STATE UNIVERSITY OF NEW YORK AT

STONY BROOK

CEAS TECHNICAL REPORT 649

Optimal Load Sharing for a Divisible Job on Bus Networks

J. Sohn & T.G. Robertazzi

December 16, 1992

Optimal Load Sharing for a Divisible Job on Bus Networks

Jeeho Sohn and Thomas G. Robertazzi, *Senior Member, IEEE*

Dept. of Electrical Engineering,

SUNY at Stony Brook,

Stony Brook, N.Y. 11794

Abstract

Optimal load allocation for load sharing a divisible job over N processors interconnected in bus-oriented networks is considered. The network has either a control processor or no control processor where processors are not equipped with front-end processors. It is analytically proved, for the first time, that a minimal solution time is achieved when the computation by each processor finishes at the same time. Closed form solutions for the minimum finish time and the optimal data allocation for each processor are also obtained.

1 Introduction

In recent years, there has been of great interest in distributed sensor networks [1]. In distributed sensor networks, measurements are made by spatially distinct sensors. The data is, then, broadcast to a site where the spatially disparate readings are fused so that meaningful decisions can be made regarding these measurements. One major issue for distributed sensor networks is the trade-off between communication and computation [2]. That is, the decision of how much time should be spent to communicate and how much time should be spent to process (compute) the measurements becomes an important problem.

Related to the distributed sensor network problem are a number of papers which deal with scheduling and load sharing in multiprocessors [3, 4]. However most work assumes that a job can be assigned to a single processor. Only recently has there been interest in multiprocessor scheduling with jobs that need to be assigned to more than one processor [5, 6, 7].

Recently there has been work on a load sharing problem involving a *divisible* job. A divisible job is a job that can be arbitrarily portioned in a linear fashion among a number of processors. Applications include the processing of very long data files as in signal and image processing and Kalman filtering.

In [8], recursive expressions for calculating the optimal load allocation for linear daisy chains of processors were presented. This is based on the simplifying premise that for an optimal allocation of load, all processors must stop processing at the same time. Intuitively, this is because otherwise some processors would be idle while others were still busy. Analogous solutions have been developed for tree networks [9] and bus networks [10, 11]. The

equivalence of first distributing load either to the left or to the right from a point in the interior of a linear daisy chain is demonstrated in [17]. Optimal sequences of load distribution in tree networks are described in [16, 18, 19, 20]. Closed form solutions for homogeneous bus and tree networks appear in [12]. Asymptotic solutions for systems with infinite number of processors appear in [13, 21].

In [14], the concept of processor equivalence was used to prove that the optimal allocation of load for a linear daisy chain of processors involves having all the processors finish computation at the same time. However, until now there has been no analytic proof available that the same type of solution is optimal for N processors interconnected through a bus type channel. What has been available, aside from intuition, is a proof for the two processors case (N=2), and computational results consistent with the *all processors stop at the same time* premise [10, 11]. In this paper an analytic proof is presented for the case of bus type networks where the network has either a control processor or no control processor and where the processors are not equipped with front-end processors. The proof shows that for the general N processors case the minimum time solution occurs when all processors finish computation at the same time. A by product of the proof are closed form solutions for the optimal load allocation when processor speeds are heterogeneous (see [12] for the case of homogeneous processor speeds).

This paper is organized as follows. In section 2, the proof for a bus network with a control processor is presented in a recursive fashion. In section 3, the proof for the case that there are no control processor and no front-end processors is examined. In section 4, generalization of the proof to an alternative architecture is discussed. The conclusion appears

in section 5.

2 Architecture 1:

Bus Network with a Control Processor

Consider a bus network that consists of N processors and a control processor that receives a burst of measurement data and distributes the processing load among the N processors to obtain the benefits of parallel processing as shown in Fig. 1. The control processor does no processing itself. It does not matter whether the processors are equipped with front-end processors or not because the load distribution is performed by the control processor. Each processor may have a different computing speed.

The following notation will be used throughout this paper:

α_i : The fraction of the entire processing load that is assigned to ith processor.

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

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

T_{cm} : The time that it takes to transmit the entire set of measurement data over the channel when $Z = 1$.

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

T_i : The finish time of the ith processor's computation, assuming the load is delivered to the originating processor (or the control processor if available) at time zero.

Fig. 2 shows the timing diagram for a bus network where there is one control processor. At time $t = 0$, the control processor transmits the first fraction of processing load to the processor 1 which takes time $\alpha_1 Z T_{cm}$. When the transmission of the first fraction of processing load is finished, the control processor then transmits the second fraction of processing load to processor 2 which takes time $\alpha_2 Z T_{cm}$. In the mean time, processor 1 starts computing the received processing load which requires $\alpha_1 w_1 T_{cp}$ of time. The process then continues on in the natural way. The equations that represent the finish time of each processor are:

$$T_1 = \alpha_1 Z T_{cm} + \alpha_1 w_1 T_{cp} \tag{1}$$

$$T_2 = (\alpha_1 + \alpha_2) Z T_{cm} + \alpha_2 w_2 T_{cp} \tag{2}$$

$$T_3 = (\alpha_1 + \alpha_2 + \alpha_3) Z T_{cm} + \alpha_3 w_3 T_{cp} \tag{3}$$

$$\vdots$$

$$T_i = (\alpha_1 + \alpha_2 + \cdots + \alpha_i) Z T_{cm} + \alpha_i w_i T_{cp} \tag{4}$$

$$\vdots$$

$$T_{N-1} = (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_{N-1} w_{N-1} T_{cp} \tag{5}$$

$$T_N = (\alpha_1 + \alpha_2 + \cdots + \alpha_N) Z T_{cm} + \alpha_N w_N T_{cp} \tag{6}$$

The fractions of total measurement load should sum to one.

$$\alpha_1 + \alpha_2 + \cdots + \alpha_i + \cdots + \alpha_N = 1 \tag{7}$$

The necessary conditions to achieve the minimum solution time will be examined through the following subsections in a recursive manner. It will first be shown that $T_1 = T_2$ for an optimal solution, then $T_1 = T_2 = T_3$ and continuing recursively until $T_1 = T_2 = \cdots = T_N$.

2.1 Consideration of T_1 and T_2

First consider T_1 and T_2, the finish times of computation of processor 1 and processor 2, respectively. On the other hand, the rest of processing finish times, $T_3, T_4, \ldots, T_N$, will not be considered yet and will be assumed to have arbitrary values. That is, the fractions, $\alpha_3, \alpha_4, \ldots, \alpha_N$, are assumed to have arbitrary constant values. They will be considered in the following subsections along with some results obtained here.

Let C_2 be the sum of $\alpha_3, \alpha_4, \ldots, \alpha_N$

$$C_2 = \alpha_3 + \alpha_4 + \cdots + \alpha_N \tag{8}$$

where C_2 is a constant. Then

$$\begin{aligned} \alpha_1 + \alpha_2 &= 1 - (\alpha_3 + \alpha_4 + \cdots + \alpha_N) \\ &= 1 - C_2 \end{aligned} \tag{9}$$

and

$$\alpha_2 = (1 - C_2) - \alpha_1 \tag{10}$$

where α_1 has its maximum value when $\alpha_2 = 0$:

$$0 \leq \alpha_1 \leq 1 - C_2 \tag{11}$$

Then, T_1 and T_2 can be represented as follows:

$$T_1 = (ZT_{cm} + w_1 T_{cp})\alpha_1 \tag{12}$$

$$T_2 = (\alpha_1 + \alpha_2)ZT_{cm} + \alpha_2 w_2 T_{cp} \tag{13}$$

$$= (1 - C_2)ZT_{cm} + [(1 - C_2) - \alpha_1]w_2 T_{cp} \tag{14}$$

$$= (1 - C_2)(ZT_{cm} + w_2 T_{cp}) - w_2 T_{cp} \cdot \alpha_1 \tag{15}$$

Now T_1 has its maximum value and T_2 has its minimum value when α_1 reaches its maximum value, that is,

$$\max(T_1) = T_1\{\ \alpha_1 = 1 - C_2\ \}$$

$$= (1 - C_2)(ZT_{cm} + w_1 T_{cp}) \tag{16}$$

$$\min(T_2) = T_2\{\ \alpha_1 = 1 - C_2\ \}$$

$$= (1 - C_2)ZT_{cm} \tag{17}$$

The optimal processing time is the time that minimizes $\max(T_1, T_2)$. As shown in Fig. 3, the optimal processing time is achieved at the crossover point of the two lines where $T_1 = T_2$. Note that there always exits a crossover point across the two lines since $\max(T_1) > \min(T_2)$, that is, $(1 - C_2)(ZT_{cm} + w_1 T_{cp}) > (1 - C_2)ZT_{cm}$.

From Eq.(12) and Eq.(13), α_2 can be expressed as a function of α_1 since $T_1 = T_2$:

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

$$= k_1 \alpha_1 \tag{18}$$

where $k_1 = \frac{\alpha_2}{\alpha_1} = \frac{w_1 T_{cp}}{ZT_{cm} + w_2 T_{cp}}$.

2.2 Consideration of T_1, T_2 and T_3

This subsection will examine the optimal processing time when T_1, T_2 and T_3 are considered. This consideration will include some information which was obtained from the previous subsection, namely $T_1 = T_2$. We will assume $\alpha_4, \alpha_5, \ldots, \alpha_N$ to have arbitrary constant values.

Let C_3 be the sum of $\alpha_4, \alpha_5, \ldots, \alpha_N$.

$$C_3 = \alpha_4 + \alpha_5 + \cdots + \alpha_N \tag{19}$$

where C_3 is a constant. Then

$$\begin{aligned} \alpha_1 + \alpha_2 + \alpha_3 &= 1 - (\alpha_4 + \alpha_5 + \cdots + \alpha_N) \\ &= 1 - C_3 \end{aligned} \tag{20}$$

and

$$\begin{aligned} \alpha_3 &= (1 - C_3) - (\alpha_1 + \alpha_2) \\ &= (1 - C_3) - (1 + k_1)\alpha_1 \end{aligned} \tag{21}$$

since $\alpha_2 = k_1\alpha_1$. Now α_1 has its maximum value when $\alpha_3 = 0$.

$$0 \leq \alpha_1 \leq \frac{1 - C_3}{1 + k_1} \tag{22}$$

Then T_1, T_2 and T_3 can be represented as follows:

$$T_1 = T_2 = (ZT_{cm} + w_1T_{cp})\alpha_1 \tag{23}$$

$$T_3 = (\alpha_1 + \alpha_2 + \alpha_3)ZT_{cm} + \alpha_3 w_3 T_{cp} \tag{24}$$

$$= (1 - C_3)ZT_{cm} + [(1 - C_3) - (1 + k_1)\alpha_1]w_3T_{cp} \tag{25}$$

$$= (1 - C_3)(ZT_{cm} + w_3T_{cp}) - (1 + k_1)w_3T_{cp} \cdot \alpha_1 \tag{26}$$

Now $T_1 = T_2$ has its maximum value and T_3 has its minimum value when α_1 reaches its maximum value, that is,

$$\begin{aligned} \max(T_1 = T_2) &= T_1\{\ \alpha_1 = \frac{1 - C_3}{1 + k_1}\ \} \\ &= \frac{1 - C_3}{1 + k_1}\,(ZT_{cm} + w_1 T_{cp}) \qquad (27) \\ \min(T_3) &= T_3\{\ \alpha_1 = \frac{1 - C_3}{1 + k_1}\ \} \\ &= (1 - C_3)ZT_{cm} \qquad (28) \end{aligned}$$

For a crossover point across the two lines to exist, $\max(T_1 = T_2)$ must be greater than $\min(T_3)$, that is,

$$\frac{1 - C_3}{1 + k_1}\,(ZT_{cm} + w_1 T_{cp}) \ > \ (1 - C_3)ZT_{cm} \qquad (29)$$

Proof: The above condition can be reduced as follows:

$$\begin{aligned} ZT_{cm} + w_1 T_{cp} &> (1 + k_1)ZT_{cm} \qquad (30) \\ w_1 T_{cp} &> k_1 ZT_{cm} \qquad (31) \\ w_1 T_{cp} &> \frac{\alpha_2}{\alpha_1}\,ZT_{cm} \qquad (32) \\ \alpha_1 w_1 T_{cp} &> \alpha_2 ZT_{cm} \qquad (33) \end{aligned}$$

From Eq.(12) and Eq.(13), the following equation must be satisfied since $T_1 = T_2$:

$$\alpha_1 w_1 T_{cp} = \alpha_2 ZT_{cm} + \alpha_2 w_2 T_{cp} \qquad (34)$$

Since $\alpha_2 w_2 T_{cp} > 0$, the above inequality, Eq.(29), is true. □

Then there exists a crossover point across the two lines and the optimal processing time is achieved at that point where $T_1 = T_2 = T_3$ as in Fig. 4.

From Eq.(13), Eq.(18) and Eq.(24), α_3 can be expressed as a function of α_2 and α_1 since $T_1 = T_2 = T_3$:

$$\begin{aligned} \alpha_3 &= \frac{w_2 T_{cp}}{Z T_{cm} + w_3 T_{cp}} \alpha_2 \\ &= k_2 \alpha_2 \\ &= k_2 k_1 \cdot \alpha_1 \end{aligned} \tag{35}$$

where $k_2 = \frac{\alpha_3}{\alpha_2} = \frac{w_2 T_{cp}}{Z T_{cm} + w_3 T_{cp}}$.

2.3 Consideration of $T_1, T_2, \ldots,$ and T_i

Based on the results of the previous subsections, one can extend the proof to show that $T_1 = T_2 = \cdots = T_i$ achieves the minimal solution time. The assumption that $\alpha_{i+1}, \alpha_{i+2}, \ldots, \alpha_N$ have some arbitrary constant values will be also hold in this subsection.

Let C_i be the sum of $\alpha_{i+1}, \alpha_{i+2}, \ldots, \alpha_N$.

$$C_i = \alpha_{i+1} + \alpha_{i+2} + \cdots + \alpha_N \tag{36}$$

where C_i is a constant. Then

$$\begin{aligned} \alpha_1 + \alpha_2 + \cdots + \alpha_i &= 1 - (\alpha_{i+1} + \alpha_{i+2} + \cdots + \alpha_N) \\ &= 1 - C_i \end{aligned} \tag{37}$$

and

$$\begin{aligned} \alpha_i &= (1 - C_i) - (\alpha_1 + \alpha_2 + \cdots + \alpha_{i-1}) \\ &= (1 - C_i) - (1 + k_1 + k_1 k_2 + \cdots + k_1 k_2 \cdots k_{i-2}) \alpha_1 \end{aligned} \tag{38}$$

where α_1 has its maximum value when $\alpha_i = 0$:

$$0 \leq \alpha_1 \leq \frac{1 - C_i}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{i-2}} \tag{39}$$

Then, $T_1, T_2, \ldots, T_i$ can be represented as follows:

$$T_1 = T_2 = \cdots = T_{i-1} = (ZT_{cm} + w_1T_{cp})\alpha_1 \tag{40}$$

$$T_i = (\alpha_1 + \alpha_2 + \cdots + \alpha_i)ZT_{cm} + \alpha_i w_i T_{cp} \tag{41}$$

Using the representation of Eq.(37) and Eq.(38) and simplifying results in:

$$\begin{aligned} T_i &= (1 - C_i)ZT_{cm} + [(1 - C_i) - (1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{i-2})\alpha_1]w_iT_{cp} \\ &= (1 - C_i)(ZT_{cm} + w_iT_{cp}) - (1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{i-2})w_iT_{cp} \cdot \alpha_1 \end{aligned} \tag{42}$$

Now $T_1 = T_2 = \cdots = T_{i-1}$ has its maximum value and T_i has its minimum value when α_1 reaches its maximum value, that is,

$$\begin{aligned} \max(T_1 = T_2 = \cdots = T_{i-1}) &= T_1\{\ \alpha_1 = \frac{1 - C_i}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{i-2}}\ \} \\ &= \frac{1 - C_i}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{i-2}}(ZT_{cm} + w_1T_{cp}) \end{aligned} \tag{43}$$

$$\begin{aligned} \min(T_i) &= T_i\{\ \alpha_1 = \frac{1 - C_i}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{i-2}}\ \} \\ &= (1 - C_i)ZT_{cm} \end{aligned} \tag{44}$$

The following condition, based on the above, must be satisfied in order for a crossover point to exist between the two lines:

$$\frac{1 - C_i}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{i-2}}(ZT_{cm} + w_1T_{cp}) > (1 - C_i)ZT_{cm} \tag{45}$$

Proof: The above condition can be reduced as follows:

$$ZT_{cm} + w_1 T_{cp} > (1 + k_1 + k_1 k_2 + \cdots + k_1 k_2 \cdots k_{i-2}) ZT_{cm} \quad (46)$$

$$w_1 T_{cp} > k_1 (1 + k_2 + k_2 k_3 + \cdots + k_2 k_3 \cdots k_{i-2}) ZT_{cm} \quad (47)$$

$$w_1 T_{cp} > \frac{\alpha_2}{\alpha_1}(1 + \frac{\alpha_3}{\alpha_2} + \frac{\alpha_3}{\alpha_2}\frac{\alpha_4}{\alpha_3} + \cdots + \frac{\alpha_3}{\alpha_2}\frac{\alpha_4}{\alpha_3} \cdots \frac{\alpha_{i-1}}{\alpha_{i-2}}) ZT_{cm} \quad (48)$$

$$\alpha_1 w_1 T_{cp} > (\alpha_2 + \alpha_3 + \cdots + \dot{\alpha}_{i-1}) ZT_{cm} \quad (49)$$

Since T_1 and T_{i-1} can be rewritten as

$$T_1 = \alpha_1 ZT_{cm} + \alpha_1 w_1 T_{cp} \quad (50)$$

$$T_{i-1} = (\alpha_1 + \alpha_2 + \cdots + \alpha_{i-1}) ZT_{cm} + \alpha_{i-1} w_{i-1} T_{cp} \quad (51)$$

and $T_1 = T_{i-1}$, the following equation is satisfied.

$$\alpha_1 w_1 T_{cp} = (\alpha_2 + \alpha_3 + \cdots + \alpha_{i-1}) ZT_{cm} + \alpha_{i-1} w_{i-1} T_{cp} \quad (52)$$

Since $\alpha_{i-1} w_{i-1} T_{cp} > 0$, the above inequality, Eq.(45), is true. □

There thus exists a crossover point across the two lines and the optimal processing time is achieved at that point where $T_1 = T_2 = \cdots = T_i$ as in Fig. 5.

One can see that this procedure can be continued up to the case including all the finish times, $T_1, T_2, \ldots, T_N$. Then $T_1 = T_2 = \cdots = T_N$ will be obtained to minimize the solution time. Hence the minimal solution time involves all processors stopping their computing at the same time.

From Eq.(41) and Eq.(51), α_i can be expressed as a function of $\alpha_{i-1}, \alpha_{i-2}, \ldots$,and α_1 since $T_1 = T_2 = \cdots = T_i$:

$$\alpha_i = \frac{w_{i-1} T_{cp}}{ZT_{cm} + w_i T_{cp}} \alpha_{i-1}$$

$$
\begin{aligned}
&= k_{i-1}\alpha_{i-1} \\
&= k_{i-1}k_{i-2}\alpha_{i-2} \\
&\vdots \\
&= k_{i-1}k_{i-2}\cdots k_1 \cdot \alpha_1 \qquad 2 \le i \le N
\end{aligned} \tag{53}
$$

where

$$
k_j = \frac{\alpha_{j+1}}{\alpha_j} = \frac{w_j T_{cp}}{Z T_{cm} + w_{j+1} T_{cp}} \qquad 1 \le j \le N-1
$$

Since the sum of α_i's must be one, α_1 can be obtained by the normalization equation.

$$
\begin{aligned}
1 &= \alpha_1 + \alpha_2 + \alpha_3 + \cdots + \alpha_N \\
&= (1 + k_1 + k_1k_2 + \cdots + k_1k_2\cdots k_{N-1})\alpha_1
\end{aligned} \tag{54}
$$

From the above the optimal values of α_i's that the originating processor (the control processor) should calculate in order to achieve the minimal solution time can be computed by the following algorithm.

1) $$k_j = \frac{w_j T_{cp}}{Z T_{cm} + w_{j+1} T_{cp}} \qquad 1 \le j \le N-1 \tag{55}$$

2) $$
\begin{aligned}
\alpha_1 &= [\, 1 + k_1 + k_1k_2 + \cdots + k_1k_2\cdots k_{N-1} \,]^{-1} \\
&= [\, 1 + \sum_{i=1}^{N-1} (\prod_{j=1}^{i} k_j) \,]^{-1}
\end{aligned} \tag{56}
$$

3) $$
\begin{aligned}
\alpha_i &= k_1k_2\cdots k_{i-1} \cdot \alpha_1 \\
&= (\prod_{j=1}^{i-1} k_j) \cdot \alpha_1 \qquad 2 \le i \le N
\end{aligned} \tag{57}
$$

Interestingly, the solution for the optimal load allocations is of a *product form*. That is, the solution of α_i (Eq.(57)) can be expressed as a product of system constants (k_i's) and a

normalization constant, α_1. The existence of a product form solution for this deterministic problem is all the more interesting as product form solutions are after associated with the stochastic environment of certain classes queueing networks [22].

3 Architecture 2: No Control Processor, Processors without Front-End Processors

The bus network to be examined in this section is one without a control processor. The processors are not equipped with front-end processors for communications off-loading. That is, the processors cannot communicate and compute at the same time. Any single processor can receive a burst of measurement data and distributes the processing load to the other processors through the bus for parallel processing. The network is shown in Fig. 6.

Fig. 7 shows the timing diagram for a bus network where there is no control processor and where processors are without front-end processors. For convenience, the originating processor which receives a burst of measurement data and distributes the processing load to the other processors is assigned processor N. At time $t = 0$, the originating processor (processor N) transmits the first fraction of processing load to the processor 1 ($\alpha_1 Z T_{cm}$). When the transmission of the first fraction of processing load is finished, processor N then transmits the second fraction of processing load to processor 2 ($\alpha_2 Z T_{cm}$). In the mean time, processor 1 starts computing the received processing load ($\alpha_1 w_1 T_{cp}$). The process then continues on in the natural way up to N-1st fraction (α_{N-1}). After completion of

N-1st fraction's transmission, processor N starts computing its own fraction of processing load ($\alpha_N w_N T_{cp}$). Naturally, the transmission of Nth fraction ($\alpha_N Z T_{cm}$) is not needed. The equations that represent the finish time of each processor are given by

$$T_1 = \alpha_1 Z T_{cm} + \alpha_1 w_1 T_{cp} \tag{58}$$

$$T_2 = (\alpha_1 + \alpha_2) Z T_{cm} + \alpha_2 w_2 T_{cp} \tag{59}$$

$$T_3 = (\alpha_1 + \alpha_2 + \alpha_3) Z T_{cm} + \alpha_3 w_3 T_{cp} \tag{60}$$

$$\vdots$$

$$T_i = (\alpha_1 + \alpha_2 + \cdots + \alpha_i) Z T_{cm} + \alpha_i w_i T_{cp} \tag{61}$$

$$\vdots$$

$$T_{N-1} = (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_{N-1} w_{N-1} T_{cp} \tag{62}$$

$$T_N = (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_N w_N T_{cp} \tag{63}$$

The fractions of total measurement load should sum to one.

$$\alpha_1 + \alpha_2 + \cdots + \alpha_i + \cdots + \alpha_N = 1 \tag{64}$$

Here one can notice that Eq.(58) through Eq.(63) are the same as Eq.(1) through Eq.(6) in the previous section except the last equations, that is,

$T_N = (\alpha_1 + \alpha_2 + \cdots + \alpha_N) Z T_{cm} + \alpha_N w_N T_{cp}$ with control processor

$T_N = (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_N w_N T_{cp}$ without control processor, without front-end processors

Therefore, the proof to show that $T_1 = T_2 = \cdots = T_{N-1}$ achieves the minimal solution

time is exactly the same as in the previous section. Thus we will look at the case involving $T_1, T_2, \ldots, T_N$.

3.1 Consideration of $T_1, T_2, \ldots,$ and T_N

We will now examine the optimal processing time when all the finish times, $T_1, T_2, \ldots, T_N$, are included. This consideration will include the previous results, namely $T_1 = T_2 = \cdots = T_{N-1}$.

Since the sum of the fractions of total measurement load is one, α_N can be rewritten by

$$\begin{aligned} \alpha_N &= 1 - (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) \\ &= 1 - (1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2})\alpha_1 \end{aligned} \tag{65}$$

where k's are defined as earlier.

Here α_1 has its maximum value when $\alpha_N = 0$:

$$0 \leq \alpha_1 \leq \frac{1}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2}} \tag{66}$$

Then, $T_1, T_2, \ldots, T_N$ can be represented as follows:

$$T_1 = T_2 = \cdots = T_{N-1} = (ZT_{cm} + w_1T_{cp})\alpha_1 \tag{67}$$

$$\begin{aligned} T_N &= (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1})ZT_{cm} + \alpha_N w_N T_{cp} \\ &= (1 - \alpha_N)ZT_{cm} + \alpha_N w_N T_{cp} \\ &= ZT_{cm} + (w_N T_{cp} - ZT_{cm})\alpha_N \end{aligned} \tag{68}$$

Here the condition for load sharing, which is $w_N T_{cp} - ZT_{cm} > 0$, for a bus network where processors are without front-end processors must be satisfied [10]. This is because

if the total communication time of the entire processing load (ZT_{cm}) is longer than the total processing time for the originating processor ($w_N T_{cp}$), then the originating processor (processor N) should not distribute the load and should compute the entire load by itself.

Using the representation of Eq.(65), Eq.(68) can be expressed as a function of α_1.

$$\begin{aligned} T_N &= ZT_{cm} + (w_N T_{cp} - ZT_{cm})[1 - (1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2})\alpha_1] \\ &= w_N T_{cp} - (1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2})(w_N T_{cp} - ZT_{cm})\alpha_1 \end{aligned} \quad (69)$$

Note that T_N has a negative slope. Now $T_1 = T_2 = \cdots = T_{N-1}$ has its maximum value and T_N has its minimum value when α_1 reaches its maximum value, that is,

$$\begin{aligned} \max(T_1 = T_2 = \cdots = T_{N-1}) &= T_1\{\ \alpha_1 = \frac{1}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2}}\ \} \\ &= \frac{1}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2}}(ZT_{cm} + w_1 T_{cp}) \quad (70) \\ \min(T_N) &= T_N\{\ \alpha_1 = \frac{1}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2}}\ \} \\ &= ZT_{cm} \quad (71) \end{aligned}$$

In order for a crossover point to exist between the two lines, Eq.(70) must be greater than Eq.(71), that is,

$$\frac{1}{1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2}}(ZT_{cm} + w_1 T_{cp}) > ZT_{cm} \quad (72)$$

Proof: The above condition can be reduced as follows:

$$\begin{aligned} ZT_{cm} + w_1 T_{cp} &> (1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-2})ZT_{cm} \quad (73) \\ w_1 T_{cp} &> k_1(1 + k_2 + k_2k_3 + \cdots + k_2k_3 \cdots k_{N-2})ZT_{cm} \quad (74) \\ w_1 T_{cp} &> \frac{\alpha_2}{\alpha_1}(1 + \frac{\alpha_3}{\alpha_2} + \frac{\alpha_3}{\alpha_2}\frac{\alpha_4}{\alpha_3} + \cdots + \frac{\alpha_3}{\alpha_2}\frac{\alpha_4}{\alpha_3} \cdots \frac{\alpha_{N-1}}{\alpha_{N-2}})ZT_{cm} \quad (75) \end{aligned}$$

$$\alpha_1 w_1 T_{cp} > (\alpha_2 + \alpha_3 + \cdots + \alpha_{N-1}) Z T_{cm} \quad (76)$$

T_1 and T_{N-1} can be rewritten as

$$T_1 = \alpha_1 Z T_{cm} + \alpha_1 w_1 T_{cp} \quad (77)$$

$$T_{N-1} = (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_{N-1} w_{N-1} T_{cp} \quad (78)$$

Since $T_1 = T_{N-1}$, we can write the following equation:

$$\alpha_1 w_1 T_{cp} = (\alpha_2 + \alpha_3 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_{N-1} w_{N-1} T_{cp} \quad (79)$$

Since $\alpha_{N-1} w_{N-1} T_{cp} > 0$, the above inequality, Eq.(72), is satisfied. □

There thus exists a crossover point across the two lines and the optimal processing time is achieved at that point where $T_1 = T_2 = \cdots = T_N$ as in Fig. 8. Hence the minimal solution time involves all processors stopping their computing at the same time.

Since $T_{N-1} = T_N$ and

$$T_{N-1} = (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_{N-1} w_{N-1} T_{cp} \quad (80)$$

$$T_N = (\alpha_1 + \alpha_2 + \cdots + \alpha_{N-1}) Z T_{cm} + \alpha_N w_N T_{cp} \quad (81)$$

we can express α_N as a function of $\alpha_{N-1}, \alpha_{N-2}, \ldots$,and α_1.

$$\begin{aligned} \alpha_N &= \frac{w_{N-1}}{w_N} \alpha_{N-1} \\ &= k_{N-1} \alpha_{N-1} \\ &= k_{N-1} k_{N-2} \alpha_{N-2} \\ &\vdots \\ &= k_{N-1} k_{N-2} \cdots k_1 \cdot \alpha_1 \end{aligned} \quad (82)$$

where $k_{N-1} = \frac{\alpha_N}{\alpha_{N-1}} = \frac{w_{N-1}}{w_N}$.

Since the sum of α's must be one, α_1 can be obtained by the normalization equation.

$$\begin{aligned} 1 &= \alpha_1 + \alpha_2 + \alpha_3 + \cdots + \alpha_N \\ &= (1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-1})\alpha_1 \end{aligned} \tag{83}$$

Therefore, the optimal values of α_i's that the originating processor N should calculate in order to achieve the minimal solution time can be computed by the following algorithm which is similar to the results of the previous section except $k_{N-1} = \frac{w_{N-1}}{w_N}$ (which appears in a slightly different form in [18]).

$$1) \qquad k_j = \begin{cases} \frac{w_j T_{cp}}{Z T_{cm} + w_{j+1} T_{cp}} & 1 \le j \le N-2 \\ \frac{w_{N-1}}{w_N} & j = N-1 \end{cases} \tag{84}$$

$$\begin{aligned} 2) \qquad \alpha_1 &= [\, 1 + k_1 + k_1k_2 + \cdots + k_1k_2 \cdots k_{N-1} \,]^{-1} \\ &= [\, 1 + \sum_{i=1}^{N-1} (\prod_{j=1}^{i} k_j) \,]^{-1} \end{aligned} \tag{85}$$

$$\begin{aligned} 3) \qquad \alpha_i &= k_1k_2 \cdots k_{i-1} \cdot \alpha_1 \\ &= (\prod_{j=1}^{i-1} k_j) \cdot \alpha_1 \qquad\qquad 2 \le i \le N \end{aligned} \tag{86}$$

4 Alternative Architecture: No Control Processor, Processors with Front-End Processors

It is possible that there may be other types of bus-oriented architectures [10]. An alternative architecture would be the case of a network without control processor and where the processors are equipped with front-end processors for communications off-loading so that the processors can communicate and compute simultaneously. Any single processor can receive a burst of measurement data and distribute the processing load amongst N processors to obtain the benefits of parallel processing.

The proof for this case that to achieve a minimum solution time all processors must finish their processing load at the same time is similar to that in this paper and is the subject of [15]. The algorithm for computing the optimal values of α_i's is the same as in the case for the network where there is a control processor (section 2).

5 Conclusion

Proofs now exist that the minimal finish time for load sharing a divisible job on a bus network and linear daisy chain network [14] involves having all the processors stop at the same time. An open problem is the demonstration of a similar result for tree networks [9].

Acknowledgement

The research in this paper was supported in part of the SDIO/IST and managed by the U.S. Office of Naval Research under grant no. N00014-91-J4063.

References

[1] R.R. Tenney and N.R. Sandell,Jr., "Detection with distributed sensors," *IEEE Transaction on Aerospace and Electronic Systems*, vol. AES-17, pp. 501–510, July 1981.

[2] C.Y. Chong, E. Tse, and S. Mori, "Distributed estimation in networks," presented at the American Control Conference, San Franciso, 1983.

[3] S.H. Bokhari, *Assignment Problems in Parallel and Distributed Computing*, Boston: Kluwer Academic Publishers, 1987.

[4] H.S. Stone, "Multiprocessor scheduling with the aid of network flow algorithms," *IEEE Transaction on Software Engineering*, vol. SE-3, no. 1, pp. 85–93, Jan. 1977.

[5] J. Du and J.Y.T. Leung, "Complexity of scheduling parallel task systems," *SIAM Journal on Discrete Mathematics*, pp. 473–487, Nov. 1989.

[6] J. Blazewicz, M. Drabowski, and J. Weglarz, "Scheduling multiprocessor tasks to minimize schedule length," *IEEE Transactions on Computers*, vol. C-35, pp. 389–398, May 1986.

[7] W. Zhao, K. Ramamritham, and J.A. Stankovic, "Preemptive scheduling under time and resource constraints," *IEEE Transactions on Computers*, vol. C-36, pp. 949–960, Aug. 1987.

[8] Y.C. Cheng and T.G. Robertazzi, "Distributed computation with communication delays," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 24, no. 6, pp. 700–712, Nov. 1988.

[9] Y.C. Cheng and T.G. Robertazzi, "Distributed computation for tree network with communication delays," *IEEE Transactions on Aerospace and Systems*, vol. 26, no. 3, pp. 511–516, May 1990.

[10] S. Bataineh and T.G. Robertazzi, "Distributed computation for a bus networks with communication delays," *Proceedings of the 1991 Conference on Information Sciences and Systems*, The Johns Hopkins University, Baltimore, pp. 709–714, March 1991.

[11] S. Bataineh and T.G. Robertazzi, "Bus oriented load sharing for a network of sensor driven processors," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, no. 5, Sept. 1991.

[12] S. Bataineh and T.G. Robertazzi, "Closed form solutions for bus and tree networks of processors load sharing a divisible job," *SUNY at Stony Brook College of Engineering and Applied Science Technical Report*, no. 627, May 1992. (Available from T. Robertazzi).

[13] S. Bataineh and T.G. Robertazzi, "Ultimate performance limits for networks of load sharing processors," *Proceedings of the 1992 Conference on Information Sciences and Systems*, Princeton, NJ, pp. 794–799, March 1992.

[14] T.G. Robertazzi, "Processor equivalence for load sharing processor daisy chains," *accepted by the IEEE Transactions on Aerospace and Electronic Systems for Oct. 1993 issue.*

[15] J. Sohn and T.G. Robertazzi, "Optimal load sharing for a divisible job on bus network," *SUNY at Stony Brook College of Engineering and Applied Science Technical Report*, no. 644, Oct. 1992. submitted for publication. (Available from T. Robertazzi).

[16] H.J. Kim, G.I. Jee, and J.G. Lee, "Optimal load distribution for tree network processors," submitted for publication.

[17] D. Ghose and V. Mani, "Distributed computation in a linear network: closed form solution and computational techniques," submitted for publication.

[18] V. Bharadwaj, D. Ghose, and V. Mani, "Closed form solutions for optimal processing time in distributed single-level tree networks with communication delays," submitted for publication.

[19] V. Bharadwaj, D. Ghose, and V. Mani, "A new strategy of load distribution in a distributed single-level tree network with communication delays," submitted for publication.

[20] V. Bharadwaj, D. Ghose, and V. Mani, "An efficient load distribution strategy for a distributed linear network of processors with communication delays," submitted for publication.

[21] D. Ghose, and V. Mani, "Distributed computation with communication delays: Asymptotic performance analysis," submitted for publication.

[22] F. Baskett, K.M. Chandy, R.R. Muntz, and F. Palacios, "Open, closed and mixed networks of queues with different classes of customers," *Journal of the ACM*, vol. 22, no. 2, pp. 248–260, April 1975.

Figure Captions

Figure 1. Bus network with a control processor.

Figure 2. Timing diagram for bus network with control processor.

Figure 3. T_1 and T_2 as a function of α_1.

Figure 4. $T_1 = T_2$ and T_3 as a function of α_1.

Figure 5. $T_1 = T_2 = \cdots = T_{i-1}$ and T_i as a function of α_1.

Figure 6. Bus network without control processor.

Figure 7. Timing diagram for bus network without control processor, processors without front-end processors.

Figure 8. $T_1 = T_2 = \cdots = T_{N-1}$ and T_N as a function of α_1.

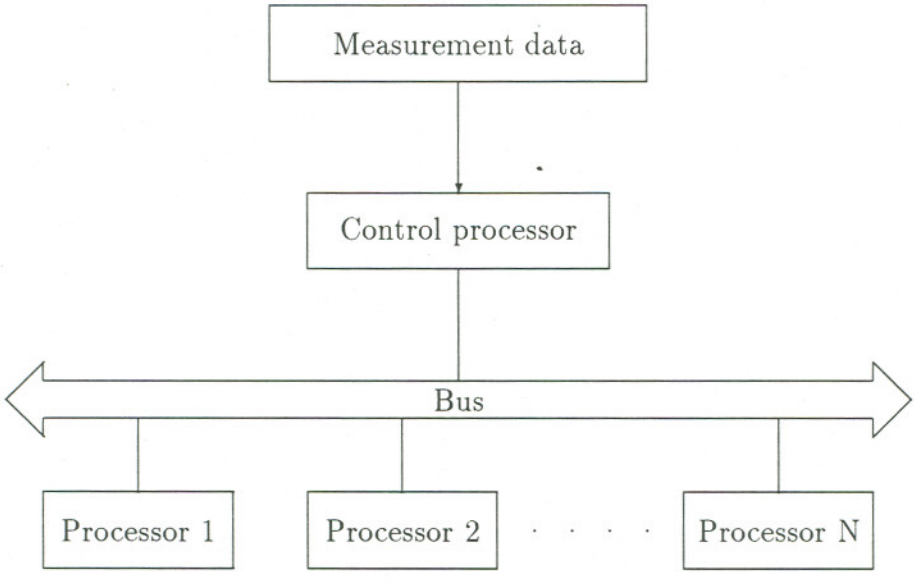

Figure 1. Bus network with a control processor.

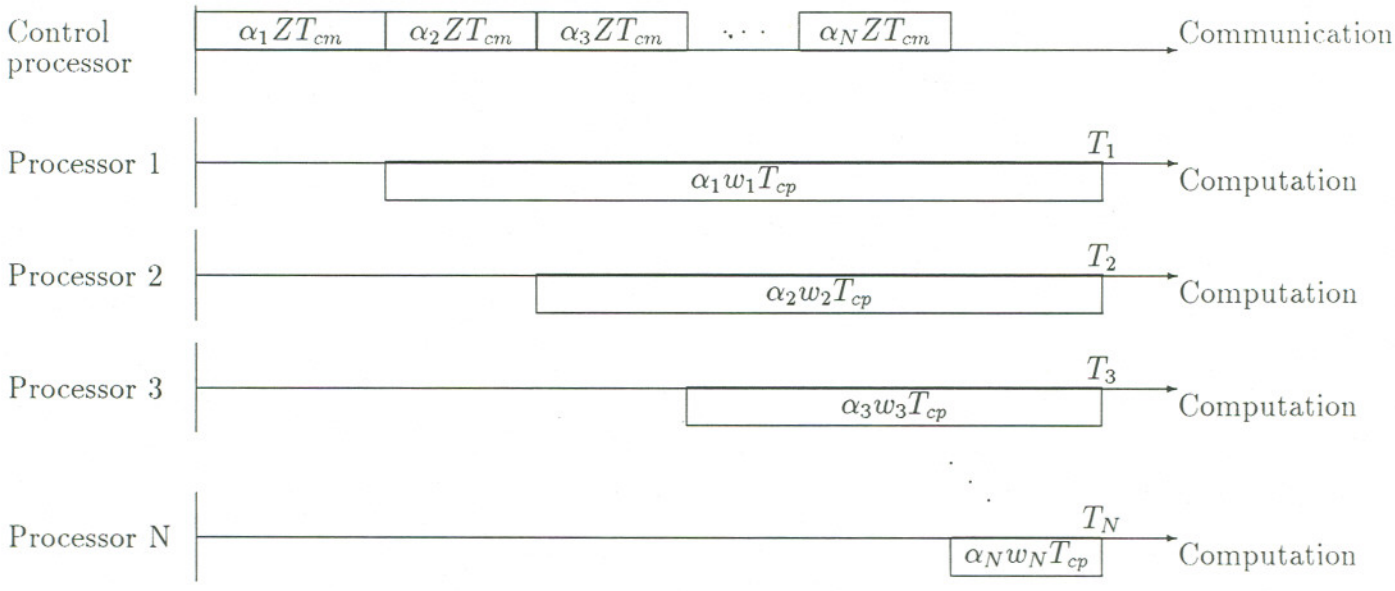

Figure 2. Timing diagram for bus network with control processor.

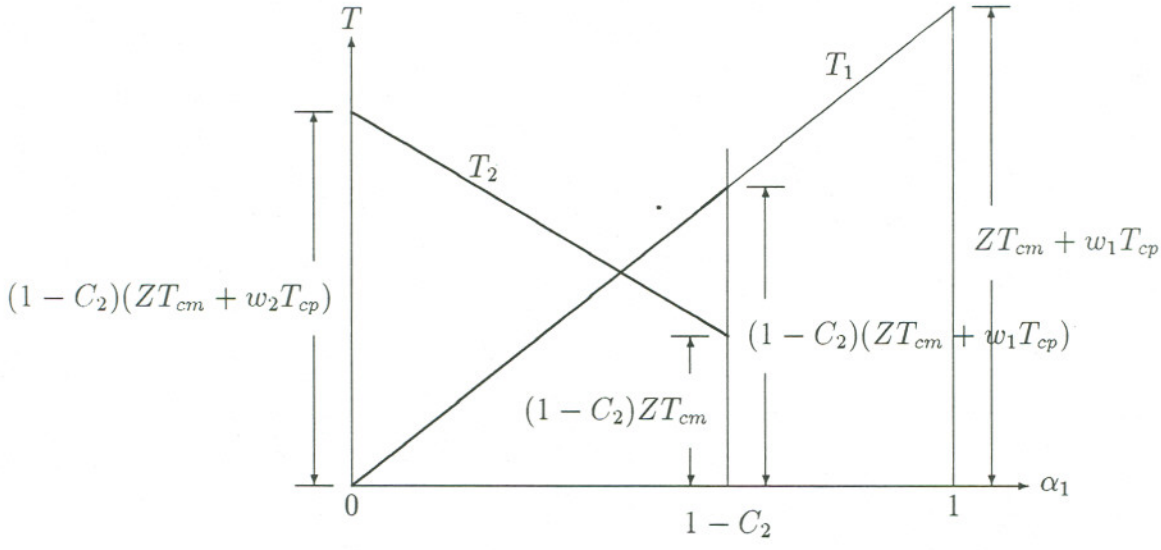

Figure 3. T_1 and T_2 as a function of α_1.

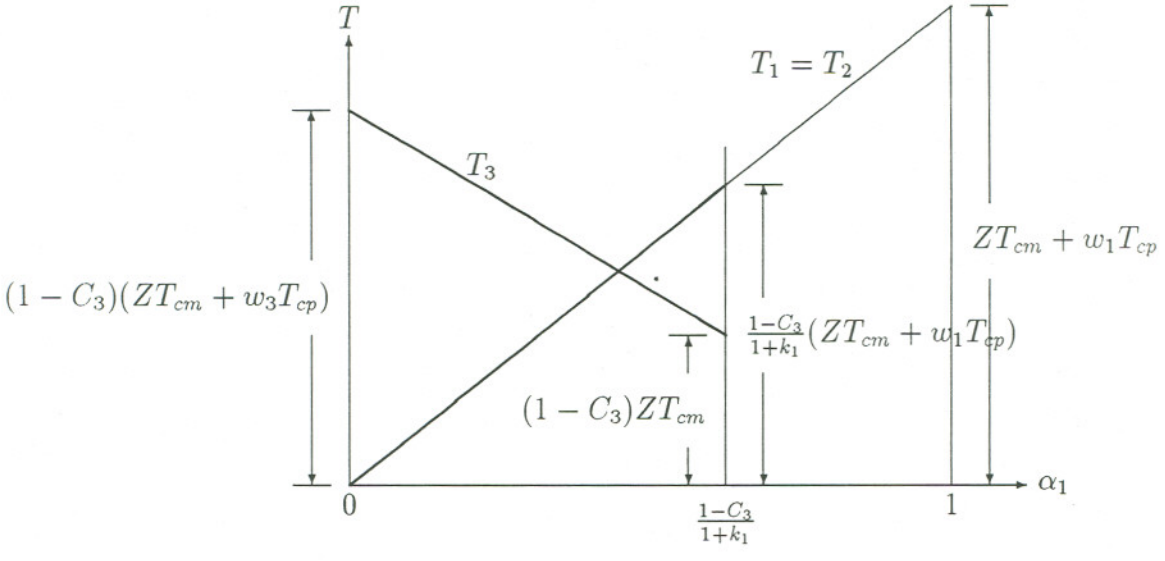

Figure 4. $T_1 = T_2$ and T_3 as a function of α_1.

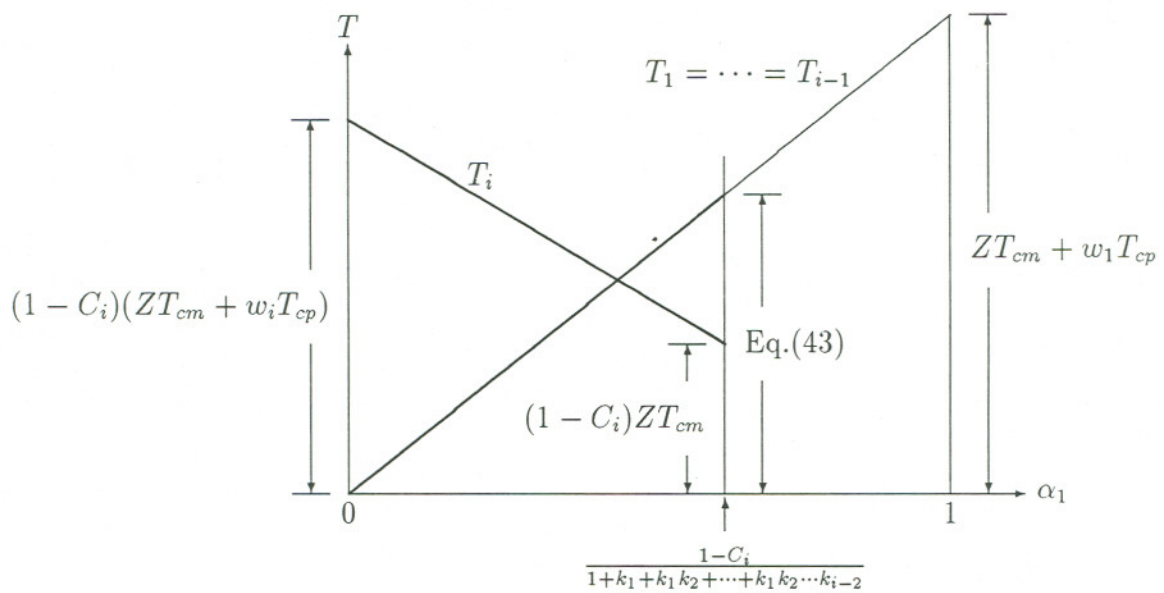

Figure 5. $T_1 = T_2 = \cdots = T_{i-1}$ and T_i as a function of α_1.

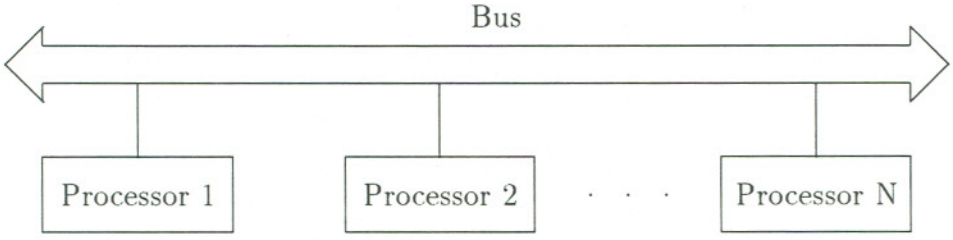

Figure 6. Bus network without control processor.

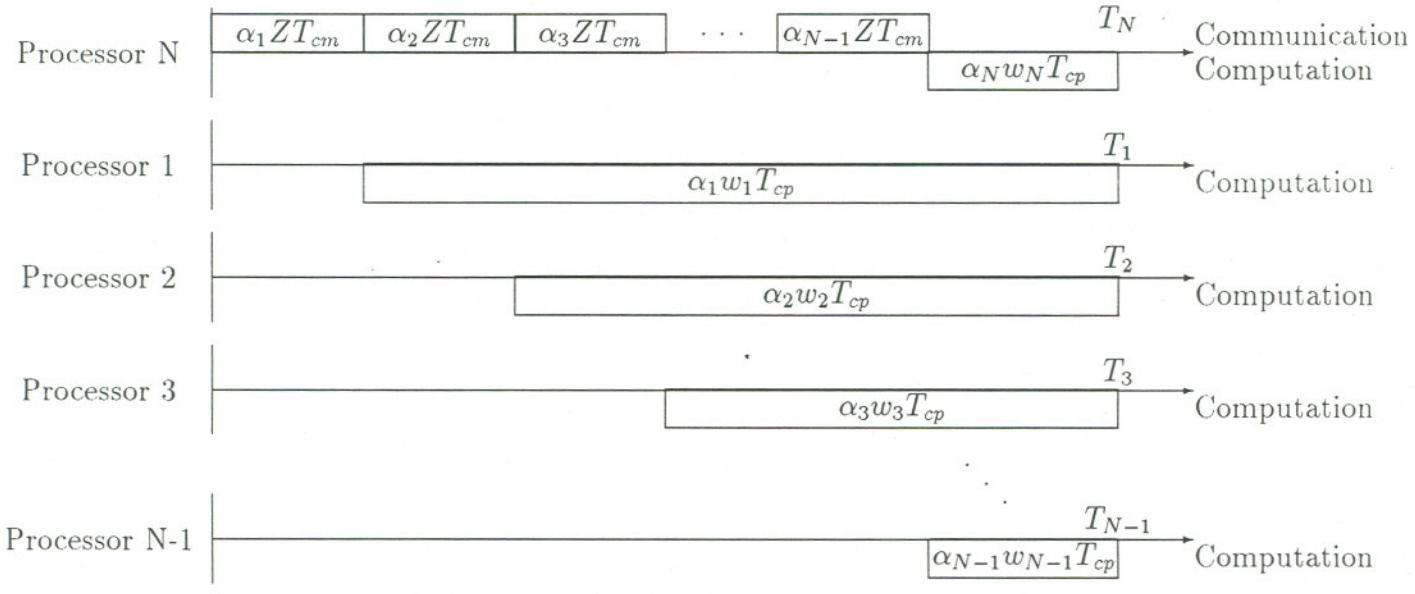

Figure 7. Timing diagram for bus network without control processor, processors without front-end processors.

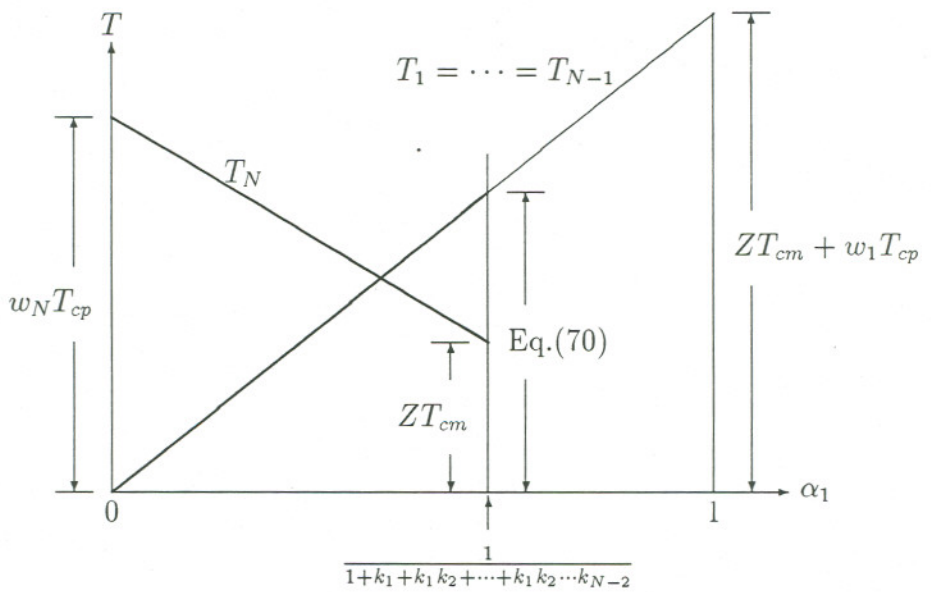

Figure 8. $T_1 = T_2 = \cdots = T_{N-1}$ and T_N as a function of α_1.