STATE UNIVERSITY OF NEW YORK AT

STONY BROOK

CEAS TECHNICAL REPORT 612

Performance Evaluation of Distributed Communication Systems for Load Balancing

T. Hsiung and T.G. Robertazzi

December 17, 1991

Performance Evaluation of Distributed Communication Systems for Load Balancing

T.Hsiung and T. G. Robertazzi
Dept. of Electrical Engineering,
SUNY at Stony Brook,
Stony Brook, NY 11794

Abstract

A load sharing problem involving the optimal allocation of measurement data among n processors interconnected through different types of communication networks is considered where the processors' architectural configuration includes front-end processors. Considered are a bus-oriented network, a linear daisy chain network, and tree networks. The objective is to evaluate the performance of each network. Comparisons are made among these networks under identical given conditions so that the most effective configurations can be determined.

1 Introduction

The efficiency of a type of parallel computation involving a number of processors which are tied together by an interconnection network are examined in this paper. The basic idea [2,3,14,15] is that one communicating processor receives a burst of measurement data(the processing load) and distributes the processing load to other processors in order to achieve a minimal solution time thru parallel processing. Here, we assume that every processor in the interconnection network has the same computational speed and each link between processors has the same channel capacity. To achieve the best performance, processors with front-end processors that can communicate and compute at the same time are used. Also, the time taken for each processor to report its solution to the starting processor is assumed to be very small.

This paper is organized as follows. In section II, III, and IV, a bus oriented network, a linear daisy chain network, and a tree network are examined, respectively. In section V, their performance will be presented. Finally section VI is the conclusion.

2 Bus Interconnection Network

Consider the case where the network model consists of n communicating processors attached to a linear bus (Fig. 1). The load may originate at any of the n homogeneous processors. The originating processor immediately begins computation on its share of the load while broadcasting the remaining load over the bus to the other processors. Each processor begins to compute its share at the moment that it finishes receiving its data.

Let us first introduce the following notation.

α_i: The fraction of measurement data that is assigned to processor i by the originating processor.

w_i : A constant that is inversely proportional to the speed of the ith processor. The ith processor can process the entire load in time $w_i T_{cp}$.

Z : A constant that is inversely proportional to the speed of the single bus. The entire load can be transmitted over the bus in time ZT_{cm}.

T_{cp} : The time that it takes the ith processor to process the entire load when $w_i = 1$.

T_{cm} : The time that it takes the processor that distributes the load to transmit all the measurement data when $Z = 1$.

T_i : The total time that elapses between the beginning of the process at $t = 0$ and the time when processor i completes its computation, $i = 1, 2, \ldots, n$. This includes, in addition to computation time, communicating time and waiting time. Waiting time is the time between the start of the communication by the originating processor and the time that the ith processor begins to receive its share of the load .

T_f : The finish time of the process is the time when the last processor finishes processing.

$$T_f = max(T_1, T_2, \cdots, T_n) \tag{2.1}$$

The timing diagram of the system appears in Fig. 2. During the period where $t = \alpha_2 ZT_{cm}$, the first processor computes its share of the load and communicates with the second processor. All other processors, processors $3, 4, 5 \ldots, n$, are idle. The equations that relate various variables and parameters together are stated below:

$$T_1 = \alpha_1 w_1 T_{cp} \tag{2.2}$$

$$T_2 = \alpha_2 ZT_{cm} + \alpha_2 w_2 T_{cp} \tag{2.3}$$

$$T_3 = (\alpha_2 + \alpha_3)ZT_{cm} + \alpha_3 w_3 T_{cp} \tag{2.4}$$

$$T_4 = (\alpha_2 + \alpha_3 + \alpha_4)ZT_{cm} + \alpha_4 w_4 T_{cp} \tag{2.5}$$

.

.

.

$$T_n = (\alpha_2 + \alpha_3 + \cdots + \alpha_n)ZT_{cm} + \alpha_n w_n T_{cp} \tag{2.6}$$

The fractions of the total measurement load should sum to one

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1 \tag{2.7}$$

The objective in analyzing the above equations is to compute the minimum finish time. It can be seen intuitively, that in order to obtain maximum parallelism and a minimum time solution, all processors must stop at the same time. This is because, otherwise, some processors would be idle while others were busy[14,15]. Another way of expressing this intuition is to say one must keep all processors utilized until the last moment; that is, all processors stop at the same time. This achieves the maximum efficiency in the system. The optimal values of α's that the originating processor should calculate in order to achieve the minimum finish time can be computed by solving recursively the following set of equations :

$$\alpha_{n-1} = \alpha_n \frac{w_n T_{cp} + ZT_{cm}}{w_{n-1} T_{cp}} \tag{2.8}$$

.

.

.

$$\alpha_3 = \alpha_4 \frac{w_4 T_{cp} + ZT_{cm}}{w_3 T_{cp}} \tag{2.9}$$

$$\alpha_2 = \alpha_3 \frac{w_3 T_{cp} + ZT_{cm}}{w_2 T_{cp}} \tag{2.10}$$

$$\alpha_1 = \alpha_2 \frac{w_2 T_{cp} + ZT_{cm}}{w_1 T_{cp}} \tag{2.11}$$

Here, α_i is solved for by equating T_i to T_{i+1}. Since we already made an assumption that all processors in the network have the same computational speed, all w_is should be

equal to each other. For simplicity, we will use w instead of the w_is in the rest of the paper. Therefore, the α_is can be expressed as follows from the above equations:

$$\alpha_{n-1} = \alpha_n \frac{wT_{cp} + ZT_{cm}}{wT_{cp}} \tag{2.12}$$

.

.

.

$$\alpha_3 = \alpha_n (\frac{wT_{cp} + ZT_{cm}}{wT_{cp}})^{n-3} \tag{2.13}$$

$$\alpha_2 = \alpha_n (\frac{wT_{cp} + ZT_{cm}}{wT_{cp}})^{n-2} \tag{2.14}$$

$$\alpha_1 = \alpha_n (\frac{wT_{cp} + ZT_{cm}}{wT_{cp}})^{n-1} \tag{2.15}$$

Let

$$r = \frac{wT_{cp} + ZT_{cm}}{wT_{cp}} \tag{2.16}$$

From equation(2.7), we get

$$\alpha_n (r^{n-1} + r^{n-2} + \cdots + r + 1) = 1 \tag{2.17}$$

$$\alpha_n = \frac{r-1}{r^n - 1} \tag{2.18}$$

From the timing diagram Fig. 2, the minimum finish time is $\alpha_1 wT_{cp}$, which is given by

$$T_f = wT_{cp} \frac{r^{n-1}(r-1)}{r^n - 1} \tag{2.19}$$

3 Linear Daisy Chain Network

For the linear daisy chain case, every processor can communicate with only its right and left immediate neighbors. The performance is affected by the position of the starting processor. In the following, the cases with origination at the boundary and from the network interior will be both examined.

3.1 Origination at Network Boundary

Suppose that the processor at left end of the chain receives a burst of measurement data and is to share the data with the other N-1 processors. The starting processor then divides the

processing load into N smaller parts optimally. It keeps the fraction $\hat{\alpha_1}$ of the received processing load for itself and transmits the remaining measurement data to its right immediate neighbor(the second processor). Upon receiving the measurement data, the second processor keeps the fraction $\hat{\alpha_2}$ of what it has received to process and transmits the remaining to its right immediate neighbor(the third processor). For the ith processor, it keeps the fraction $\hat{\alpha_i}$ of what it has just received and transmits the remaining to the i+1st processor. The process repeats itself until the Nth processor is reached. The timing diagram of the entire process is shown in Fig. 3. Here Z is the (common) link speed.

Again, in order to obtain maximun parallelism and a minimum time solution all the processors must stop computing at the same time. The starting processor should compute its fraction of the processing load during the entire processing period, so that the total processing time T_t equals the processing time of the starting processor. From Fig. 3 it can also be seen that the processing time $\alpha_i w T_{cp}$ of the ith processor equals the transmission time , $(1-\alpha_1-\alpha_2-\cdots-\alpha_i)ZT_{cm}$, from the ith processor to the i+1st processor plus the processing time, $\alpha_{i+1} w T_{cp}$, of its right immediate neighbor(the i+1st processor). Here the α_i's are the actual fraction of processing load of the ith processor and can be expressed as a function of $\hat{\alpha_i}$'s:

$$\alpha_1 = \hat{\alpha_1} \tag{3.1}$$

$$\alpha_i = \hat{\alpha_i} \prod_{j=1}^{i-1}(1-\hat{\alpha_j}), i = 2, 3, \ldots, N-1. \tag{3.2}$$

The total computing time of the ith processor equals

$$\alpha_i w T_{cp} = (1-\alpha_1-\alpha_2-\cdots-\alpha_i)ZT_{cm} + \alpha_{i+1} w T_{cp} \tag{3.3}$$

substituting (3.1)&(3.2) into (3.3) yields

$$\begin{aligned} \hat{\alpha_i} w T_{cp} \prod_{j=1}^{i-1}(1-\hat{\alpha_j}) &= [1-\hat{\alpha_1}-\hat{\alpha_2}(1-\hat{\alpha_1})-\cdots-\hat{\alpha_i}\prod_{j=1}^{i-1}(1-\hat{\alpha_j})]ZT_{cm} \\ &\quad + \hat{\alpha_{i+1}} w T_{cp} \prod_{j=1}^{i}(1-\hat{\alpha_j}) \qquad (3.4) \\ &= (1-\hat{\alpha_1})[1-\hat{\alpha_2}-\hat{\alpha_3}(1-\hat{\alpha_2})-\cdots-\hat{\alpha_i}\prod_{j=2}^{i-1}(1-\hat{\alpha_j})]ZT_{cm} \end{aligned}$$

$$+ \hat{\alpha_{i+1}} wT_{cp} \prod_{j=1}^{i}(1-\hat{\alpha_j}) \tag{3.5}$$

$$= (1-\hat{\alpha_1})(1-\hat{\alpha_2})[1-\hat{\alpha_3}-\hat{\alpha_4}(1-\hat{\alpha_3})-\cdots-\hat{\alpha_i}\prod_{j=3}^{i-1}(1-\hat{\alpha_j})]ZT_{cm}$$

$$+ \hat{\alpha_{i+1}} wT_{cp} \prod_{j=1}^{i}(1-\hat{\alpha_j}) \tag{3.6}$$

.

.

.

$$= (1-\hat{\alpha_1})(1-\hat{\alpha_2})(1-\hat{\alpha_3})\cdots(1-\hat{\alpha_i})ZT_{cm}$$

$$+ \hat{\alpha_{i+1}} wT_{cp} \prod_{j=1}^{i}(1-\hat{\alpha_j}) \tag{3.7}$$

$$= (ZT_{cm} + \hat{\alpha_{i+1}} wT_{cp}) \prod_{j=1}^{i}(1-\hat{\alpha_j}) \tag{3.8}$$

$$\hat{\alpha_i} wT_{cp} = (1-\hat{\alpha_i})ZT_{cm} + (1-\hat{\alpha_i})\hat{\alpha_{i+1}} wT_{cp}, i = 1, 2, \ldots, N-2. \tag{3.9}$$

when $i = N-1$ from (3.3)

$$\alpha_{N-1} wT_{cp} = (1-\alpha_1-\cdots-\alpha_{N-1})ZT_{cm} + \alpha_N wT_{cp} \tag{3.10}$$

$$= \alpha_N ZT_{cm} + \alpha_N wT_{cp} \tag{3.11}$$

$$= \alpha_N (ZT_{cm} + wT_{cp}) \tag{3.12}$$

Again, substituting (3.1)&(3.2) into (3.12)

$$\hat{\alpha_{N-1}} wT_{cp} \prod_{j=1}^{N-2}(1-\hat{\alpha_j}) = \hat{\alpha_N}(ZT_{cm} + wT_{cp}) \prod_{j=1}^{N-1}(1-\hat{\alpha_j}) \tag{3.13}$$

$$\hat{\alpha_{N-1}} wT_{cp} = \hat{\alpha_N}(1-\hat{\alpha_{N-1}})(ZT_{cm} + wT_{cp}) \tag{3.14}$$

But $\hat{\alpha_N} = 1$,

$$\hat{\alpha_{N-1}} wT_{cp} = (1-\hat{\alpha_{N-1}})(ZT_{cm} + wT_{cp}) \tag{3.15}$$

From (3.9),(3.15) and through some simple algebra, $\hat{\alpha_i}$ may be expressed as

$$\hat{\alpha_i} = \frac{ZT_{cm} + \hat{\alpha_{i+1}} wT_{cp}}{wT_{cp} + ZT_{cm} + \hat{\alpha_{i+1}} wT_{cp}} \tag{3.16}$$

$$\hat{\alpha_{N-1}} = \frac{ZT_{cm} + wT_{cp}}{2wT_{cp} + ZT_{cm}} \tag{3.17}$$

The $\hat{\alpha_i}$s can be solved recursively through (3.16) and (3.17). The total processing time is

$$T_t = \alpha_1 w T_{cp} \tag{3.18}$$

3.2 Origination from Network Interior

Suppose that the originating processor is in the middle instead of at the end of the linear daisy chain network. It shares the measurement data with N_l other processors to its left and N_r processors to its right. Since every processor in the network has only one front end processor, it will transmit in only one direction at a time. As the process starts, it first divides the processing load into smaller parts, then transmits the fraction β_l of the total processing load to its left immediate neighbor and the fraction β_r of the total processing load to its right immediate neighbor, and keeps the remaining fraction $1 - \beta_r - \beta_l$ for itself to compute at the same time. Upon receiving the data, the left first processor transmits the fraction $1 - \hat{\alpha_{l1}}$ of what it has received to its left immediate neighbor and keeps the remaining load for itself to compute. The whole process at the left repeats itself until the N_lth processor is reached. The same operation is performed by the right side of network until the N_rth processor is reached. The timing diagram of the entire process is shown in Fig. 4. Note that Z is the (common) link speed.

As in the above cases, a minimum solution time would be achieved when all the processors stop computing at the same instant. It follows that the total processing time is equal to the processing time of the originating processor. The timing equations can be constructed naturally by simply examining the timing diagram:

$$(1 - \beta_r - \beta_l) w T_{cp} = \beta_l Z T_{cm} + \beta_l \hat{\alpha_{l1}} w T_{cp} \tag{3.19}$$

$$\beta_l \hat{\alpha_{l1}} w T_{cp} = \beta_r Z T_{cm} + \beta_r \hat{\alpha_{r1}} w T_{cp} \tag{3.20}$$

where $\hat{\alpha_{li}}$ and $\hat{\alpha_{lr}}$ are the fraction of what the left and right ith processors have received for themselves to compute, respectively. The 0th processor is the starting processor. From (3.19) & (3.20), both β_l and β_r can be expressed as

$$\beta_l = \frac{w T_{cp}(Z T_{cm} + \hat{\alpha_{r1}} w T_{cp})}{(w T_{cp} + Z T_{cm} + \hat{\alpha_{l1}} w T_{cp})(Z T_{cm} + \hat{\alpha_{r1}} w T_{cp}) + \hat{\alpha_{l1}} w^2 {T_{cp}}^2} \tag{3.21}$$

$$\beta_r = \frac{\hat{\alpha_{l1}} w^2 {T_{cp}}^2}{(w T_{cp} + Z T_{cm} + \hat{\alpha_{l1}} w T_{cp})(Z T_{cm} + \hat{\alpha_{r1}} w T_{cp}) + \hat{\alpha_{l1}} w^2 {T_{cp}}^2} \tag{3.22}$$

Here,$\hat{\alpha_{li}}$ and $\hat{\alpha_{ri}}$ can be obtained from (3.16) and (3.17) since from Fig. 4 we can easily see that both the left side and the right side of the linear network are identical to the unidirectional case discussed earlier in section 3.1 with the left first processor and the right first processor as their starting processor,respectively. From (3.1) and (3.2) the actual fraction α_{li} and α_{ri} of the total processing load of the left and right ith processor can be calculated and from (3.21) and (3.22) β_l and β_r can be solved. The total processing time is then

$$T_t = (1 - \beta_l - \beta_r) w T_{cp} \tag{3.23}$$

4 Tree Network

Consider a tree network of communicating processors. Each processor can only communicate with its parent processor and children processors. Suppose that the root processor of the network receives a burst of measurement data and is to share the data with the other N-1 processors. It first keeps some fraction of data for its self to compute and distributes the remaining fraction of data to its children processors. Each child processor of the root processor keeps some fraction of what it has received and distributes the remaining load to its children processors(grandchildren of the root processor). The process continues until the processors in the lowest level are reached. The required numerical process proceeds in calculating the optimal allocation of processing load from the bottom of the tree to the top. The process is repeated until the fraction of the data that the root processor processes is determined. There are two basic types of subtrees in the network: those whose children processors are terminal nodes and those whose children processors are not terminal nodes. A terminal node has no children. An example of tree network is given in Fig. 5.

Consider a subtree of the network that consists of one parent processor and i-1 children processors that are terminal nodes of the network. The parent processor which has received some data D from its parent processor keeps a fraction α_i of D for itself to compute and transmits the remainder to its children in turn. The first child receives a fraction α_1 of D, the second child receives a fraction α_2 of D,..., and the i-1st child receives a fraction α_{i-1} of D. The timing diagram of the entire process is shown in Fig.6.

From the timing diagram, the relationships among the processors in the subtree can

be expressed by the following equations:

$$\alpha_i w T_{cp} = \alpha_1 Z T_{cm} + \alpha_1 w T_{cp} \quad (4.1)$$

$$\alpha_j w T_{cp} = \alpha_{j+1} Z T_{cm} + \alpha_{j+1} w T_{cp}, j = 1, 2, \cdots, i-2 \quad (4.2)$$

and

$$\alpha_1 + \alpha_2 + \cdots + \alpha_i = 1 \quad (4.3)$$

There are a total of i linear equations and i unknowns. The α_js can thus be determined. Note that Z is the (common) link speed.

Consider a subtree of the network consisting of one parent processor and k-1 children processors that are not terminal nodes of the network. As above, the parent processor which has received some data D from its parent processor keeps β_k fraction of D for itself to compute and transmits the remainder to its children in turn. The first child receives β_1 fraction of D, the second child receives β_2 fraction,..., and the k-1st child receives a fraction β_{k-1} of D. Upon receiving the β_1 fraction of D, the first child keeps γ_1 fraction of what it has just received and transmits the remainder to its l_1 children processors. For the jth child, it keeps γ_j fraction of what it has just received and transmits the remaining to its l_j children processors. Here l_j is the number of children of the jth child processor. The timing diagram is shown in Fig. 7. Again, equations can be constructed as follows from the timing diagram.

$$\beta_k w T_{cp} = \beta_1 Z T_{cm} + \beta_1 \gamma_1 w T_{cp} \quad (4.4)$$

$$\beta_j \gamma_j w T_{cp} = \beta_{j+1} Z T_{cm} + \beta_{j+1} w \gamma_{j+1} T_{cp} \quad (4.5)$$

and

$$\beta_1 + \beta_2 + \cdots + \beta_k = 1 \quad (4.6)$$

The β_j can be solved by the above k linear equations. The γ_js would have been determined from the next level below. If the next level is the lowest level of the tree, the value of γ_j can be obtained from (4.1)-(4.3) where α_1 corresponds to γ_j. If the next level is not the lowest level of the tree, γ_j can be obtained from (4.4)- (4.6) where γ_j corresponds to β_1.

If a subtree of the network consists of one parent processor and both terminal node and nonterminal node children, the γ_js of the terminal node processors should equal

to 1, which indicates these processors keep all of what they have received. The γ_js of the nonterminal node processors can be obtained either from (4.1)-(4.3) or from (4.4)-(4.6) depending on whether their next levels are the lowest or not.

5 Performance Evaluation

Load allocation for the three networks was implemented by running a computer program in order to evaluate their performances. This program was written in the C language and is based on the equations derived earlier in the previous three sections. Each architecture's performance was evaluated by computing the minimum total finish time. For the tree network, three types of tree configurations were investigated in the program. Refering to Fig.8, Fig.9, and Fig.10, these are the fully developed binary tree, left tree, and right tree respectively. We assumed that all parent processors in each tree start distributing data load from left to right. Comparisons were made amongst these five architectures under various situations.

The minimum total finish time of each network with $T_{cm} = 0.5$, $T_{cp} = 1.0$, $w = 1.0$, and $Z = 0.1, 0.2, 0.5, 1.0, 10$ and 20 is shown in table 1 thru table 6. Table 7 thru table 12 gives the minimum total finish time for $T_{cm} = 1.0$, $T_{cp} = 1.0$, $w = 1.0$, and $Z = 0.1, 0.2, 0.5, 1.0, 10$ and 20 and table 13 thru table 18 for $T_{cm} = 1.0$, $T_{cp} = 0.5$, $w = 1.0$ and $Z = 0.1, 0.2, 0.5, 1.0, 10$ and 20. Let n be the number of processors. Examing all cases from these tables, we found that the five networks have the same performance for $n \leq 2$. For $n = 3$, all networks except the linear daisy chain with origination from boundary have the same processing speed. It is apparent that all networks have the same configurations when $n \leq 2$ and all but the linear daisy chain retain the same structures when $n = 3$. If there are more than three processors, the bus oriented network has the best efficiency and the linear daisy chain has the worst efficiency . The binary tree would take the second place. The right tree and the left tree have the same processing speed. As a matter of fact, these results are as expected when we inspect their timing diagrams carefully. Some qualitative comments are made below.

Consider first the bus oriented network and the linear daisy chain network. The second processor in the bus oriented network starts computation immediately after receiving

its own processing load. The wait time for the second processor in the bus oriented network is $\alpha_2 ZT_{cm}$. While the second processor in the linear daisy chain network can not start computing until it completes receiving the total processing load except for the fraction that the first processor keeps for itself. If the linear daisy chain network has the same division of processing load as the bus oriented network, the second processor in the linear daisy chain network must wait for a longer amount of communication time, which is $(1-\alpha_1)ZT_{cm}$ or $(\alpha_2+\alpha_3+\cdots+\alpha_n)ZT_{cm}$, than its counterpart in the bus oriented network. The third processor, the forth processor,..., etc all have to suffer the longer communication delay for the certain fractions of processing load that are repeatedly transmitted over the channel until the nth processor is reached. This is illustrated in Fig.11. In this case the total finish time for the linear daisy chain would be $\alpha_1 wT_{cp}$ plus the extra time spent over the communication channel. This is not the minimum solution time because all processors would not stop at the same time.

To achieve minimum total finish time, the linear daisy chain has to use the optimal division scheme which has been introduced in section 3.1. This scheme would minimize the unnecessary communication delay in the linear daisy chain network by distributing a larger fraction of processing load to the first processor and a smaller fraction of processing load to the second processor and even smaller fractions of processing load to the third processor, and so forth. Denote $\alpha.daisy_i$ as the fraction of measurment data that is assigned to processor i by the originating processor in the linear daisy chain network with optimal division of load and $\alpha.bus_i$ as the same in the bus oriented network with optimal division of load. The previous statement can be mathematically expressed as follows:

$$\begin{aligned} \alpha.daisy_1 &> \alpha.bus_1 \\ (1-\alpha.daisy_1)ZT_{cm} &< (1-\alpha.bus_1)ZT_{cm} \end{aligned}$$

where $(1-\alpha.daisy_1)$ is the wait time of the second processor in the linear daisy chain with optimal division of load and $(1-\alpha.bus_1)$ is used as the wait time of the second processor in the linear daisy chain. Similarly the wait time of each processor in the linear daisy chain with optimal division of load is always less than that with the same division of load as the bus oriented network. Therefore the total wait time of the linear daisy chain network is

effectively reduced by the optimal division scheme so as to improve the efficiency of the network. Obviously this scheme has made a balance between the communication delay and computation time in the linear daisy chain network.

As discussed in the previous sections, a general solution time for each network with the optimal division of load used is found to be wT_{cp} multiplied by the fraction of data reserved for the starting processor to compute. A larger fraction of computation load is given to the first processor to compensate for the communication delay in the linear daisy chain. Therefore, the linear daisy chain could never be faster than the bus oriented network.

As far as tree network is concerned, it can be considered to be a combination of the bus oriented network and the linear daisy chain network. The closer it is to the bus oriented network, the less minimum total finish time it takes and vice versa. Consider the binary tree case where each node processor of the network is numbered in the same way as given in Fig.8. The second processor starts computation as soon as the fraction of measurement data for itself and its offspring have been completely received from the root processor. Again, if the binary tree network and the linear daisy chain network both have the same division of load as the bus oriented network, the wait time of transmission for the second processor in the binary tree network is $(\alpha_2 + \alpha_4 + \alpha_5 + \cdots)ZT_{cm}$. As we know,

$$\begin{aligned} \alpha_2 ZT_{cm} &< (\alpha_2 + \alpha_4 + \alpha_5 + \cdots)ZT_{cm} \\ &< (1 - \alpha_2)ZT_{cm} \end{aligned}$$

The second processor of the binary tree network spends a smaller amount of time on waiting for the arrival of data than the one in the linear daisy chain network but more time than the one in the bus oriented network. Similar analysis of communication delay can be applied to the rest of node processors. Therefore, since each processor in the binary tree network spends more time than its counterpart in the bus oriented network and less time than its counterpart in the linear daisy chain in waiting for the arrival of measurement data, the total time the binary tree network spends on the communication delay is hence shorter than the linear daisy chain and longer than the bus oriented network. Nevertheless, the optimal division scheme would not allow the situation to happen that occured in the linear daisy chain case. All processors in the network must stop at the same time to achieve the

minimum processing time. Again the root processor of the binary tree network has to take over a larger fraction of load than the starting processor in the bus oriented network but not as much fraction as the one in the linear daisy chain to make up the communication delay. The computation time of the root processor would fall in between the comutation time of the starting processors in the bus oriented network and in the linear daisy chain. Thus the bus oriented network is more efficient than the binary tree network which is more efficient than the linear daisy chain.

Comparing the binary tree and the right tree network, there are no difference between them when $n < 6$. This is revealed by their identical architectures. Consider the case that $n > 5$, there would be only five processors at level 3(level 1 is the root) in the left tree network but more than five(up to seven) processors in the binary tree network starting computation(the rest are waiting). By level m, there are $2m - 1$ processors in the left tree network and $2^m - 1$ processors in the binary tree network which could have already begun computation. Since $2^m - 1 > 2m - 1$ for $m > 2$, the binary tree network is more concurrent than the left tree network and hence faster.Also note that $2^m - 1$ increases much more rapidly than $2m - 1$ as m increases. The larger the m is, the bigger the diffence of their efficiency would be. For example, from table 7 where $Z = 0.1$,$w = T_{cp} = T_{cm} = 1.0$, the minimum processing time is 0.226928 and 0.238169 for $n = 6$. For $n = 20$, the minimum processing time is 0.147592 and 0.204233. The former figure is for the binary tree network and the latter one is for the left tree network.

Now let us focus on the left tree and right tree networks themselves. Doubtless, the second processor in the left tree network has to suffer a longer communication delay than the one in the right tree network because it waits for the completion of receiving the measurement data from the root processor, not only for itself, but also for its offspring. This mean that the third processor in the right tree network can start receiving the data from the root processor for itself and its offspring earlier than the third processor in the left tree network so that it can complete receiving the process at the same time as the third processor in the left tree network does. This is illustrated in Fig.12 where $n = 4$. The individual fraction of load for each processor in the left tree and right tree networks with $w = Z = T_{cp} = T_{cm} = 1.0$ are as follow: $\alpha_{l1} = 5/9$, $\alpha_{l2} = 2/9$, $\alpha_{l3} = 1/9$, $\alpha_{l4} = 1/9$, $\alpha_{r1} = 5/9$, $\alpha_{r2} = 5/18$, $\alpha_{r3} = 1/9$,

$\alpha_{r4} = 1/18$. The basic idea is that the third processors in these two networks finish receiving data from the root processor(start computation) simultaneously and have equal fractions of computation load for any n. They thus terminate computation at the same moment.

Furthermore, we may approach this problem by considering the bus oriented network as a type of tree network. The starting processor corresponds to the root processor which distributes the measurement data to its n-1 children processors optimally. These children processors of the root processor have no children. Similarly, the linear daisy chain networks with different originations can be treated as various types of tree networks. The originating processor is always the root processor of the tree. For example, the one that originates load at the network boundary is an unary tree network, where the root processor passes down the data to its single child processor which continues to pass down the data to its single child. The distribution process proceeds until the nth generation is reached(there are n processors in the network). The case where load is originates from the network interior is the type of tree network where each parent processor has only one child except that the root processor has two children. Inspecting all the tree architectures we had discussed in this paper, we found that basically the type of tree network with expansion in breadth is more efficient than the type of tree network with expansion in depth. This is because the former one achieves a higher degree of paralellism. Therefore, for these types of tree networks, we can determine intuitively their relative performance by simply comparing their architectures.

Finally, in Fig.43 the minimum total processing time is plotted against the position of the processors in a linear daisy chain network of 21 processors with $w = 1$, $T_{cp} = 1$, $T_{cm} = 0.5$, and five performance curves are obtained with $Z = 0.1, 0.2, 1, 5, and 10$, respectively. As shown in this figure, the total processing time is minimized when the starting processor is at the center of the linear network. This is because the entire network breaks into two equally spaced linear daisy chain when originating at the center.

6 Conclusion

In this paper five architectures are examined in the context of a particular load sharing problem. For the five architectures the optimal processing time is achieved when all processors stop at the same time. The best processing time is obtained for the bus oriented architecture where the processing load are transmitted over the channel only once(assuming an error free channel). The worst case is when the load are repeatedly transmitted from one processor to another such as in the linear daisy chain. Also, as observed from the tables, the longer the transmission delay is, the longer the total processing time is.

References

[1] Chair, Z. and Varshney, P.K., "Optimum Data Fusion in Multiple Sensor Detection Systems", IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-22, Jan. 1986, pp. 98-101.

[2] Cheng, Y.C. and Robertazzi, T.G., "Distributed Computation with Communication Delay", IEEE Transactions on Aerospace and Electronic Systems, vol. AES-24, No. 6, Nov. 1988, pp. 700-712.

[3] Cheng, Y.C. and Robertazzi, T.G., "Distributed Computation for a Tree Network with Communication Delay", IEEE Transactions on Aerospace and Electronic Systems, vol. AES-26, No. 3, July 1990, pp. 511-516.

[4] Chong, C.Y., Tse, E. and Mori, S., "Distributed Estimation in Networks", American Control Conference, San Francisco, CA, 1983.

[5] Coffman, E.G. and Denning, P., " Operating System Theory " , Prentice-H all, Inc., Englewood Cliffs, N.J. 1973.

[6] Hwang, K. and Briggs, F., " Computer Architecture and Parallel Processing rocessing " , McGraw-Hill, Inc., 1984.

[7] Reibman, A.R. and Nolte, L.W., "Optimal Detection and Performance of Distributed Sensor Systems", IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-23, Jan. 1987, pp. 24-30.

[8] Reibman, A.R. and Nolte, L.W., "Design and Performance Comparison of Distributed Detection Networks", IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-23, Nov. 1987, pp. 789-797.

[9] Sadjadi, F., "Hypothesis Testing in a Distributed Environment", IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-22, March 1986, pp. 134-137.

[10] Srinivasan, R., "Distributed Radar Detection Theory", IEE Proceedings, Vol. 133, Pt. F, No.1, Feb. 1986, pp. 55-60.

[11] Tenney, R.R. and Sandell, N.R., Jr., "Detection with Distributed Sensors", IEEE Transactions on Aerospace and Electronic Systems, Vol. AES-17, July 1981, pp. 501-510.

[12] Thomopoulos, S.C.A., Viswanathan, R. and Bougoulias, D.P., "Optimal Decision Fusion in Multiple Sensor Systems", IEEE Transactions on Aerospace and Electronic Systems, vol. AES-23, Sept. 1987, pp. 644-653.

[13] Tsitsiklis, J. and Athans, M., "On the Complexity of Distributed Decision Problems", IEEE Transactions on Automatic Control, Vol. AC-30, May 1985, pp. 440-446.

[14] Bataineh, S. and Robertazzi,T.G., "Distributed Computation for a Bus Network with Communication delays ", Proceedings of the 1991 Conference on Information Sciences Systems, The Johns Hopkins University, Baltimore MD, March 1991, pp. 709-714.

[15] Bataineh, S. and Robertazzi, T.G., "Bus Oriented Load Sharing for a Network of Sensor Driven Processors", IEEE Transaction on Systems, Man and Cybernetics special issue on Distributed Sensor Networks, Sept. 1991, Vol 21, no. 5.

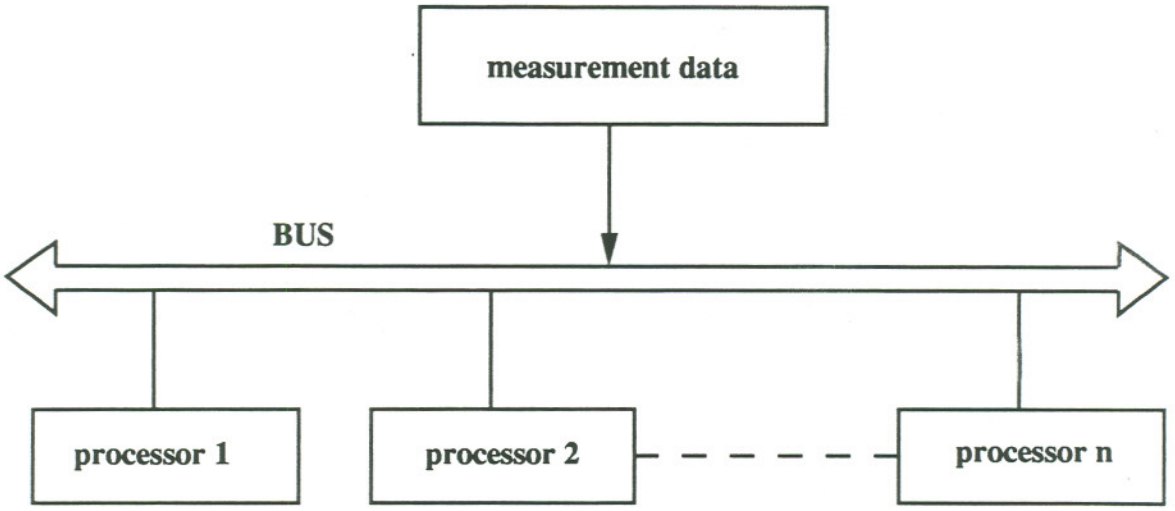


Fig. 1.

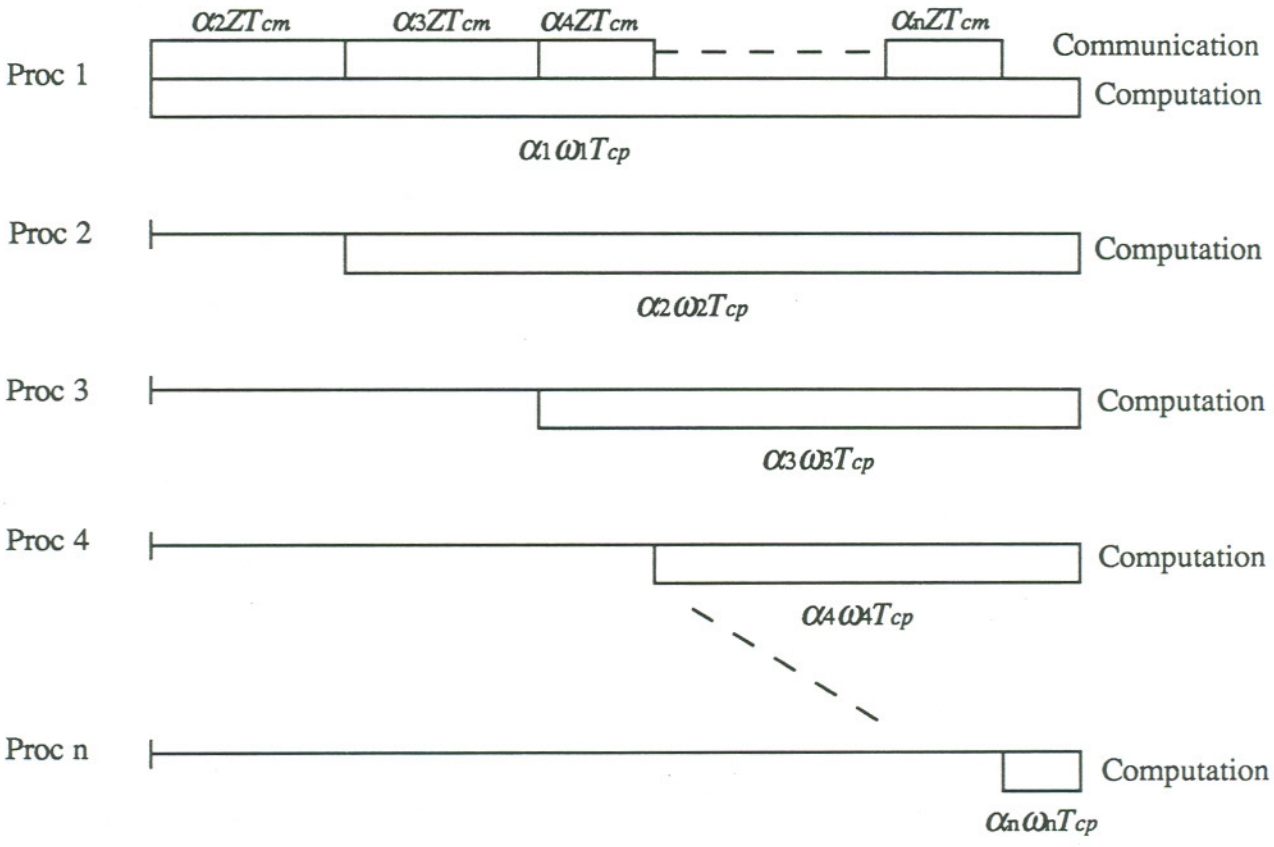


Fig. 2. Timing diagram for bus oriented network.

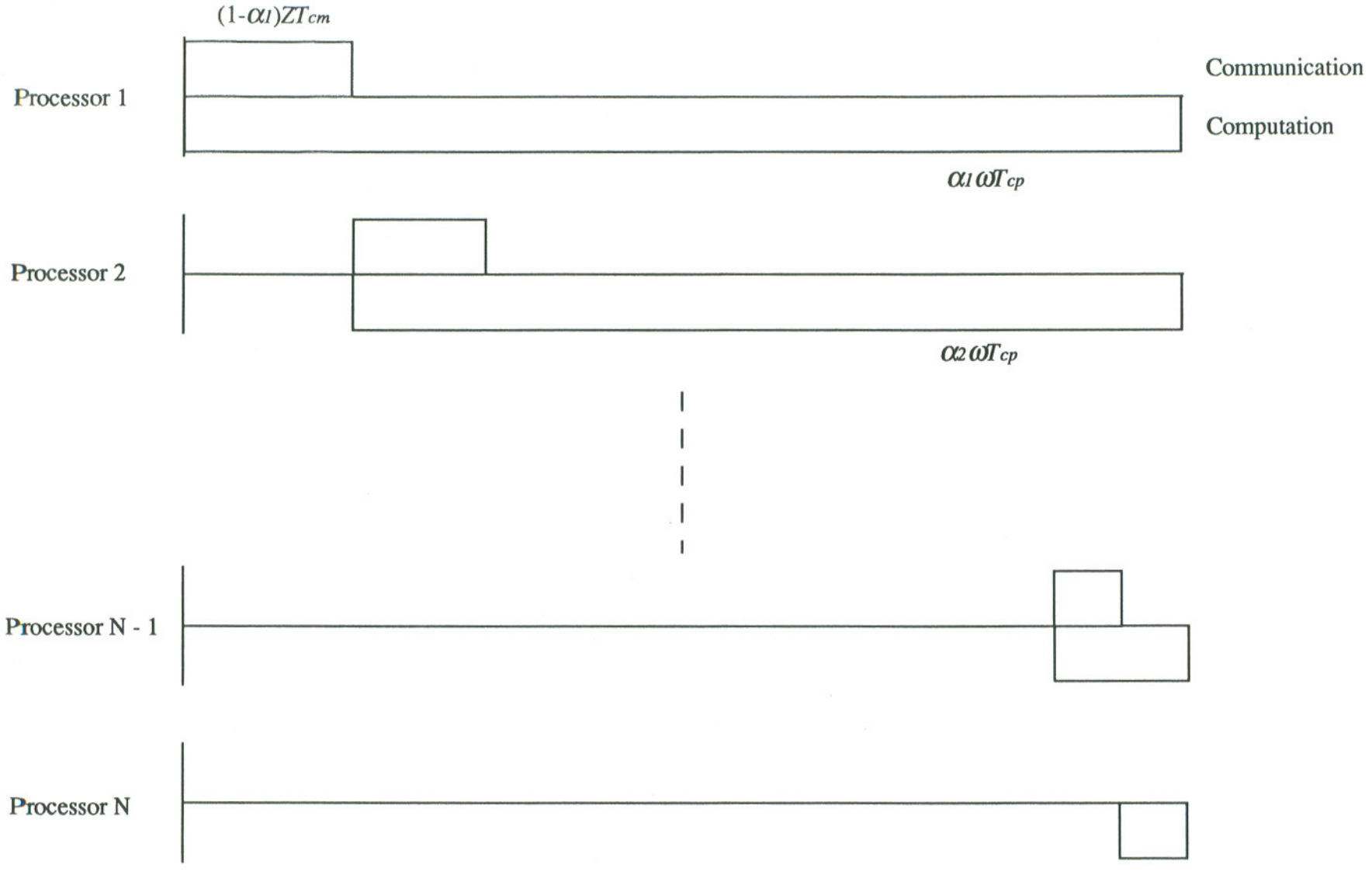


Fig. 3. Linear daisy chain network timing diagram: origination at network boundary.

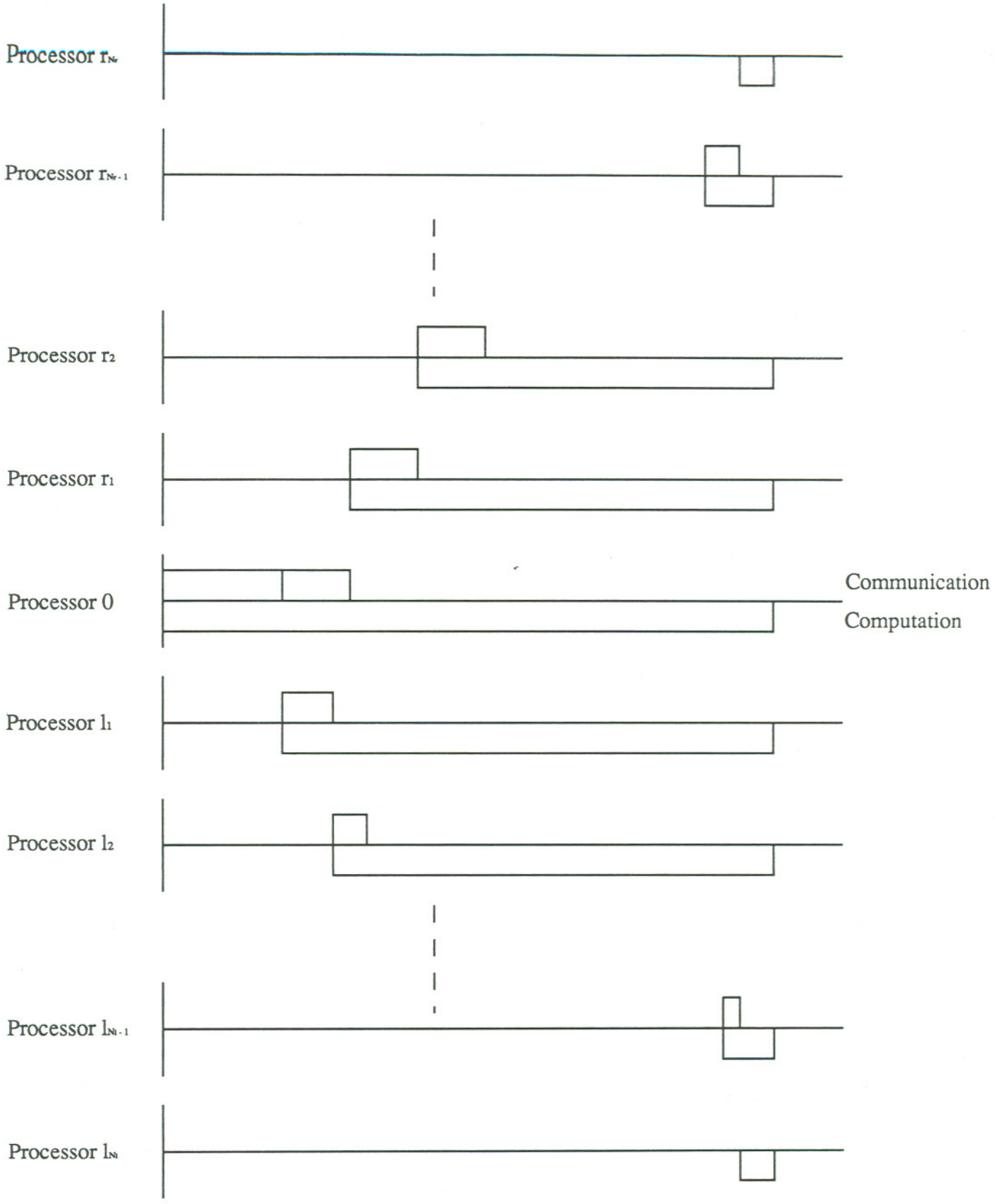


Fig. 4. Linear daisy chain timing diagram: origination from network interior.

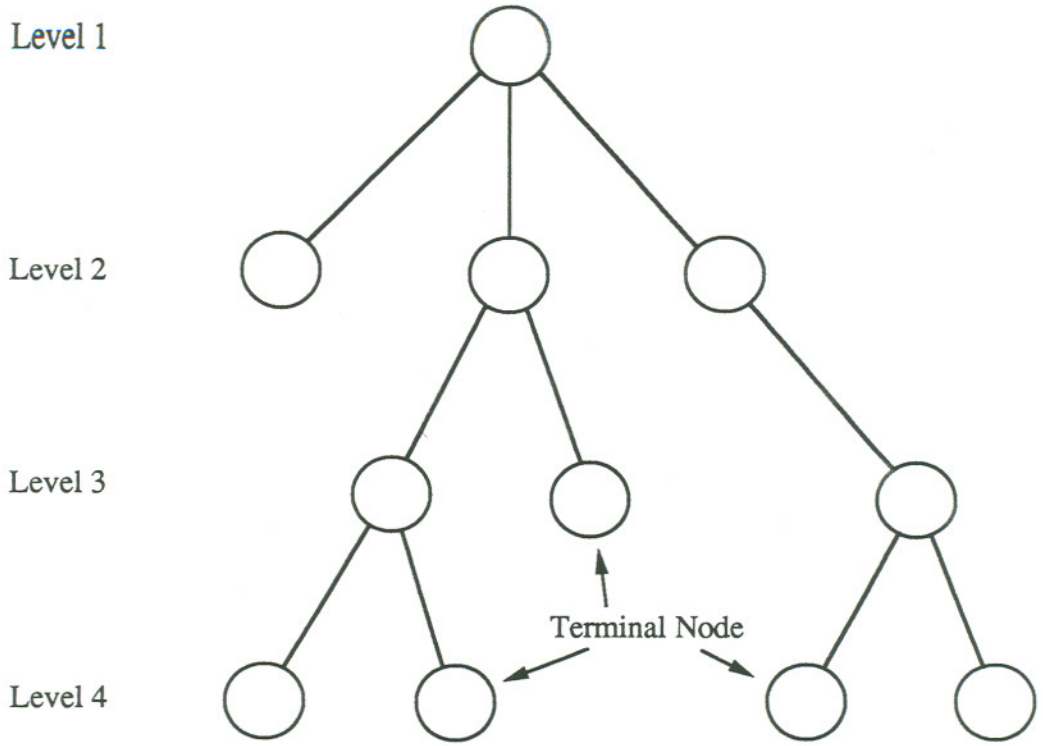


Fig. 5. Sample tree network.

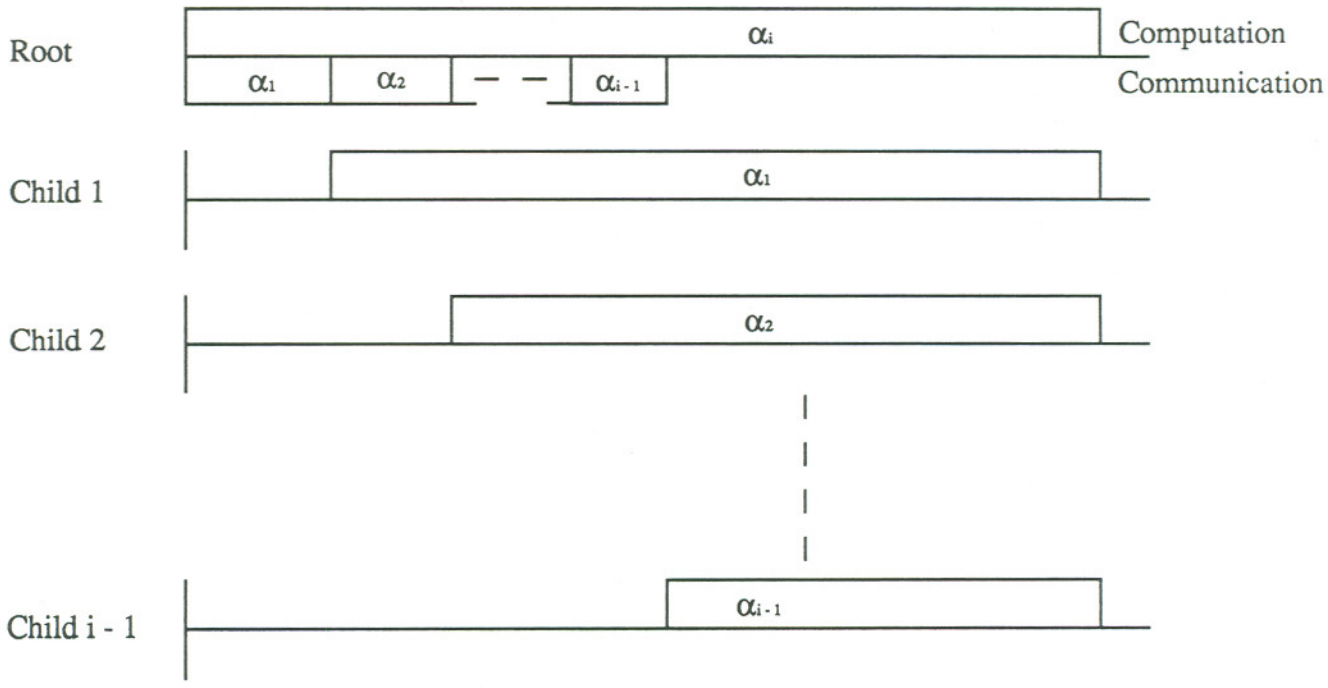


Fig. 6. Terminal node processor with front end processor.

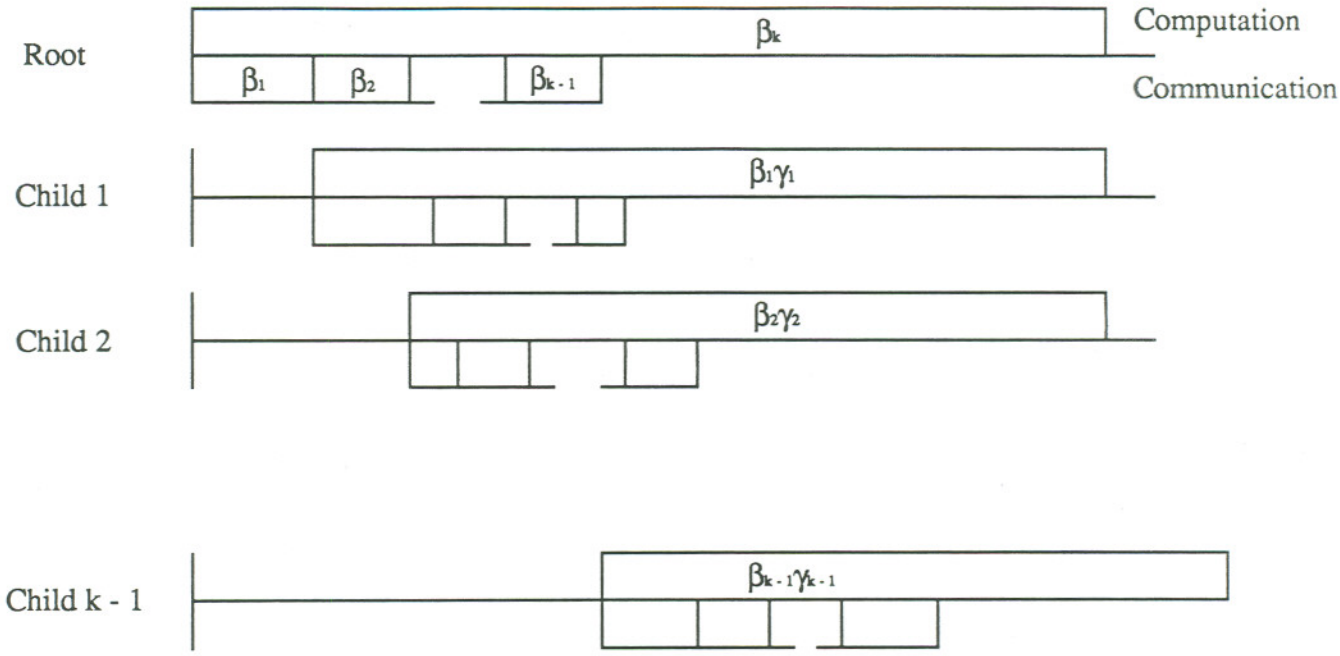


Fig. 7. Nontermional node processor with front end processor.

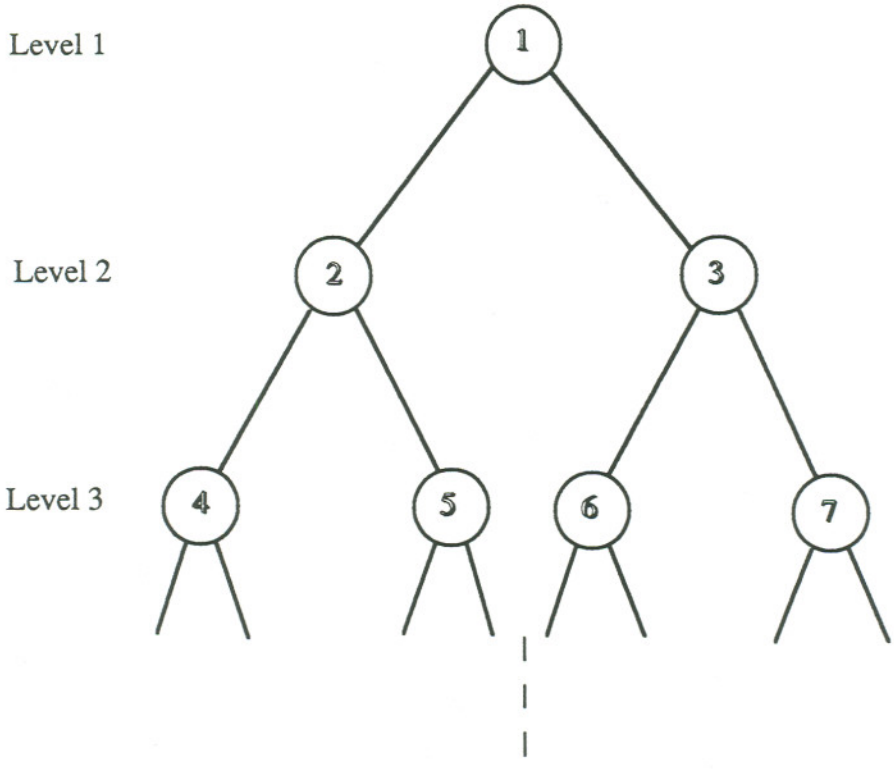
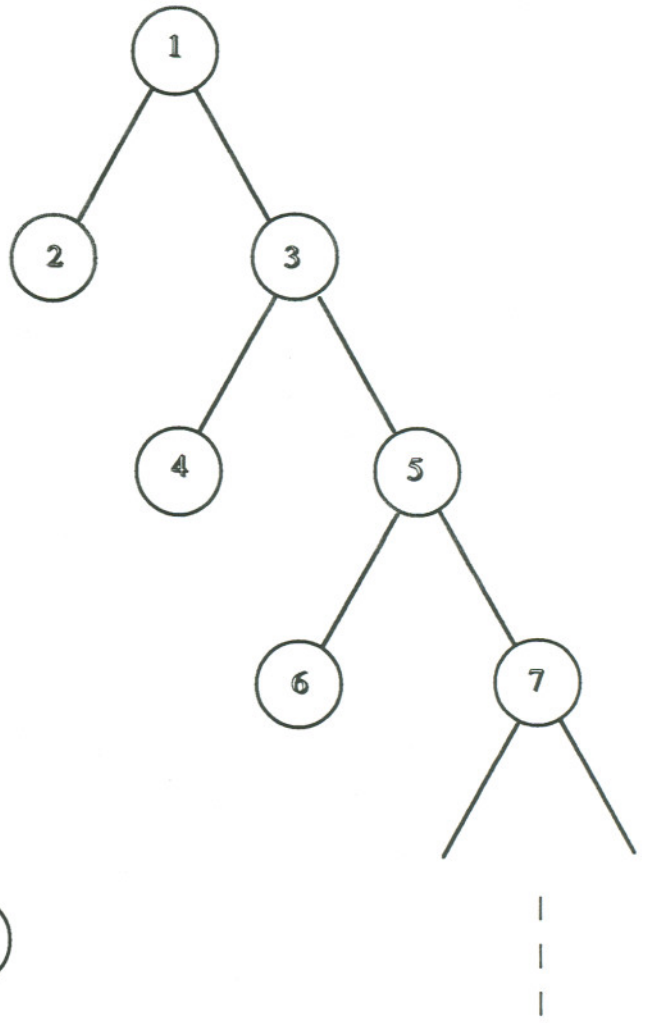


Fig. 8. Binary tree network.

Fig. 9. Left tree network.



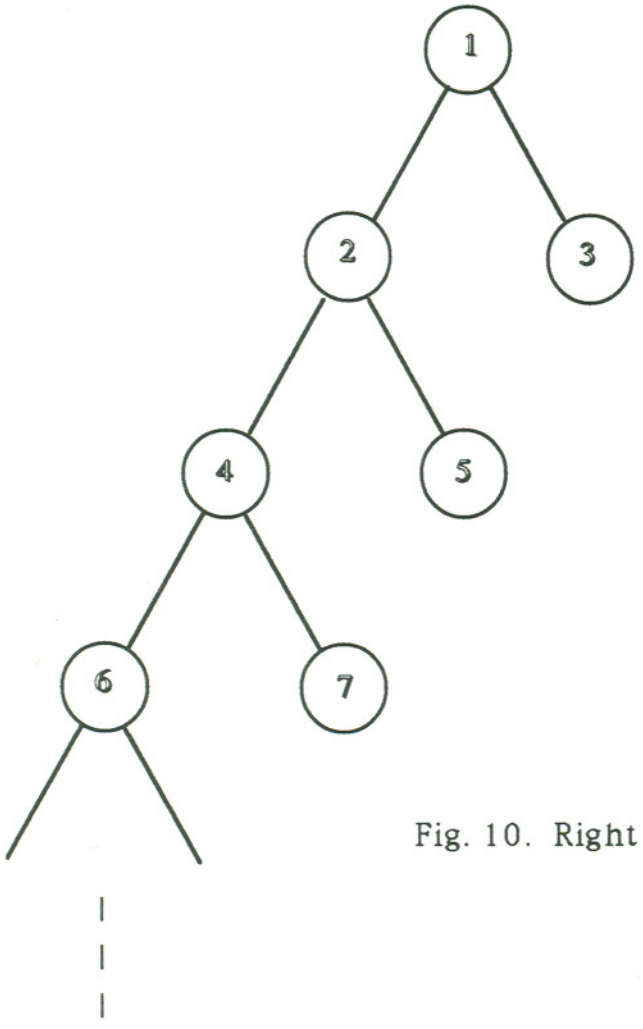


Fig. 10. Right tree network

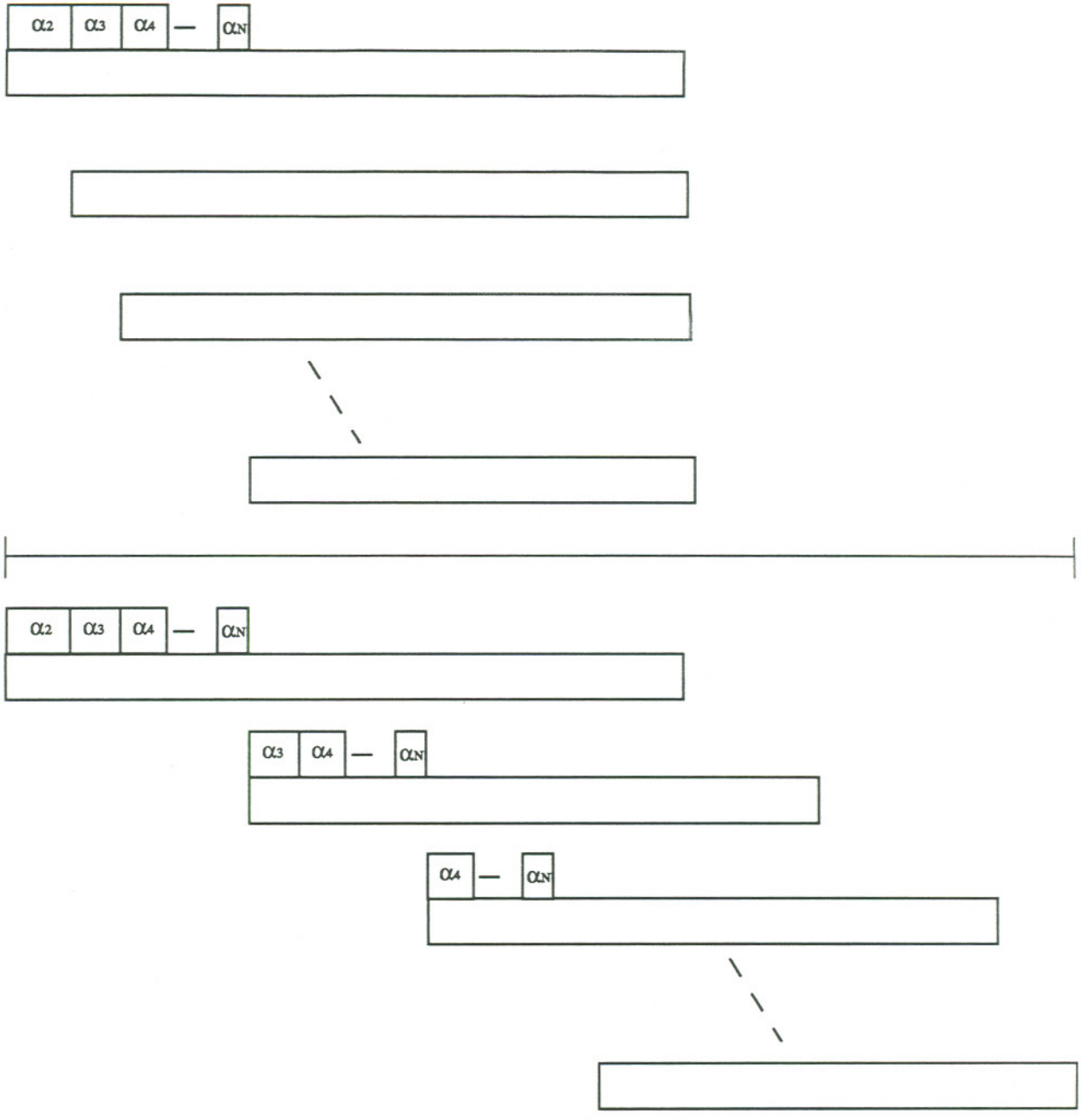


Fig. 11. Comparison between bus oriented network and linear daisy chain network.

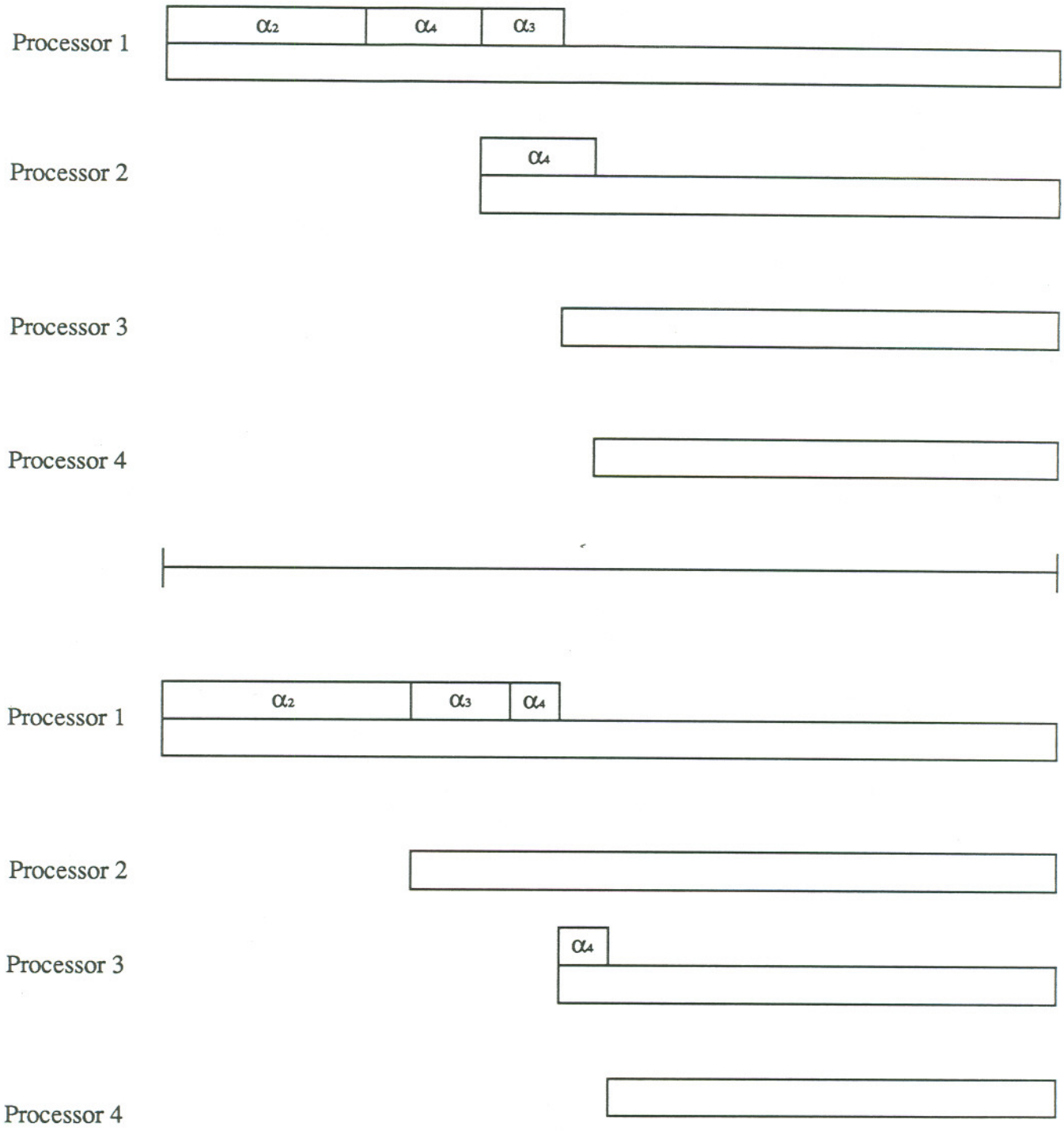


Fig. 12. Comparison between left tree network and right tree network when n = 4.

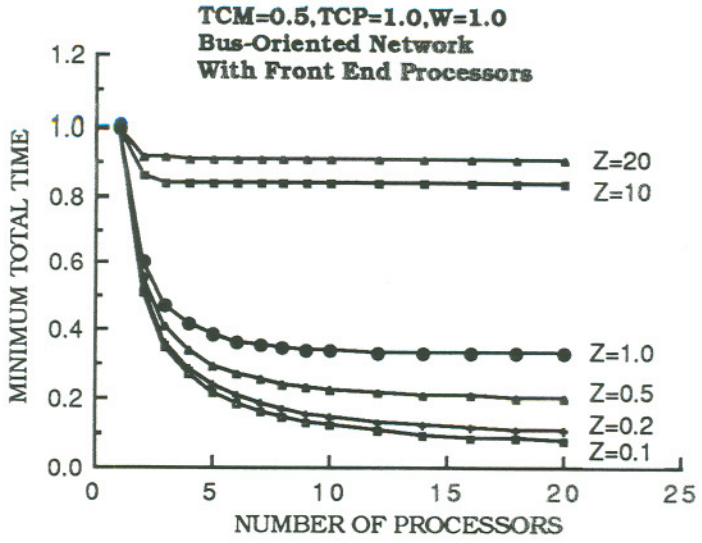


Fig.13

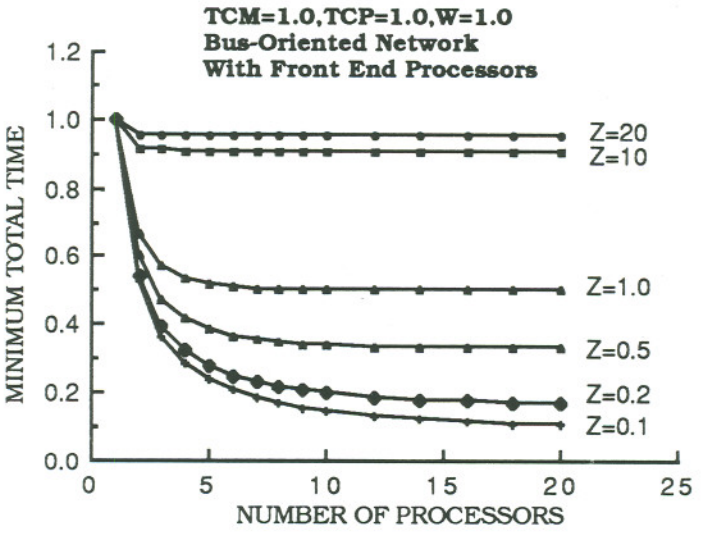


Fig.14

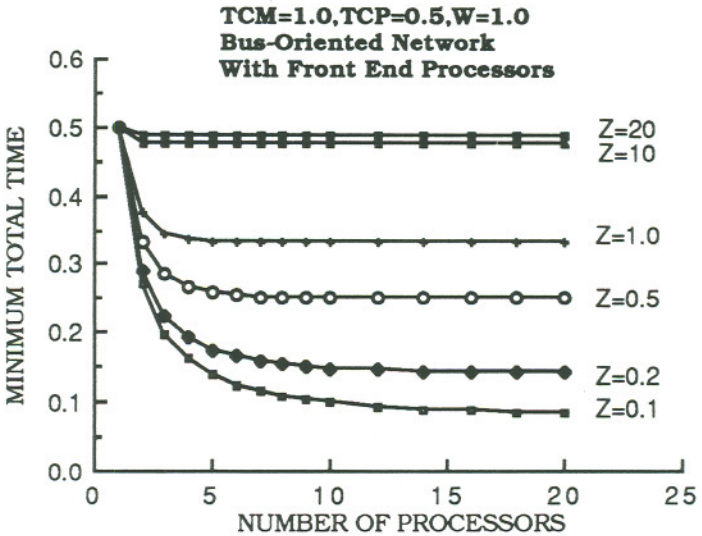


Fig.15

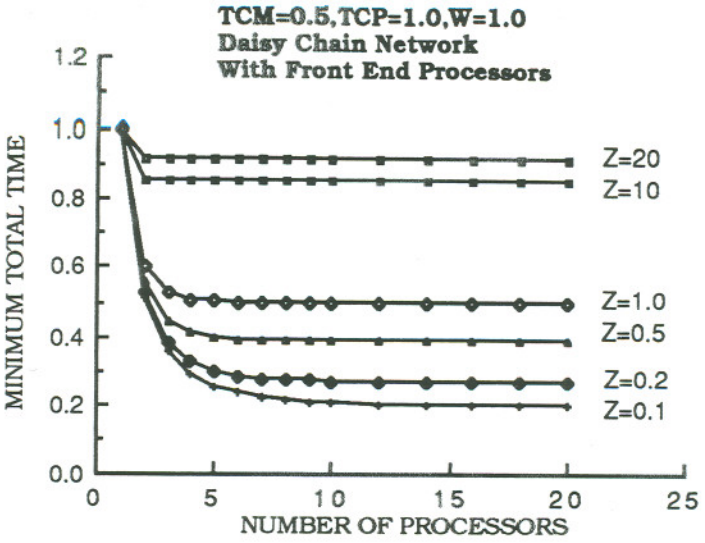


Fig.16

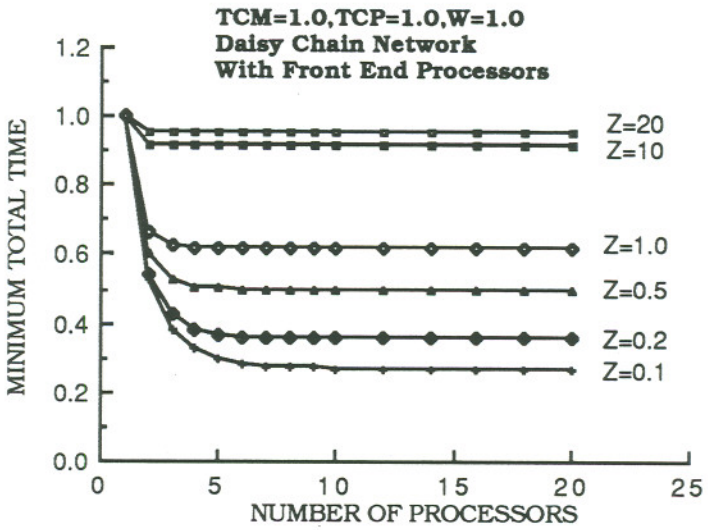


Fig.17

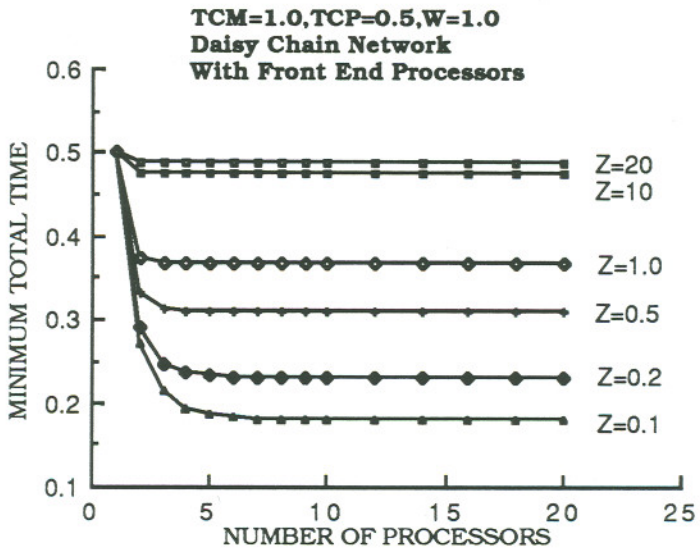


Fig.18

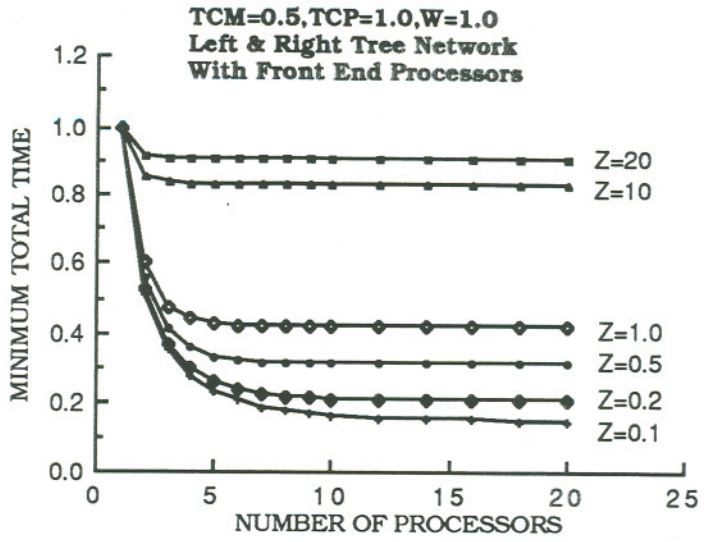


Fig.19

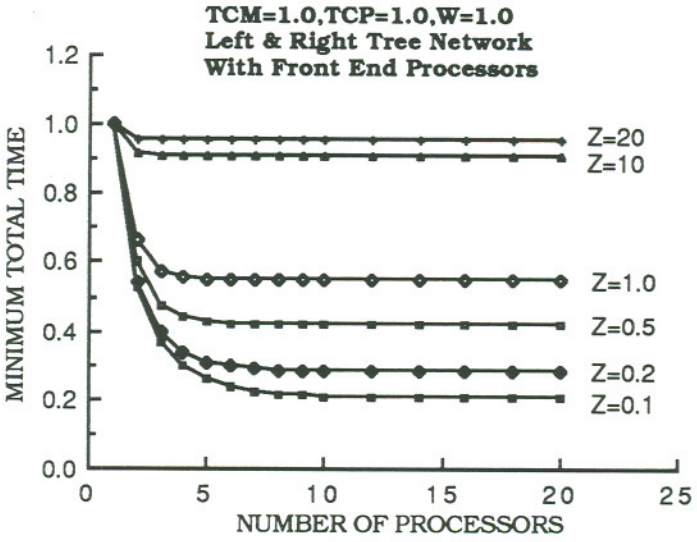


Fig.20

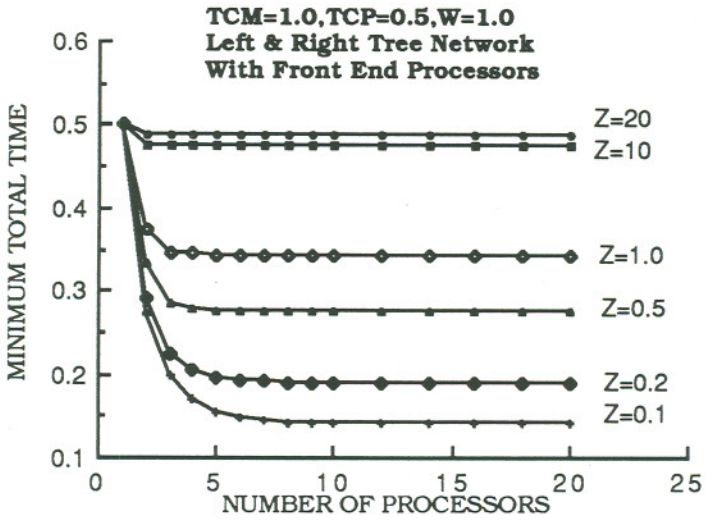


Fig.21

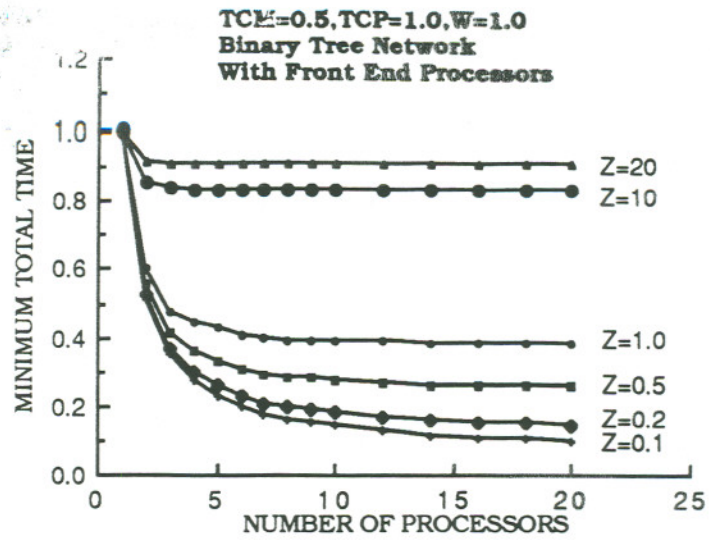


Fig.22

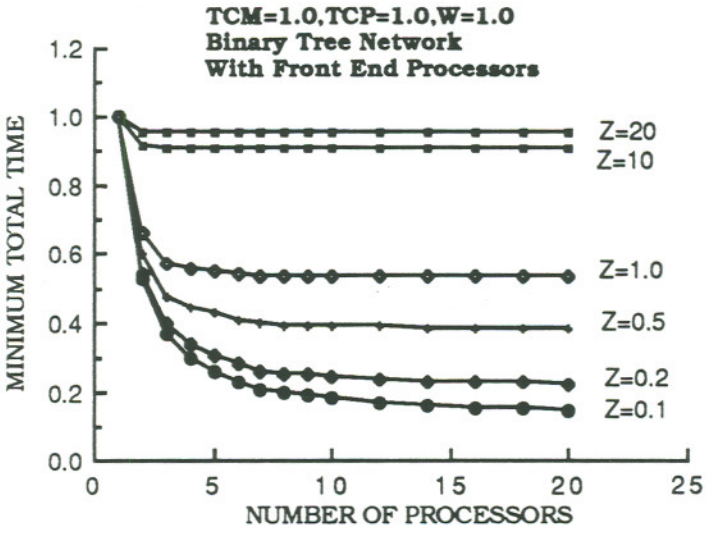


Fig.23

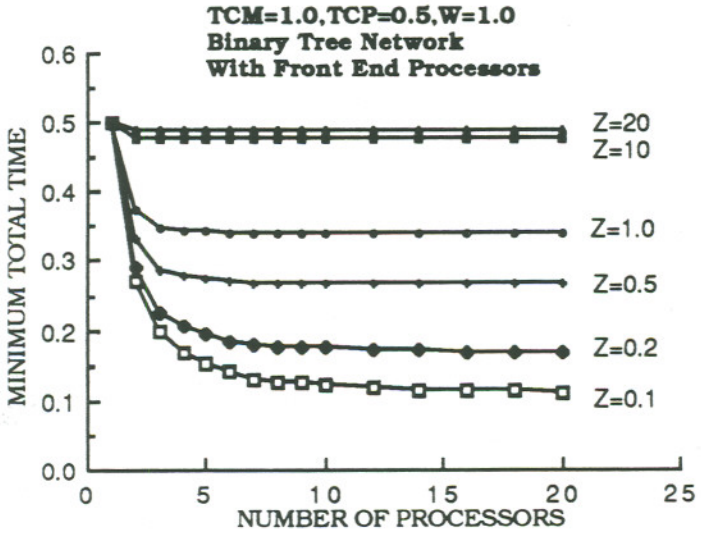


Fig.24

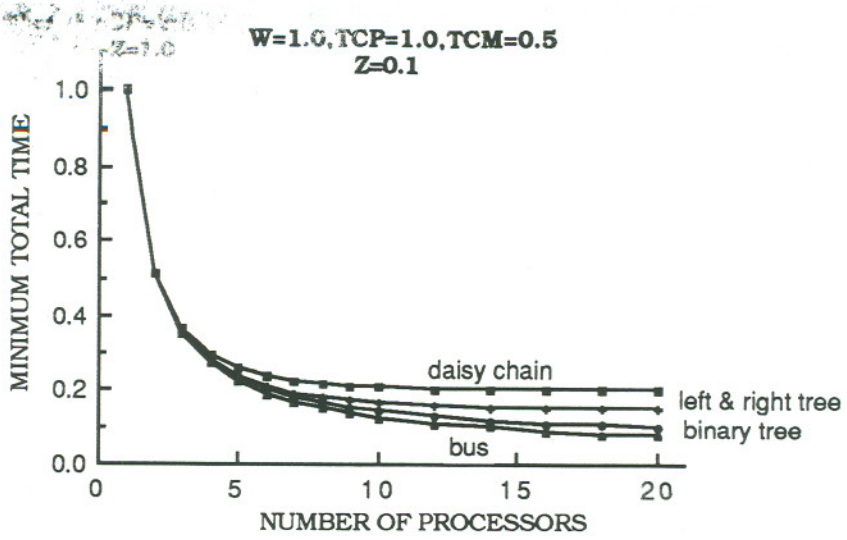


Fig.25

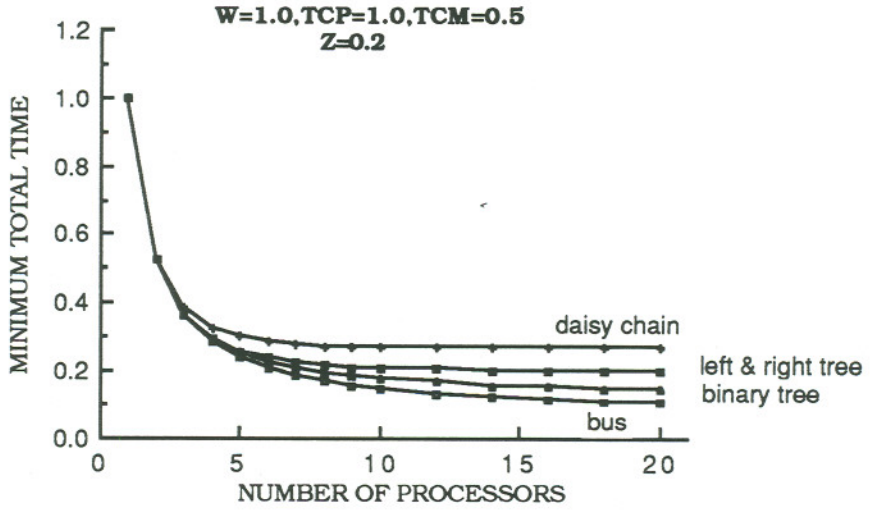


Fig.26

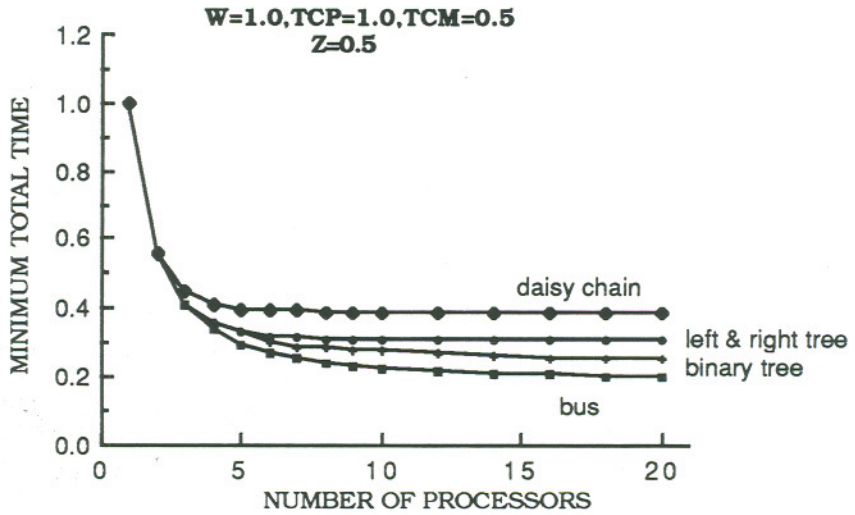


Fig.27

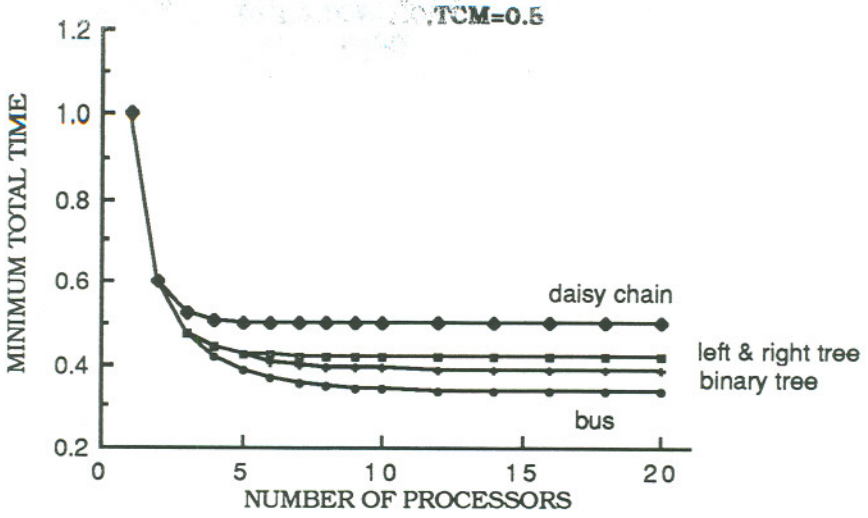


Fig.28

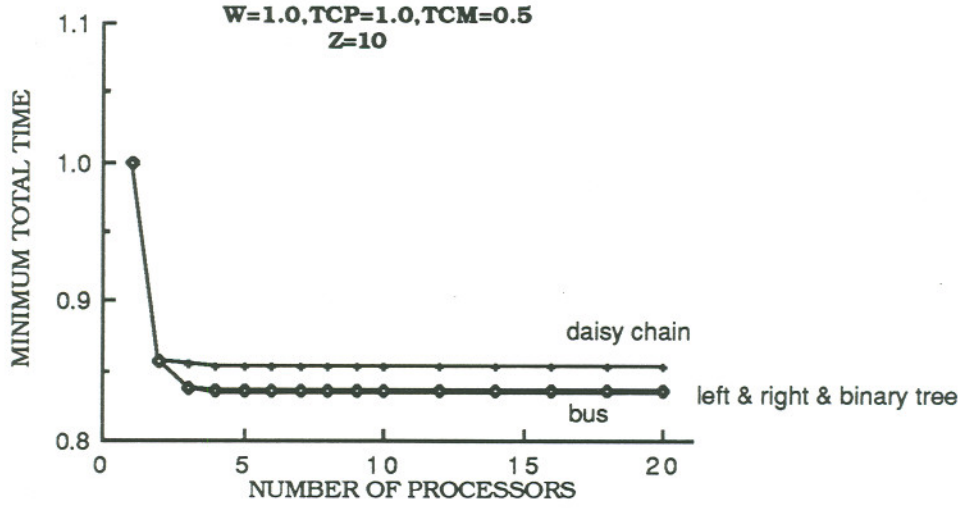


Fig.29

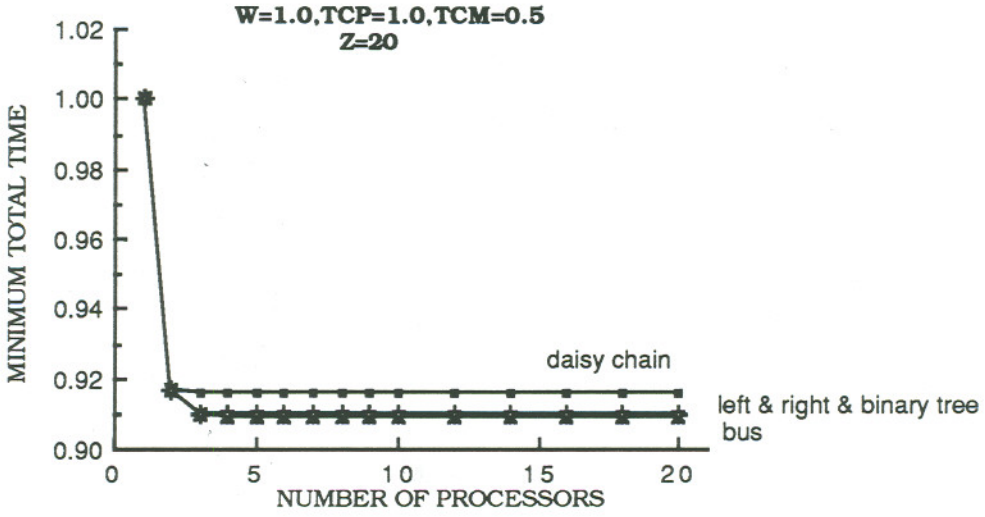


Fig.30

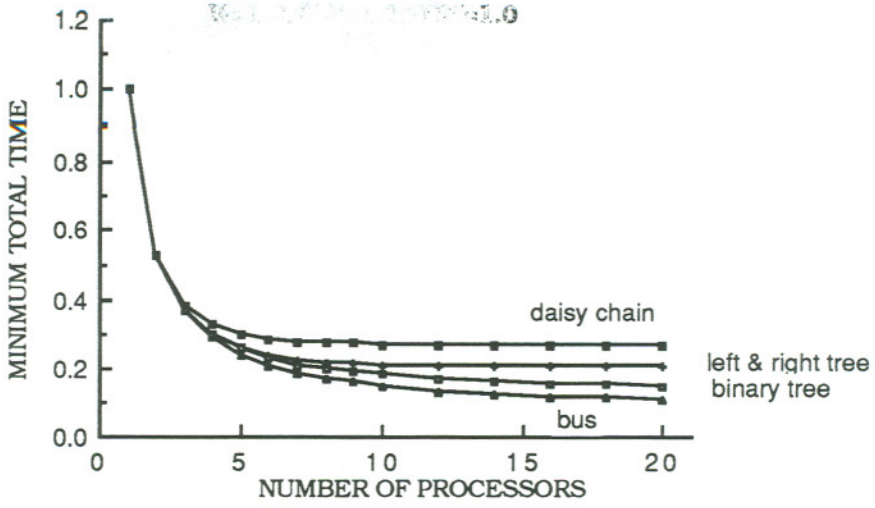


Fig.31

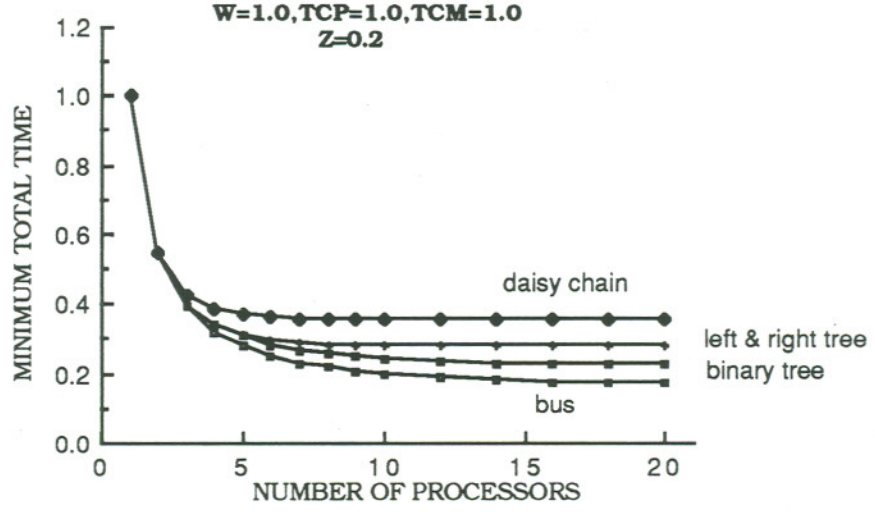


Fig.32

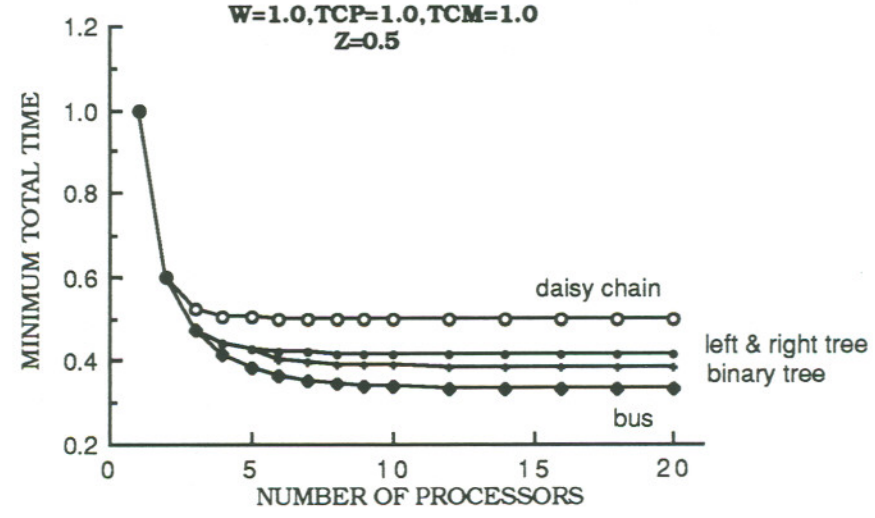


Fig.33

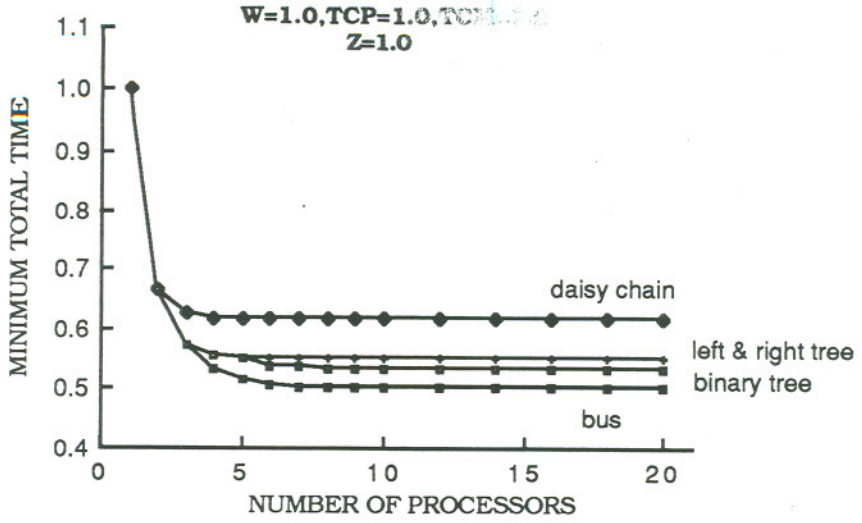


Fig.34

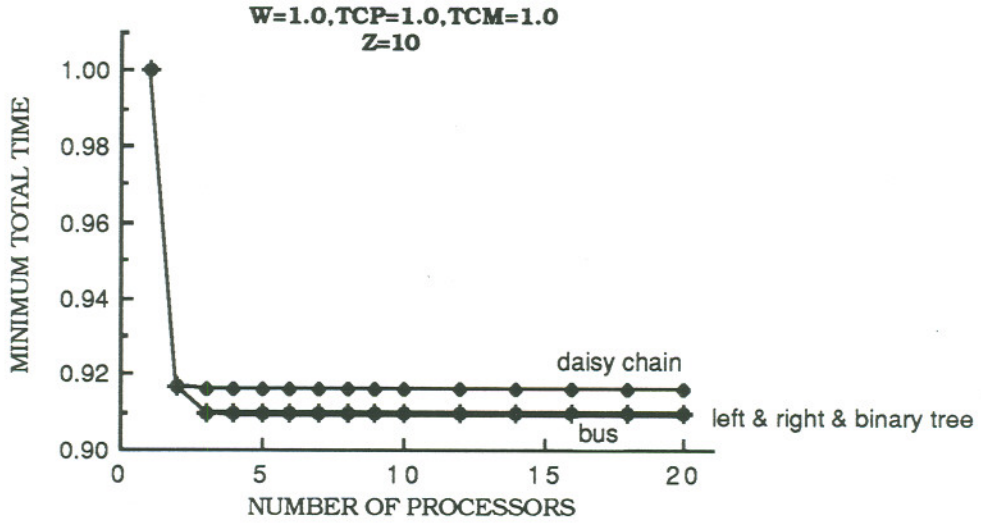


Fig.35

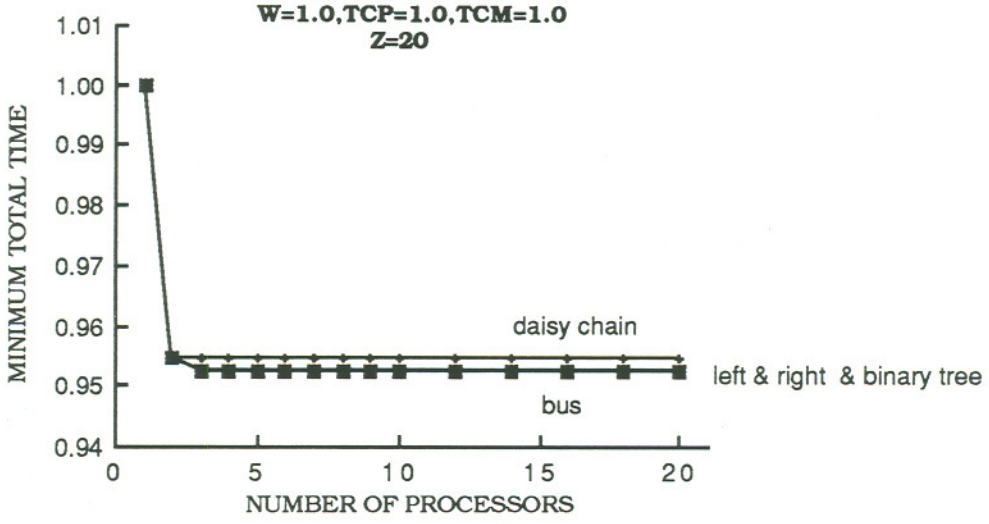


Fig.36

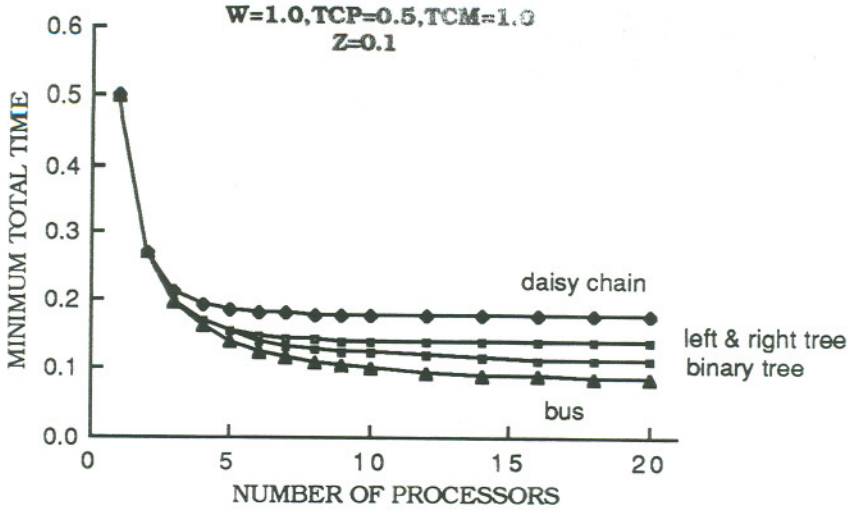


Fig.37

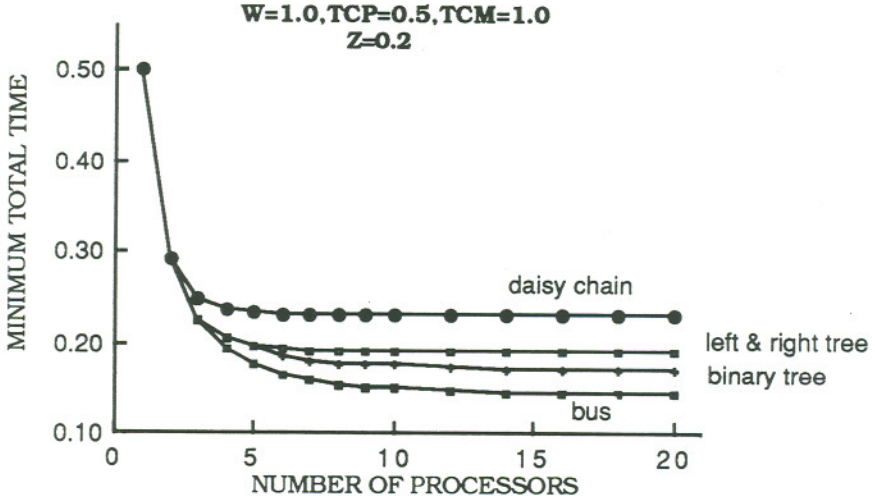


Fig.38

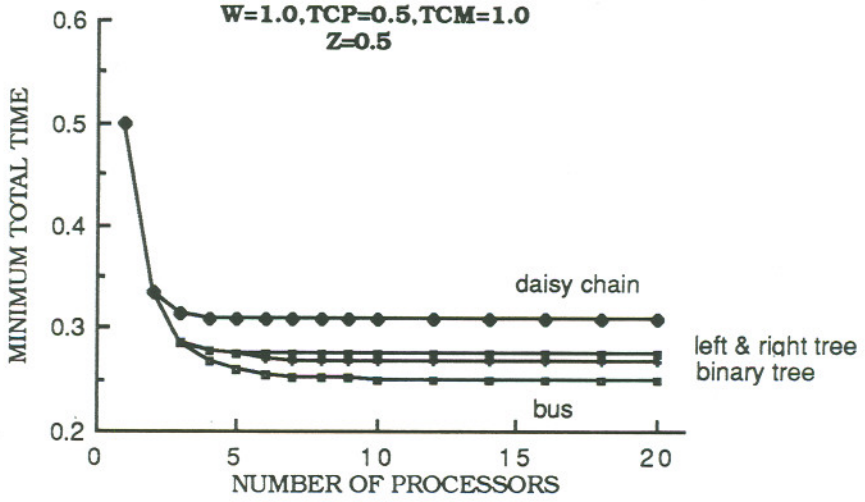


Fig.39

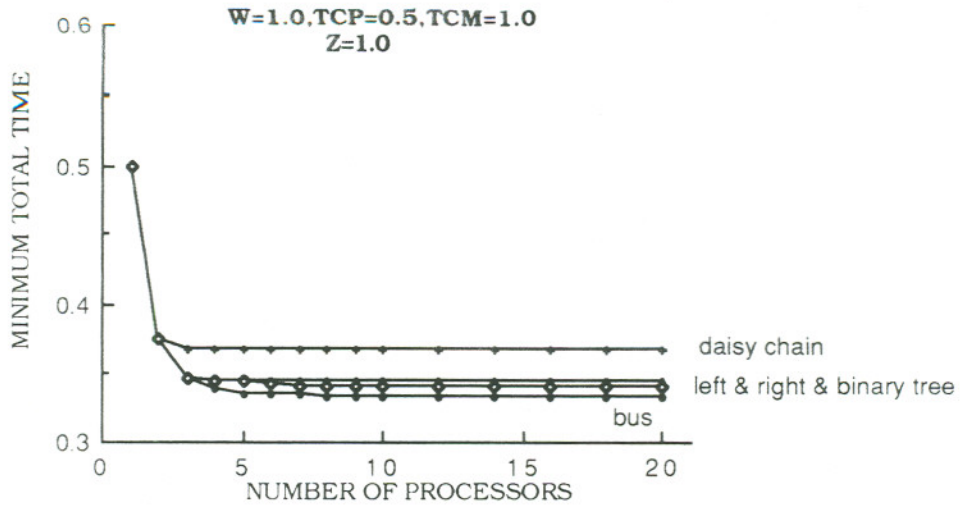


Fig.40

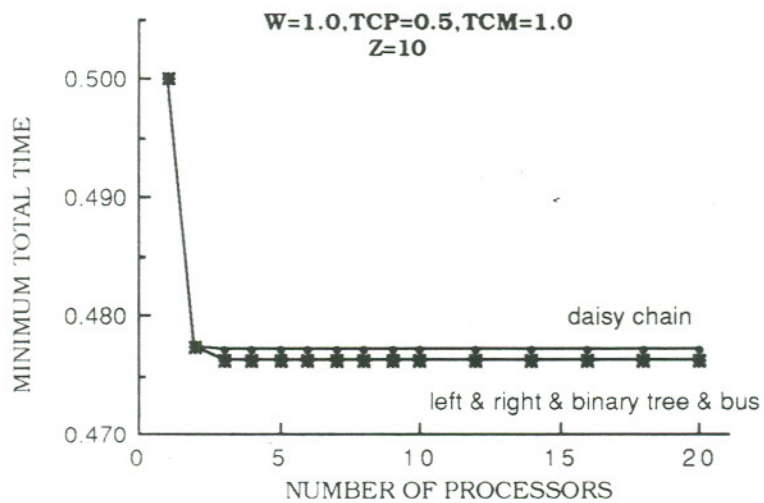


Fig.41

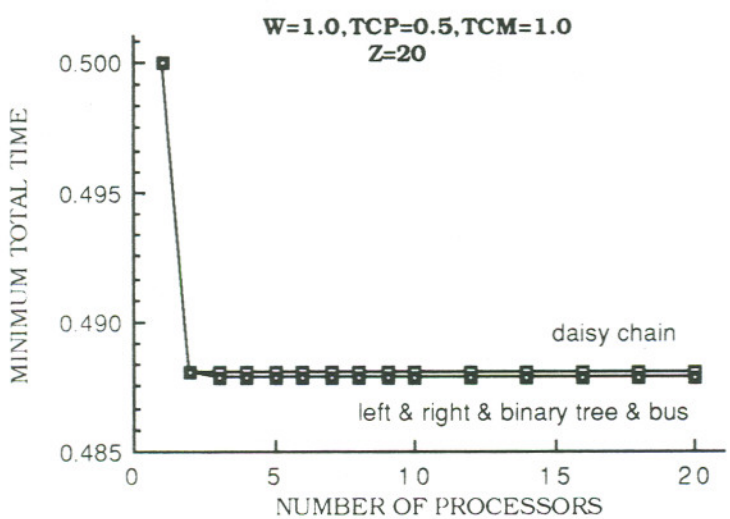


Fig.42

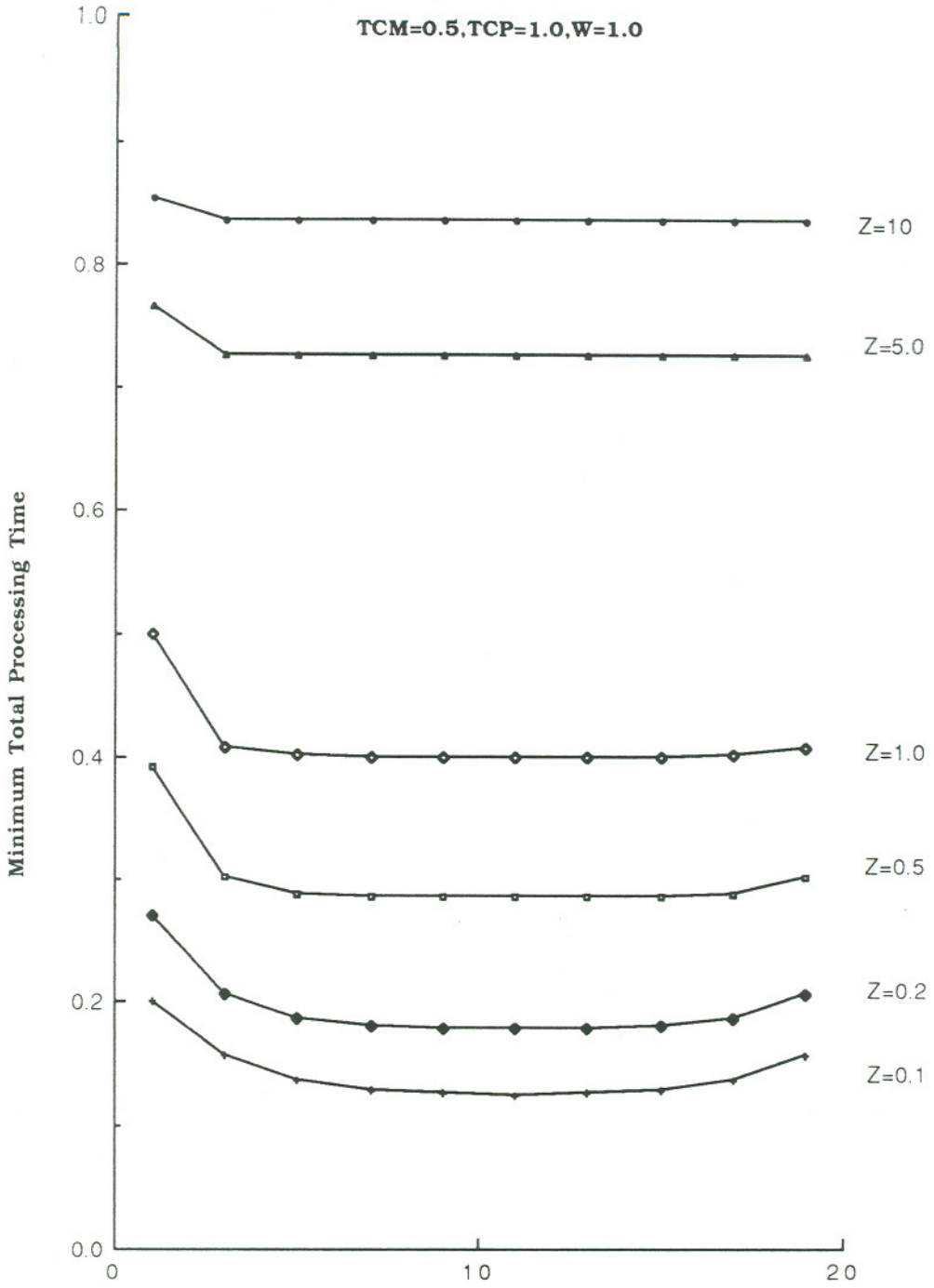

TCM=0.5,TCP=1.0,W=1.0
Minimum Total Processing Time
1.0
0.8
0.6
0.4
0.2
0.0
0
10
20
Z=10
Z=5.0
Z=1.0
Z=0.5
Z=0.2
Z=0.1
The Posotion of the Starting Processor
(linear network of 21 processors)
Origination from network interior with front end processors

Fig.43

Table 1.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.512195	0.512195	0.512195	0.512195	0.512195
3	0.349722	0.359875	0.349722	0.349722	0.349722
4	0.268583	0.290717	0.274242	0.274242	0.274242
5	0.219976	0.254131	0.230681	0.230681	0.230681
6	0.187636	0.233206	0.204514	0.204514	0.197707
7	0.164590	0.220702	0.187074	0.187074	0.175730
8	0.147354	0.213033	0.175613	0.175613	0.161288
9	0.133991	0.208255	0.167515	0.167515	0.151084
10	0.123338	0.205249	0.161978	0.161978	0.142171
11	0.114656	0.203345	0.157957	0.157957	0.135544
12	0.107453	0.202135	0.155154	0.155154	0.127697
13	0.101368	0.201364	0.153090	0.153090	0.121897
14	0.096213	0.200872	0.151638	0.151638	0.116647
15	0.091755	0.200558	0.150561	0.150561	0.112627
16	0.087876	0.200357	0.149799	0.149799	0.109714
17	0.084475	0.200228	0.149232	0.149232	0.107509
18	0.081473	0.200146	0.148830	0.148830	0.105473
19	0.078805	0.200093	0.148530	0.148530	0.103887
20	0.076421	0.200060	0.148317	0.148317	0.101925

w = 1.0; Z = 0.1; Tcp = 1.0; Tcm = 0.5

Table.2.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.523810	0.523810	0.523810	0.523810	0.523810
3	0.365559	0.384164	0.365559	0.365559	0.365559
4	0.286792	0.326220	0.297052	0.297052	0.297052
5	0.239816	0.298846	0.258948	0.258948	0.258948
6	0.208734	0.285125	0.238169	0.238169	0.226928
7	0.186732	0.278044	0.225137	0.225137	0.206845
8	0.170404	0.274334	0.217516	0.217516	0.195101
9	0.157855	0.272375	0.212535	0.212535	0.187425
10	0.147950	0.271336	0.209546	0.209546	0.180189
11	0.139967	0.270784	0.207561	0.207561	0.175226
12	0.133421	0.270490	0.206358	0.206358	0.167877
13	0.127980	0.270334	0.205554	0.205554	0.162947
14	0.123406	0.270251	0.205065	0.205065	0.158205
15	0.119522	0.270207	0.204737	0.204737	0.154898
16	0.116197	0.270183	0.204537	0.204537	0.152828
17	0.113331	0.270171	0.204403	0.204403	0.151417
18	0.110846	0.270164	0.204322	0.204322	0.150041
19	0.108679	0.270160	0.204267	0.204267	0.149070
20	0.106781	0.270158	0.204233	0.204233	0.147592

w = 1.0; Z = 0.2; Tcp = 1.0; Tcm = 0.5

Table 3.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.555556	0.555556	0.555556	0.555556	0.555556
3	0.409836	0.446154	0.409836	0.409836	0.409836
4	0.338753	0.410431	0.358025	0.358025	0.358025
5	0.297447	0.397747	0.331959	0.331959	0.331959
6	0.271056	0.393111	0.320951	0.320951	0.304268
7	0.253073	0.391398	0.314994	0.314994	0.289281
8	0.240319	0.390763	0.312386	0.312386	0.282721
9	0.231005	0.390528	0.310951	0.310951	0.279110
10	0.224058	0.390440	0.310317	0.310317	0.275019
11	0.218794	0.390407	0.309967	0.309967	0.272682
12	0.214758	0.390395	0.309812	0.309812	0.267626
13	0.211635	0.390391	0.309726	0.309726	0.264826
14	0.209201	0.390389	0.309689	0.309689	0.261635
15	0.207293	0.390389	0.309668	0.309668	0.259807
16	0.205793	0.390388	0.309658	0.309658	0.258982
17	0.204607	0.390388	0.309653	0.309653	0.258522
18	0.203669	0.390388	0.309651	0.309651	0.257995
19	0.202924	0.390388	0.309650	0.309650	0.257692
20	0.202333	0.390388	0.309649	0.309649	0.257028

w = 1.0; Z = 0.5; Tcp = 1.0; Tcm = 0.5

Table 4.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.600000	0.600000	0.600000	0.600000	0.600000
3	0.473684	0.523810	0.473684	0.473684	0.473684
4	0.415385	0.505882	0.440000	0.440000	0.440000
5	0.383886	0.501466	0.425287	0.425287	0.425287
6	0.365414	0.500366	0.420896	0.420896	0.404975
7	0.354055	0.500092	0.418907	0.418907	0.395779
8	0.346868	0.500023	0.418305	0.418305	0.392994
9	0.342236	0.500006	0.418031	0.418031	0.391728
10	0.339216	0.500001	0.417948	0.417948	0.389925
11	0.337232	0.500000	0.417910	0.417910	0.389088
12	0.335922	0.500000	0.417899	0.417899	0.386642
13	0.335055	0.500000	0.417894	0.417893	0.385528
14	0.334479	0.500000	0.417892	0.417892	0.383941
15	0.334096	0.500000	0.417891	0.417891	0.383204
16	0.333842	0.500000	0.417891	0.417891	0.382978
17	0.333672	0.500000	0.417891	0.417891	0.382875
18	0.333559	0.500000	0.417891	0.417891	0.382728
19	0.333484	0.500000	0.417891	0.417891	0.382659
20	0.333434	0.500000	0.417891	0.417891	0.382459

w = 1.0; Z = 1.0; Tcp = 1.0; Tcm = 0.5

Table 5.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.857143	0.857143	0.857143	0.857143	0.857143
3	0.837209	0.854167	0.837209	0.837209	0.837209
4	0.833977	0.854103	0.836735	0.836735	0.836735
5	0.833441	0.854102	0.836667	0.836667	0.836667
6	0.833351	0.854102	0.836665	0.836665	0.836259
7	0.833336	0.854102	0.836665	0.836665	0.836200
8	0.833334	0.854102	0.836665	0.836665	0.836199
9	0.833333	0.854102	0.836665	0.836665	0.836199
10	0.833333	0.854102	0.836665	0.836665	0.836197
11	0.833333	0.854102	0.836665	0.836665	0.836197
12	0.833333	0.854102	0.836665	0.836665	0.836196
13	0.833333	0.854102	0.836665	0.836665	0.836196
14	0.833333	0.854102	0.836665	0.836665	0.836195
15	0.833333	0.854102	0.836665	0.836665	0.836194
16	0.833333	0.854102	0.836665	0.836665	0.836194
17	0.833333	0.854102	0.836665	0.836665	0.836194
18	0.833333	0.854102	0.836665	0.836665	0.836194
19	0.833333	0.854102	0.836665	0.836665	0.836194
20	0.833333	0.854102	0.836665	0.836665	0.836194

w = 1.0; Z = 10; Tcp = 1.0; Tcm = 0.5

Table 6.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.916667	0.916667	0.916667	0.916667	0.916667
3	0.909774	0.916084	0.909774	0.909774	0.909774
4	0.909153	0.916080	0.909722	0.909722	0.909722
5	0.909097	0.916080	0.909718	0.909718	0.909718
6	0.909091	0.916080	0.909718	0.909718	0.909670
7	0.909091	0.916080	0.909718	0.909718	0.909666
8	0.909091	0.916080	0.909718	0.909718	0.909666
9	0.909091	0.916080	0.909718	0.909718	0.909666
10	0.909091	0.916080	0.909718	0.909718	0.909666
11	0.909091	0.916080	0.909718	0.909718	0.909666
12	0.909091	0.916080	0.909718	0.909718	0.909666
13	0.909091	0.916080	0.909718	0.909718	0.909666
14	0.909091	0.916080	0.909718	0.909718	0.909666
15	0.909091	0.916080	0.909718	0.909718	0.909666
16	0.909091	0.916080	0.909718	0.909718	0.909666
17	0.909091	0.916080	0.909718	0.909718	0.909666
18	0.909091	0.916080	0.909718	0.909718	0.909666
19	0.909091	0.916080	0.909718	0.909718	0.909666
20	0.909091	0.916080	0.909718	0.909718	0.909666

w = 1.0; Z = 20.0; Tcp = 1.0; Tcm = 0.5

Table 7.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.523810	0.523810	0.523810	0.523810	0.523810
3	0.365559	0.384164	0.365559	0.365559	0.365559
4	0.286792	0.326220	0.297052	0.297052	0.297052
5	0.239816	0.298846	0.258948	0.258948	0.258948
6	0.208734	0.285125	0.238169	0.238169	0.226928
7	0.186732	0.278044	0.225137	0.225137	0.206845
8	0.170404	0.274334	0.217516	0.217516	0.195101
9	0.157855	0.272375	0.212535	0.212535	0.187425
10	0.147950	0.271336	0.209546	0.209546	0.180189
11	0.139967	0.270784	0.207561	0.207561	0.175226
12	0.133421	0.270490	0.206358	0.206358	0.167877
13	0.127980	0.270334	0.205554	0.205554	0.162947
14	0.123406	0.270251	0.205065	0.205065	0.158205
15	0.119522	0.270207	0.204737	0.204737	0.154898
16	0.116197	0.270183	0.204537	0.204537	0.152828
17	0.113331	0.270171	0.204403	0.204403	0.151417
18	0.110846	0.270164	0.204322	0.204322	0.150041
19	0.108679	0.270160	0.204267	0.204267	0.149070
20	0.106781	0.270158	0.204233	0.204233	0.147592

w = 1.0; Z = 0.1; Tcp = 1.0; Tcm = 1.0

Table 8.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.545455	0.545455	0.545455	0.545455	0.545455
3	0.395604	0.427083	0.395604	0.395604	0.395604
4	0.321908	0.385403	0.338843	0.338843	0.338843
5	0.278650	0.369246	0.309361	0.309361	0.309361
6	0.250588	0.362751	0.295871	0.295871	0.280114
7	0.231187	0.360103	0.288237	0.288237	0.263564
8	0.217175	0.359017	0.284586	0.284586	0.255649
9	0.206733	0.358570	0.282473	0.282473	0.251069
10	0.198769	0.358386	0.281451	0.281451	0.246160
11	0.192587	0.358311	0.280856	0.280856	0.243203
12	0.187721	0.358279	0.280567	0.280567	0.237397
13	0.183850	0.358267	0.280398	0.280398	0.234004
14	0.180744	0.358261	0.280317	0.280317	0.230339
15	0.178235	0.358258	0.280269	0.280269	0.228118
16	0.176197	0.358258	0.280245	0.280245	0.227015
17	0.174533	0.358258	0.280232	0.280232	0.226364
18	0.173171	0.358258	0.280225	0.280225	0.225655
19	0.172052	0.358258	0.280221	0.280221	0.225223
20	0.171130	0.358258	0.280220	0.280220	0.224363

w = 1.0; Z = 0.2; Tcp = 1.0; Tcm = 1.0

Table 9.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.600000	0.600000	0.600000	0.600000	0.600000
3	0.473684	0.523810	0.473684	0.473684	0.473684
4	0.415385	0.505882	0.440000	0.440000	0.440000
5	0.383886	0.501466	0.425287	0.425287	0.425287
6	0.365414	0.500336	0.420896	0.420896	0.404975
7	0.354055	0.500092	0.418907	0.418907	0.395779
8	0.346868	0.500023	0.418305	0.418305	0.392994
9	0.342236	0.500006	0.418031	0.418031	0.391728
10	0.339216	0.500001	0.417948	0.417948	0.389925
11	0.337232	0.500000	0.417910	0.417910	0.389088
12	0.335992	0.500000	0.417899	0.417899	0.386642
13	0.335055	0.500000	0.417894	0.417894	0.385528
14	0.334479	0.500000	0.417892	0.417892	0.383941
15	0.334096	0.500000	0.417891	0.417891	0.383204
16	0.333842	0.500000	0.417891	0.417891	0.382978
17	0.333672	0.500000	0.417891	0.417891	0.382875
18	0.333559	0.500000	0.417891	0.417891	0.382728
19	0.333484	0.500000	0.417891	0.417891	0.382659
20	0.333434	0.500000	0.417891	0.417891	0.382459

w = 1.0; Z = 0.5; Tcp = 1.0; Tcm = 1.0

Table 10.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.666667	0.666667	0.666667	0.666667	0.666667
3	0.571429	0.625000	0.571429	0.571429	0.571429
4	0.533333	0.619048	0.555556	0.555556	0.555556
5	0.516129	0.618182	0.550000	0.550000	0.550000
6	0.507937	0.618056	0.549020	0.549020	0.539216
7	0.503937	0.618037	0.548673	0.548673	0.535398
8	0.501961	0.618034	0.548611	0.548611	0.534722
9	0.500978	0.618034	0.548589	0.548589	0.534483
10	0.500489	0.618034	0.548586	0.548585	0.534014
11	0.500244	0.618034	0.548584	0.548584	0.533846
12	0.500122	0.618034	0.548584	0.548584	0.533202
13	0.500061	0.618034	0.548584	0.548584	0.532973
14	0.500031	0.618034	0.548584	0.548584	0.532526
15	0.500015	0.618034	0.548584	0.548584	0.532366
16	0.500008	0.618034	0.548584	0.548584	0.532338
17	0.500004	0.618034	0.548584	0.548584	0.532328
18	0.500002	0.618034	0.548584	0.548584	0.532308
19	0.500001	0.618034	0.548584	0.548584	0.532301
20	0.500000	0.618034	0.548584	0.548584	0.532274

w = 1.0; Z = 1.0; Tcp = 1.0; Tcm = 1.0

Table 11.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.916667	0.916667	0.916667	0.916667	0.916667
3	0.909774	0.916084	0.909774	0.909774	0.909774
4	0.909153	0.916080	0.909722	0.909722	0.909722
5	0.909097	0.916080	0.909718	0.909718	0.909718
6	0.909091	0.916080	0.909718	0.909718	0.909670
7	0.909091	0.916080	0.909718	0.909718	0.909666
8	0.909091	0.916080	0.909718	0.909718	0.909666
9	0.909091	0.916080	0.909718	0.909718	0.909666
10	0.909091	0.916080	0.909718	0.909718	0.909666
11	0.909091	0.916080	0.909718	0.909718	0.909666
12	0.909091	0.916080	0.909718	0.909718	0.909666
13	0.909091	0.916080	0.909718	0.909718	0.909666
14	0.909091	0.916080	0.909718	0.909718	0.909666
15	0.909091	0.916080	0.909718	0.909718	0.909666
16	0.909091	0.916080	0.909718	0.909718	0.909666
17	0.909091	0.916080	0.909718	0.909718	0.909666
18	0.909091	0.916080	0.909718	0.909718	0.909666
19	0.909091	0.916080	0.909718	0.909718	0.909666
20	0.909091	0.916080	0.909718	0.909718	0.909666

w = 1.0; Z = 10.0; Tcp = 1.0; Tcm = 1.0

Table 12.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	1.000000	1.000000	1.000000	1.000000	1.000000
2	0.954545	0.954545	0.954545	0.954545	0.954545
3	0.952484	0.954451	0.952484	0.952484	0.952484
4	0.952386	0.954451	0.952479	0.952479	0.952479
5	0.952381	0.954451	0.952479	0.952479	0.952479
6	0.952381	0.954451	0.952479	0.952479	0.952475
7	0.952381	0.954451	0.952479	0.952479	0.952475
8	0.952381	0.954451	0.952479	0.952479	0.952475
9	0.952381	0.954451	0.952479	0.952479	0.952475
10	0.952381	0.954451	0.952479	0.952479	0.952475
11	0.952381	0.954451	0.952479	0.952479	0.952475
12	0.952381	0.954451	0.952479	0.952479	0.952475
13	0.952381	0.954451	0.952479	0.952479	0.952475
14	0.952381	0.954451	0.952479	0.952479	0.952475
15	0.952381	0.954451	0.952479	0.952479	0.952475
16	0.952381	0.954451	0.952479	0.952479	0.952475
17	0.952381	0.954451	0.952479	0.952479	0.952475
18	0.952381	0.954451	0.952479	0.952479	0.952475
19	0.952381	0.954451	0.952479	0.952479	0.952475
20	0.952381	0.954451	0.952479	0.952479	0.952475

w = 1.0; Z = 20.0; Tcp = 1.0; Tcm = 1.0

Table 13.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	0.500000	0.500000	0.500000	0.500000	0.500000
2	0.272727	0.272727	0.272727	0.272727	0.272727
3	0.197802	0.213542	0.197802	0.197802	0.197802
4	0.160954	0.192702	0.169421	0.169421	0.169421
5	0.139325	0.184623	0.154680	0.154680	0.154680
6	0.125294	0.181376	0.147935	0.147935	0.140057
7	0.115593	0.180051	0.144118	0.144118	0.131782
8	0.108587	0.179508	0.142293	0.142293	0.127824
9	0.103366	0.179285	0.141237	0.141237	0.125535
10	0.099384	0.179193	0.140726	0.140726	0.123080
11	0.096293	0.179155	0.140428	0.140428	0.121601
12	0.093860	0.179140	0.140283	0.140283	0.118699
13	0.091925	0.179133	0.140199	0.140199	0.117002
14	0.090372	0.179131	0.140158	0.140158	0.115170
15	0.089118	0.179130	0.140134	0.140134	0.114059
16	0.088098	0.179129	0.140123	0.140123	0.113507
17	0.087267	0.179129	0.140116	0.140116	0.113182
18	0.086586	0.179129	0.140113	0.140113	0.112828
19	0.086026	0.179129	0.140111	0.140111	0.112611
20	0.085565	0.179129	0.140110	0.140110	0.112181

w = 1.0; Z = 0.1; Tcp = 0.5; Tcm = 1.0

Table 14.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	0.500000	0.500000	0.500000	0.500000	0.500000
2	0.291667	0.291667	0.291667	0.291667	0.291667
3	0.224771	0.247899	0.224771	0.224771	0.224771
4	0.193131	0.236259	0.204861	0.204861	0.204861
5	0.175486	0.232980	0.195689	0.195689	0.195689
6	0.164736	0.232041	0.192573	0.192573	0.184146
7	0.157830	0.231772	0.191068	0.191068	0.178573
8	0.153241	0.231694	0.190546	0.190546	0.176640
9	0.150123	0.231672	0.190292	0.190292	0.175700
10	0.147973	0.231665	0.190204	0.190204	0.174468
11	0.146474	0.231663	0.190161	0.190161	0.173853
12	0.145422	0.231663	0.190146	0.190146	0.172218
13	0.144680	0.231663	0.190139	0.190139	0.171421
14	0.144154	0.231662	0.190136	0.190136	0.170375
15	0.143781	0.231662	0.190135	0.190135	0.169851
16	0.143516	0.231662	0.190135	0.190135	0.169667
17	0.143327	0.231662	0.190134	0.190134	0.169577
18	0.143193	0.231662	0.190134	0.190134	0.169458
19	0.143097	0.231662	0.190134	0.190134	0.169399
20	0.143028	0.231662	0.190134	0.190134	0.169240

w = 1.0; Z = 0.2; Tcp = 0.5; Tcm = 1.0

Table 15.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	0.500000	0.500000	0.500000	0.500000	0.500000
2	0.333333	0.333300	0.333333	0.333333	0.333333
3	0.285714	0.312500	0.285714	0.285714	0.285714
4	0.266667	0.309524	0.277778	0.277778	0.277778
5	0.258065	0.309091	0.275000	0.275000	0.275000
6	0.253986	0.309028	0.274510	0.274510	0.269608
7	0.251969	0.309019	0.274336	0.274336	0.267699
8	0.250980	0.309017	0.274306	0.274306	0.267361
9	0.250489	0.309017	0.274295	0.274295	0.267241
10	0.250244	0.309017	0.274293	0.274293	0.267007
11	0.250122	0.309017	0.274942	0.274942	0.266923
12	0.250061	0.309017	0.274292	0.274292	0.266601
13	0.255031	0.309017	0.274292	0.274292	0.266487
14	0.250015	0.309017	0.274292	0.274292	0.266263
15	0.250008	0.309017	0.274292	0.274292	0.266183
16	0.250004	0.309017	0.274292	0.274292	0.266169
17	0.250002	0.309017	0.274292	0.274292	0.266164
18	0.250001	0.309017	0.274292	0.274292	0.266154
19	0.250000	0.309017	0.274292	0.274292	0.266151
20	0.250000	0.309017	0.274292	0.274292	0.266137

w = 1.0; Z = 0.5; Tcp = 0.5; Tcm = 1.0

Table 16.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	0.500000	0.500000	0.500000	0.500000	0.500000
2	0.375000	0.375000	0.375000	0.375000	0.375000
3	0.346154	0.366667	0.346154	0.346154	0.346154
4	0.337500	0.366071	0.343750	0.343750	0.343750
5	0.334711	0.366029	0.343137	0.343137	0.343137
6	0.333791	0.366026	0.343085	0.343085	0.341312
7	0.333486	0.366025	0.343072	0.343072	0.340846
8	0.333384	0.366025	0.343071	0.343071	0.340806
9	0.333350	0.366025	0.343070	0.343070	0.340796
10	0.333339	0.366025	0.343070	0.343070	0.340766
11	0.333335	0.366025	0.343070	0.343070	0.340758
12	0.333334	0.366025	0.343070	0.343070	0.340719
13	0.333334	0.366025	0.343070	0.343070	0.340709
14	0.333333	0.366025	0.343070	0.343070	0.340679
15	0.333333	0.366025	0.343070	0.343070	0.340671
16	0.333333	0.366025	0.343070	0.343070	0.340671
17	0.333333	0.366025	0.343070	0.343070	0.340670
18	0.333333	0.366025	0.343070	0.343070	0.340670
19	0.333333	0.366025	0.343070	0.343070	0.340670
20	0.333333	0.366025	0.343070	0.343070	0.340669

w = 1.0; Z = 1.0; Tcp = 0.5; Tcm = 1.0

Table 17.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	0.500000	0.500000	0.500000	0.500000	0.500000
2	0.477273	0.477273	0.477273	0.477273	0.477273
3	0.476242	0.477226	0.476242	0.476242	0.476242
4	0.476193	0.477226	0.476240	0.476240	0.476240
5	0.476191	0.477226	0.476240	0.476240	0.476240
6	0.476190	0.477226	0.476240	0.476240	0.476237
7	0.476190	0.477226	0.476240	0.476240	0.476237
8	0.476190	0.477226	0.476240	0.476240	0.476237
9	0.476190	0.477226	0.476240	0.476240	0.476237
10	0.476190	0.477226	0.476240	0.476240	0.476237
11	0.476190	0.477226	0.476240	0.476240	0.476237
12	0.476190	0.477226	0.476240	0.476240	0.476237
13	0.476190	0.477226	0.476240	0.476240	0.476237
14	0.476190	0.477226	0.476240	0.476240	0.476237
15	0.476190	0.477226	0.476240	0.476240	0.476237
16	0.476190	0.477226	0.476240	0.476240	0.476237
17	0.476190	0.477226	0.476240	0.476240	0.476237
18	0.476190	0.477226	0.476240	0.476240	0.476237
19	0.476190	0.477226	0.476240	0.476240	0.476237
20	0.476190	0.477226	0.476240	0.476240	0.476237

w = 1.0; Z = 10.0; Tcp = 0.5; Tcm = 1.0

Table 18.

No. of Processors	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1	0.500000	0.500000	0.500000	0.500000	0.500000
2	0.488095	0.488095	0.488095	0.488095	0.488095
3	0.487812	0.488088	0.487812	0.487812	0.487812
4	0.487805	0.488088	0.487812	0.487812	0.487812
5	0.487805	0.488088	0.487812	0.487812	0.487812
6	0.487805	0.488088	0.487812	0.487812	0.487812
7	0.487805	0.488088	0.487812	0.487812	0.487812
8	0.487805	0.488088	0.487812	0.487812	0.487812
9	0.487805	0.488088	0.487812	0.487812	0.487812
10	0.487805	0.488088	0.487812	0.487812	0.487812
11	0.487805	0.488088	0.487812	0.487812	0.487812
12	0.487805	0.488088	0.487812	0.487812	0.487812
13	0.487805	0.488088	0.487812	0.487812	0.487812
14	0.487805	0.488088	0.487812	0.487812	0.487812
15	0.487805	0.488088	0.487812	0.487812	0.487812
16	0.487805	0.488088	0.487812	0.487812	0.487812
17	0.487805	0.488088	0.487812	0.487812	0.487812
18	0.487805	0.488088	0.487812	0.487812	0.487812
19	0.487805	0.488088	0.487812	0.487812	0.487812
20	0.487805	0.488088	0.487812	0.487812	0.487812

w = 1.0; Z = 20.0; Tcp = 0.5; Tcm = 1.0

Table 19.

w	Z	Tcp	Tcm	Tmin_Bus	Tmin_Chain	Tmin_Left	Tmin_Right	Tmin_Binary
1.0	0.1	1.0	0.5	1.000000	1.367096	1.185890	1.185890	1.076925
1.0	0.2	1.0	0.5	1.000000	1.474483	1.247679	1.247679	1.116801
1.0	0.5	1.0	0.5	1.000000	1.468965	1.240583	1.240583	1.135616
1.0	1.0	1.0	0.5	1.000000	1.334709	1.156484	1.156484	1.099297
1.0	10.0	1.0	0.5	1.000000	1.021899	1.003317	1.003317	1.002904
1.0	20.0	1.0	1.0	1.000000	1.006840	1.000579	1.000579	1.000537
1.0	0.1	1.0	1.0	1.000000	1.474483	1.247679	1.247679	1.116801
1.0	0.2	1.0	1.0	1.000000	1.491584	1.255531	1.255531	1.138141
1.0	0.5	1.0	1.0	1.000000	1.334709	1.156484	1.156484	1.099297
1.0	1.0	1.0	1.0	1.000000	1.184766	1.071239	1.071239	1.050410
1.0	10.0	1.0	1.0	1.000000	1.006864	1.000579	1.000579	1.000537
1.0	20.0	1.0	1.0	1.000000	1.001945	1.000087	1.000087	1.000083
1.0	0.1	0.5	1.0	1.000000	1.491584	1.255531	1.255531	1.138184
1.0	0.2	0.5	1.0	1.000000	1.384663	1.187083	1.187083	1.114441
1.0	0.5	0.5	1.0	1.000000	1.184766	1.071239	1.071239	1.050410
1.0	1.0	0.5	1.0	1.000000	1.082694	1.023217	1.023217	1.018190
1.0	10.0	0.5	1.0	1.000000	1.001945	1.000087	1.000087	1.000083
1.0	20.0	0.5	1.0	1.000000	1.000522	1.000012	1.000012	1.000012

Table 20.

position of starting processor	Z=0.1	Z=0.2	Z=0.5	Z=1.0	Z=5.0	Z=10.0
1	0.200038	0.270157	0.390388	0.500000	0.765564	0.854102
3	0.155742	0.205276	0.301358	0.407407	0.725822	0.836309
5	0.136182	0.184714	0.286977	0.400468	0.725861	0.836300
7	0.127986	0.178695	0.284944	0.400029	0.725861	0.836300
9	0.124782	0.177007	0.284663	0.400002	0.725861	0.836300
11	0.123946	0.176655	0.284629	0.400000	0.725861	0.836300
13	0.124782	0.177007	0.284663	0.400002	0.725861	0.836300
15	0.127986	0.178695	0.284944	0.400029	0.725861	0.836300
17	0.136182	0.184714	0.286977	0.400468	0.725861	0.836300
19	0.155742	0.205276	0.301358	0.407407	0.725822	0.836309