THE USE OF A SYNCHRONIZER YIELDS MAXIMUM COMPUTATION RATE IN DISTRIBUTED NETWORKS

by

S. Even and S. Rajsbaum

Technical Report #659

December 1990
The Use of a Synchronizer Yields Maximum Computation Rate in Distributed Networks

Shimon Even** and Sergio Rajsbaum***

Department of Computer Science
Technion- Israel Institute of Technology
Haifa, Israel 32000

ABSTRACT

In a previous paper we analyzed the performance of networks with negligible transmission delay, whose operation is controlled by a simple synchronizer. It was shown that full speed is achieved, for any wake-up pattern, by letting the network run free, without the use of a "firing squad" mechanism or a scheduler.

In this paper we investigate the effect of fixed delays in the communication channels on the performance of a network in which there is a global clock, but there is no global start-up signal. We show that here too, maximum rate of computation is always reached, just by using the synchronizer and letting the network run free. To a certain extent, the wake-up pattern may influence the length of the transitory stage and the periodicity of the steady state, but not the ultimate rate.

1. Introduction

Background

Assume a program has been written for a given synchronous network of processors. On a global start-up signal, all processors start computing simultaneously. On every beat of the global clock each processor, according to its program, performs one computational step and sends messages to some of its neighbors. The transmission delay in the communication channels (edges) is such that all messages arrive at their destinations in time to be used in the next computational step. The exact nature of the program and its purpose, is of no concern to us. Also, we assume the

** Part of this work was done while the authors visited the Computer Science Program, University of Texas at Dallas, Richardson, TX.
*** Supported by the Fund for the Promotion of Research at the Technion.

Tel: (972-4) 294 450

Csnet: event@cs.technion.ac.il

Bitnet: even@TECHSEL

Now visiting Bellcore, Morristown, N.J.

Tel: (972-4) 294 359

Csnet: raisbaum@acmit.technion.ac.il

Bitnet: raisbaum@75CHUNIX

Technion - Computer Science Department - Technical Report CS0659 - 1990
processors, and the communication channels are completely reliable.

If one wants to run the same program on an asynchronous network of the same topology, where no global start-up signal exists, and where transmission delays are unpredictable, certain measures must be taken to ensure the correctness of the computation.

Following Myhill (see [M]), who posed the Firing Squad Problem in 1957, it has been widely believed that such a start-up mechanism is necessary to compensate for the lack of a global start-up signal.

The use of synchronizers was suggested by Awerbuch [A] to overcome the difficulties which stem from the lack of a global clock and the unpredictability of the transmission delays. His paper proposes several specific methods for implementing a simulation of this type, and studies their complexities.

The highest rate of computation which can be achieved by a network, in which the delays are fixed and known, was studied by Reiter [R] in 1968. He showed that the rate is bounded by the inverse of the maximum cycle mean delay of the network and that it is possible to schedule the computational steps of the processors so that this maximum rate is achieved. However, this implies the use of a centralized scheduler.

Our Approach

We consider the use of a simple synchronizer, which is inspired by the model of marked graphs (see Geneich [GE] or Commoner, et al. [CHEP]). A similar mechanism was used by Chandy and Lamport [CL]. Each processor waits for messages to arrive from all its in-neighbors before it performs the next computational step. (It is assumed, for example, that every message is followed by an "end-of-message" marker, even if the message is empty.) Messages which arrive through an edge are stored in a FIFO buffer. The validity of the computation is obvious, but the rate of computation of the whole network remains to be analyzed.

The synchronizers suggested by Awerbuch [A] require that all edges are bidirectional. Our synchronizer is similar to synchronizer alpha in [A], but can be used also in directed networks. We believe that our synchronizer is the simplest one to capture the essence of the synchronizer methodology.

Observe that this simple synchronizer is a distributed device, which lets the network run free, locally delaying the computation only as long as necessary. We argue that this local device is sufficient to take care of the lack of a start-up signal and to guarantee that the optimal rate of computation is reached within a reasonable transitory time.

We break the general problem to subcases according to the following parameters:

(1) Clocks.

(1.1) A global clock exists.

(1.2) There is no global clock but the individual clocks, which may not be synchronized, all run at the same rate.
Our Previous Results

In our previous paper, [ER], we analyzed the case of networks in which the transmission delays are positive, but negligible (case (2.1)).

In the absence of a global start-up signal, we showed that the use of the synchronizer guarantees that within $2V1$ units of time (in a network of $V1$ processors) full rate operation is reached. This holds when there is a global clock (case (1.1)), and also when the local clocks run at the same rate, but are not synchronized (case (1.2)). In the former case the processors end up working in unison, as if there was a global start-up signal, proving that the use of a firing-squad mechanism is superfluous.

When no assumption is made about the rate of the local clocks (case (1.3)), we showed that after a transitory period, the rate of operation of the network is not slower than the rate of any sluggish clock. (A sluggish clock is such that between two consecutive beats of it, every clock in the network beats at least once.)

The New Results

In this paper we assume case (1.1) and investigate the effect of fixed delays (case (2.2)), on the performance of the network. However, similar results hold if instead of assuming case (1.1), one assumes case (1.2). We show that here too, maximum rate of computation is always reached, just by using the synchronizer. We present an $O(1V1\cdot1E1)$ algorithm to compute the rate of the network. It follows that the rate of the network is at least $1 / \Delta$, where $\Delta$ is the greatest delay of any edge, regardless of the number of processors, or the topology of the network.

To a certain extent, the wake-up pattern may influence the length of the transitory stage and the periodicity of the steady state, but not the ultimate rate. We show that the number of messages sent on each of the edges during the transitory stage is $O(\Delta \cdot1V1^3)$, and that this bound is tight, even if there is a global start-up signal. This is in contrast with the case of negligible delays (2.1), where the length of the transitory stage is $O(1V1)$.

We show that the periodicity may be exponential in the number of processors and that the number of messages, sent in one period, is related to the least common multiple of the lengths of certain cycles in the network.
Our results concerning the rate apply not only to systems with fixed delays. If upper bounds on the delays are known, then our results yield lower bounds on the performance of the network. If the average delays are known, regardless of their distribution, our results yield upper bounds on the average performance of the network; see [RS], where (2.3) is studied.

The Model

The network, \(N\), consists of a finite, directed and strongly connected graph \((V, E)\). Each vertex is a processor running its own program, and each edge is a communication link. The processors communicate by sending messages along the communication links. The beats of the global clock are heard simultaneously by all the processors (case (1.1)). For simplicity, we assume that the clock beats on every integral time. Let the time at which the first processor of \(N\) wakes up (and sends message \(W\)) be equal to 0.

Initially, all processors are in a quiescent state, \(Q\), in which they send no messages, and stay in this state until either the processor wakes up spontaneously (this may be caused by a message to the processor from the outside world, not included in our model) or upon receiving a wake-up message, \(W\), through one of its incoming edges. The processor becomes active and never returns to be quiescent. On the next beat, the processor sends \(W\) on all its outgoing edges; i.e. the wake-up signal propagates by broadcast.

Let us denote by \(M_i\) the messages sent as a result of the \(i\)-th computational step. Note that the contents of the messages is of no interest to us; what matters is to which step of the computation the message corresponds.

An active processor \(p\), after having sent \(W\), on the next beat sends on all its outgoing edges its initial computational messages \(M_0\). Assuming that \(p\) has already sent \(M_i, i \geq 0\), once \(M_i\) has arrived on all its incoming edges, on the next beat, \(p\) produces messages \(M_{i+1}\), and sends them on all its outgoing edges. If \(M_i\) has been sent, but \(M_{i+1}\) has not been received on all its incoming edges, then on the next beat no computational step takes place, and no messages are sent; in this case we say that the beat is skipped.

The delay of an edge \(p \rightarrow q\) is the integer \(d(p, q) \geq 1\), or simply \(d\), if whenever \(p\) sends a message at time \(t\), \(q\) receives it in the time-interval \([t + d - 1, t + d]\).
Example

A directed cycle $p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_1$, where $d(p_1, p_2) = d(p_3, p_1) = 1$, $d(p_2, p_3) = 2$, and $p_1$ is the only one to wake up spontaneously.

<table>
<thead>
<tr>
<th>beat</th>
<th>$p_1$</th>
<th>$p_2$</th>
<th>$p_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>1</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>-</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>3</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>-</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>4</td>
<td>-</td>
<td>4</td>
</tr>
</tbody>
</table>

3.1 Analysis of the Ultimate Computational Rate

For a processor $p \in N$, denote by $N(p, t)$ the number of beats during which $p$ has sent $M_i$ messages (i.e., messages different from $W$), up to time $t$, inclusive. Clearly, for every $p$, $N(p, 0) = 0$.

Let $P = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{n+1}$ be a path such that the delay from $p_i$ to $p_{i+1}$, $1 \leq i \leq n$, is $d_i$. We define the delay of $P$, $D(P)$ or simply $D$, as

$$D = \sum_{i=1}^{n} d_i$$

and the rate of $P$, $R(P)$, as

$$R(P) = \frac{n}{D}.$$ 

We shall say that the rate of computation of a network $N, R(N)$, is

$$R(N) = \lim_{t \to \infty} \frac{N(p, t)}{t}$$

where $p$ is a processor in $N$. It will be shown that the limit exists and that its value is independent of $p$. 

Technion - Computer Science Department - Technical Report CS0659 - 1990
Let \( C = p_0 \rightarrow p_1 \rightarrow \cdots \rightarrow p_{n-1} \rightarrow p_0 \) be a directed cycle in \( N \). Let us denote by \( D \) the delay of \( C \).

**Lemma 1**

For every \( p \) in \( C \) and every \( t_1 > 0 \),

\[
N(p, t_1 + D) - N(p, t_1) \leq n.
\]

**Proof**

If \( p \) does not send any \( M_i \) messages in the time-interval \( (t_1, t_1 + D) \) the statement is trivial. Thus, denote by \( t_1 < t \leq t_1 + D \) the first time in the interval, at which \( p \) sends a message \( M_i \).

Without loss of generality, assume \( p = p_0 \). Thus, \( p_1 \) cannot send \( M_{i+1} \) before time \( t + d_0 \). But then, \( p_2 \) cannot send \( M_{i+2} \) before time \( t + d_0 + d_1 \), etc. We see that \( p \) cannot send \( M_{i+n} \) before time \( t + \sum_{i=0}^{n-1} d_i = t + D \). Thus, in the interval \( [t, t_1 + D] \subset [t, t + D) \), \( p \) can send at most \( n \) messages. \( \square \)

**Corollary 1**

For every \( p \) in \( C \) and every \( k \geq 0 \),

\[
N(p, kD) \leq kn.
\]

A simple cycle of \( N \) is called **critical** if no other simple cycle in \( N \) has lower rate. Let \( \hat{C} \) be a critical cycle, and denote its rate by \( \hat{R} \), its delay by \( \hat{D} \) and its length by \( \hat{L} \).

**Theorem 1**

Let \( p \) be a processor of \( N \). Then

\[
\sup_{t \rightarrow \infty} \frac{N(p, t)}{t} \leq \hat{R}.
\]

**Proof**

First we shall prove the theorem for \( p \) on \( \hat{C} \). Let \( I_k = (kD, (k+1)\hat{D}) \), for \( k \geq 0 \).

For \( t \in I_k \), Lemma 1 implies that

\[
\frac{N(p, t)}{t} \leq \frac{N(p, k\hat{D}) + \hat{L}}{k\hat{D}},
\]

and by Corollary 1,

\[
\frac{N(p, t)}{t} \leq \frac{(k+1)\hat{L}}{k\hat{D}}.
\]
We have that

$$\lim_{k \to \infty} \frac{(k + 1)L}{kD} = \frac{L}{D} = \hat{R}.$$ 

Therefore,

$$\sup_{t \to \infty} \frac{N(p, t)}{t} \leq \hat{R}.$$ 

It remains to show that the theorem is true for $p$ not on $\hat{C}$. Let $q$ be a node on $\hat{C}$. Since the network is strongly connected, there is a path from $q$ to $p$, and thus, in the limit, $p$ cannot run faster than $q$. □

Let $t_k(p)$, $k \geq 0$ be the time at which processor $p$ sends $M_k$. Define $t_0$ as the last time on which a message $M_0$ is sent by a processor in $H$:

$$t_0 = \max_{p \in V} \{ t_0(p) \}.$$ 

Theorem 2

Let $C = P_0 \rightarrow P_1 \rightarrow ... \rightarrow P_{n-1} \rightarrow P_0$ be a network which is a simple directed cycle. The rate of computation of this network is $\hat{R}(C)$.

Proof

Without loss of generality, assume that $t_0(p_0) = t_0$, i.e. $P_0$ is a processor of $C$ that is last to send $M_0$. We first show that for all $k \geq 0$, $0 \leq i \leq n - 1$,

$$t_k(p_i) \leq t_0 + \sum_{j=1}^{k} d_{i-j},$$

where summation in the indices is taken modulo $n$.

By induction on $k$. For the induction basis, observe that by definition of $t_0$, for every $0 \leq i \leq n - 1$, $t_0(p_i) \leq t_0$.

Suppose that

$$t_k(p_i) \leq t_0 + \sum_{j=1}^{k} d_{i-j}$$

and

$$t_k(p_{i+1}) \leq t_0 + \sum_{j=1}^{k} d_{i+1-j}.$$ 

Then, $p_{i+1}$ receives $M_k$ by time $t_0 + \sum_{j=1}^{k} d_{i-j} + d_i$, which is equal to $t_0 + \sum_{j=1}^{k+1} d_{i+1-j}$. Since by that time $p_{i+1}$ has already sent $M_k$. 


$t_k(p_{i+1}) < t_0 + \sum_{j=1}^{k+1} d_{i+1-j}$,

then at that time, if not earlier, $p_{i+1}$ sends $M_{k+1}$.

Thus, we have that, for all $k \geq 0$, $0 \leq i \leq n-1$,

$$t_{kn}(p_i) \leq t_0 + kD,$$

and therefore,

$$ N(p_i, t_0 + kD) \geq kn. $$

For $t_0 + kD \leq t < t_0 + (k+1)D$, 

$$ N(p_i, t) \geq N(p_i, t_0 + kD) - \frac{kn}{t} \geq \frac{kn}{t_0 + (k + i)D} $$

But

$$ \lim_{k \to \infty} \frac{kn}{t_0 + (k + i)D} = \frac{n}{D}, $$

hence,

$$ \inf_{t \to \infty} \frac{N(p_i, t)}{t} \geq \frac{n}{D} $$

and by Theorem 1, the rate of computation of the network is $\frac{n}{D} = R(C)$. $\square$

We shall show that $R$, the rate of $N$, exists and is equal to $\hat{R}$. By Theorem 1, the rate of $N$, if it exists, is at most $\hat{R}$. Theorem 3 will provide the remaining claims, but first we prove three lemmas.

For processors $p$, $q$ in $N$, and every nonnegative integer $k$, define $F_k(p, q)$ to be the maximum delay of a path of length $k$ from $p$ to $q$. If no such path exists, then $F_k(p, q) = -\infty$. Also, $F_0(p, p) = 0$. Define $F_k(q)$ to be the maximum of $F_k(p, q)$, over all $p \in N$.

Lemma 2

For every processor $q$ in $N$, $q$ sends $M_k$ by time $t_0 + F_k(q)$, that is,

$$ N(q, t_0 + F_k(q)) \geq k + 1. $$

Proof

By induction on $k$. 

Technion - Computer Science Department - Technical Report CS0659 - 1990
For the basis, observe that \( F_0(q) = 0 \).

Let \( P \) the set of processors which have an edge towards \( q \), and let \( d(p, q) \) be the delay of the edge from \( p \) to \( q \).

Suppose that for every processor \( r \), \( t_k(r) \leq t_0 + F_k(r) \). Define \( \tau \) by

\[
\tau = \max_{p \in P} \{ t_0 + F_k(p) + d(p, q) \}.
\]

It is easy to see that

\[
\tau = t_0 + F_{k+1}(q).
\]

From the definition of \( \tau \), it follows that \( q \) will receive \( M_k \) from all processors in \( P \) before \( \tau \). It remains to show that \( q \) will send \( M_{k+1} \), i.e. \( t_{k+1}(q) \leq t_0 + F_{k+1}(q) \).

By the induction hypothesis, \( t_k(q) \leq t_0 + F_k(q) \). But \( t_0 + F_k(q) < t_0 + F_{k+1}(q) \), since the delays on the edges are positive. Hence, by \( t_0 + F_{k+1}(q) \), \( q \) will send \( M_{k+1} \), i.e. \( t_{k+1}(q) \leq t_0 + F_{k+1}(q) \).

**Lemma 3**

For every (not necessarily simple) cycle \( P = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_n \rightarrow p_1 \)

\[
R(P) = \hat{R}.
\]

**Proof**

If \( P \) is simple, then the Lemma follows from the definition of \( \hat{R} \). Now, assume \( P \) is not simple. Remove a simple cycle from \( P \) in the following way. Let \( j \leq n \) be the least index such that \( p_j = p_1 \), \( i < j \). Clearly, \( C_1 = p_i \rightarrow p_{i+1} \rightarrow \cdots \rightarrow p_j \) is a simple cycle. Remove from \( P \) all the edges of \( C_1 \) to obtain a shorter cycle. Repeat this procedure until no more edges are left, obtaining simple cycles \( C_2, \ldots, C_k \). Denote by \( L(C_i), i \leq i \leq k \), the number of edges of \( C_i \).

We could have done this procedure on a path which is not a cycle, say from \( s \) to \( t \). In this case, after extracting the simple cycles from the path, a simple path from \( s \) to \( t \) would remain. We shall refer to this procedure as a decomposition of the path.

For each cycle \( C_i \)

\[
D(C_i) \leq \frac{L(C_i)}{\hat{R}}.
\]

Each edge of \( P \) appears in exactly one cycle. Thus, the rate of cycle \( P \) is

\[
R(P) \geq \frac{k L(C_i)}{\sum_{i=1}^{k} \frac{L(C_i)}{\hat{R}}}.
\]
Lemma 4

For any processor \( p \),

\[ N(p, t_0 + k\hat{D} + \Delta |V|) \geq k\hat{L} + |V| + 1. \]

That is, \( p \) sends at least \( k\hat{L} + |V| + 1 \) messages by time \( t_0 + k\hat{D} + \Delta |V| \).

Proof

By Lemma 2, it is sufficient to show that \( F_{\hat{L} + |V|} (p) \leq k\hat{D} + \Delta |V| \), i.e. any path \( P \) of length \( k\hat{L} + |V| \) ending in \( p \) has a delay less than or equal to \( k\hat{D} + \Delta |V| \).

Extract from \( P \) cycles \( C_1, \ldots, C_j \), as in the proof of Lemma 3, ending with a simple path \( P' \), whose start and end processors are the same as those of \( P \). Clearly,

\[ L(P) = L(P') + \sum_{i=1}^{j} L(C_i) \quad \text{and} \quad D(P) = D(P') + \sum_{i=1}^{j} D(C_i). \]

We know that the rate of any simple cycle is bounded by \( \hat{R} \), thus,

\[ D(P) \leq D(P') + \frac{1}{\hat{R}} \sum_{i=1}^{j} L(C_i) \]

\[ = D(P') + \frac{1}{\hat{R}} (k\hat{L} + |V| - L(P')) \]

\[ \leq \Delta L(P') + k\hat{D} + \frac{|V| - L(P')}{\hat{R}} \]

\[ = \Delta L(P') + k\hat{D} + \frac{\hat{D}}{\hat{L}} (|V| - L(P')) \]

\[ \leq \Delta L(P') + k\hat{D} + \Delta (|V| - L(P')) \]
A = k\hat{D} + A I V I ,
which completes the proof of the lemma. □

Theorem 3
If N is strongly connected then R (N) = \hat{R}.

Proof
By Lemma 4, for any processor p
\[ N(p, t_0 + k\hat{D} + \Delta |V|) \geq k\hat{L} + |V| + 1. \]
For \( t_0 + k\hat{D} + \Delta |V| \leq t < t_0 + (k + 1)\hat{D} + \Delta |V| \),
\[ N(p, t) \geq N(p, t_0 + k\hat{D} + \Delta |V|), \]
and
\[ \frac{N(p, t)}{t} \geq \frac{k\hat{L} + |V| + 1}{t_0 + (k + 1)\hat{D} + \Delta |V|}. \]
But
\[ \lim_{t \to \infty} \frac{k\hat{L} + |V| + 1}{t_0 + k\hat{D} + \Delta |V|} = \frac{\hat{L}}{\hat{D}} = \hat{R}. \]
Therefore,
\[ \inf_{t \to \infty} \frac{N(p, t)}{t} \geq \hat{R}. \]
This statement, jointly with Theorem 1, implies that \( \lim_{t \to \infty} \frac{N(p, t)}{t} \) exists, and its value is \( \hat{R} \).
□

Consider the meaning of Theorem 3, in the case of undirected networks, in which the delay of each edge in both directions is the same. We make the following simple observation.

Corollary 2
The rate of an undirected network is 1 / \Delta.
3.2 An Algorithm to Compute the Rate of a Network

Karp [K] describes an $O((|V| \cdot |E|)$ dynamic programming algorithm to find the minimum cycle mean in a strongly connected, directed network $N$ with integral (not necessarily positive) delays. A similar algorithm solves the problem of finding $\hat{\lambda} (= \frac{1}{R})$, the maximum cycle mean. Thus, by Theorem 3, $\hat{\lambda} = \frac{1}{R(N)}$. The validity of this algorithm is based on the following Theorem.

**Theorem K**

Let $N$ be a strongly connected network. For any vertex $q$

$$\hat{\lambda} = \max_{p \in V} \min_{0 \leq k < |V|} \left\{ \frac{F_{|V|}(p, q) - F_k(p, q)}{|V| - k} \right\}. \tag{3.1}$$

**Proof**

Case 1: $\hat{\lambda} = 0$.

In this case, there exists a cycle of zero delay, and there exists no cycle of positive delay. Since there are no positive cycles, there is a maximum-delay path from $p$ to $q$ of length less than $|V|$. Let this maximum delay be $\pi(p, q)$. Then $F_{|V|}(p, q) \leq \pi(p, q)$. Also,

$$\pi(p, q) = \max_{0 \leq k \leq |V| - 1} \{ F_k(p, q) \}. \tag{3.2}$$

Thus,

$$\min_{0 \leq k \leq |V| - 1} \{ F_{|V|}(p, q) - F_k(p, q) \} = F_{|V|}(p, q) - \max_{0 \leq k \leq |V| - 1} \{ F_k(p, q) \} = F_{|V|}(p, q) - \pi(p, q) \leq 0,$n

and

$$\min_{0 \leq k \leq |V| - 1} \left\{ \frac{F_{|V|}(p, q) - F_k(p, q)}{|V| - k} \right\} \leq 0. \tag{3.3}$$

Let us show that equality holds in (3.2) if and only if $F_{|V|}(p, q) = \pi(p, q)$.

If $F_{|V|}(p, q) = \pi(p, q)$, then for all $0 \leq k \leq |V| - 1$, $F_{|V|}(p, q) \geq F_k(p, q)$, and therefore

$$\min_{0 \leq k \leq |V| - 1} \left\{ \frac{F_{|V|}(p, q) - F_k(p, q)}{|V| - k} \right\} \geq 0,$n

and equality in (3.2) follows.
On the other hand, if \( F_1(P, q) < \pi(p, q) \), then there exists a \( 0 \leq k \leq |V|-1 \), such that 
\[
F_1(P, q) < F_k(P, q)
\]
For this \( k \)
\[
\left\{ \frac{F_1(P, q) - F_k(P, q)}{|V| - k} \right\} < 0,
\]
and strict inequality holds in (3.2).

Hence we can complete the proof by showing that there exists a \( p \) such that 
\( F_1(P, q) = \pi(p, q) \).

Let \( \tilde{C} \) be a critical cycle, and let \( w \) be a vertex in \( \tilde{C} \). Let \( P(w, q) \) be a path of delay \( \pi(w, q) \) from \( w \) to \( q \). Then \( P(w, q) \), preceded by any number of repetitions of \( \tilde{C} \), is also a maximum-delay path from \( w \) to \( q \). Hence, any final part of such a path must be a maximum-delay path from its start point to \( q \). After sufficiently many repetitions of \( \tilde{C} \), such an initial part of length \( |V| \) will occur; let its start vertex be \( w' \). Then \( F_1(w', q) = \pi(w', q) \). Choosing \( p = w' \), the proof is complete.

Case 2: \( \hat{A} \neq 0 \).

Consider the effect of reducing each edge delay \( d(e) \) by a constant \( c \). Clearly \( \hat{A} \) is reduced by \( c \), \( (F_1(P, q) - F_k(P, q)) / (|V| - k) \) is reduced by \( c \), and
\[
\max_{p \in V} \min_{0 \leq k \leq |V|-1} \left\{ \frac{F_1(P, q) - F_k(P, q)}{|V| - k} \right\}
\]
is reduced by \( c \). Hence both sides of the expression in the theorem are affected equally when the function \( d \) is translated by a constant. Choosing that translation which makes \( \hat{A} \) zero, and then applying the result proved for \( \hat{A} = 0 \), the proof is complete.

One can compute the quantities \( F_k(p, q) \) by the recurrence
\[
F_k(p, q) = \max_{p \rightarrow s \in E} \{ d(p,s) + F_{k-1}(s, q) \}
\]

\( k = 1, 2, ..., |V| \), with the initial conditions
\[
F_0(q, q) = 0; \quad F_0(r, q) = -\infty, \quad r \neq q.
\]
The computation requires \( O(|V| \cdot |E|) \) operations, and once the quantities \( F_k(p, q) \) have been tabulated, one can compute
\[
\hat{A} = \max_{p \in V} \min_{0 \leq k < |V|} \left\{ \frac{F_1(P, q) - F_k(P, q)}{|V| - k} \right\}
\]
3.3 Analysis of the Transitory Stage and the Periodicity

We shall say that a processor \( p \) of \( N \) is \( k/s \)-periodic at time \( t_1 \), if for all \( t \geq t_1 \), if \( p \) sends \( M_f \) at \( t \), then \( p \) will send \( M_{f+k} \) at \( t+s \). Also, if at \( t_1 \) all processors in \( N \) are \( k/s \)-periodic, we shall say that \( N \) is \( k/s \)-periodic at \( t_1 \).

Observe that if \( N \) is \( k/s \)-periodic at \( t \), then its rate is \( k/s \). By Theorem 3, \( k/s = \hat{R} \). Thus, it suffices to say that \( N \) is \( k \)-periodic, or that the periodicity of \( N \) is \( k \). Moreover, in every interval of length \( s \) which starts at \( t \) or after, every processor in \( N \) sends exactly \( k \) messages.

For all processors \( p \) and \( n \geq 0 \), let

\[
\tau_n(p) = \max_{q \in P} \left( \tau_n(q) + F_1(q, p) \right);
\]

that is, the first integral time on which \( p \) having received \( M_n \) from all its in-neighbors, can send

* Up to here, this section has been taken from Karp's paper, almost verbatim. He continues: "If the actual cycle yielding the minimum cycle mean is desired, it can be computed by selecting the minimizing \( V \) and \( k \) in (1), finding a minimum-weight edge progression of length \( n \) from \( s \) to \( V \), and extracting a cycle of length \( n - k \) occurring within that edge progression." (The reader should ignore the difference in notation, the fact that we discuss maximum cycle mean, while Karp discussed minimum cycle mean, and the fact that he looks at paths starting at \( s \) while we look at paths ending in \( q \). These are not consequential.) The validity of this statement was questioned by various people, and R. M. Karp himself was in doubt. The reason for the uncertainty is the "possibility" that the optimizing path of length \( |V| \) may make use of some heavy (light) edges which will compensate for the cycle in it \( \forall n \) being optimal. We show that Karp's statement is, in fact, correct.
Lemma 5

For $n \geq 0$, either $t_{n+1}(p) = \tau_n(p)$, or $t_{n+1}(p) = \tau_n(p) + 1$.

Proof

Let $n$ be the smallest index for which the claim is false, i.e., at $\tau_n(p)$, $p$ sends neither $M_{n+1}$ nor $M_n$. Then, $p$ could do one of three things at $\tau_n(p)$:

1. send $M_i, i > n + 1$,
2. skip the beat $\tau_n(p)$,
3. send $M_i, i < n$, or $W$.

$\tau_n(p)$ is the earliest time at which $p$ can send $M_{n+1}$, since before that it has not received $M_n$ via all its incoming edges. Thus, (1) is impossible. However, by time $\tau_n(p)$, $p$ has received $M_n$ along all its incoming edges, and if it does not send $M_{n+1}$, it is busy sending an earlier message. Thus, the only remaining case is (3).

Assume $n > 0$. Clearly, $\tau_{n-1}(p) \leq \tau_n(p) - 1$. Also, $t_{n-1}(p) \geq \tau_n(p)$, by (3). Thus, $t_{n-1}(p) \geq \tau_{n-1}(p) + 1$, and $n$ is not the smallest index for which the claim fails.

Assume $n = 0$. Case (3) states that $p$ sends $W$ at $\tau_0(p)$, which is clearly impossible, since $p$ sends $W$ before that time.

Lemma 6

For $n \geq 0$, if $t_n(p) = \tau_n(p)$ then for every $0 \leq i \leq n$, $t_{n-i}(p) = \tau_{n-i}(p) = t_n(p) - i$.

Proof

By induction on $i$. The case $i = 0$ is assumed. Let us show that if the Lemma holds for $i \geq 0$, then it holds for $i + 1$. Observe that $\tau_{n-i-1}(p) \leq t_{n-i}(p) - 1$. By Lemma 5, $t_{n-i}(p) \leq \tau_{n-i-1}(p) + 1 \leq \tau_{n-i}(p)$. By the inductive hypothesis $t_{n-i}(p) = \tau_{n-i}(p) = t_n(p) - i$, and therefore $\tau_{n-i-1}(p) = t_n(p) - i - 1$.

By Lemma 5, $t_{n-i}(p) \leq \tau_{n-i-1}(p) + 1$. Since $t_{n-i-1}(p) \leq \tau_{n-i-1}(p) - 1$, it follows that $t_{n-i-1}(p) \leq \tau_{n-i-1}(p)$. If equality holds, the proof is complete. Assume that $t_{n-i-1}(p) < \tau_{n-i-1}(p)$. This means that $p$ sent $M_{n-i-1}$ before $\tau_{n-i-1}(p)$, and thus at $\tau_{n-i-1}(p)$ it can send $M_{n-i}$, namely, $t_{n-i}(p) = \tau_{n-i}(p) = \tau_{n-i-1}(p)$, a contradiction to the inductive hypothesis.

With these tools we are now ready to characterize the times on which a processor $p$ sends a message $M_n$, i.e., the times $\tau_n(p)$. 

-15-
Theorem 4

For every \( p \in V, n \geq |V|, k \geq 1, \)

\[ t_{n+k}(p) = \max_{q \in V} \{ t_n(q) + F_k(q, p) \} . \]

Proof

First we shall prove the Theorem for \( k = 1. \)

Assume that there exists a \( p \in V \) and an \( n \geq |V| \) for which

\[ t_{n+1}(p) = \max_{q \in V} \{ t_n(q) + F_1(q, p) \} = \tau_n(p). \]

By Lemma 5, \( t_{n+1}(p) = \tau_n(p) + 1, \) and hence \( t_n(p) = \tau_n(p). \) By Lemma 6, \( p \) does not skip at beats \( \tau_n(q) - i, \) \( 0 \leq i \leq n. \) Also, for every in-neighbor \( q, t_l(q) < \tau_l(p) = t_l(p). \)

If the distance from \( p \) to \( q \) is \( l \) \( (|V| - 1 \leq n - 1) \) then \( t_l(r) \geq t_l(p); \) the proof is by induction on \( l. \) Assume the distance from \( p \) to \( q \) is \( l + 1, \) \( r \to s. \) Then, \( t_{l+1}(s) \geq t_l(r) + 1, \) and by the inductive hypothesis \( t_{l+1}(s) \geq t_l(p) + 1. \) Since for \( n + 1 \) consecutive beats \( p \) does not skip, \( t_{l+1}(p) = t_l(p) + 1. \)

Let \( l \) be the distance from \( p \) to \( q. \) Then \( t_l(q) \geq t_l(p). \) However, \( l \leq n - 1, \) so \( t_l(q) < t_l(p). \)

A contradiction, proving the Theorem for \( k = 1. \)

Let us prove that if the theorem holds for \( k \) then it holds for \( k + 1. \) We can write \( t_{n+k+1}(p) \) as \( t_{n+k+1}(p), \) and thus

\[ t_{n+k+1}(p) = \max_{q \in V} \{ t_{n+k}(q) + F_k(q, p) \}. \]

By the inductive hypothesis,

\[ t_{n+k}(q) = \max_{r \in V} \{ t_n(r) + F_k(r, q) \} \]

By the definition of \( F \) we get the desired result:

\[ t_{n+k+1}(p) = \max_{r \in V} \{ t_n(r) + F_{k+1}(r, p) \} \]

\( \square \)

We begin by studying the periodicity of a simple cycle.

Theorem 5

A simple directed cycle \( C = p_0 \to p_1 \to \cdots \to p_{|V|-1} \to p_0, \) where \( p_0 \) satisfies \( t_0(p_0) = t_0, \) is \( |V| \)-periodic by time \( t_0 + D. \)
Proof

For \( n \geq |V|, 0 \leq i \leq |V| - 1 \),

\[ t_{n+|V|}(p_i) = t_n(p_i) + D, \]

because by Theorem 4,

\[ t_{n+|V|}(p_i) = \max_{q \in V} \{ t_n(q) + F_{|V|}(q, p_i) \} = t_n(p_i) + D. \]

We have shown that every processor \( p_i \) is \(|V|/D\)-periodic by time \( t_n(p_i), n \geq |V| \). It remains to show that \( t_{|V|}(p_i) \leq t_0 + D \). We know from the proof of Theorem 2, that

\[ t_k(p_i) \leq t_0 + \sum_{j=1}^k d_{i-j}. \]

Letting \( k = |V| \), we get

\[ t_{|V|}(p_i) \leq t_0 + \sum_{j=1}^{|V|} d_{i-j} = t_0 + D. \]

\[ \square \]

Let \( N' = (V', E') \) be the subgraph obtained by removing from \( N \) all edges and vertices which are not in some critical cycle. The graph \( N' \) consists of several connected components, or critical components, each one strongly connected. Let \( S \) be the set of vertices in one of the critical components. As we shall now see, there exist an \( l \) such that every \( p \in S \) is eventually \( l \)-periodic (in \( N \)). Moreover, \( l \leq gl \), where \( gl \) is the greatest common divisor (gcd) of the lengths of all critical cycles of \( S \).

In the next theorem we show that the periodicity of processors in \( V' \) is at most \(|V| \). Later we shall present an example of a network \((G_k)\), in which the periodicity of processors not in \( V' \), may be exponential in \(|V| \).

**Theorem 6**

There exist an \( l, l \leq gl \), such that every processor \( p \in S \) is \( l \)-periodic.

**Proof**

Assume \( p \) is in some critical cycle \( \hat{C} \) of length \( \hat{L} \) and delay \( \hat{D} \). We first prove that there exists an \( n_0 \geq 0 \), s.t. for all \( n \geq n_0 \),

\[ t_n(p) = t_n(p) + \hat{D}. \]

By Lemma 4, for all \( k \geq 0 \)

\[ N(p, t_0 + k\hat{D} + \Delta |V|) \geq k\hat{L} + |V| + 1. \] (2.1)
Assume that $N(p, t_0 + \Delta |V|) = m$. By Lemma 1, for every $t_1 > 0$

$$N(p, t_1 + \hat{D}) - N(p, t_1) \leq \hat{L}.$$  

Let us show that there is a finite number of intervals of the form

$$(t_0 + k\hat{D} + \Delta |V|, t_0 + (k + 1)\hat{D} + \Delta |V|),$$

in which $p$ sends less than $\hat{L}$ messages. Suppose that among the first $k$ intervals after $t_0 + \Delta |V|$, there have been $c$ intervals during which $p$ sends less than $\hat{L}$ messages. Then

$$N(p, t_0 + k\hat{D} + \Delta |V|) \leq m + (k - c)\hat{L} + c(\hat{L} - 1) = m + k\hat{L} - c.$$  

By (2.1) $m + k\hat{L} - c \geq k\hat{L} + |V| + 1$, or

$$c \leq m - |V| - 1,$$

proving that the number of such intervals is bounded. Hence, there exist a $k_0$, such that if $k \geq k_0$ then

$$N(p, t_0 + (k + 1)\hat{D} + \Delta |V|) - N(p, t_0 + k\hat{D} + \Delta |V|) = \hat{L}.$$  

And by Lemma 1, there exists an $n_0$, s.t. for $n \geq n_0$, $t_{n+1}(p) = t_n(p) + \hat{D}$, i.e. $p$ is $\hat{L}/\hat{D}$-periodic. Moreover, there exists an $n_0$ such that for every $C$ in $N$, every $p$ in $\hat{C}$ and every $n \geq n_0$, $t_{n+1}(p) = t_n(p) + \hat{D}$.

It follows that if $p \rightarrow q$, is an edge of $N'$, then for $n \geq n_0$, $t_{n+1}(q) = t_n(p) + d(p, q)$; that is, a message $M_p$ sent along an edge of $N'$, is used (in $N$) as soon as it arrives. Hence, if $p$ is $l/d$-periodic, then so is $q$. Since the connected component of $N'$ spanned by $S$ (i.e. the critical component) is strongly connected, all its processors are eventually $l/d$-periodic in $N$.

Let $l$ be the least integer such that the processors of $S$ are $l/d$-periodic. If $p \in S$ is in a critical cycle of length $\hat{L}$ and delay $\hat{D}$, then $p$ is $\hat{L}/\hat{D}$-periodic. It follows that $l$ divides $\hat{L}$, and therefore $l$ divides $gl$. A similar statement, for $d$ and $gd$, follows.

Theorem 6 implies that if $gl = 1$, then regardless of which processors wake up spontaneously, every $p \in S$ is $l$-periodic, or $1/\hat{L}$-periodic.

We now turn to the question of the length of the transitory stage before periodicity is reached, and prove the periodicity, as a by-product.

Recall that $\Delta$ is the greatest edge-delay, and $\hat{A} = 1 / \hat{\Delta}$. Denote by $\hat{\Delta}$ the smallest edge-delay. Let us shift the delays of the edges in such a way that the delay of any critical cycle becomes 0. For every edge $e$, the shifted delay is $\hat{d}(e) = d(e) - \hat{\Delta}$. Hence, $\hat{\Delta} = \Delta - \hat{\Delta}$ is the greatest shifted delay, while $\hat{\Delta} = \hat{\Delta} - \hat{\Delta}$ is the smallest shifted delay. The functions $\hat{D}(\cdot)$, and $\hat{F}_k(\cdot; \cdot)$ are defined as before, but in reference to the shifted delays. In order to analyze the times
\( t_k(p) \),

\( k \geq |V| \), we define shifted times, \( \tilde{t}_k(p) \), as follows

\[
\tilde{t}_k(p) = \max_{q \in V} \{ t_{|V|}(q) + \tilde{F}_k(q, p) \}
\]

\[
= t_{|V|+k}(p) - k \hat{A}.
\]

For a path \( P \) from \( q \) to \( p \), let

\[
\bar{T}(P) \triangleq t_{|V|(q)} + \bar{D}(P).
\]

Since \( \hat{A} \) is an average delay (of some critical cycle), it is the ratio of two positive integers.

Let \( a \) and \( b \) be two relatively prime, positive integers such that

\[
\hat{A} = \frac{a}{b}.
\]

Observe that \( b \leq |V| \).

Let \( f \) be the absolute value of the greatest shifted delay of a non-critical cycle. (Notice that the shifted delay of a non-critical cycle is negative.)

\textbf{Lemma 7}

\[
f \geq \frac{1}{|V|}.
\]

\textbf{Proof}

Let \( C \) be a non-critical cycle which determines \( f \). Clearly

\[
D < L \hat{A} = L \frac{a}{b}.
\]

Thus,

\[
D \leq \left\lfloor L \frac{a}{b} \right\rfloor - 1,
\]

and

\[
-f = \bar{D} \leq \left\lfloor L \frac{a}{b} \right\rfloor - L \frac{a}{b} - 1 < 0.
\]

Therefore,

\[
f \geq L \frac{a}{b} - \left\lfloor L \frac{a}{b} \right\rfloor + 1 > 0.
\]
It follows that
\[ f \geq \frac{1}{b} \geq \frac{1}{|V|}. \]
\[ \square \]

We next quantify the effect of adding the quantities \( t_{|V|}(\cdot) \) to the function \( \overline{F}(\cdot, \cdot) \). Let \( T \) be defined by
\[
T = \max_{p,q \in V} |t_{|V|}(p) - t_{|V|}(q)|.
\]

Lemma 8
\[ T \leq |V| (\Delta - 1). \]

Proof
For every processor \( r \), there exists a simple path from a processor which sent \( M_0 \) at \( t_0 \) to \( r \). Let its length be \( l \). Clearly \( l \leq |V| - 1 \). Also, \( \delta(r) \geq t_0 + l \) (easily proved by induction on \( l \); see also [ER]). Therefore \( \delta(r) \geq t_0 + k \) for every \( k \geq l \). It follows that \( t_{|V|}(r) \geq t_0 + |V| - 1 \).

On the other hand, by Lemma 2, for every \( r \), \( t_{|V|}(r) \leq t_0 + F_{|V|}(r) \). Thus,
\[ |V| - F_{|V|}(p) \leq t_{|V|}(p) - t_{|V|}(q) \leq F_{|V|}(p) - |V| \]
or,
\[ |t_{|V|}(p) - t_{|V|}(q)| \leq F_{|V|}(p) - |V|. \]

However, \( F_{|V|}(p) \leq \Delta \cdot |V| \), and the Lemma follows. \( \square \)

Denote by \( S_i \), \( 1 \leq i \leq c \), the critical components of \( N \), and by \( g_{l_i} \) the \( gcd \) of all critical cycles of \( S_i \). Define
\[ \lambda = \frac{\text{lcm}(g_{l_1}, g_{l_2}, \ldots, g_{l_c})}{f}, \]

where \( \text{lcm} \) denotes the least common multiple operation. As we shall see, \( \lambda \) is related to the periodicity of the network. The number of messages sent during the transitory stage is related to \( I_0 \), where
\[ I_0 = \frac{|V|}{f} [(\Delta - \delta) (|V| - 1) + T] + |V| - 1. \]

Let \( s \) be a processor in \( N \). Define
\[ \overline{D}_r = \max \{ \overline{D}(s) : k \equiv r \ (mod \ \lambda), \ k > I_0 \} \]
and let \( P_r \) be a path for which the maximum is attained. Thus, \( P_r \) ends in \( s \), has length \( L(P_r) > I_0, L(P_r) \equiv r \ (mod \ \lambda) \), and
\[ \overline{D}_r = t_{|V|}(w) + \overline{F}_{L(P_r)}(w, s) = \overline{t}(P_r), \]

for some \( w \in V \).

Let us show that \( \overline{D}_r \) is defined. Since there are paths leading to \( s \), of every length \( k \), \( k > l_0 \), and \( k \equiv r \mod \lambda \), it suffices to show that every increasing sequence of shifted delays of such paths is finite. The shifted delay of any edge is a multiple of \( 1/b \), and therefore, the shifted delay of any path is also a multiple of \( 1/b \). It follows that in an increasing sequence of shifted paths delays, the increment from one path to the next is at least \( 1/b \). Since there are no cycles of positive shifted delay, the shifted delay of any path is bounded by \( \overline{D}(1|V| - 1) \), proving that such a sequence is finite.

Let the set of edges in \( P_r \), which do not belong to critical cycles of the decomposition (for the meaning of decomposition, see the proof of Lemma 3), be called the non-critical part of \( P_r \). Let \( L_r \) be the number of edges in the non-critical part of \( P_r \).

**Lemma 9**

\[ L_r \leq l_0. \]

**Proof**

The shifted delay of \( P_r \) is equal to the shifted delay of its simple path, plus the sum of the shifted delays of its simple cycles (per the decomposition). A noncritical cycle has negative delay of at most \( -f \). Then,

\[ \overline{D}_r \leq \overline{D}(1|V| - 1) + \frac{L_r - (1|V| - 1)}{|V|} (-f) + \max_{q \in V} \{ t_{|V|}(q) \}, \]

where the first term is an upper bound on the shifted delay of the simple path and the second term is an upper bound on the cumulative delay of the non-critical cycles (a lower bound on the number of non-critical cycles, times the maximum shifted delay of such cycles).

Let us construct a path \( P_r \) to \( s \) of length greater than \( l_0 \), congruent to \( r \) modulo \( \lambda \), and passes through a vertex of a critical cycle \( \hat{C} \), as follows. Take a simple path from a vertex in \( \hat{C} \) to \( s \) which does not have any other vertices in common with \( \hat{C} \). Then extend the initial part of this path by turning backwards around \( \hat{C} \) until a length congruent to \( r \) modulo \( \lambda \) is obtained. Thus,

\[ \overline{D}(P_r) \geq \overline{t}(1|V| - 1). \]

Hence

\[ \overline{D}_r \geq \overline{D}(P_r) + \min_{q \in V} \{ t_{|V|}(q) \}. \]

Therefore

\[ \overline{D}(1|V| - 1) + \frac{L_r - (1|V| - 1)}{|V|} (-f) + \max_{q \in V} \{ t_{|V|}(q) \} \geq \overline{D}(1|V| - 1) + \min_{q \in V} \{ t_{|V|}(q) \}. \]
Since $\max\{ t_{|V|}(q) \} - \min\{ t_{|V|}(q) \} \leq T$ it follows that

$$L_T \leq \frac{|V|}{T} \left( \frac{1}{\bar{d}} - \frac{1}{\bar{d}} \right) + |V| - 1$$

\[\square\]

In the proof of Lemma 11, we use Lemma 10, as well as the following theorem, which is a direct consequence of Theorem 1 in Brauer, [B].

**Theorem B**

Let $L_1 \leq L_2 \leq \ldots \leq L_c$ be positive integers and $g = \gcd(L_1, L_2, \ldots, L_c)$. For every $a$ such that

1. $g$ divides $a$, and
2. $a \geq (L_1 - 1)(L_c - 1)$,

the equation

$$\sum_{i=1}^{c} x_i L_i = a$$

has a solution in nonnegative integers.

For a simple, strongly connected digraph $G (V, E)$, define a Spanning Eulerian Derivative, $(SED)$ of $G$, to be a digraph $H (V', E')$ for which the following three conditions hold:

1. $V' = V$.
2. For every edge $u \rightarrow v$ in $E'$ there exists an edge $u \rightarrow v$ in $E$. Note that there are no parallel edges in $G$, but there may be parallel edges in $H$.
3. $H$ is Eulerian, i.e., there exists a directed circuit in $H$ which includes all its vertices, and every edge of $H$ appears exactly once in the circuit.

**Lemma 10**

If $G (V, E)$ is a simple, strongly connected finite digraph then $G$ has a $SED, H (V, E')$, such that $|E'| < |V|^2$. 
Proof  
Choose a spanning tree $T$, of $G$ (the direction of the edges in $T$ is immaterial). For each edge $e \in T$, find a simple directed cycle $C$ in $G$, which goes through $e$, and put all edges of $C$ into $E'$; if an edge is already in $E'$, add a duplicate. It is easy to see that the resulting graph $H$ is a SED, and $|E'| < |V|^2$. □

We continue our study of $P_r$.

Lemma 11

Let $n_0 \hat{=} l_0 + 2|V|^2$. For every $n \geq n_0$ there exists a directed path $P$ ending in $s$, such that $L(P) = n$ and $\overline{T}(P) = \overline{D}_r$, where $r \equiv n \pmod{\lambda}$.

Proof

Let $P_r$ be a path as defined following Lemma 8, and denote by $L_r$ the length of the non-critical part of $P_r$. This non-critical part is obtained by removing some critical cycles from $P_r$. In fact, by Lemma 9, at least one critical cycle is removed. Denote by $S_1, S_2, \ldots, S_m$, $m \geq 1$, the critical components each of which contains a critical cycle removed from $P_r$. Construct for each $S_i$, $1 \leq i \leq m$ a SED, with a set of edges $E_i$, such that $|E_i| < |S_i|^2$. Consider the digraph consisting of the non-critical part of $P_r$, which by Lemma 9 has at most $l_0$ edges, and all SED’s $(S_i, E_i)$, $1 \leq i \leq m$. Clearly,

$$\sum_{i=1}^{m} |E_i| < \sum_{i=1}^{m} |S_i|^2 \leq \left( \sum_{i=1}^{m} |S_i| \right)^2 \leq |V|^2.$$

Thus, the resulting digraph $G'(V', E')$ has less than $l_0 + |V|^2$ edges. Also, $G'$ has an Eulerian path $P'$, which starts at $w$ and ends at $s$. The corresponding $\overline{T}$ is given by

$$\overline{T}(P') = t_{|V|^2}(w) + \overline{T}_{L(P')} (w, s).$$

and is equal to $\overline{D}_r$ of the path $P_r$ we started with.

Assume

$$L(P') \equiv r' \pmod{\lambda}.$$

Thus,

$$L(P_r) - L(P') \equiv r - r' \pmod{\lambda}.$$

Let $L_1 \leq L_2 \leq \ldots \leq L_c$ be the different lengths of all the critical cycles of the components $S_1, S_2, \ldots, S_m$. It follows that the number $L(P')$ satisfies

$$L(P_r) = L(P') + \sum_{i=1}^{c} x_i L_i,$$

where $x_1, x_2, \ldots, x_c$ are integers, because $P'$ is obtained from $P_r$ by removing and adding
critical cycles (An Eulerian graph is equal to the union of disjoint simple cycles. All cycles in a critical component are critical).

Let $M \Delta n - L(P')$. Clearly,

$$M \equiv r - r' \pmod{\lambda}$$

Also, since $n \geq l_0 + 2|V|^2$, and $L(P') < l_0 + |V|^2$, it follows that

$$M \geq |V|^2 \geq (l_1 - 1)(l_2 - 1).$$

Let $g = \gcd(L_1, L_2, \ldots, L_c)$. Clearly $g \mid \lambda$. By equation (2.2)

$$\sum_{i=1}^{c} x_i L_i = L(P_r) - L(P')$$

has an integral solution, and thus $g \mid L(P_r) - L(P')$. Also,

$$M \equiv r - r' \equiv L(P_r) - L(P') \pmod{\lambda}$$

and therefore the congruence holds modulo $g$ as well. It follows that $g \mid M$ and thus, by Theorem B, the equation

$$\sum_{i=1}^{c} x_i L_i = M \quad (2.3)$$

has a nonnegative solution. Use this solution to define $P$ by adding to $P'$ the cycles, as specified by the $x_i$'s in (2.3). Thus,

$$L(P) = L(P') + M = n$$

and since the cycles added to $P'$ are critical,

$$\overline{T}(P) = D_r.$$ 

We are now ready to prove the main result: The number of messages sent on each edge during the transitory stage is $O\left(\Delta |V|^3\right)$, while the periodicity is at most $\lambda$.

Theorem 7

Any processor $s$ is $\lambda$-periodic after sending $|V| + l_0 + 2|V|^2 \left( = O\left(\Delta |V|^3\right) \right)$ messages on each of its outgoing edges.
Proof

Lemma 11 implies that for every \( n \geq n_0 \),

\[ T_{n+\lambda}(s) = T_n(s). \]

since each is equal to \( D_r \).

It follows that

\[ t_{|V|+\lambda+n}(s) - (n+\lambda)\hat{A} = t_{|V|+n}(s) - n\hat{A} \]

and

\[ t_{|V|+n+\lambda}(s) = t_{|V|+n}(s) + \lambda\hat{A}. \]  \hspace{1cm} (2.4)

Therefore, \( s \) is \( \lambda \)-periodic after sending \(|V| + l_0 + 2|V|^2\) messages on each of its outgoing edges. \( \Box \)

The following bound for the time to reach periodicity follows by applying Lemma 2 to the result of Theorem 7, and noting that \( t_0 \leq \Delta |V| \).

Corollary 3

The network \( N \) is \( \lambda \)-periodic by time \( t_0 + \Delta (|V| + l_0 + 2|V|^2) = O(\Delta^2 |V|^3) \).

Observe that formula (2.4), in addition to describing the periodicity of the computation, also yields the eventual computational rate, and therefore provides an alternative (but more involved) proof of Theorem 3.

In closing, we prove two statements: there are graphs with an exponential period, and there are graphs for which the bound of Theorem 7 on the transitory stage (and of Corollary 3) is tight.

For the first statement, we show now a family of networks for which \( \lambda \) is exponential in the number of processors, and if all processors wake-up simultaneously then the period is exactly \( \lambda \). Observe that if all processors wake-up simultaneously, and if the time they all send \( M_0 \) is defined to be \( 0 \), then, by Lemma 6, for every \( k \geq 1 \),

\[ t_k(p) = \max_{q \in V} \{ F_k(q,p) \}. \]

For every \( m \geq 2 \), let \( G_m \) consist of a vertex \( s \) and \( i \) simple directed cycles \( C_j \), \( j = 1, 2, \ldots, i \), where \( i \) is the number of primes less than or equal to \( m \). Also, \( L(C_j) = p_j \), where \( p_j \) is the \( j \)th prime. One vertex, \( v_j \), of each cycle \( C_j \) is connected to \( s \) by two antiparallel edges; the one going into \( v_j \) is of normalized delay \(-p_j\), the one going into \( s \) is of normalized delay \( 0 \). The normalized delay of the edge in the cycle \( C_j \), entering \( v_j \), has delay \( p_j - 1 \). All other edges have normalized delay \(-1 \).
Let $n_m$ be the number of vertices in $G_m$. Then

$$n_m = 1 + \sum_{j=1}^{i} p_j < m^2.$$  

Observe that all the cycles $C_j$ are critical, i.e., have normalized delay 0. Assume $t_0(v) = 0$, for all $v \in V$. Then, for $n \geq 1$,

$$\bar{\tau}_n(p) = \max_{q \in V} (\bar{F}_n(q, p))$$

$$= t_n(p) - n \cdot \bar{A}.$$  

Now, if the period is less than $\lambda$, say $\lambda'$, then $\lambda'$ divides $\lambda$. Observe that every heaviest path ending in $s$ has non-negative normalized delay, and hence, no heaviest path ending in $s$ uses an edge of normalized delay $-p_j$. Let $n \equiv 1 \pmod{\lambda}$, $n \geq n_0$ ($n_0$ as defined in Lemma 11). It follows that

$$\bar{\tau}_n(s) = 0,$$

because a heaviest path ending in $s$ of length $n$ starts in a vertex $v_j$ then turns $(n - 1) / p_j$ times around $C_j$, and then ends in $s$. If the period is $\lambda'$, then

$$\bar{\tau}_{n+\lambda'}(s) = \bar{\tau}_n(s).$$  

But observe that

$$\bar{\tau}_{n+\lambda'}(s) > 0,$$
because there exists a \( 1 \leq k \leq l \) such that \( p_k \) does not divide \( \lambda' \), and thus, there exists a path of length \( n + \lambda' \) starting in a vertex of \( C_k \) different from \( v_k \) and ending in \( s \); this path has normalized delay greater than 0. Thus, the period is exactly \( \lambda \).

If \( p \) denotes a prime number, then

\[
\lambda = \prod_{p \mid m} p,
\]

and hence

\[
\lambda > 2^{\frac{m}{4}}
\]

(see, for example, [HW] pp. 341). On the other hand, \( n_m \) is less than \( m^2 \). Hence, the periodicity of \( G_m \) is at least \( 2^{\frac{m}{4}} \).

Let us now show that the bound on the transitory stage, given in Theorem 7, is tight. Construct for every \( k \geq 2 \), a network \( H_k \) as follows. The graph \( H_k \) consist of two simple, directed cycles \( \hat{C} \) and \( C \), joined to a vertex \( s \) by two simple paths, formed by antiparallel edges. Let \( u \) be a vertex of \( \hat{C} \), and \( v \) a vertex of \( C \). One simple path joins \( s \) and \( u \), and the other joins \( s \) and \( v \). The length of \( \hat{C} \) and of the two simple paths is \( k \); the length of \( C \), \( k + 1 \). Thus, the number of vertices of \( H_k \), \( n_k \), is \( 4k \). Let \( d \geq 1 \) be an integer. The delays of the edges in the simple path joining \( s \) and \( u \) are as follows. The edges in the direction from \( s \) to \( u \) have normalized delay \(-2d - 1 / k\); the edges in the direction from \( u \) to \( s \) have normalized delay \(-1 / k\). The normalized delays of the other path are \(-2d - 1 / k\), in the direction from \( s \) to \( v \), and \( d - 1 / k \) on the edges from \( v \) to \( s \). Finally, the delays of the edges in the simple cycles are \(-1 / k\) except for two edges, which have delay \((k - 1) / k \); the one going out of \( u \) and the one going out of \( v \).

Observe that \( \overline{D}(\hat{C}) = 0 \), and \( \overline{D}(C) = -1 / k \). One can construct a network with delays that correspond to the normalized delays of \( H_k \), by choosing \( \hat{A} = 3d + 1 / k \), for example, and adding \( \hat{A} \) to the normalized delay of each edge. In this case \( \Delta \) is equal to \( 4d \).
Consider a heaviest path ending in \( s \), of length \( n = k + ck(k+1) \), \( c \geq 0 \). It is easy to see that the heaviest path of length \( n \) that uses vertices of \( \hat{C} \) is of the form: starts in \( u \), then goes around \( \hat{C} \), \( c(k+1) \) times, and then goes to \( s \) along the corresponding simple path; this path, \( P_1 \), has normalized delay \(-1\). By Lemma 9, if \( n \geq n_0 \) (\( n_0 \) as defined in Lemma 11), such a heaviest path must use vertices of \( \hat{C} \). Therefore, using \( P_1 \),

\[
\overline{t}_{n+k(k+1)}(s) = \overline{t}_n(s) = -1,
\]

since \( \hat{C} \) is the only critical cycle.

Consider the following path, \( P_2 \): starts in \( v \), turns \( ck \) times around \( C \), and then goes to \( s \) along the corresponding simple path. The normalized delay of \( P_2 \) is:

\[
kd -1 + ck(-\frac{1}{k}).
\]

Thus, while

\[
c < kd
\]

the path \( P_2 \) is heavier than \( P_1 \), that is, for \( c < kd \)

\[
\overline{t}_{k+ck(k+1)}(s) > -1,
\]
and thus, if all processors wake up simultaneously, $s$ will not enter periodicity before sending $k + dk^2(k + 1)$ messages. If $A = 3d + 1 / k$, this is $O(\Delta n_k^2)$, since $n_k = 4k$ and $\Delta = 4d$.

Acknowledgement

The authors wish to thank Avi Wigderson and Richard M. Karp for their helpful comments.

References


