}^{\lfloor k/2 \rfloor} \binom{k}{q-1} \right]
\end{aligned}$$

$$= N2^N \sum_{k=0}^{N} 2^{-k} \left[\sum_{q=0}^{\lfloor k/2 \rfloor} \binom{k}{q} - \sum_{q=0}^{\lfloor k/2 \rfloor - 1} \binom{k}{q} \right]$$

$$= N2^N \sum_{k=0}^{N} 2^{-k} \binom{k}{\lfloor k/2 \rfloor}$$

$$= N2^N \left[\sum_{\substack{k=1 \\ k \text{ odd}}}^{N} 2^{-k} \binom{k}{\lfloor k/2 \rfloor} + \sum_{\substack{k=0 \\ k \text{ even}}}^{N} 2^{-k} \binom{k}{\lfloor k/2 \rfloor} \right]$$

$$= N2^N \left[\sum_{l=1}^{\left\lfloor \frac{N+1}{2} \right\rfloor} 2^{-(2l-1)} \binom{2l-1}{l} + \sum_{l=0}^{\lfloor N/2 \rfloor} 2^{-2l} \binom{2l}{l} \right]$$

$$= N2^N \left[\sum_{l=1}^{\left\lfloor \frac{N+1}{2} \right\rfloor} 2^{-2l} \binom{2l}{l} + \sum_{l=0}^{\lfloor N/2 \rfloor} 2^{-2l} \binom{2l}{l} \right]$$

By Lemma 4.2,

$$\sum_{s \in L(N)} D(s) = N2^N \left[\left[\sum_{i=1}^{2} \frac{2\,N_i+1}{2^{2\,N_i}} \binom{2\,N_i}{N_i} \right] - 1 \right]$$

where $N_1 = \left\lfloor \frac{N+1}{2} \right\rfloor$ and $N_2 = \left\lfloor \frac{N}{2} \right\rfloor$.

By Stirling's approximation,

$$\binom{2n}{n} = \frac{2^{2n}}{\sqrt{\pi\, n}} \left(1 + O\left(\frac{1}{n}\right) \right)$$

and therefore

$$\sum_{s \in L(N)} D(s) = 2^N \left[\left[\sum_{i=1}^{2} \frac{N\,(2N_i+1)}{\sqrt{\pi\, N_i}} \left(1 + O\left(\frac{1}{N_i} \right) \right) \right] - N \right].$$

Since $N_1 = \dfrac{N}{2} + O(1)$ and $N_2 = \dfrac{N}{2} + O(1)$, we obtain

$$\sum_{s \in L(N)} D(s) = 2^N \left[2\sqrt{2/\pi}\ N^{3/2} - N + O(\sqrt{N}) \right]$$

$$= 2^N \left[2\sqrt{2/\pi}\ N^{3/2} + O(N) \right]$$

[]

We are now ready to compute $avg(N)$, the average number of messages sent by the algorithm in rings of N processes. The average is computed over all 2^N ring configurations.

4.4. Theorem. For any integer $N>0$,

$$avg(N) = 4\sqrt{2/\pi}\ N^{3/2} + O(N).$$

Proof. We first count the total number of messages sent clockwise in all ring configurations of a fixed size N. By symmetry, the same total number of messages is sent counterclockwise.

As in the proof of 3.2, we assign labels $1,2,\ldots,N$, in that order, to the processes around the ring in a clockwise direction, and we define s_i, $1 \le i \le N$, to be one or zero when the right channel in the process labeled i points clockwise or counterclockwise, respectively.

It follows from the definition of the algorithm in section 2 that a message $[v,d]$ is sent clockwise by the process labeled k if and only if, for the sequence $\alpha \in Pref(N)$ such that the length of α is d and the last element of α is k, $s[\alpha]$ is weakly dominating (in that case $v = \#_1\alpha - \#_0\alpha$). Moreover, two different messages sent (clockwise) correspond this way to two different sequences in $Pref(N)$.

Thus, the total number of messages sent clockwise in all ring configurations is

$$\sum_{s \in L(N)} D(s),$$

and the total number of messages sent both clockwise and counter-clockwise is

$$2 \sum_{s \in L(N)} D(s).$$

Thus,

$$avg(N) = \frac{1}{2^N} \, 2 \sum_{s \in L(N)} D(s)$$

and by Lemma 4.3

$$avg(N) = 4 \sqrt{2/\pi} \; N^{3/2} + O(N)$$

[]

References.

[1] C. Attiya, M. Snir, and M. Warmuth, *Computing on an Anonymous Ring*, Technical Report UCSC-CRL-85-3, Computer Research Lab, Department of Computer Science, University of California, Santa Cruz, November 1, 1985.

[2] N. Dershowitz and S. Zaks, *The Cycle Lemma and Some Applications*, July 1984.

[3] A. Dvoretzky and T. Motzkin, *A Problem of Arrangements*, Duke Mathematical Journal, Volume 14, pp. 305-313, 1947.

[4] H.W. Gould, *Combinatorial Identities*, Morgantown, West Virginia, Morgantown Printing and Binding Company, 1972.