STATE UNIVERSITY OF NEW YORK AT

STONY BROOK

CEAS TECHNICAL REPORT 596

BUS ORIENTED LOAD SHARING FOR A NETWORK OF INTELLIGENT SENSORS

S. Bataineh and T.G. Robertazzi

February 1, 1991

Bus Oriented Load Sharing for a Network of Intelligent Sensors

Sameer Bataineh and Thomas 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 amongst n intelligent sensors interconnected through a bus type communication medium is considered for three distinct architectural configurations. It is found that a minimal time solution can be achieved if the computation by each sensor ends simultaneously. Simple recursions for the determination of the optimal allocation of load are presented. It is shown that a small number of intelligent sensors can be almost as effective as a larger number. These bus oriented architectures produce faster solutions than a previously published linear daisy chain architecture.

Keywords: Sensor Network, Load Sharing, Load Balancing, Scheduling.

Address all corresspondence to T. Robertazzi

Phone: (516)632-8412/messages 8400

E-mail: tom@sbee.sunysb.edu

1 Introduction

The problem of the fusion of data from distributed sensors has received an increasing amount of attention [1, 7, 8, 9, 10, 12, 13] since the publication of a paper by Tenney and Sandell in 1981 [11]. The basic idea is that measurements are made by spatially separated sensors, of possibly different types. The data, or better yet statistically meaningful summaries of the data, are then relayed to a site where the spatially disparate readings are fused so that meaningful decisions can be made regarding these measurements.

One important issue for distributed fusion concerns the trade-off between communication and computation [4]. That is, how much computation should take place at the sensor and how much and what information should be relayed to the fusion site. In this paper we examine the tradeoff between communication and computation in the context of a load sharing problem involving an "intelligent sensor" network. An intelligent sensor can make measurements, perform computation and communicate with neighboring sensors. In this problem it is assumed that a single sensor receives a burst of measurement data that requires processing in time proportional to the length of the data. It is further assumed that the data can be divided amongst multiple intelligent sensors to achieve a faster solution thru parallel processing. We also wish to take the time that it takes to transmit the data, or fractional parts of it, between intelligent sensors into account. The objective is to deduce the fraction of the data that should be allocated to each intelligent sensor so that the data can be processed in a minimal amount of time.

A network of such sensors arranged in a linear daisy chain was examined in [2] and a tree type configuration was examined in [3]. In this paper a broadcast bus is used to connect the sensors. This offers the possibility of a faster solution compared to the linear daisy chain architecture. Three configurations utilizing a broadcast bus are considered. In the first, the originating sensor acts as a control unit, distributing load but not processing it. In the other two configurations the originating sensor can perform computation on a portion of the load. These latter two configurations differ in whether or not a front-end sub-processor is included in a sensor for communication off-loading (so the sensor may perform computation and communication simultaneously).

In what follows it is assumed that each intelligent sensor is under the control of its own processor. Thus in the discussion that follows, the term " processor " will be used interchangeably with " Intelligent sensor ".

2 Architecture 1: Linear Network with Control Processor

Consider the case where the network model consists of one control processor and n communicating processors. As shown in Fig. 1, the control processor receives the measurement data and communicates it through a broadcast bus to the processors. The communication time for processor i, $i = 1, 2, \ldots, n$, is proportional to the amount of measurement data that has to be assigned to that processor. Two major factors determine the amount of measurement data assigned to each processor. The first is the computational speed of processor i in the network . Faster processors will receive more data than the slower ones. The second is the order of the load distribution among the processors in the network. Processor i starts computation immediately after receiving its share of the load from the control processor, $i = 1, 2, \ldots, n$. The timing diagram of the system is depicted in Fig. 2.

Let us first introduce the following notation.

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

w_i : *A constant that is inversely proportional to the speed of the ith processor.*

Z : *A constant that is inversely proportional to the speed of the bus between the control processor and communicating processor i in the network.*

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

T_{cm} : *The time it takes for control processor 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 *completed its computation. This include, in addition to computation time , communicating time with the control processor and waiting time. Waiting time is the time that processor* i *has to wait before being able to communicate with the control processor,* $i = 1, 2, \ldots, n$.

T_f : *The finish time of the process.*

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

The timing diagram, Fig. 2, shows that at $t = 0$, *the communicating processors are all idle and the control processor has completed receiving the measurement data and starts to communicate with the first processor in the system.*

The equations that govern the relations among various variables and parameters in the system are

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

$$T_2 = (\alpha_1 + \alpha_2) Z T_{cm} + \alpha_2 w_2 T_{cp} \tag{2.3}$$

$$T_3 = (\alpha_1 + \alpha_2 + \alpha_3) Z T_{cm} + \alpha_3 w_3 T_{cp} \tag{2.4}$$

$$T_4 = (\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4) Z T_{cm} + \alpha_4 w_4 T_{cp} \tag{2.5}$$

.

.

.

$$T_n = (\alpha_1 + \alpha_2 + \cdots + \alpha_n) Z T_{cm} + \alpha_n w_n T_{cp} \tag{2.6}$$

The fraction of total measurement load should sum to one

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

The important point of interest is the optimal total processing time , T_{M_1}. In order to examine the necessary condition to achieve the minimum time solution, let us consider a simple system with two processor, $n = 2$.

In this system:

$$\alpha_1 + \alpha_2 = 1 \tag{2.8}$$

$$T_1 = \alpha_1 Z T_{cm} + \alpha_1 w_1 T_{cp} \tag{2.9}$$

$$T_2 = (\alpha_1 + \alpha_2) Z T_{cm} + \alpha_2 w_2 T_{cp} \tag{2.10}$$

$$= Z T_{cm} + (1 - \alpha_1) w_2 T_{cp} \tag{2.11}$$

The optimal processing time, T_m, is:

$$T_m = min\,(max(T_1, T_2)) \tag{2.12}$$

As shown in Fig. 3 the min max function, T_m is optimized at the cross over point of the two lines. That is, where $T_1 = T_2$.

The control processor must find the optimal values for α_1 and α_2. An interesting problem would be extending this proof to higher dimensions. It would however, be difficult to identify the minimum point in three or higher dimensions. It can be seen intuitively though, that in order to obtain the maximum parallelism and minimum time solution, all processors must stop at the same time. This is because, otherwise, some processors would be idle while others were busy. Another way of expressing this intuition is to say one must keep all processors utilized up till the last moment; that is, all processors stop at the same time. This achieves the maximum efficiency in the system.

$$\eta(efficiency) = \frac{average\ finish}{finish\ time}$$

$$\eta = \frac{\frac{\sum_{i=1}^{n} T_i}{n}}{T_f}$$

$$\eta = \frac{\sum_{i=1}^{n} T_i}{n T_f} \tag{2.13}$$

Where n: is the total number of processors

$$T_f = max(T_1, T_2, T_3, \ldots, T_n)$$

$$\eta \longrightarrow 1 \quad if \quad |T_i - T_j| \longrightarrow 0 \tag{2.14}$$

$$i \neq j, \quad i = 1, 2, \ldots, n \quad j = 1, 2, \ldots, n$$

To verify the fact that all the processors must stop at the same time so as to obtain minimum processing (finish) time, two tables of data were obtained and their values were compared. Table one is obtained by running an exhaustive search program on different values of w *and* Z *with* $T_{cm} = 0.5, T_{cp} = 1$ *and* $n = 4$. *the program finds the best values of* $\alpha's$ *for the parameters searched within and the minimum processing time. There are four processors used to obtain table 1. The second table is obtained by solving recursively equations (2.2) to (2.5) based on previous idea that*

$$T_1 = T_2 = T_3 = T_4$$

The optimal values of $\alpha's$ *for n processors can be computed by solving recursively the following set of recursions:*

$$\alpha_1 + \alpha_2 + \alpha_3 + \cdots + \alpha_n = 1 \tag{2.15}$$

$$\alpha_{n-1} = \frac{\alpha_n(w_n T_{cp} + Z T_{cm})}{w_{n-1} T_{cp}} \tag{2.16}$$

$$\alpha_{n-2} = \frac{\alpha_{n-1}(w_{n-1} T_{cp} + Z T_{cm})}{w_{n-2} T_{cp}} \tag{2.17}$$

.

.

.

$$\alpha_2 = \frac{\alpha_3(w_3 T_{cp} + Z T_{cm})}{w_2 T_{cp}} \tag{2.18}$$

$$\alpha_1 = \frac{\alpha_2(w_2 T_{cp} + Z T_{cm})}{w_1 T_{cp}} \tag{2.19}$$

α_i *is solved for by equating* T_i *to* T_{i+1}, *that is* $T_i = T_{i+1}$. *As a comparison of the two tables reveals, the values of* $\alpha's$ *and the minimum processing time is almost the same for the same values of* $w's$, Z, T_{cm} *and* T_{cp}.

The optimal minimum processing time (finish time) function,T_{M_1} *is given by :*

$$T_{M_1} = \frac{\prod_{j=1}^{j=n} Z T_{cm} + w_j T_{cp}}{\sum_{i=1}^{n} (Z T_{cm})^{n-i} (T_{cp})^{i-1} (W_1 + W_2 + \cdots + W_{\binom{n}{i-1}})} \tag{2.20}$$

Where W_k is a product of terms:

$$W_k = (w_{k_1} w_{k_2} \cdots w_{k_{i-1}})$$

and

$$k_p \neq k_q$$

$$k_p = 1, 2, 3, \ldots, n \qquad k_q = 1, 2, 3, \ldots, n$$

The optimal minimum processing time function is symmetric in $w_1, w_2, \ldots w_n$. This can be seen from the expression of T_{M_1} where each term in the denominator contains $\binom{n}{i-1}$ components of W; each of which consists of $i-1$ combination of w. To clarify the point of symmetricity an example where $n = 3$ is provided on the (Appendix). So the position of the processors on the bus is not important, that is, any processor can be placed any where on the bus or any two processor can be exchanged without affecting the minimum processing time, T_{M_1}. In Fig. 4 the optimal minimum total processing time is plotted against the number of the processors in the linear network with $T_{cm} = 0.5$, $T_{cp} = 1.0$, $w_i = 1$ where $i = 1, 2, 3, \cdots n$ and four performance curves were obtained for Z_i =0.1, 0.2, 0.5 and 1 . As shown in Fig. 4, the minimum processing time, T_{M_1}, levels off to ZT_{cm} as more processor are added. This can be proved as following:

$$T_n = ZT_{cm} + \alpha_n w_n T_{cp} \tag{2.21}$$

It is clear to see that as the number of processors increases, the load that will be assigned to each processor will become smaller. In other words if

$$n \longrightarrow \infty \quad \Longrightarrow \quad \alpha_n \longrightarrow 0$$

therefore,

$$\lim_{n \to \infty} T_n = ZT_{cm} \tag{2.22}$$

Fig. 4 implies that only few numbers of processors are needed to come close to the minimum processing time . Fig. 4 also verifies some intuitive results such as : the minimum processing time increases as the value of Z increases and decreases as the number of processors increases.

In Fig. 5 we show the effect of various speeds of processors with a fixed speed of the bus. In this figure the optimal minimum processing time is plotted against the number of

processors in the network with $T_{cm} = 0.5$, $T_{cp} = 1.0$, $Z = 1.0$ and four performance curves were obtained for $w_i = 0.05, 0.1, 0.2, 0.5$ and 1 where $i = 1, 2, 3 \ldots, n$. As Fig. 5 reveals, if a large number of processors is used, the optimal processing time is independent of the speed of the processors. That is , for any processing speed, the minimum processing time will level off to ZT_{cm}. The second interesting point to observe is that using a small number of fast processors to solve the computational problem is better than using a large number of slow processors.

3 Architecture 2: No Control Processor, Processors with Front-End Processors

In order to improve the optimal minimum finish time, we consider another linear network topology where there is no control processor . Rather, the load may originate at any of the n homogeneous processors. Moreover, each processor contains a front-end processor for communications off-loading. That is, with the inclusion of the front-end processor, each processor may compute and communicate at the same time . The load may be originated at any one of these processors. The processor that originates the load is now performing both computation and communication simultaneously. Thus, it 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 it finishes receiving its data. As mentioned in the second section, two major factors determine the amount of measurement data assigned to each processor. The first is the computational speed of the processor. Faster processors will receive more data than slower ones. The second is the order of the processor for load distribution. The timing diagram of the system is plotted in Fig. 6. Between $t = 0$ and $t = \alpha_2 ZT_{cm}$ 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. In general, in the period between $t = 0$ and $t = (\alpha_2 + \alpha_3 + \cdots \alpha_i) ZT_{cm}$, $n - i$ processors would be idle and $i - 1$ processors perform computation; $i = 2, 3, 4, \ldots, n$. This fact serves to increase the minimum finish time.

In the following we will use the same definitions for α_i, w_i, T_{cp}, and T_f as in section 2. However, Z, T_{cm} and T_i are defined slightly different as following.

T_{cm} : *The time it takes for 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 *completed its computation,* $i = 1, 2, \ldots, n$. *This includes, in addition to computation time, communicating time and waiting time. Waiting time is the time that processor* i *has to wait before being able to communicate with the processor that distributes the load.*

Z : *A constant that is inversely proportional to the speed of the bus.*

With these definitions, the equations that relate various variables and parameters together are stated below:

$$T_1 = \alpha_1 w_1 T_{cp} \tag{3.1}$$

$$T_2 = \alpha_2 Z T_{cm} + \alpha_2 w_2 T_{cp} \tag{3.2}$$

$$T_3 = (\alpha_2 + \alpha_3) Z T_{cm} + \alpha_3 w_3 T_{cp} \tag{3.3}$$

$$T_4 = (\alpha_2 + \alpha_3 + \alpha_4) Z T_{cm} + \alpha_4 w_4 T_{cp} \tag{3.4}$$

$$\vdots$$

$$T_n = (\alpha_2 + \alpha_3 + \cdots + \alpha_n) Z T_{cm} + \alpha_n w_n T_{cp} \tag{3.5}$$

The fraction of the total measurement should sum to one

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1 \tag{3.6}$$

The objective in analyzing the above equations is to compute the optimal minimum processing time and compare it with the result that was obtained in the previous section.

The same intuition as in the previous section can be adopted to show that the optimal minimum finish time would be achieved when all processors stop at the same time,

that is when:

$$T_1 = T_2 = T_3 = \cdots = T_n$$

similarly, two tables of data, table 3 and table 4, were obtained and their values were compared in order to verify the fact that all processors must stop at the same time. Table 3 is obtained by running an exhaustive search program on different values of w and Z with $T_{cm} = 0.5$, $T_{cp} = 1$ and $n = 4$. The program finds the best values of $\alpha's$ for the parameters searched within and the minimum processing time. Four processors were used to obtain Table 3. Table 4 is obtained by solving recursively equations (3.1) to (3.4) based on previous idea that

$$T_1 = T_2 = T_3 = T_4$$

The optimal values of $\alpha's$ that the original processor should calculate in order to achieve the minimum processing time can be computed by solving recursively the following set of equations :

$$\alpha_{n-1} = \alpha_n \frac{w_n T_{cp} + Z T_{cm}}{w_{n-1} T_{cp}} \tag{3.7}$$

.

.

.

$$\alpha_3 = \alpha_4 \frac{w_4 T_{cp} + Z T_{cm}}{w_3 T_{cp}} \tag{3.8}$$

$$\alpha_2 = \alpha_3 \frac{w_3 T_{cp} + Z T_{cm}}{w_2 T_{cp}} \tag{3.9}$$

$$\alpha_1 = \alpha_2 \frac{w_2 T_{cp} + Z T_{cm}}{w_1 T_{cp}} \tag{3.10}$$

Here, α_i is solved for by equating T_i to T_{i+1}, that is $T_i = T_{i+1}$. A comparison of the tables shows that the values of $\alpha's$ and the optimal minimum processing time is almost the same for the same values of $w's$, T_{cm}, T_{cp} and Z.

The minimum processing time function, T_{M_2}, is given by:

$$T_{M_2} = w_1 T_{cp} \frac{\prod_{j=2}^{n} (w_j T_{cp} + Z T_{cm})}{\sum_{i=1}^{n} (Z T_{cm})^{n-i} (T_{cp})^{i-1} (W_1 + W_2 + W_3 + \cdots + W_{\binom{n}{i-1}})} \tag{3.11}$$

and the maximum throughput(γ) is:

$$\gamma = \frac{1}{T} \tag{3.12}$$

Where

$$W_x = (w_{k_1} w_{k_2} \cdots w_{k_{i-1}})$$

and

$$k_p \neq k_q \qquad k_p = 1, 2, 3, \ldots, n \qquad k_q = 1, 2, 3, \ldots, n$$

Unlike the minimum processing time function, T_{M_1}, in the previous section the minimum processing time function, T_{M_2}, is symmetric in $w_2, w_3, w_4, \ldots, w_n$ but not symmetric in $w_1, w_2, w_3, \ldots, w_n$. This implies that the choice of the processor where the load is originated is important. To achieve the best processing time in such a system, the fastest processor would have to distribute the data. This is verified by table 3 and table 4.

In Fig. 7 the minimum total processing time function,T_{M_2}, is plotted against the number of the processors in the network with $T_{cm} = 0.5$, $T_{cp} = 1.0$, $w_i = 1$ where $i = 1, 2, 3, \ldots, n$ and six performance curves were obtained for $Z_i = 0.1$, 0.2, 0.5, 1, 10 and 20. As shown in the figure, the optimal minimum total processing time, T_{M_2}, decreases as the number of processors increases. However, the curves for large Z levels off quickly after a small number of processors. This is because it is almost fast for a small number of processors to solve the problem as it is to take time to communicate the problem to a large number of processors. In that case, it is better to use only smaller number of processors .

4 Architecture 3: No Control Processor, Processors without Front-End Processors

The network topology that is discussed in this section is similar to that discussed in the previous one except for the fact that each of n homogeneous processors in the network contains no front-end processor for communicating off-loading. That is, each processor may either communicate or compute but not do both at the same time. The load may be originated at any one of these processors. The processor that originates the load broadcasts to each processor in the network its share of the load before its starts to compute its own share. Each processor begins to compute its share of the load at the moment that it finishes receiving its data. As stated previously, two major factors determine the amount of measurement data assigned to each processor, the computational speed of the processor and the order of the

processor for load distribution. Faster processors will receive more data than slower ones. The timing diagram of the system is plotted in Fig. 8. Between $t = 0$ *and* $\alpha_2 Z T_{cm}$*, none of the processors performs computation, the first processor communicates data to the second processor and processors* $3, 4, 5, \ldots, n$ *are all idle. In general, in the period between* $t = 0$ *and* $t = (\alpha_1 + \alpha_2 + \cdots + \alpha_i) Z T_{cm}$*, only* $i - 2$ *processors perform computation and* $n - i$ *processors are idle,* $i = 2, 3, \ldots, n$*. This fact serves to increase the minimum finish time.*

In the following we will use the same definitions for α_i*,* w_i*,* Z*,*T_{cm}*,* T_{cp}*,* T_i *and* T_f *as in previous section. With these definitions, the equations that relate the various variables and parameters together are stated below:*

$$T_1 = (1 - \alpha_1) Z T_{cm} + \alpha_1 w_1 T_{cp} \tag{4.1}$$

$$T_2 = \alpha_2 Z T_{cm} + \alpha_2 w_2 T_{cp} \tag{4.2}$$

$$T_3 = (\alpha_2 + \alpha_3) Z T_{cm} + \alpha_3 w_3 T_{cp} \tag{4.3}$$

$$T_4 = (\alpha_2 + \alpha_3 + \alpha_4) Z T_{cm} + \alpha_4 w_4 T_{cp} \tag{4.4}$$

$$\vdots$$

$$T_n = (1 - \alpha_1) Z T_{cm} + \alpha_n w_n T_{cp} \tag{4.5}$$

The fractions of the total measurement load should sum to one

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1 \tag{4.6}$$

We can use the same intuition results as in the previous section in order to show that the optimal minimum finish time would be achieved when all processors stop at the same time, that is when:

$$T_1 = T_2 = T_3 = \cdots = T_n$$

As before, in order to verify the fact that all processors must stop at the same time in order to achieve the optimal minimum finish time, two tables of data were obtained and their values were compared . To obtain table 5, an exhaustive search program on different

values of w and Z with $T_{cm} = 0.5$, $T_{cp} = 1.0$ and $n = 4$. The program finds the best values of $\alpha's$ for the parameters searched within and the minimum processing time. There are four processors for table 5. Table 6 was obtained by solving recursively the equations from (4.1) to (4.4) based on previous idea that

$$T_1 = T_2 = T_3 = T_4$$

The originating processor should calculate the optimal values of $\alpha's$. These values can be computed by solving recursively the following set of equations:

$$\alpha_{n-1} = \alpha_n \frac{w_n T_{cp} + Z T_{cm}}{w_{n-1} T_{cp}} \tag{4.7}$$

.

.

.

$$\alpha_3 = \alpha_4 \frac{w_4 T_{cp} + Z T_{cm}}{w_3 T_{cp}} \tag{4.8}$$

$$\alpha_2 = \alpha_3 \frac{w_3 T_{cp} + Z T_{cm}}{w_2 T_{cp}} \tag{4.9}$$

$$\alpha_1 = \alpha_2 \frac{w_n}{w_1} \tag{4.10}$$

Except for α_1, here also, α_i is solved for by equating T_i to T_{i+1}, that is, $T_i = T_{i+1}$. α_1 is solved for by equating T_1 to T_n. A comparison of the tables reveals that the values of $\alpha's$ and the minimum processing time is about the same for the same values of $w's$, T_{cm}, T_{cp} and Z, except when slow communication makes it faster to use a single processor (see below).

The optimal minimum processing time function, T_{M_3}, for this network topology, is given by:

$$T_{M_3} = \frac{w_1 \prod_{j=2}^{n} (Z T_{cm} + w_j T_{cp})}{w_1 \sum_{i=2}^{n-1} (Z t_{cm})^{n-i} (T_{cp})^{i-2} (W_1 + W_2 + \cdots + W_{\binom{n-1}{i-2}}) + (T_{cp})^{n-2} (Z_1 + Z_2 + \cdots + Z_n)} \tag{4.11}$$

and the maximum throughput(γ) is:

$$\gamma = \frac{1}{T} \tag{4.12}$$

Where

$$W_x = (w_{k_1} w_{k_2} \ldots w_{k_{i-2}})$$

$k_p = 2, 3, 4, \ldots, n \qquad k_q = 2, 3, 4, \ldots, n$

And

$$Z_y = (w_{k_1} w_{k_2} \ldots w_{k_{n-1}})$$

$k_p = 1, 2, 3, \ldots, n \qquad k_q = 1, 2, 3, \ldots, n$

And

$$n \geq 3$$

For $n = 2$, the minimum processing time function is given by:

$$T_{M_3} = \frac{w_1(ZT_{cm} + w_2 T_{cp})}{w_1 + w_2} \tag{4.13}$$

The expression of the optimal minimum processing time function, for architecture 3, T_{M_3}, implies that T_{M_3} is not symmetric in $w_1, w_2, \ldots, w_n$; however, it is symmetric in $w_2, w_3, \ldots, w_n$. The symmetricity of T_{M_3} is obvious if one observes that each term of the summation in the denominator contains $\binom{n-1}{i-2}$ components of W, each of which consists of $i-2$ elements of w. The second term of the denominator, that is not included in the summation, contains n components of Z each of which consists of $n-1$ elements of w. This implies that the choice of the processor where the load is originated is important as it was the case in the previous section. Again, in order to achieve the best processing time in this system, the fastest processor would have to distribute the data. This fact is also verified by the table 5 and table 6. However, processor $2, 3, 4, \ldots, n$ can receive data in any order without affecting the minimum processing time. In Fig. 9 the optimal minimum processing time is plotted against the number of the processors in the network with $T_{cm} = 0.5$, $T_{cp} = 1.0$, $w_i = 1$; $i = 1, 2, 3, \ldots, n$, and seven performance curves were obtained for $Z = 0.1$, 0.2, 0.5, 1.0, 1.8, 2.0 and 2.1. From Fig. 9, there are two important observations. The first is the optimal minimum processing time function levels off to a certain value after few number of processors. The number of processors at which T_{M_3} levels off increases as the speed of communication decreases. The second is that there is a thresh- hold value that limits the speed of communication. After this value, using more than one processor to compute the load would take more time than if only one processor was used. This is because of the time that is wasted in slow communication.

In the following we will calculate the value at which T_{M_3} levels off and the thresh-hold value of Z. Based on previous result that the optimal minimum processing time is achieved when all processors stop at the same time, the following equation holds:

$$\begin{aligned} T_1 &= (1-\alpha_1)ZT_{cm} + \alpha_1 w_1 T_{cp} = T_{M_3} \\ &= ZT_{cm} + \alpha_1(w_1 T_{cp} - ZT_{cm}) \end{aligned} \tag{4.14}$$

As

$$n \longrightarrow \infty \qquad \Longrightarrow \qquad \alpha_1 \longrightarrow 0$$

and so $T_{M_3} \longrightarrow ZT_{cm}$

The following inequality must hold; otherwise, the optimal minimum processing time will increase as we use more processors to process the load.

$$\begin{aligned} w_1 T_{cp} - ZT_{cm} &\geq 0 \\ Z &\leq \frac{w_1 T_{cp}}{T_{cm}} \end{aligned} \tag{4.15}$$

This implies that the thresh-hold value of Z is:

$$Z_{th} = \frac{w_1 T_{cp}}{T_{cm}} \tag{4.16}$$

Both previous results are verified in Fig. 9.

5 Conclusion

In this paper three bus oriented architecture are examined in the context of a load sharing problem. For the three architectures it was shown that the optimal processing time is achieved when all processors stop at the same time. The best processing time is obtained for the architecture where the processors have front end processors for communication off loading and the fastest processor distributes the load. If the control processor distributes the load, the location of the processors on the bus is not important. The interaction between communication and computation has also been examined. If the speed of the bus slow, the use of more than a few processors will not substantially improve the performance of the

system. This is because it is as fast for a small number of processors to solve the problem as it is to take the time to communicate the problem to a larger number of processors. It was also observed that using a smaller number of fast processor to solve a computationl problem can be better than using a large number of slow processors.

Acknowledgements

The research reported in this paper was supported by National Science Foundation under Grant number NCR-8703689 and by the SDIO/IST and by the U.S Office of Naval Research under Grant number N00014-85-K 0610.

Appendix

We will clarify the symmetricity in equation (2.20) by using an example for a system with three processors , $n = 3$. For such system the optimal minimum time function is given by:

$$T_{M_1} = \frac{(ZT_{cm} + w_1T_{cp})(ZT_{cm} + w_2T_{cp})(ZT_{cm} + w_3T_{cp})}{(ZT_{cm})^2(T_{cp})^0(W_1) + (ZT_{cm}T_{cp})(W_1 + W_2 + W_3) + (ZT_{cm})^0(T_{cp})^2(W_1 + W_2 + W_3)} \quad (A.1)$$

$$T_{M_1} = \frac{(ZT_{cm} + w_1T_{cp})(ZT_{cm} + w_2T_{cp})(ZT_{cm} + w_3T_{cp})}{(ZT_{cm})^2 + (ZT_{cm}T_{cp})(w_1 + w_2 + w_3) + (T_{cp})^2(w_1w_2 + w_1w_3 + w_2w_3)} \quad (A.2)$$

The symmetricity is obvious in the product terms of the numerator of equation (A.1). The denominator has three sum of product terms. The first, the second and the third terms are obtained for $i = 1, 2$ and 3 respectively. As mentioned in the text each W in the denominator consists of $i - 1$ product terms of w. In this example $i = 1, 2, 3$. Therefore, in the first term of the denominator W consists of $(1 - 1 = 0)$ of w. In the second term each W consists of one w, while in the third term each W consists of a product of two different w. Applying the previous steps on equation (A.1), We obtain equation (A.2). One can readily see the symmetricity in equ. (A.2), since ($w_1 + w_2 + w_3$) and ($w_1w_2 + w_1w_3 + w_2w_3$) are symmetric in w_1, w_2 and w_3.

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, 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, July 1990.

[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 rocessing 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 Se"sor 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.

Table 1: Result of Search Program for Architecture 1

Z	w_1	w_2	w_3	w_4	α_1	α_2	α_3	α_4	T_{min}
0.1	1.0	1.0	1.0	1.0	0.2700	0.2600	0.2400	0.2300	0.2865
0.5	1.0	1.0	1.0	1.0	0.3400	0.2700	0.2200	0.1700	0.4275
1.0	1.0	1.0	1.0	1.0	0.4100	0.2800	0.1800	0.1300	0.6300
2.0	1.0	1.0	1.0	1.0	0.5400	0.2600	0.1400	0.0600	1.080
1.0	0.5	0.2	0.8	1.0	0.5200	0.4000	0.0600	0.0200	0.5400
1.0	1.0	0.2	0.8	0.5	0.3400	0.5200	0.0800	0.0600	0.5340
1.0	1.0	0.8	0.2	0.5	0.3500	0.2700	0.3200	0.0600	0.5340

Table 2: Result Obtained by Solving a Set of Recursive Equations for Architecture 1

Z	w_1	w_2	w_3	w_4	α_1	α_2	α_3	α_4	T_{min}
0.1	1.0	1.0	1.0	1.0	0.2686	0.2558	0.2436	0.2320	0.2820
0.5	1.0	1.0	1.0	1.0	0.3388	0.2710	0.2168	0.1734	0.4234
1.0	1.0	1.0	1.0	1.0	0.4154	0.2769	0.1846	0.1231	0.6231
2.0	1.0	1.0	1.0	1.0	0.5333	0.2667	0.1333	0.0667	1.067
1.0	0.5	0.2	0.8	1.0	0.5311	0.3794	0.0584	0.0311	0.5311
1.0	1.0	0.2	0.8	0.5	0.3541	0.5058	0.0778	0.0623	0.5311
1.0	1.0	0.8	0.2	0.5	0.3541	0.2724	0.3113	0.0623	0.5311

Table 3: Result of Search Program for Architecture 2

Z	w_1	w_2	w_3	w_4	α_1	α_2	α_3	α_4	T_{min}
0.1	1.0	1.0	1.0	1.0	0.2600	0.2600	0.2400	0.24000	0.2770
0.5	1.0	1.0	1.0	1.0	0.3400	0.2600	0.2200	0.1800	0.3450
1.0	1.0	1.0	1.0	1.0	0.4200	0.2800	0.1800	0.1200	0.4200
2.0	1.0	1.0	1.0	1.0	0.5400	0.2600	0.1400	0.0600	0.5400
1.0	0.5	0.2	0.8	1.0	0.5300	0.3800	0.0600	0.0300	0.2680
1.0	1.0	0.2	0.8	0.5	0.3600	0.5000	0.0800	0.0600	0.3600
1.0	1.0	0.8	0.2	0.5	0.3600	0.2600	0.3200	0.0600	0.3600

=

Table 4: Result Obtained by Solving a Set of Recursive Equations for Architecture 2

Z	w_1	w_2	w_3	w_4	α_1	α_2	α_3	α_4	T_{min}
0.1	1.0	1.0	1.0	1.0	0.2686	0.2558	0.2436	0.2320	0.2686
0.5	1.0	1.0	1.0	1.0	0.3388	0.2710	0.2168	0.1734	0.3388
1.0	1.0	1.0	1.0	1.0	0.4154	0.2769	0.1846	0.1231	0.4154
2.0	1.0	1.0	1.0	1.0	0.5333	0.2667	0.1333	0.0667	0.5333
1.0	0.5	0.2	0.8	1.0	0.5311	0.3794	0.0584	0.0311	0.2656
1.0	1.0	0.2	0.8	0.5	0.3541	0.5058	0.0778	0.0623	0.3541
1.0	1.0	0.8	0.2	0.5	0.3541	0.2724	0.3113	0.0623	0.3541

Table 5: Result of Search Program for Architecture 3

Z	w_1	w_2	w_3	w_4	α_1	α_2	α_3	α_4	T_{min}
0.1	1.0	1.0	1.0	1.0	0.2400	0.2600	0.2600	0.2400	0.2860
0.5	1.0	1.0	1.0	1.0	0.2200	0.3200	0.2600	0.2000	0.4150
1.0	1.0	1.0	1.0	1.0	0.1800	0.3800	0.2600	0.1800	0.5900
2.0	1.0	1.0	1.0	1.0	1.0000	0.0000	0.0000	0.0000	1.000
1.0	0.5	0.2	0.8	1.0	1.0000	0.0000	0.0000	0.0000	0.5000
1.0	1.0	0.2	0.8	0.5	0.0400	0.7400	0.1200	0.1000	0.5300
1.0	1.0	0.8	0.2	0.5	0.0400	0.4000	0.4600	0.1000	0.5300

Table 6: Result Obtained by Solving a Set of Recursive Equations for Architecture 3

Z	w_1	w_2	w_3	w_4	α_1	α_2	α_3	α_4	T_{min}
0.1	1.0	1.0	1.0	1.0	0.2408	0.2655	0.2529	0.2408	0.2788
0.5	1.0	1.0	1.0	1.0	0.2078	0.3247	0.2595	0.2078	0.4058
1.0	1.0	1.0	1.0	1.0	0.1739	0.3913	0.2609	0.1739	0.5870
2.0	1.0	1.0	1.0	1.0	0.1250	0.5000	0.2500	0.1250	1.000
1.0	0.5	0.2	0.8	1.0	0.1172	0.7143	0.1099	0.0586	0.5000
1.0	1.0	0.2	0.8	0.5	0.04598	0.7471	0.1149	0.0920	0.5230
1.0	1.0	0.8	0.2	0.5	0.04598	0.4023	0.4598	0.0920	0.5230

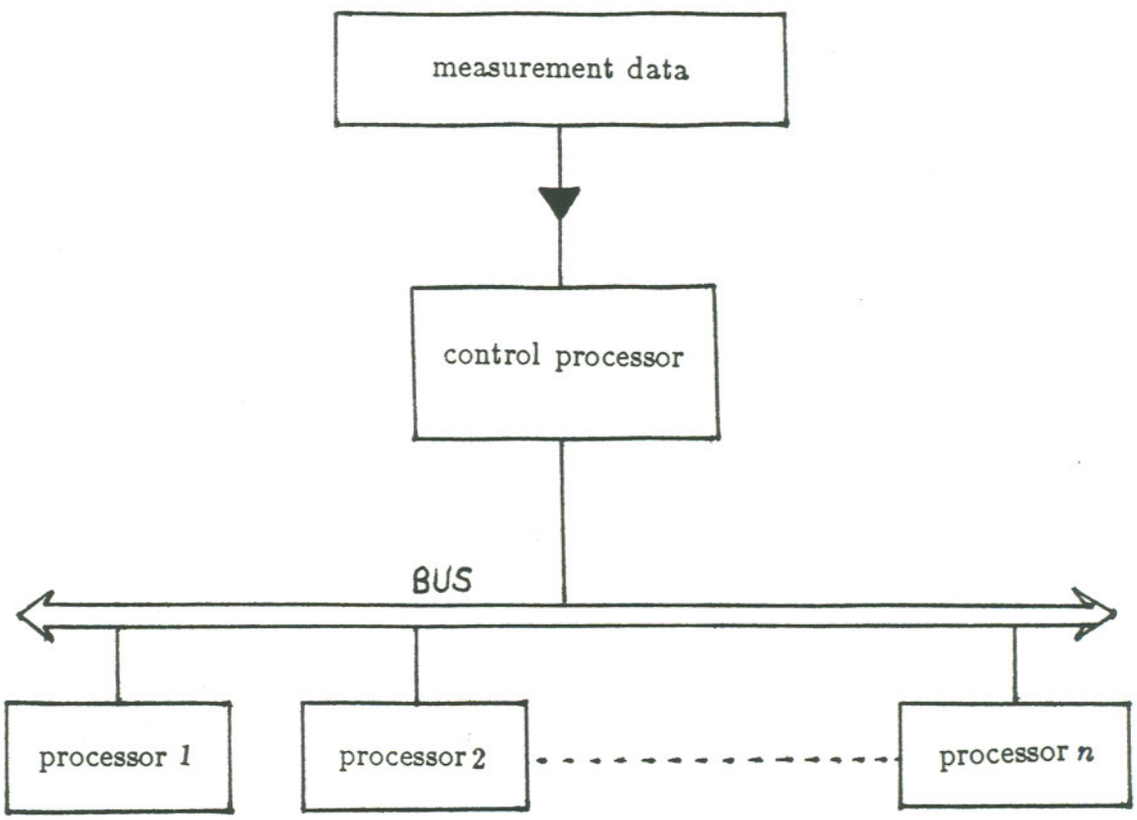
measurement data
control processor
BUS
processor 1
processor 2
processor n

Fig. 1

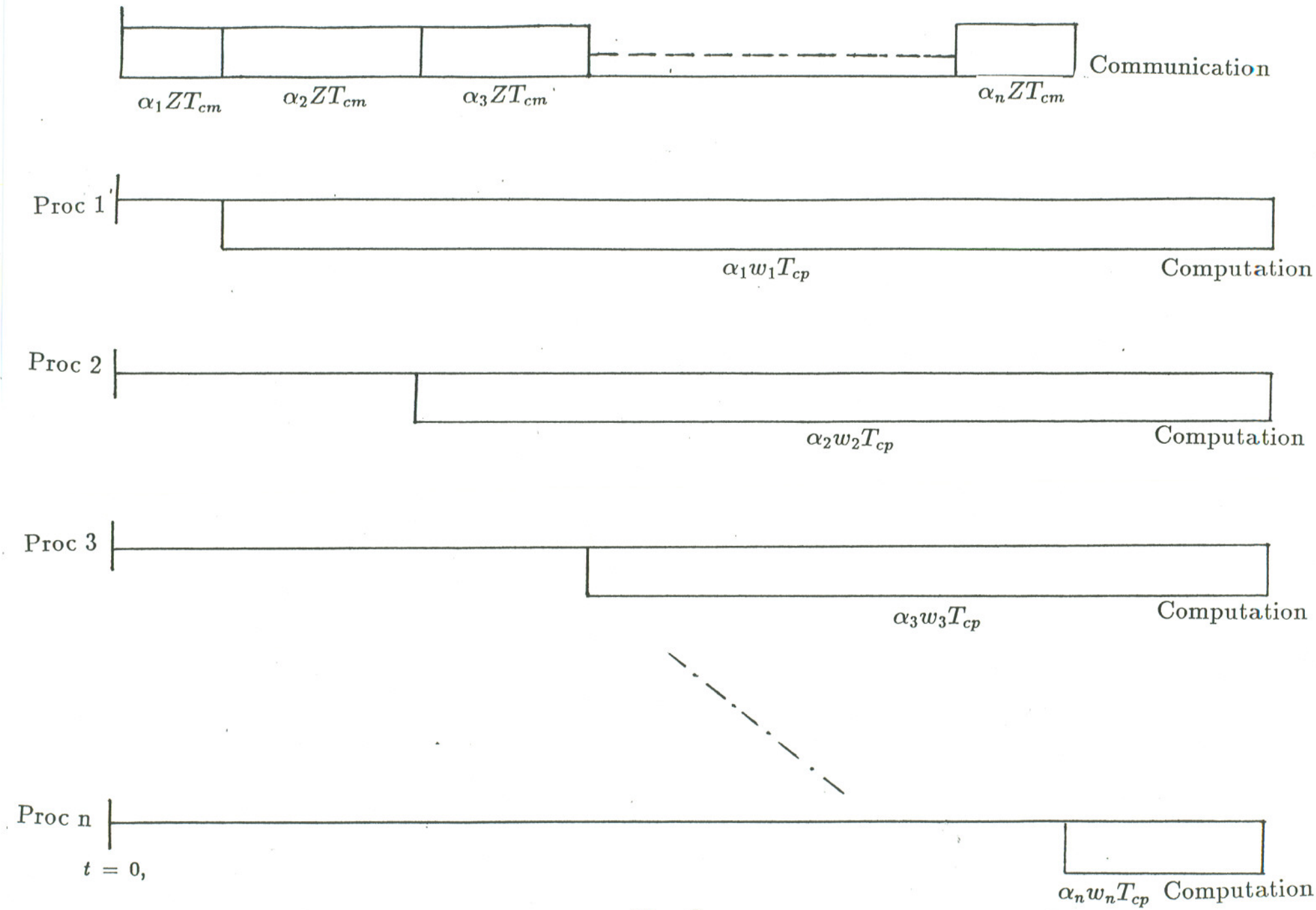
Communication
$\alpha_1 ZT_{cm}$
$\alpha_2 ZT_{cm}$
$\alpha_3 ZT_{cm}$
$\alpha_n ZT_{cm}$
Proc 1
$\alpha_1 w_1 T_{cp}$
Computation
Proc 2
$\alpha_2 w_2 T_{cp}$
Computation
Proc 3
$\alpha_3 w_3 T_{cp}$
Computation
Proc n
$t = 0$,
$\alpha_n w_n T_{cp}$
Computation

Fig. 2

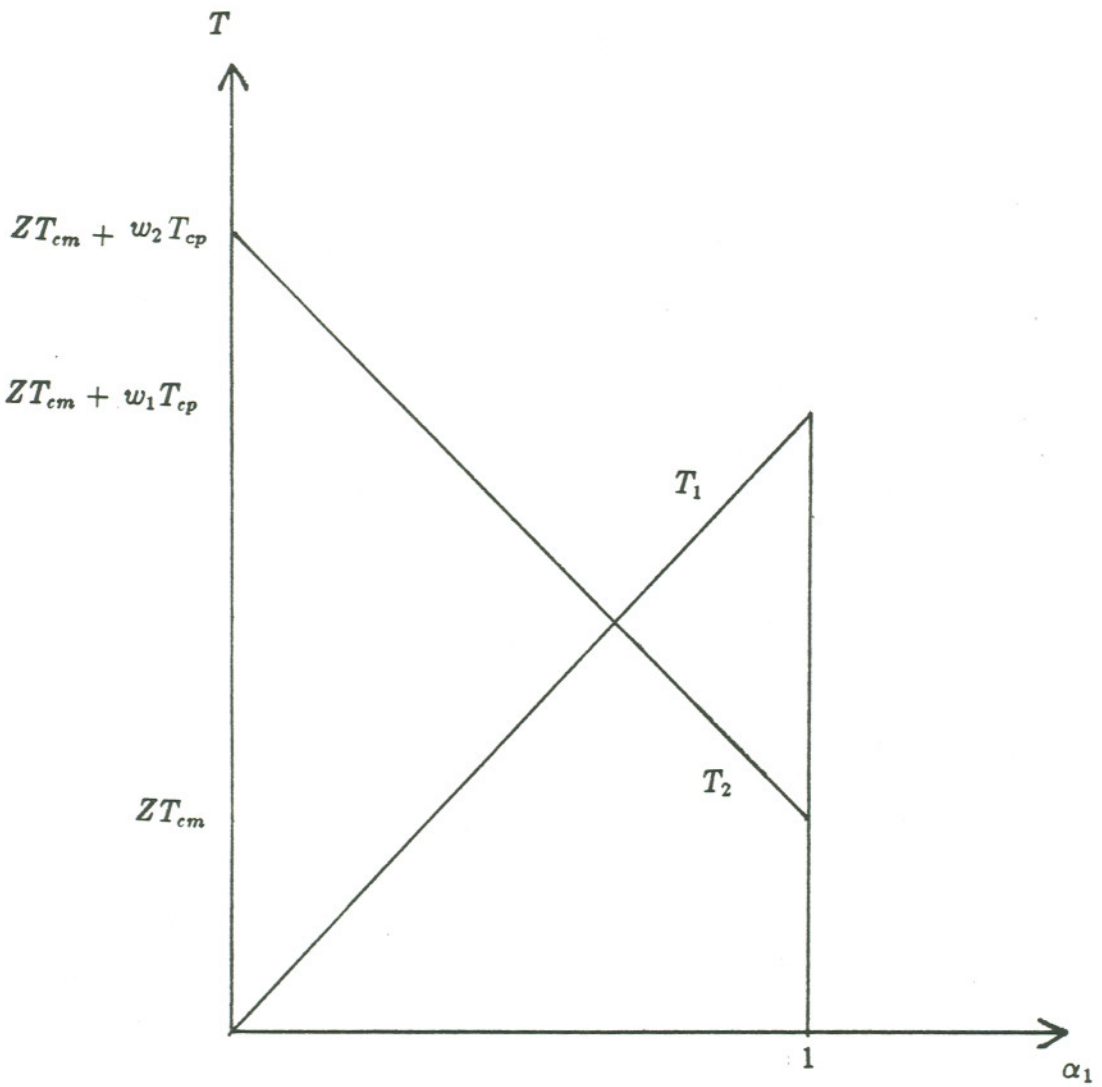

Fig. 3

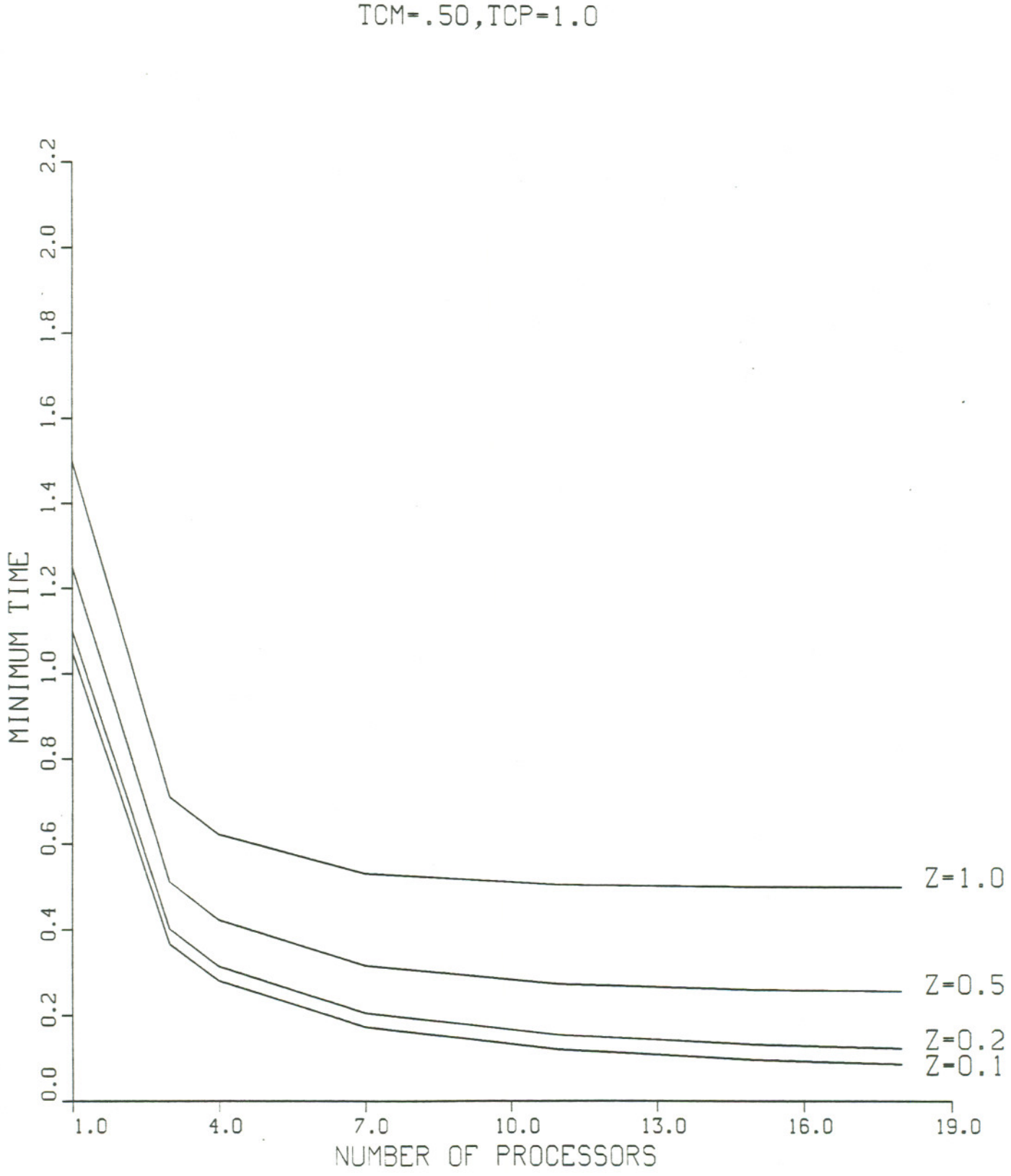
TCM=.50,TCP=1.0
MINIMUM TIME
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
1.0
4.0
7.0
10.0
13.0
16.0
19.0
NUMBER OF PROCESSORS
Z=1.0
Z=0.5
Z=0.2
Z=0.1

Fig. 4

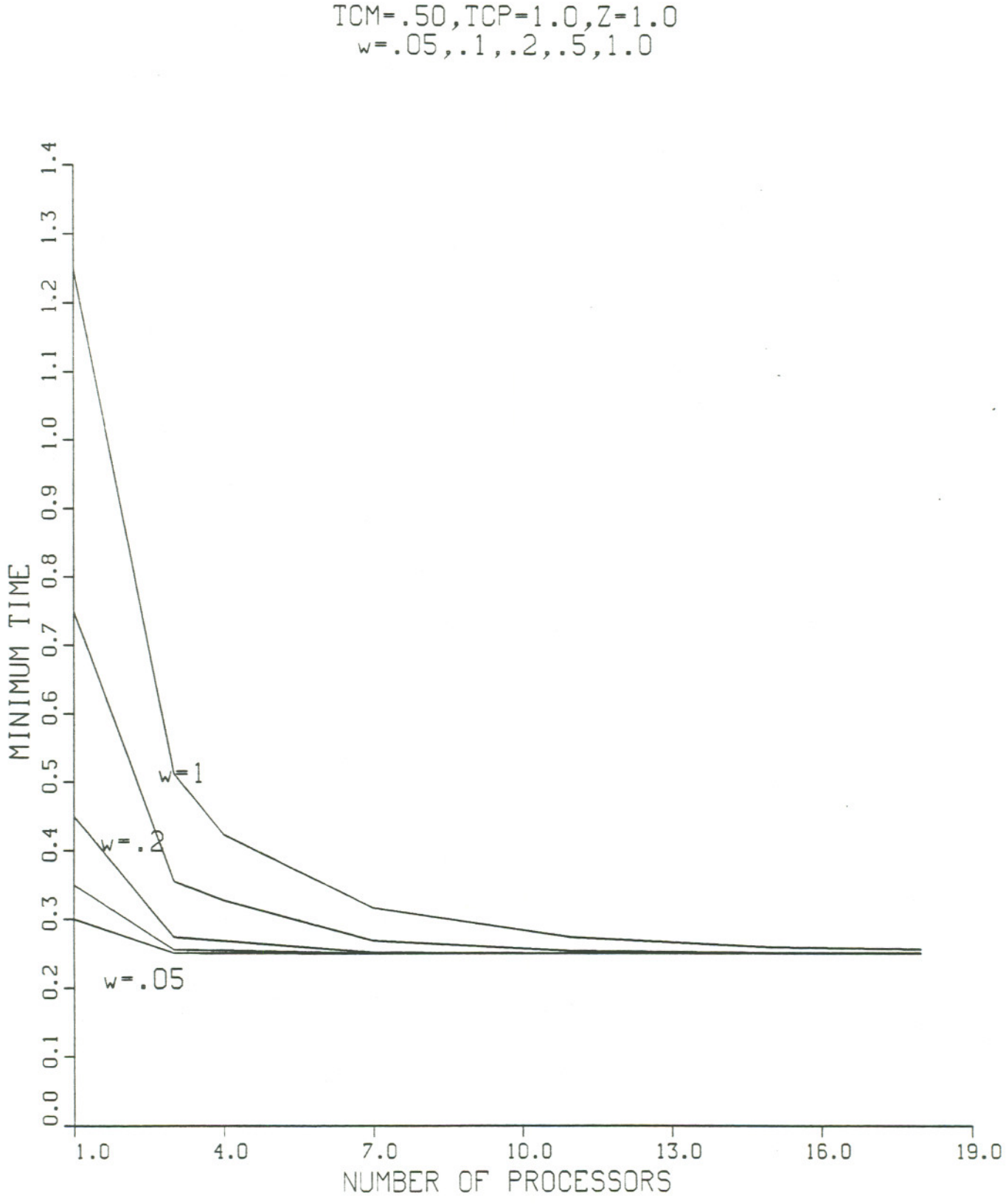
TCM=.50,TCP=1.0,Z=1.0
w=.05,.1,.2,.5,1.0
MINIMUM TIME
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.2
1.3
1.4
w=1
w=.2
w=.05
1.0
4.0
7.0
10.0
13.0
16.0
19.0
NUMBER OF PROCESSORS

Fig. 5

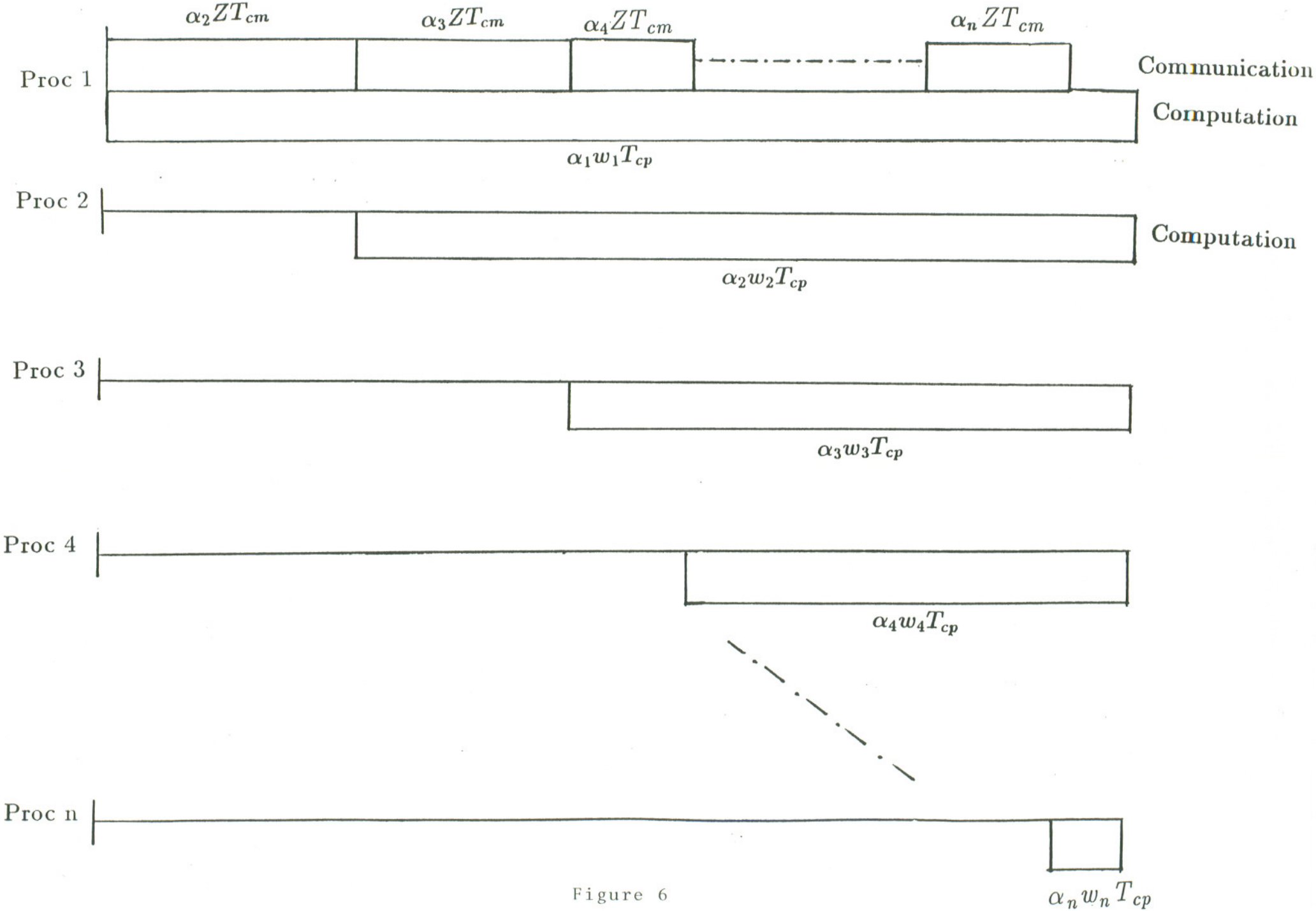
$\alpha_2 ZT_{cm}$
$\alpha_3 ZT_{cm}$
$\alpha_4 ZT_{cm}$
$\alpha_n ZT_{cm}$
Proc 1
Communication
Computation
$\alpha_1 w_1 T_{cp}$
Proc 2
Computation
$\alpha_2 w_2 T_{cp}$
Proc 3
$\alpha_3 w_3 T_{cp}$
Proc 4
$\alpha_4 w_4 T_{cp}$
Proc n
$\alpha_n w_n T_{cp}$

Figure 6

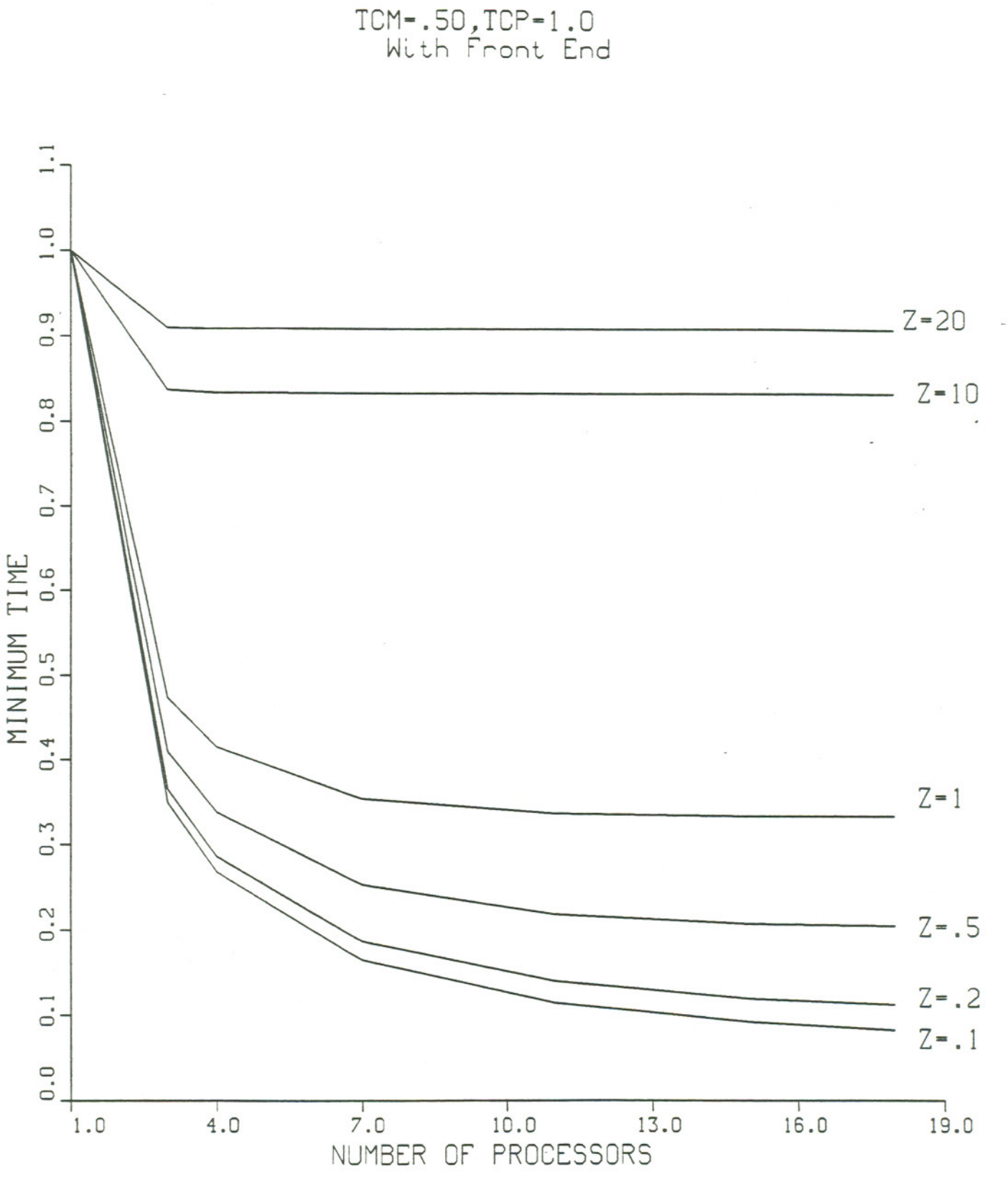
TCM=.50,TCP=1.0
With Front End
MINIMUM TIME
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
1.0
4.0
7.0
10.0
13.0
16.0
19.0
NUMBER OF PROCESSORS
Z=20
Z=10
Z=1
Z=.5
Z=.2
Z=.1

Fig. 7

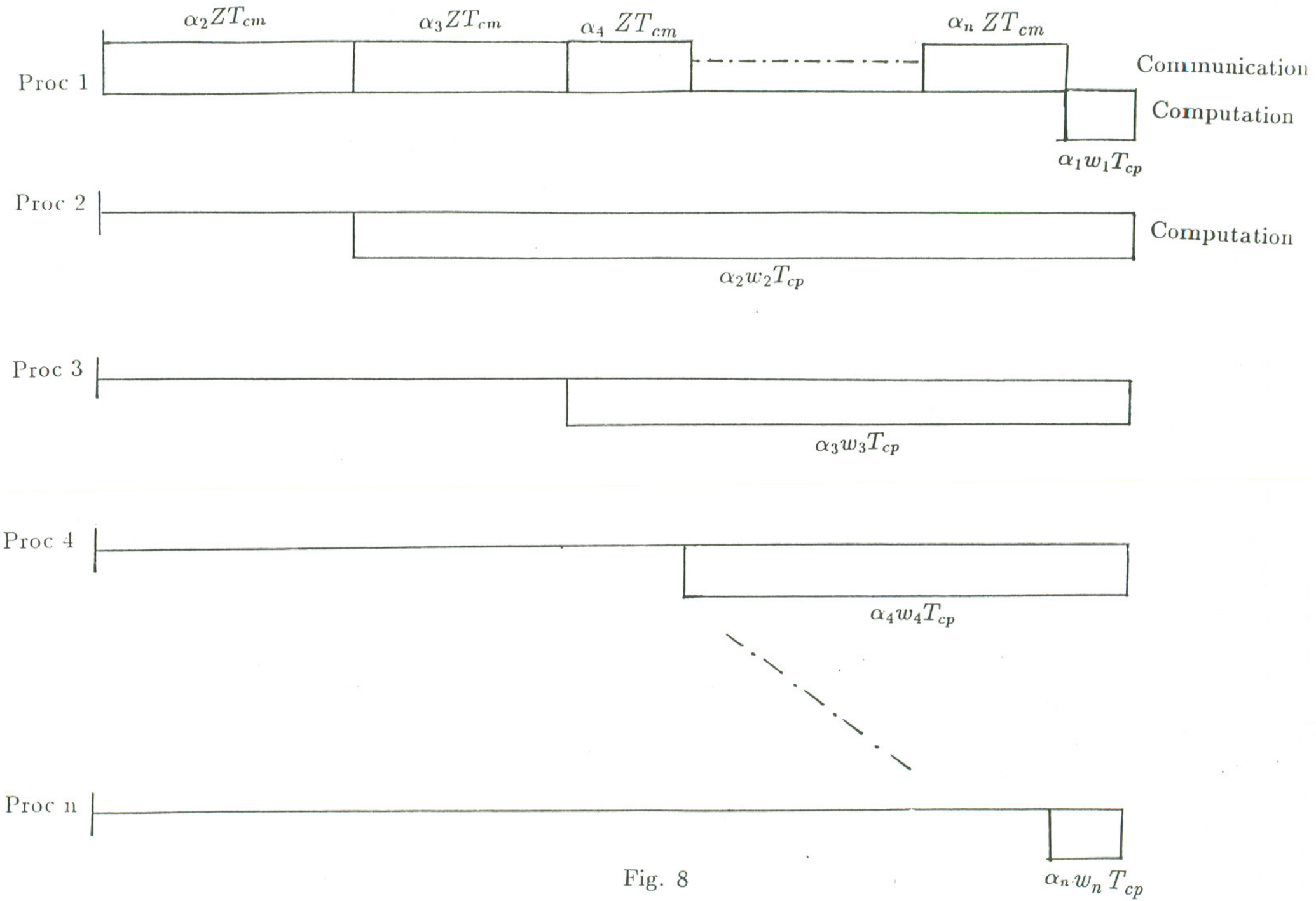
$\alpha_2 ZT_{cm}$
$\alpha_3 ZT_{cm}$
$\alpha_4\ ZT_{cm}$
$\alpha_n\ ZT_{cm}$
Communication
Proc 1
Computation
$\alpha_1 w_1 T_{cp}$
Proc 2
Computation
$\alpha_2 w_2 T_{cp}$
Proc 3
$\alpha_3 w_3 T_{cp}$
Proc 4
$\alpha_4 w_4 T_{cp}$
Proc n
$\alpha_n w_n\ T_{cp}$

Fig. 8

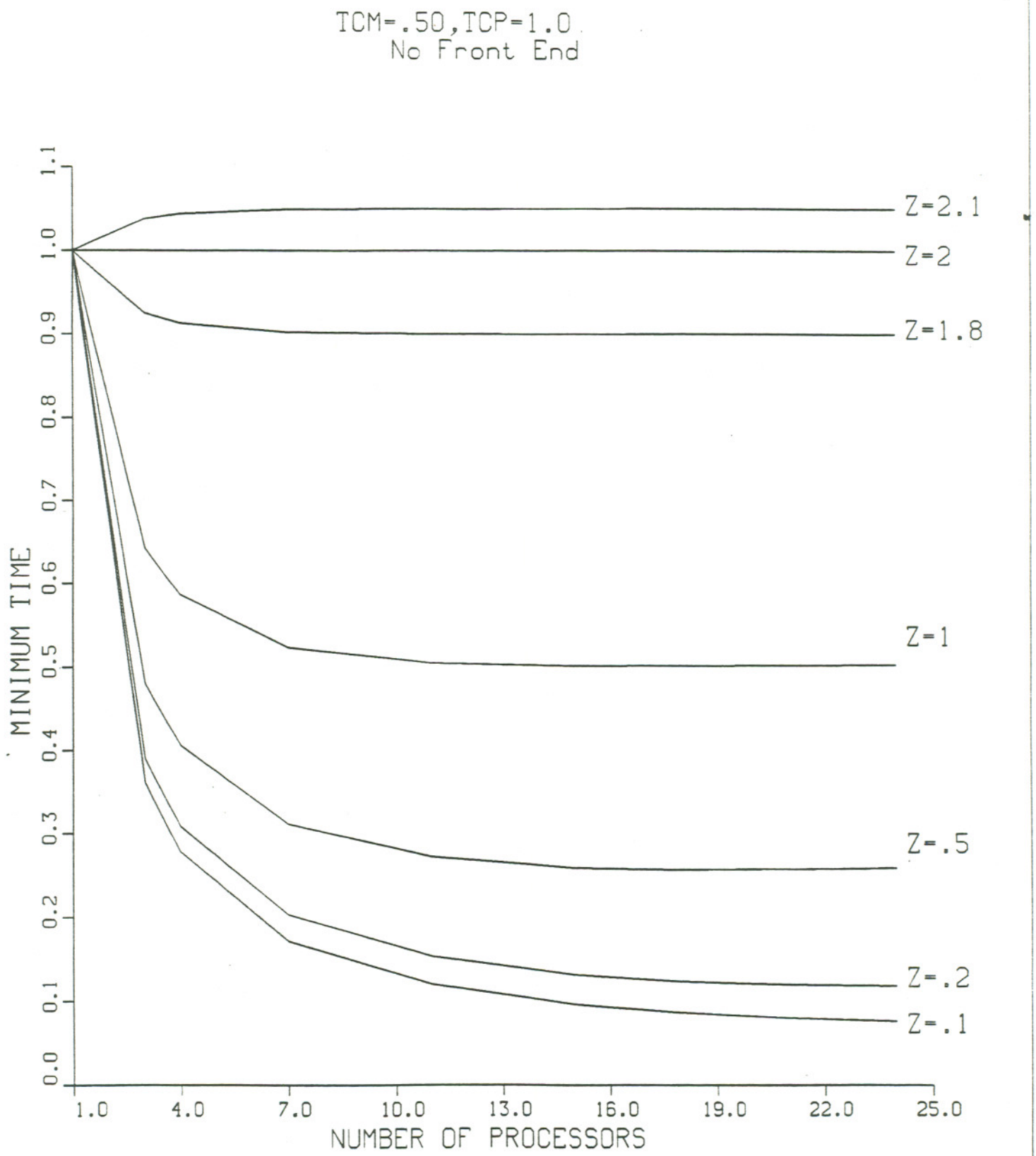
TCM=.50,TCP=1.0
No Front End
MINIMUM TIME
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
1.1
Z=2.1
Z=2
Z=1.8
Z=1
Z=.5
Z=.2
Z=.1
1.0
4.0
7.0
10.0
13.0
16.0
19.0
22.0
25.0
NUMBER OF PROCESSORS

Fig. 9