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

We study the performance of networks whose operation is controlled by a simple synchronizer. In a previous paper we analyzed the performance of networks with negligible transmission delay. 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. In any case, the length of the transitory stage is bounded polynomially, and the bound is tight, while the period may be of exponential length.

* An extended abstract of this paper appeared in STOC'90. Part of this work was done while the authors visited the Computer Science Program, University of Texas at Dallas, Richardson, TX.

** Now visiting Bell Laboratories, Lucent Technologies, Murray Hill, NJ. Was supported by the Fund for the Promotion of Research at the Technion.
E-address: even@cs.technion.ac.il

*** Current address: Instituto de Matematicas - U.N.A.M., Ciudad Universitaria, D.F. 04510, MEXICO.
E-address: rajsbaum@servidor.unam.mx

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 tick 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 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, namely, to guarantee that each processor is performing the same 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 transmission delays. His paper proposes several specific methods for implementing a simulation of this type, and studies their complexities. Synchronizers are useful as a design tool of efficient algorithms, e.g. [A85]. Various aspects and applications thereof were studied in [A85,AS88,CGZ86,FLS87,PU89].

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 Genrich [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. This simple synchronizer is the basis of several distributed algorithms and, as was shown by [MR91] it can be used to model the synchronizers in [A], the schedulers [BG89,MMZ88], and any marked graph [CHEP]. Therefore the results in this paper can be used to analyze the performance of many algorithms under specific transmission delays, as opposed to the usual unit time delays assumption.

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.

(1.3) Each processor has its own clock, and the rate of the clocks may differ.

(1.4) The operation of the processors is message driven.

(2) Transmission delays.

(2.1) Positive but negligible delays.

(2.2) Fixed non-negligible delays.

(2.3) Changing non-negligible delays.

Our aim is to study the behavior of networks under all possible combinations of these parameters (except the (1.4) and (2.1) combination).

Previous and Related Results

In our previous paper, [ER], we analyzed the behavior of the network in the absence of a global start-up signal and when the transmission delays are positive, but negligible (case (2.1)).

In the absence of a global start-up signal, but when (1,1) or (1,2) are assumed, we showed that the use of the synchronizer guarantees that within $2|V|$ units of time (in a network of $|V|$ processors) full rate operation is reached. If (1,1) holds, 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 if there is no need for the processors to know when unison is reached.

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 ticks of it, every clock in the network ticks at least once.)

In [RS] the case (2.3) of random delays was studied. Some results are presented for arbitrary distributions and then upper and lower bounds are presented for the case of identically distributed exponential delays.

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) or (1.4) (for further discussion see [ER]). We believe that the techniques developed here may help to deal with case (1.3). We show that, as in case (2.1), in case (2.2) a maximum rate of computation is always reached, just by using the synchronizer. We present an $O(|V| \cdot |E|)$ algorithm to compute the rate of the network. It follows that the rate of the network is at least $1 / \Delta$, where Δ 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 $O(\Delta \cdot |V|^3)$, and that this bound is tight, even if there is a global start-up signal. This implies a tight bound on the time to reach full speed, namely $O(\Delta^2 \cdot |V|^3)$. This is in contrast with the case of negligible delays (2.1), where the length of the transitory stage is $O(|V|)$.

In [MR91] it has been shown that our results generalize to a larger family of protocols, which includes marked graphs [CHEP] and distributed schedulers [MMZ88,BG89]. Thus, our results imply a polynomial bound on the transitory stage of the schedulers, solving an open question left in [MMZ88] and [BG89].

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].

The Model

The network, $\boldsymbol{N}$ 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. Thus, it is natural to assume that (V, E) has no self-loops and no parallel edges. The processors communicate by sending messages along the communication links.

The ticks of the global clock are heard simultaneously by all the processors (case (1.1)). For simplicity, we assume that the clock ticks on every integral time. Let the time at which the first processor of $\boldsymbol{N}$ wakes up (and sends message W) be equal to 0.

Initially, each processor is in a *quiescent state*, Q, in which it sends no messages, and stays 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 tick, the processor sends *W* on all its outgoing edges; i.e. the wake-up signal propagates by broadcast.

Let us denote by $M_{p,i}$ the messages sent by p as a result of its i-th computational step, or simply M_i when p is clear from the context. Note that the contents of the messages are 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 tick 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 tick, p produces messages M_{i+1}, and sends them on all its outgoing edges. If M_i has been sent, but M_i has not been received on all its incoming edges, then on the next tick no computational step takes place, and no messages are sent; in this case we say that the tick 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 $p_0 \rightarrow p_1 \rightarrow p_2 \rightarrow p_0$, where $d(p_0, p_1) = d(p_2, p_0) = 1$, $d(p_1, p_2) = 2$, and p_0 is the only one to wake up spontaneously.

tick	p_0	p_1	p_2
0	W		
1	0	W	
2	-	0	
3	-	1	W
4	-	-	0
5	1	-	1
6	2	2	2
7	3	3	-
8	-	4	3
9	4	-	4

The rest of the paper is organized as follows. In Section 2 we derive an expression for the ultimate computational rate. In Section 3 we present an efficient algorithm to compute the computational rate. In Section 4 we present a more elegant and interesting proof for the ultimate computational rate of Section 2. The proof is more involved but it also provides a polynomial bound on the time to reach the ultimate computational rate, and shows that once the rate is reached, the network has a periodic behavior. In Section 5 we show that the periodicity may be of exponential length, and that the polynomial upper bound on the length of the transitory stage is essentially optimal.

2. Analysis of the Ultimate Computational Rate

For a processor $p \in \boldsymbol{N}$, denote by $N(p,\ t)$ the number of ticks 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 $P = p_1 \to p_2 \to \cdots \to p_{n+1}$ be a path such that the delay from p_i to p_{i+1}, $1 \le i \le n$, is d_i. We define the *delay* of P, $D(P)$ or simply D, as

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

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

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

We shall say that the *rate of computation of a network* $\boldsymbol{N}$, $\boldsymbol{R}(\boldsymbol{N})$, is

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

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

Let $C = p_0 \to p_1 \to \cdots \to p_{n-1} \to p_0$ be a directed cycle in $\boldsymbol{N}$. Let us denote by D the delay of C.

Lemma 1

For every p in C and every $t_1 \ge 0$,

$$N(p,\ t_1 + D\ ) - N(p,\ t_1) \le n.$$

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, $t_1 < t \le t_1 + D$, 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. □

Corollary 1

For every p on C and every $k \geq 0$,

$$N(\ p,\ kD\ ) \leq kn.$$

A (simple) cycle of $\boldsymbol{N}$ is called *critical* if no other cycle in $\boldsymbol{N}$ has a 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 $\boldsymbol{N}$. Then

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

Proof

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

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

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 directed 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 $\boldsymbol{N}$:

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

Theorem 2

Let $C = \; p_0 \rightarrow p_1 \rightarrow \cdots \rightarrow p_{n-1} \rightarrow p_0$ be a network which is a (simple directed) cycle. The rate of computation of such a network is $R(C)$.

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 arithmetic in the subscripts of d 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}. This completes the proof of the induction.

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,

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

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

$$N(p_i,\ t) \ge N(p_i,\ t_0 + kD)$$

$$\frac{N(p_i,\ t)}{t} > \frac{kn}{t_0 + (k+1)D}$$

But

$$\lim_{k \to \infty} \frac{kn}{t_0 + (k+1)D} = \frac{n}{D},$$

hence,

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

By Theorem 1, the rate of computation of such a network is well defined (i.e. the limit exists) and its value is $\frac{n}{D} = R(C)$. □

We now return to a general, strongly connected network $\boldsymbol{N}$, and $\hat{R}$, the rate of a critical cycle. We shall show that $\boldsymbol{R}$, the rate of $\boldsymbol{N}$, exists and is equal to $\hat{R}$. By Theorem 1, the rate of $\boldsymbol{N}$, 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 $\boldsymbol{N}$, 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 $\boldsymbol{N}$, q sends M_k by time $t_0 + F_k(q)$; that is, $t_k(q) \le t_0 + F_k(q)$, or $N(q,\ t_0 + F_k(q)) \ge k + 1$.

Proof

By induction on k. 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, $t_k(r) \le t_0 + F_k(r)$. Define τ by

$$\tau \stackrel{\Delta}{=} \max_{p \in P} \ \{\ t_0 + F_k(p) + d(p,\ q)\ \}.$$

It is easy to see that

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

From the definition of τ, it follows that q has received M_k from all processors in P before τ. It remains to show that q has sent M_k before τ.

By the induction hypothesis, $t_k(q) \le t_0 + F_k(q)$. 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. $t_{k+1}(q) \le t_0 + F_{k+1}(q)$. □

Lemma 3

For every (not necessarily simple) directed circuit $P = p_0 \to p_1 \to \cdots \to p_{n-1} \to p_0$

$$R(P) \ge \hat{R}.$$

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 $j \le n-1$ be the least index such that $p_j = p_i$, $i < j$. Clearly, $C_1 = p_i \to p_{i+1} \to \cdots \to p_j$ is a cycle. Remove from P all the edges of C_1 to obtain a shorter circuit. Repeat this procedure until a (simple) cycle C_k remains. Thus, P has been decomposed into k cycles $C_1, \ldots, C_k$. Denote by $L(C_i)$, $1 \le i \le k$, the number of edges of C_i.

We could have performed this procedure on any directed path, which is not necessarily a circuit. In this case, after extracting the cycles from the path, a simple nonempty path would remain. We shall refer to this procedure as a *decomposition* of the path.

For each cycle C_i

$$D(C_i) \le \frac{L(C_i)}{\hat{R}}.$$

Each occurrence of an edge of P appears in exactly one cycle of the decomposition. Thus, the rate of cycle P is

$$R(P) \ge \frac{\sum_{i=1}^{k} L(C_i)}{\sum_{i=1}^{k} \frac{L(C_i)}{\hat{R}}},$$

or

$$R(P) \ge \hat{R}.$$

□

Denote by Δ the greatest delay of an edge in $\boldsymbol{N}$.

Lemma 4

For every processor p and every nonnegative integer k,

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

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 $F_{k\hat{L}+|V|}(p) \leq k\hat{D} + \Delta|V|$, i.e. any path P of length $k\hat{L} + |V|$ ending in p has a delay less than or equal to $k\hat{D} + \Delta|V|$.

As in the proof of Lemma 3, decompose P into $C_1, \ldots, C_j$, ending with a simple path P', whose start and end processors are the same as those of P. Clearly,

$$L(P) = L(P') + \sum_{i=1}^{j} L(C_i) \quad \text{and}$$

$$D(P) = D(P') + \sum_{i=1}^{j} D(C_i)\ .$$

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

$$\begin{aligned}
D(P) &\leq D(P') + \frac{1}{\hat{R}} \sum_{i=1}^{j} L(C_i) \\
&\leq \Delta\, L(P') + k\hat{D} + \frac{|V| - L(P')}{\hat{R}} \\
&= \Delta\, L(P') + k\hat{D} + \frac{\hat{D}}{\hat{L}} (|V| - L(P')) \\
&\leq \Delta\, L(P') + k\hat{D} + \Delta\, (|V| - L(P')) \\
&= k\hat{D} + \Delta|V|,
\end{aligned}$$

which completes the proof of the lemma. □

Theorem 3

$\boldsymbol{R}(\boldsymbol{N}) = \hat{R}$.

Proof

By Lemma 4, for every processor p and every nonnegative integer k,

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

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

$$\frac{N(p,\ t)}{t} > \frac{k\hat{L} + |V| + 1}{t_0 + (k+1)\hat{D} + \Delta|V|}$$

But

$$\lim_{k \to \infty} \frac{k\hat{L} + |V| + 1}{t_0 + (k+1)\hat{D} + \Delta|V|} = \frac{\hat{L}}{\hat{D}} = \hat{R}.$$

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. 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 $\boldsymbol{N}$ with weights on its edges. By using the delays as weights, a similar algorithm solves the problem of finding $\hat{A}$ $(= 1 / \hat{R})$, the maximum cycle mean. Thus, by Theorem 3, $\hat{A} = 1 / \boldsymbol{R}(\boldsymbol{N})$. The validity of this algorithm is based on the following Theorem.

Theorem K

Let $\boldsymbol{N}$ be a strongly connected network. For any vertex q

$$\hat{A} = \max_{p \in V} \min_{0 \le k < |V|} \left\{ \frac{F_{|V|}(p, q) - F_k(p, q)}{|V| - k} \right\}. \tag{1}$$

Proof

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

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 $F_{|V|}(p, q) \le \pi(p, q)$. Also,

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

Thus,

$$\begin{aligned} \min_{0 \le k \le |V|-1} \{F_{|V|}(p, q) - F_k(p, q)\} &= F_{|V|}(p, q) - \max_{0 \le k \le |V|-1} \{F_k(p, q)\} \\ &= F_{|V|}(p, q) - \pi(p, q) \\ &\le 0, \end{aligned}$$

and

$$\min_{0 \le k \le |V|-1} \left\{ \frac{F_{|V|}(p, q) - F_k(p, q)}{|V| - k} \right\} \le 0. \tag{2}$$

Let us show that equality holds in (2) if and only if $F_{|V|}(p, q) = \pi(p, q)$.

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

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

and equality in (2) follows.

On the other hand, if $F_{|V|}(p, q) < \pi(p, q)$, then there exists a $0 \le k \le |V|-1$, such that $F_{|V|}(p, q) < F_k(p, q)$. For this k

$$\left\{ \frac{F_{|V|}(p, q) - F_k(p, q)}{|V| - k} \right\} < 0 ,$$

and strict inequality holds in (2).

Hence we can complete the proof by showing that there exists a p such that

$F_{|V|}(p, q) = \pi(p, q)$.

Let $\hat{C}$ be a critical cycle, and let w be a vertex in $\hat{C}$. 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 $\hat{C}$, 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 $\hat{C}$, such an initial part of length $|V|$ will occur; let its start vertex be w'. Then $F_{|V|}(w', q) = \pi(w', q)$. The choice of $p = w'$ completes the proof of Case 1.

Case 2: $\hat{A} \neq 0$.

Consider the effect of decreasing each edge delay $d(e)$ by a constant c. Clearly $\hat{A}$ is decreased by c, $(F_{|V|}(p, q) - F_k(p, q)) / (|V| - k)$ is decreased by c, and

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

is decreased 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

$$F_k(p, q) = \max_{p \to s \in E} \{ d(p,s) + F_{k-1}(s, q) \}$$

$k = 1, 2, \ldots, |V|$, with the initial conditions

$$F_0(q, q) = 0 \; ; \qquad F_0(r, q) = -\infty, \quad r \neq q \; .$$

The computation requires $O(|V| \cdot |E|)$ operations, and once the quantities $F_k(p, q)$ have been tabulated, one can compute

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

in $O(|V|^2)$ further operations. Since $\boldsymbol{N}$ is strongly connected, $|V| \leq |E|$, and the over-all computation time is $O(|V| \cdot |E|)$.*

We proceed to show that the data computed above suffices to find a critical cycle. There are several ways to do it, but we shall describe here the method indicated by Karp.

---

* 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 $|V|(n)$ may make use of some heavy (light) edges which will compensate for the cycle in it not being optimal. We show that Karp's statement is, in essence, correct.

Let p be the vertex for which the maximum is reached in (1), and let P be a directed path of length $|V|$ whose delay is $F_{|V|}(p,\ q)$. We shall prove that every (simple directed) cycle in P is critical.

Like Case 1, in the proof of Theorem K, first assume $\hat{A}\ =\ 0$. As was shown in that proof, in this case $F_{|V|}(p,\ q)\ =\ \pi(p,\ q)$. Thus, P cannot include any noncritical cycles, since their delay is negative, and by omitting them the delay of the path increases, contradicting the maximality of $\pi(p,\ q)$.

Now, if $\hat{A} \neq 0$, we show that the same claim holds. By subtracting $\hat{A}$ from the delay of every edge, the new $\hat{A}$ becomes zero, while the path P continues to maximize (1), since it yields a value zero for $(F_{|V|}(p,\ q)\ -\ F_k(p,\ q)\ )\ /\ (|V|\ -\ k)$. If P contains, originally, a noncritical cycle, then now it contains a negative-delay cycle, in contradiction to the claim in Case 1.

4. Analysis of the Transitory Stage and the Periodicity

We say that a processor p of $\boldsymbol{N}$ is *$k/s-$periodic* at time t_1, if for all $t\ \geq\ t_1$, if p sends M_i at t, then p will send M_{i+k} at $t\ +\ s$. Also, if at t_1 all processors in $\boldsymbol{N}$ are $k/s-$periodic, we say that $\boldsymbol{N}$ is *k/s-periodic* at t_1.

Observe that if at any time $\boldsymbol{N}$ is k/s-periodic, then its rate is k/s. By Theorem 3, $k/s\ =\ \hat{R}$. Thus, the specification of k implies that $s\ =\ k/\hat{R}$, and it suffices to say that $\boldsymbol{N}$ is *k-periodic*.

For every processor p and $n\ \geq\ 0$, let

$$\tau_n(p)\ \stackrel{\Delta}{=}\ \max_{q \to p\ \in\ E}\ \{\ t_n(q)\ +\ F_1(q,p)\ \};$$

that is, the first integral time on which p can send M_{n+1}, if it is not busy sending an earlier message. Therefore, $t_{n+1}(p)\ \geq\ \tau_n(p)$. In what follows, we show that if p is busy sending an earlier message, then this message is M_n, and M_{n+1} is sent on the next tick. That is, p lags behind its incoming messages by at most one tick. Lemma 6 states that if p lags behind its incoming messages, it has been lagging since the time it has sent M_0. Even when it sends M_0, in a sense it is lagging, since M_0 is sent at $\tau_0(p)$, when actually the incoming messages already allow p to send M_1. Furthermore, Lemma 6 shows that if M_n is lagging, then it and all messages preceeding it appear in one burst, with no tick skipped in between.

Lemma 5

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

Proof

Let n be the least index for which the claim is false. Therefore, $\tau_n(p)\ +\ 1\ \ <\ \ t_{n+1}(p)$. This implies that at time $\tau_n(p)\ +\ 1$ p is busy sending an earlier message. Thus, it suffices to show that

$$\tau_n(p) + 1 \leq t_n(p) \tag{3}$$

is false.

Assume $n > 0$. Clearly, $\tau_{n-1}(p) + 1 \leq \tau_n(p)$. Thus, (3) implies that $\tau_{n-1}(p) + 2 \leq t_n(p)$, contradicting the assumption that n is the least index for which the claim is false.

If $n = 0$, (3) reduces to $\tau_0(p) + 1 \leq t_0(p)$, contradicting the ground rule that M_0 is sent out one tick after W: The last M_0 to reach p arrives at time τ_0. Therefore, the first W to reach p arrives by time $\tau_0(p) - 1$. It follows that p sends W by time $\tau_0(p) - 1$, and therefore sends M_0 by $\tau_0(p)$. A contradiction. □

Note that in this proof we have also shown that

$$\tau_n(p) \geq t_n(p) , \tag{4}$$

and if equality holds then M_n and M_{n+1} are both lagging.

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 trivial. Let us show that if the claim holds for $0 \leq i < n$, then it holds for $i + 1$. Observe that $\tau_{n-i-1}(p) \leq \tau_{n-i}(p) - 1$. 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 (4), $t_{n-i-1}(p) \leq \tau_{n-i-1}(p)$. 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, $t_{n-i}(p) = \tau_{n-i-1}(p) < \tau_{n-i}(p)$, in contradiction to the inductive hypothesis. □

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

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

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 ticks $\tau_n(p) - i$, $0 \leq i \leq n$. Also, for every in-neighbor q, $t_i(q) < \tau_i(p) = t_i(p)$.

If the distance from p to r is l ($\leq |V| - 1 \leq n - 1$) then $t_l(r) \geq t_l(p)$. The proof of this claim is by induction on l: Assume the distance from p to s is $l + 1$, and the last edge on some shortest path from p to s is $r \rightarrow s$. 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 ticks p does not skip, $t_{l+1}(p) = t_l(p) + 1$. This completes the proof of the claim.

For an in-neighbor q of p, 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 already know that

$$t_{n+k+1}(p) = \max_{q \in V} \{t_{n+k}(q) + F_1(q, p)\}.$$

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:

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

□

We begin by studying the periodicity of a cycle.

Theorem 5

If $\boldsymbol{N}$ is a (simple directed) cycle $C = \quad p_0 \rightarrow p_1 \rightarrow \cdots \rightarrow p_{|V|-1} \rightarrow p_0$, where p_0 satisfies $t_0(p_0) = t_0$, then $\boldsymbol{N}$ is $|V|$-periodic at time $t_0 + D$.

Proof

For $n \geq |V|$, $0 \leq i \leq |V| - 1$, by Theorem 4,

$$\begin{aligned} t_{n+|V|}(p_i) &= \max_{q \in V} \{\ t_n(q) + F_{|V|}(q, p_i)\ \} \\ &= t_n(p_i) + D. \end{aligned}$$

Thus, every processor p_i is $|V|/D-$periodic at time $t_{|V|}(p_i)$. 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

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

□

Let $\boldsymbol{N}' = (V', E')$ be the subgraph obtained by removing from $\boldsymbol{N}$ all edges and vertices which are not in some critical cycle. The graph $\boldsymbol{N}'$ consists of (disjoint) strongly connected components, called *critical components*. A critical component, S, will denote a subnet (consisting of a directed graph and delays assigned to its edges) as well as just the set of vertices in it.

Lemma 7

Every cycle in a critical component is critical in $\boldsymbol{N}$.

Proof

Let $C = p_0 \to p_1 \to \cdots \to p_{n-1} \to p_0$ be a cycle in a critical component S. For each edge $p_i \to p_{i+1}$ there is a directed path Π_i both in $\boldsymbol{N}$ and in S, which completes $p_i \to p_{i+1}$ into a critical cycle. The average delay of each of these cycles is $\hat{A}$.

Allowing repetitions, the ratio of the sum of the delays of all edges in these cycles, divided by the number of edges, is exactly $\hat{A}$. Therefore, if the average delay of C is less than $\hat{A}$, then the directed circuit which consists of chaining all Π_i's, $0 \leq i \leq n-1$, has an average delay higher than $\hat{A}$. Therefore, there is a cycle in $\boldsymbol{N}$ whose average delay is higher than $\hat{A}$. A contradiction. □

Let S be a critical component. We shall show that there exists an l such that every $p \in S$ is eventually $l-$periodic (in $\boldsymbol{N}$). Moreover, $l \leq gl$, where gl is the greatest common divisor (*gcd*) of the lengths of all critical cycles in S.

In the next theorem we show that the periodicity of each processor in V' is at most $|V|$. Later we shall present networks (G_k), in which the periodicity of processors not in V', is exponential in $|V|$.

Theorem 6

For each S there exists 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 $n \geq n_0$,

$$t_{n+\hat{L}}(p) = t_n(p) + \hat{D}.$$

By Lemma 4, for all $k \geq 0$

$$N(p,\ t_0 + k\hat{D} + \Delta|V|) \geq k\hat{L} + |V| + 1. \tag{5}$$

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 are 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 (5), $m\ +\ k\hat{L}\ -\ c \geq k\hat{L}\ +\ |V|\ +\ 1$, 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 $n\ \geq\ n_0,\ t_{n+\hat{L}}(p)\ =\ t_n(p)\ +\ \hat{D}$, i.e. p is $\hat{L}/\hat{D}$-periodic. Moreover, there exists an n_0 such that for every $\hat{C}$ in ***N***, every p in $\hat{C}$ and every $n\ \geq\ n_0$, $t_{n+\hat{L}}(p)\ =\ t_n(p)\ +\ \hat{D}$.

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_n 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 critical component S 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 1-periodic (or $1/\hat{A}$-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 Δ is the greatest edge-delay, and $\hat{A}\ =\ 1\ /\ \hat{R}$. Denote by δ 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 $\bar{d}(e)\ \ \stackrel{\Delta}{=}\ \ d(e)\ -\ \hat{A}$. Hence, $\bar{\Delta}\ =\ \Delta\ -\ \hat{A}$ is the greatest shifted delay, while $\bar{\delta}\ =\ \delta\ -\ \hat{A}$ is the smallest shifted delay. The functions $\bar{D}(\cdot)$, and $\bar{F}_k(\cdot,\cdot)$ 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*, $\bar{t}_k(p)$, as follows

$$\overline{t}_k(p) \ \stackrel{\Delta}{=}\ \max_{q \in V} \{ \ t_{|V|}(q) + \overline{F}_k(q,p) \ \}.$$

By Theorem 4,

$$\overline{t}_k(p) = t_{|V|+k}(p) - k \cdot \hat{A}.$$

For a path P from q to p, let

$$\overline{T}(P) \ \stackrel{\Delta}{=}\ t_{|V|}(q) + \overline{D}(P) \ .$$

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 $b \le |V|$.

If there are noncritical cycles in ***N***, let $f > 0$ be the minimum absolute value of the shifted delays of the noncritical cycles. (Notice that the shifted delay of every noncritical cycle is negative.) If all cycles in ***N*** are critical then f is not defined.

Lemma 8

If f is defined then

$$f \ge \frac{1}{|V|} \ .$$

Proof

Let C be a noncritical cycle of length L and delay D which determines f. Clearly

$$D < L \cdot \hat{A} = L \cdot \frac{a}{b}.$$

Thus,

$$D \le \left\lceil L \frac{a}{b} \right\rceil - 1,$$

and

$$-f = \overline{D} \ \stackrel{\Delta}{=}\ D - L \cdot \hat{A} \le \left\lceil L \frac{a}{b} \right\rceil - L \frac{a}{b} - 1 \ .$$

Therefore,

$$f \ge L \cdot \frac{a}{b} - \left\lceil L \frac{a}{b} \right\rceil + 1 > 0.$$

It follows that

$$f \geq \frac{1}{b} \geq \frac{1}{|V|} .$$

□

Let T be defined by

$$T \ \stackrel{\Delta}{=} \ \max_{p,q \in V} \mid t_{|V|}(p) - t_{|V|}(q) \mid .$$

Lemma 9

$$T \leq |V| \cdot (\Delta - 1) .$$

Proof

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

On the other hand, by Lemma 2, for every r, $t_{|V|}(r) \leq t_0 + F_{|V|}(r)$. Thus, for every pair of processors, p and q, one has

$$|V| - F_{|V|}(q) \ \leq \ t_{|V|}(p) - t_{|V|}(q) \ \leq \ F_{|V|}(p) - |V|$$

or,

$$\mid t_{|V|}(p) - t_{|V|}(q) \mid \ \leq \ \max\{ F_{|V|}(p) , F_{|V|}(q) \} - |V| .$$

However, $\max\{ F_{|V|}(p) , F_{|V|}(q) \} \leq \Delta \cdot |V|$, and the Lemma follows. □

Denote by S_i, $1 \leq i \leq c$, the critical components of $\boldsymbol{N}$, and by gl_i the *gcd* of the lengths of all critical cycles of S_i. Define

$$\lambda \ \stackrel{\Delta}{=} \ lcm(gl_1 , gl_2 , . . . , gl_c) ,$$

where *lcm* denotes the least common multiple operation. As we shall see, λ is related to the periodicity of the network. Also, we will show that the number of messages sent during the transitory stage is related to l_0, where

$$l_0 \ \stackrel{\Delta}{=} \begin{cases} \dfrac{|V|}{f} \left[(\Delta - \delta)(|V| - 1) + T \right] + |V| - 1 & \text{if } f \text{ is defined,} \\ |V| - 1 & \text{if } f \text{ is not defined.} \end{cases}$$

Let s be a processor in $\boldsymbol{N}$. In the remainder of this section s remains fixed. Define

$$\overline{D}_r \overset{\Delta}{=} \max \{ \ \overline{t}_k(s) \ : \ k \ \equiv \ r \ (\text{mod } \lambda) \ , \ k > l_0 \ \},$$

and let P_r be a path for which the maximum is attained; i.e., P_r starts in some $w \in V$, ends in s, has length $L(P_r) > l_0$, $L(P_r) \equiv r \ (\text{mod } \lambda)$, and

$$\overline{D}_r = t_{|V|}(w) + \overline{F}_{L(P_r)}(w, s) = \overline{T}(P_r).$$

Let us show that $\overline{D}_r$ is defined. For every k ($> l_0$ and $\equiv r \ (\text{mod } \lambda)$), there is a path of length k leading to s. 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 $\overline{\Delta}(|V| - 1)$, proving that such a sequence is finite.

Assume a decomposition of P_r is performed (as in the proof of Lemma 3). Discard all the critical cycles. Let the *noncritical part* of P_r be the remaining graph, which may have noncritical cycles and may not be connected. Denote the number of edges of the noncritical part of P_r by L_r. Note that if all cycles are critical then P_r is a simple path and therefore $L_r \le |V| - 1$.

Lemma 10

$$L_r \le l_0 .$$

Proof

If all cycles are critical the Lemma follows from the comment above. It remain to prove the Lemma when f is defined.

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 cycles (per the decomposition). A noncritical cycle has a negative delay of at most $-f$. Thus,

$$\overline{D}_r \le \max_{q \in V} \{ \ t_{|V|}(q) \ \} + \overline{\Delta}(|V| - 1) + \frac{L_r - (|V| - 1)}{|V|} \cdot (-f),$$

where the second term is an upper bound on the shifted delay of the simple path and the third term is an upper bound on the cumulative shifted delay of the noncritical cycles (a lower bound on the number of noncritical cycles, times an upper bound on the shifted delay of such a cycle).

Let us construct a path P'_r to s, of length greater than l_0 and congruent to r modulo λ, which 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 this path by turning backwards around $\hat{C}$ until a length congruent to r modulo λ is obtained. Thus,

$$\overline{D}(P'_r) \ge \overline{\delta}(|V| - 1).$$

Also,

$$\overline{D}_r \geq \overline{D}(P'_r) + \min_{q \in V} \{ t_{|V|}(q) \} ,$$

and

$$\max_{q \in V} \{ t_{|V|}(q) \} + \overline{\Delta}(|V| - 1) + \frac{L_r - (|V| - 1)}{|V|} \cdot (-f) \geq \overline{\delta}(|V| - 1) + \min_{q \in V} \{ t_{|V|}(q) \}.$$

Since $\max_{q \in V} \{ t_{|V|}(q) \} - \min_{q \in V} \{ t_{|V|}(q) \} = T$ it follows that

$$L_r \leq \frac{|V|}{f} \left[(\overline{\Delta} - \overline{\delta})(|V| - 1) + T \right] + |V| - 1$$

□

In the proof of Lemma 12, we use Lemma 11, as well as the following theorem, which is a direct consequence of Theorem 1 in Brauer, [B].

Theorem B

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

(1) g divides a, and

(2) $a \geq (L_1 - g)(L_c - g)$,

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 $u \xrightarrow{e'} v$ in E' there exists an edge $u \xrightarrow{e} v$ 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 11

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 (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 $|E'| < |V|^2$. $\square$

We continue our study of P_r.

Lemma 12

Let $n_0 \triangleq l_0 + 2|V|^2$. 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 9, and denote by L_r the length of the noncritical part of P_r. This noncritical part is obtained by removing some critical cycles from P_r. In fact, by Lemma 10, at least one critical cycle is removed. Denote by $S_1, S_2, \ldots, S_m$, $m \geq 1$, the sets of vertices of critical components of $\boldsymbol{N}$, each of which contains a critical cycle removed from P_r. As in Lemma 11, construct for each S_i, $1 \leq i \leq m$ a *SED*, with a set of edges E_i, such that $|E_i| < |S_i|^2$. By Lemma 7, every cycle in such a *SED* is critical. Consider the digraph, $G'(V', E')$, consisting of the noncritical 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, $G'(V', E')$ 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

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

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 $L_1 \leq L_2 \leq \cdots \leq L_c$ be the different lengths of all (critical) cycles of the components

$S_1, S_2, \ldots, S_m$. It follows that the number $L(P')$ satisfies

$$L(P_r) = L(P') + \sum_{i=1}^{c} x_i L_i \ , \tag{6}$$

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 circuit is a union of edge disjoint cycles.)

Let $M \overset{\Delta}{=} n - L(P')$. Since $n \geq l_0 + 2|V|^2$, and $L(P') < l_0 + |V|^2$, it follows that $M > |V|^2$.

Let $g = gcd(L_1, L_2, \ldots, L_c)$. Therefore,

$$M > (L_1 - g)(L_c - g) \ .$$

Also,

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

Since $g \mid \lambda$,

$$M \equiv L(P_r) - L(P') \pmod{g} \ .$$

Equation (6) implies that $g \mid (L(P_r) - L(P'))$. Thus,

$$g \mid M \ .$$

By Theorem B, the equation

$$\sum_{i=1}^{c} x_i L_i = M \tag{7}$$

has a nonnegative solution. Use this solution to define P by adding to P' the cycles, as specified by the x_i's in (7). Thus,

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

and since the cycles added to P' are critical,

$$\overline{T}(P) = \overline{D}_r .$$

□

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

Theorem 7

Every processor s is $\lambda-$periodic after sending $|V| + l_0 + 2|V|^2$ $(= O(\Delta|V|^3))$ messages.

Proof

Lemma 12 implies that for every $n \geq n_0$,

$$\overline{t}_{n+\lambda}(s) \;=\; \overline{t}_n(s),$$

since each is equal to $\overline{D}_r$.

It follows that

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

and

$$t_{|V|+n+\lambda}(s) \;=\; t_{|V|+n}(s) + \lambda\hat{A}. \tag{8}$$

Therefore, s is $\lambda-$periodic after sending $|V| + l_0 + 2|V|^2$ messages. □

The following bound on the time to reach periodicity is derived by applying Lemma 2 to the result of Theorem 7, and noting that $t_0 \leq \Delta\cdot|V|$.

Corollary 3

The network ***N*** is λ-periodic by time $t_0 + \Delta\cdot(|V| + l_0 + 2|V|^2)\,(\,=\,O(\,\Delta^2|V|^3)\,)$.

Observe that formula (8), 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.

5. Lower Bounds on Transitory Stage and Periodicity

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 λ is exponential in the number of processors, and if all processors wake-up simultaneously then the period is exactly λ.

Assume that all processors wake-up simultaneously, and that the time they all send M_0 is 0. Under these circumstances, we show two more lemmas.

Lemma 13

If $t_0(p) = 0$ for every p, then $t_{n+1}(p) = \tau_n(p)$, for every p and $n \geq 0$.

Proof:

Let p be a processor and let n be the least integer for which the claim is false. By Lemma 5, $t_{n+1}(p) = \tau_n(p) + 1$. This implies that $t_n(p) = \tau_n(p)$. By Lemma 6, $t_0(p) = \tau_0(p)$. This contradicts the fact that $t_0(p) = 0$. □

Lemma 14

If $t_0(p) = 0$ for every p, then for every p, $n \geq 0$ and $k \geq 1$,

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

This Lemma is similar to Theorem 4, with the additional condition that $t_0(p) = 0$ for every p, but relaxing the condition that $n \geq |V|$.

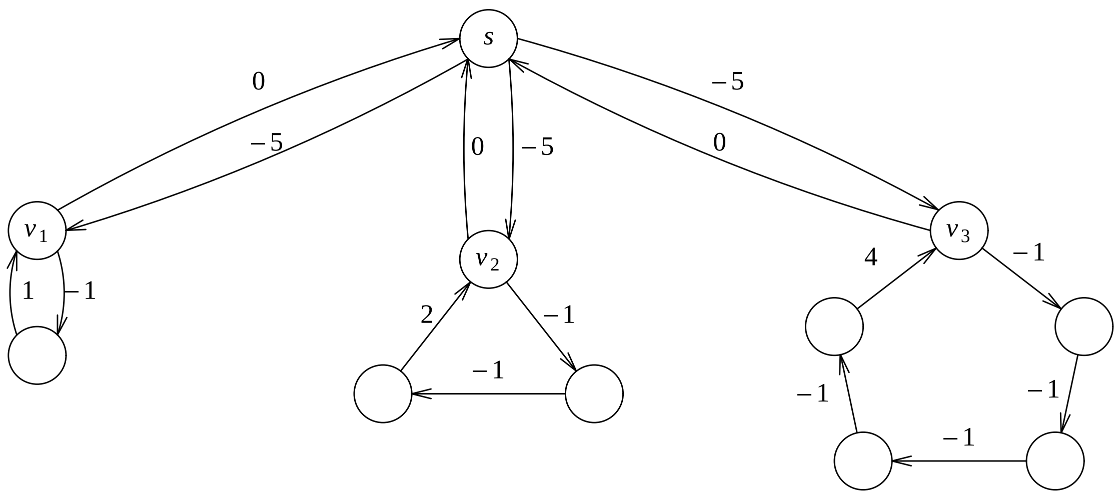
Proof:

The proof is similar to that of Theorem 4, with the following change. The case of $k = 1$ is provided by Lemma 13. There is no change in the proof of the inductive step. □

Now, Lemma 14, if $n = 0$, implies that

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

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 jth 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 shifted delay $-p_i$, the one going into s is of shifted delay 0. The shifted delay of the edge in the cycle C_j, entering v_j, has delay $p_j - 1$. All other edges have shifted delay -1.



The Network G_5

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 shifted delay 0. Assume $t_0(v) = 0$, for all vertices v in G_m. Then, for $n \geq 1$,

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

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

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

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 λ', then

$$\bar{t}_{n+\lambda'}(s) = \bar{t}_n(s) .$$

But observe that

$$\bar{t}_{n+\lambda'}(s) > 0,$$

because there exists a $1 \leq k \leq i$ such that p_k does not divide λ', 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 shifted delay greater than 0. Thus, the period is exactly λ.

If p denotes a prime number, then

$$\lambda = \prod_{p \leq m} p,$$

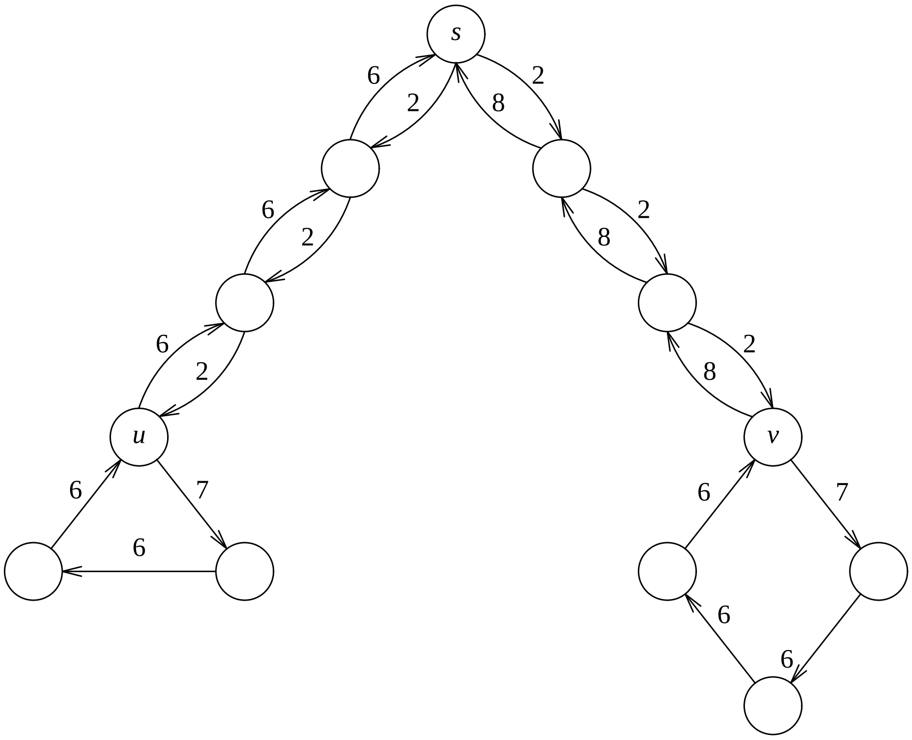
and hence

$$\lambda > 2^{\frac{m}{4}}$$

(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 $2^{\frac{n_m^{1/2}}{4}}$).

Next, it is shown 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 shifted delay $-2d - \frac{1}{k}$; the edges in the direction from u to s have shifted delay $-\frac{1}{k}$. The shifted delays of the other path are $-2d - \frac{1}{k}$, in the direction from s to v, and $d - \frac{1}{k}$ on the edges from v to s. Finally, the delays of the edges in the cycles are $-\frac{1}{k}$ except for two edges, which have delay $\frac{k-1}{k}$: the one going out of u and the one going out of v.

Observe that $\overline{D}(\hat{C}) = 0$, and $\overline{D}(C) = -\frac{1}{k}$. One can construct a network with delays that correspond to the shifted delays of H_k, by choosing $\hat{A} = 3d + \frac{1}{k}$, for example, and adding $\hat{A}$ to the shifted delay of each edge. In this case Δ is equal to $4d$.



The Network H_3 with $d = 2$ and $\hat{A} = 6\ \frac{1}{3}$

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 shifted delay -1. By Lemma 10, if $n \geq n_0$ (n_0 as defined in Lemma 12), such a heaviest path must use vertices of $\hat{C}$. Therefore, using P_1,

$$\bar{t}_{n+k(k+1)}(s) = \bar{t}_n(s) = -1,$$

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 shifted delay of P_2 is :

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

Thus, while $kd > c$, the path P_2 is heavier than P_1. Namely,

$$\bar{t}_{k+ck(k+1)}(s) > -1,$$

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

Acknowledgement

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

References

[A] B. Awerbuch, "Complexity of Network Synchronization", JACM, Vol. 32, No. 4, Oct. 1985, pp. 804-823.

[A85] B. Awerbuch, "Reducing Complexities of Distributed Maximum Flow and Breadth-First Search Algorithms by Means of Network Synchronization", Networks, Vol. 15, No. 4, Winter 1985, pp. 425-437.

[AS88] B. Awerbuch, M. Sipser, "Dynamic Networks are As Fast As Static Networks". In Proc. 29th IEEE Symp. on Foundations of Computer Science, October 1988, pp. 206-220.

[B] A. Brauer, "On a Problem of Partitions", American J. Math., Vol. 64, 1942, pp. 299-312.

[BG89] V.C. Barbosa, E. Gafni, "Concurrency in Heavily Loaded Neighborhood-Constrained Systems", ACM Trans. on Programming Languages and Systems, Vol. 11, No. 4, Oct. 1989, pp. 562-584.

[CGZ86] C.T. Chou, I.S. Gopal, S. Zaks, "Synchronizing Asynchronous Bounded-Delay Networks". Research Report RC-12274, IBM Yorktown, October 1986.

[CL] K. M. Chandy and L. Lamport, "Distributed Snapshots: Determining Global States of Distributed Systems", ACM Trans. on Computer Systems, Vol. 3, No. 1, Feb. 1985.

[CHEP] F. Commoner, A.W. Holt, S. Even, A. Pnueli, "Marked Directed Graphs", J. of Computer and System Sciences, Vol. 5, No. 5, Oct. 1971, pp. 511-523.

[ER] S. Even, S. Rajsbaum, "Unison, Canon and Sluggish Clocks in Networks Controlled by a Synchronizer", to appear in Mathematical Systems Theory. The first part of this paper, "Unison in Distributed Networks", appeared in *Sequences, Combinatorica, Compression, Security, and Transmission*, R.M. Capocelli (ed.), Springer-Verlag, 1990, pp. 479-487.

[FLS87] A. Fekete, N. Lynch, L. Shrira, "A Modular Proof of Correctness for a Network Synchronizer", In Proc. of the 2nd Workshop on Distributed Algorithms, CWI, July 1987, pp. 219-256.

[HW] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*, 4-th edition, 1968, Oxford University Press.

[GE] H. J. Genrich, "Einfache Nicht-Sequentielle Prozesse", Gesellschaft fur Mathematik und Datenverarbeitung, Birlinghoven, West Germany, 1970.

[K] R. M. Karp, "A Characterization of the Minimum Cycle Mean in a Digraph", Discrete Mathematics, Vol. 23, 1978, pp. 309-311.

[M] E.F. Moore, "The Firing Squad Synchronization Problem", in Sequential Machines, Selected Papers, Addison-Wesley, 1964, pp. 213-214.

[MMZ88] Y. Malka, S. Moran, S. Zaks, "A Lower Bound on the Period Length of a Distributed Scheduler", Algorithmica, No. 10, 1994, pp. 383-398.

[MR91] Y. Malka, S. Rajsbaum, "Analysis of Distributed Algorithms Based on Recurrence Relations," in Proc. of 5th Int. Workshop on Distributed Algorithms, Delphi, Greece, Oct. 7--9: S. Toueg et.al. (Eds.), LNCS, Vol. 579, Springer Verlag, 1991, pp. 242-253.

[PU89] D. Peleg, J. Ullman, "An Optimal Synchronizer for the Hypercube", SIAM J. on Computing, Vol. 18, No. 2, 1989, pp. 740-747.

[R] R. Reiter, "Scheduling Parallel Computations", J. of the ACM, Vol. 15, No. 4, Oct. 1968, pp. 590-599.

[RS] S. Rajsbaum, M. Sidi, "On the Performance of Synchronized Programs in Distributed Networks with Random Processing Times and Transmission Delays", IEEE Trans. on Parallel and Distributed Systems, Vol. 5, No. 9, Sept. 1994, pp. 939-950.