Closed Form Solutions for Bus and Tree Networks of Processors Load Sharing a Divisible Job

Sameer Bataineh, Te-Yu Hsiung, and Thomas G. Robertazzi, *Senior Member, IEEE*

***Abstract*—Optimal load allocation for load sharing a divisible job over processors interconnected in either a bus or a tree network is considered. The processors are either equipped with front-end processors or not so equipped. Closed form solutions for the minimum finish time and the optimal data allocation for each processor are obtained. The performance of large symmetric tree networks is examined by aggregating the component links and processors into a single equivalent processor. This allows an easy examination of large tree networks. In addition, it becomes possible to find a closed form solution for the optimal amount of data that is to be assigned to each processor in the tree network in order to achieve the minimum finish time.**

***Index Terms*— Load sharing, load balancing, divisible job, multiprocessors.**

I. Introduction

A DIVISIBLE job is a job that can be arbitrarily split in a linear fashion among a number of processors. Applications include the processing of very large data files such as occurs in signal and image processing, Kalman filtering and experimental data processing. Most work to date on load sharing has involved indivisible jobs, that is jobs that can only be assigned to a single processor [6]–[13]. Only a small amount of recent work has examined jobs that can be assigned to multiple processors [17]–[19].

Load sharing of a divisible job among a number of processors which are connected together by an interconnection network such as a tree network and a bus network was examined in detail in [2]–[4], [15], [20], [21], respectively. A set of recursive equations were developed to calculate the optimal fractions of the load that have to be assigned to each processor in the network in order to achieve the minimum finish time. The processors were assumed to have different speeds.

In this paper, the processors are all assumed to have the same speed. This enables one to find a closed form equation by which one can calculate the optimal fractions of the load that has to be assigned to each processor in the network in order to achieve the minimum finish time. Moreover, compact simple expressions for the minimum finish time for different networks are also obtained.

This paper is organized as follows. In the second section, bus oriented networks are examined while in the third section, we examine tree networks. Performance evaluation curves are presented in Section IV. Section V contains the conclusion.

Manuscript received May 29, 1992; revised April 10, 1993. This work was supported by the BMDO/IST and managed by the Office of the Naval Research under grant No. N00014-91-J4063.

S. Bataineh is with the Department of Electrical Engineering, Jordan University of Science and Technology, Irbid, P.O. Box 3030, Jordan.

T.-Y. Hsiung and T. G. Robertazzi are with the Department of Electrical Engineering, SUNY at Stony Brook, Stony Brook, NY 11794 USA; e-mail: tom@sbee.sunysb.edu.

IEEE Log Number 9401636.

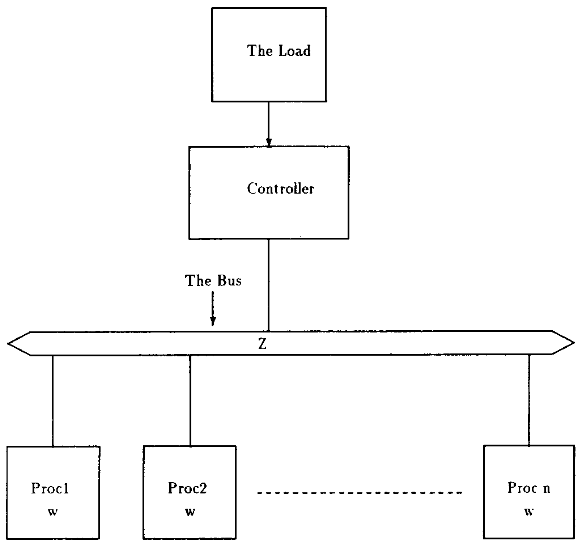

Fig. 1. Bus network with controller.

II. Bus Network

A. Architecture 1: Bus 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, $i = 1, 2, \cdots, n$, is proportional to the amount of measurement data that has to be assigned to that processor. Each processor begins to compute its share of the load once the share has been completely received. Bus propagation delay is ignored. 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 originating processor.

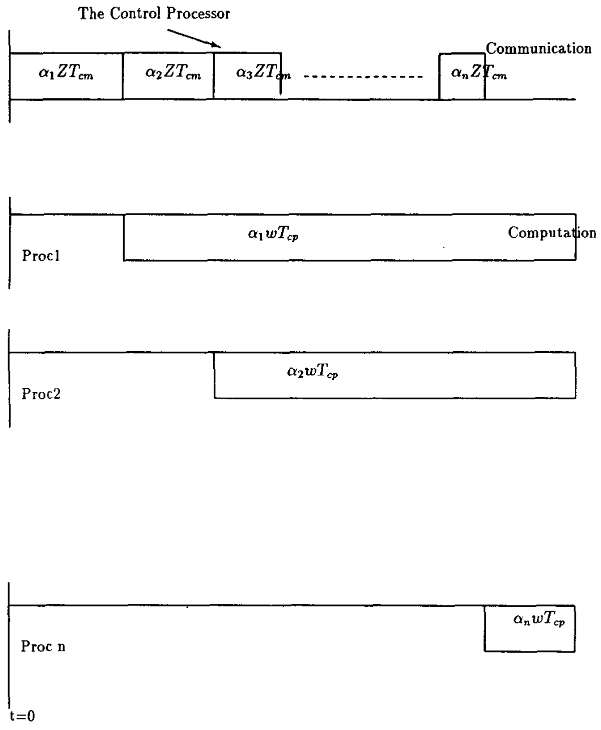

Fig. 2. Timing diagram for bus network with controller.

$T_{\rm cp}$: The time that it takes the ith processor to process the entire load when $w = 1$. The time for arbitrary w is $wT_{\rm cp}$.

$T_{\rm cm}$: The time that it takes the control processor to transmit all the measurement data over the bus when $Z = 1$. The time for arbitrary Z is $ZT_{\rm cm}$.

w : A constant that is inversely proportional to the computation speed of any processor in the network. Any processor can process the entire load in time $wT_{\rm cp}$.

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 $ZT_{\rm cm}$.

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, $i = 1, 2, \cdots, n$. 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, Fig. 2, shows that at $t = 0$, the 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

$$T_1 = \alpha_1 ZT_{\rm cm} + \alpha_1 wT_{\rm cp} \tag{2.2}$$
$$T_2 = (\alpha_1 + \alpha_2)ZT_{\rm cm} + \alpha_2 wT_{\rm cp} \tag{2.3}$$
$$T_3 = (\alpha_1 + \alpha_2 + \alpha_3)ZT_{\rm cm} + \alpha_3 wT_{\rm cp} \tag{2.4}$$
$$T_4 = (\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4)ZT_{\rm cm} + \alpha_4 wT_{\rm cp} \tag{2.5}$$
$$\vdots$$
$$T_n = (\alpha_1 + \alpha_2 + \cdots + \alpha_n)ZT_{\rm cm} + \alpha_n wT_{\rm cp}. \tag{2.6}$$

The fractions of the total measurement load should sum to one

$$\alpha_1 + \alpha_2 + \alpha_3 + \cdots + \alpha_n = 1. \tag{2.7}$$

The important point of interest is the minimum finish time, T_f which occurs when all processors stop at the same time [3], [14], [16]. Intuitively this can be proved by contradiction; if the processors do not all stop at the same time some will be idle while others are busy and the finish time can be improved by transfering load to the idle processors. Based on this we can write the following set of equations:

$$\alpha_{n-1} = \alpha_n r \tag{2.8}$$
$$\alpha_{n-2} = \alpha_{n-1} r \tag{2.9}$$
$$\vdots$$
$$\alpha_2 = \alpha_3 r \tag{2.10}$$
$$\alpha_1 = \alpha_2 r \tag{2.11}$$

where $r = \frac{(wT_{\rm cp} + ZT_{\rm cm})}{wT_{\rm cp}}$.

Here α_i is solved for by equating T_i to T_{i+1}. Using the above set of equations, we can now write α's as a function of only α_n and r.

$$\alpha_i = \alpha_n r^{n-i} \tag{2.12}$$

where $i = 1, 2, 3, \cdots, n-1$.

Using (2.7) and (2.12), one can write

$$\alpha_n(r^{n-1} + r^{n-2} + r^{n-3} + \cdots + r + 1) = 1 \tag{2.13}$$
$$\alpha_n\left(\sum_{i=1}^{n} r^{n-i}\right) = 1 \tag{2.14}$$
$$\alpha_n\left(\frac{r^n - 1}{r - 1}\right) = 1. \tag{2.15}$$

This implies that:

$$\alpha_n = \frac{r - 1}{r^n - 1}. \tag{2.16}$$

Knowing the value of α_n, the control processor can simply compute the amount of data that has to be assigned to each processor in the network by using (2.12).

The minimum finish time is given by (from (2.2), (2.12), (2.16)):

$$T_{f1} = (ZT_{\rm cm} + wT_{\rm cp})\left(\frac{r^n - r^{n-1}}{r^n - 1}\right) \tag{2.17}$$

and the maximum throughput is

$$\gamma = \frac{1}{T_{f1}}. \tag{2.18}$$

As mentioned earlier, with these closed form solutions one can do some mathematical operations to find some parameters of interest. For instance, we know that as n approaches ∞, $T_{f1} \longrightarrow ZT_{\text{cm}}$ [3], [4]. In the following, we will prove this result analytically:
As $n \longrightarrow \infty$ $(\frac{r^n - r^{n-1}}{r^n - 1}) \longrightarrow \frac{r-1}{r}$.
Substituting the definition of r in the above and substituting the result back in T_{f1} result in

$$T_{f1} = ZT_{\text{cm}}$$

B. Architecture 2: No Control Processor, Processors Without Front-End Processors

The network architecture that is discussed in this section is similar to that discussed in the previous one except for the fact that there is no control processor. Each of n homogeneous processors in the network also 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 originate 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. Bus propagation delay is neglected. The timing diagram of the system is plotted in Fig. 3. Between $t = 0$ and $\alpha_2 ZT_{\text{cm}}$, none of the processors performs computation, the first processor communicates data to the second processor and processors $3, 4, 5, \cdots, n$ are all idle. In general, in the period between $t = 0$ and $t = (\alpha_2 + \alpha_3 + \cdots + \alpha_i)ZT_{\text{cm}}$, only $(i-2)$ processors perform computation, $(n-i)$ processors are idle, $i = 2, 3, \cdots, n$, and two are communicating. This facts serves to increase the minimum finish time.

In the following, we will use the same definitions for α_i, w, Z, T_i, T_{cp}, and T_f as in previous section. T_{cm} is defined slightly differently as following:

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

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

$$T_1 = (1 - \alpha_1)ZT_{\text{cm}} + \alpha_1 wT_{\text{cp}} \tag{2.19}$$
$$T_2 = \alpha_2 ZT_{\text{cm}} + \alpha_2 wT_{\text{cp}} \tag{2.20}$$
$$T_3 = (\alpha_2 + \alpha_3)ZT_{\text{cm}} + \alpha_3 wT_{\text{cp}} \tag{2.21}$$
$$T_4 = (\alpha_2 + \alpha_3 + \alpha_4)ZT_{\text{cm}} + \alpha_4 wT_{\text{cp}} \tag{2.22}$$
$$\vdots$$
$$T_n = (1 - \alpha_1)ZT_{\text{cm}} + \alpha_n wT_{\text{cp}}. \tag{2.23}$$

The fractions of the total measurement load should sum to one

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1. \tag{2.24}$$

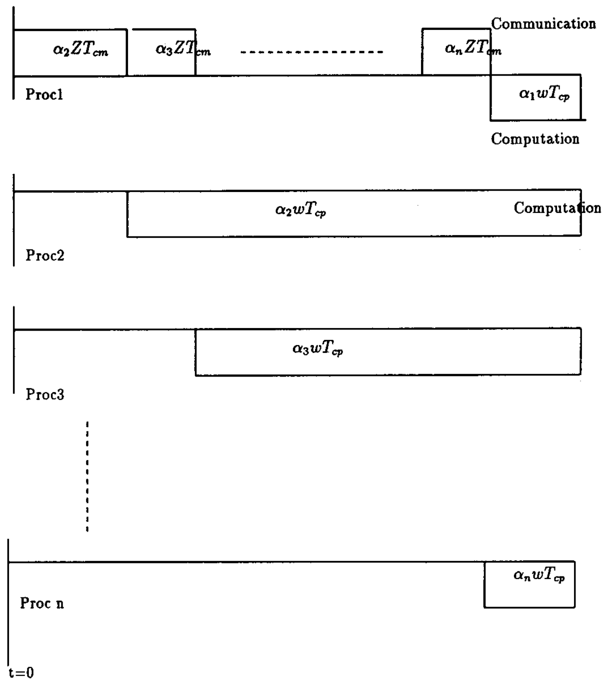

Fig. 3. Timing diagram for homogeneous bus network without front-end processors.

As mentioned earlier, that the minimum finish time is achieved when all processors stop at the same time [3], [14], [16], that is when:

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

The originating processor should calculate the optimal values of α's. To find these values, one should first write the following set of equations:

$$\alpha_{n-1} = \alpha_n r \tag{2.25}$$
$$\vdots$$
$$\alpha_3 = \alpha_4 r \tag{2.26}$$
$$\alpha_2 = \alpha_3 r \tag{2.27}$$
$$\alpha_1 = \alpha_n \tag{2.28}$$

where $r = \frac{wT_{\text{cp}} + ZT_{\text{cm}}}{wT_{\text{cp}}}$
Here α_i is solved for by setting

$$T_i = T_{i+1}, \quad \text{for } i = 2, 3, \cdots, n-1$$
$$T_i = T_n, \quad \text{o.w.}$$

From the above equations the optimal values of α's can be written in terms of α_n and r as follows:

$$\alpha_j = \begin{cases} \alpha_n r^{n-j}, & \text{if } j = 2, 3, \cdots, n-1 \\ \alpha_n, & \text{if } j = 1. \end{cases} \tag{2.29}$$

It is apparent from the above equation that if the optimal value of α_n can be found, the optimal values of other α's can be readily computed using equation (2.29). Using (2.24) and (2.29), one can find the optimal value of α_n in terms of r as follows:

$$\alpha_n(1 + r^{n-2} + r^{n-3} + \cdots + r + 1) = 1 \tag{2.30}$$

$$\alpha_n\left(\sum_{i=1}^{n} r^{n-i} + 1 - r^{n-1}\right) = 1 \tag{2.31}$$

$$\alpha_n = \left(\frac{r-1}{r^{n-1} + r - 2}\right). \tag{2.32}$$

From (2.19) the minimum finish time function, T_{f_2}, for this network architecture, is given by

$$T_{f_2} = ZT_{\text{cm}} + \left(\frac{r-1}{r^{n-1} + r - 2}\right)(wT_{\text{cp}} - ZT_{\text{cm}}) \tag{2.33}$$

and the maximum throughput(γ) is

$$\gamma = \frac{1}{T_{f_2}}. \tag{2.34}$$

Conditions when it is economical to distribute load are discussed in [3], [16].

C. Architecture 3: No Control Processor, Processors With Front-End Processors

The network architecture 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 a front-end processor for communicating off-loading. That is, with the inclusion of front-end processors, each processor may communicate and compute at the same time. The load may originate at any 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. The timing diagram of the system is plotted in Fig. 4. Between $t = 0$ and $t = \alpha_2 ZT_{\text{cm}}$ the first processor computes its share of the load and communicates with the second processor. All other processors, processors $3, 4, 5 \cdots, n$, are idle. In general, in the period of between $t = 0$ and $t = (\alpha_2 + \alpha_3 + \cdots \alpha_i)ZT_{\text{cm}}$, $(n-i)$ processors would be idle, $(i-1)$ processors perform computation; $i = 2, 3, 4, \cdots, n$, and one is communicating. In the following we will use the same definitions for α_i, w, T_{cp}, Z, T_{cm}, T_i and T_f as in the previous section.

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

$$T_1 = \alpha_1 wT_{\text{cp}} \tag{2.35}$$

$$T_2 = \alpha_2 ZT_{\text{cm}} + \alpha_2 wT_{\text{cp}} \tag{2.36}$$

$$T_3 = (\alpha_2 + \alpha_3)ZT_{\text{cm}} + \alpha_3 wT_{\text{cp}} \tag{2.37}$$

$$T_4 = (\alpha_2 + \alpha_3 + \alpha_4)ZT_{\text{cm}} + \alpha_4 wT_{\text{cp}} \tag{2.38}$$

$$\vdots$$

$$T_n = (\alpha_2 + \alpha_3 + \cdots + \alpha_n)ZT_{\text{cm}} + \alpha_n wT_{\text{cp}}. \tag{2.39}$$

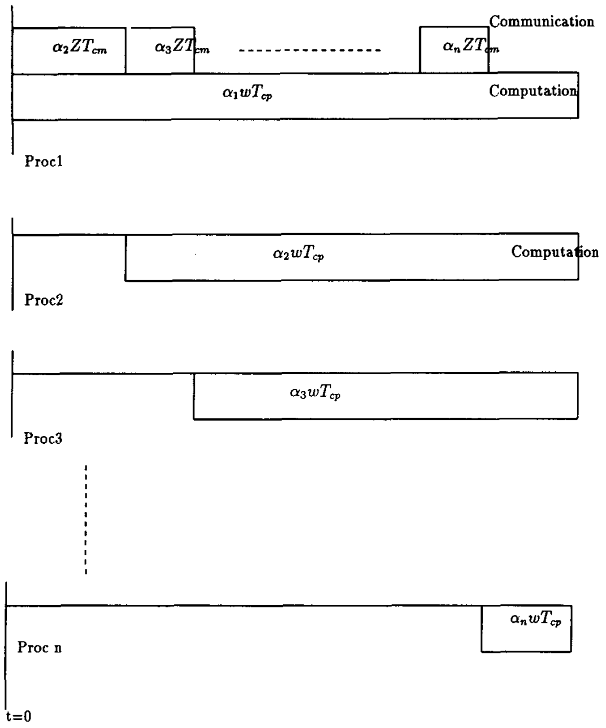

Fig. 4. Timing diagram for homogeneous bus network with front-end processors.

The fractions of the total measurement load should sum to one:

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1. \tag{2.40}$$

The objective in analyzing the above equations is to compute the minimum finish time and compare it with the results that was obtained in the previous sections. The minimum finish time would be achieved when all processors stop at the same time, that is when:

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

[4], [14], [16].

The optimal values of α's that the originating processor should calculate in order to achieve the minimum finish time can be computed by finding first the following set of equations:

$$\alpha_{n-1} = \alpha_n r \tag{2.41}$$

$$\vdots$$

$$\alpha_3 = \alpha_4 r \tag{2.42}$$

$$\alpha_2 = \alpha_3 r \tag{2.43}$$

$$\alpha_1 = \alpha_2 r \tag{2.44}$$

$$\tag{2.45}$$

where $r = \frac{wT_{\text{cp}} + ZT_{\text{cm}}}{wT_{\text{cp}}}$. Here α_i is solved for by equating T_i to T_{i+1}. From the above equations, the optimal values of

$\alpha's$ can be obtained in terms of α_n and r as shown in the following equation:

$$\alpha_i = \alpha_n r^{n-i} \tag{2.46}$$

where $i = 1, 2, 3, \cdots, n-1$.

Again, as before, using (2.40) and (2.46), one can find α_n in terms of r. The steps to do that are presented in the following equations [5]:

$$\alpha_n(r^{n-1} + r^{n-2} + r^{n-3} + \cdots + r + 1) = 1 \tag{2.47}$$

$$\alpha_n\left(\sum_{i=1}^{n} r^{n-i}\right) = 1 \tag{2.48}$$

$$\alpha_n\left(\frac{r^n - 1}{r - 1}\right) = 1 \tag{2.49}$$

$$\alpha_n = \frac{r-1}{r^n - 1}. \tag{2.50}$$

Knowing the optimal value of α_n, the originating processor can now simply compute all other optimal values of α's by using equation (2.46). The minimum finish time function, T_{f_3}, can be calculated from (2.35) [5]:

$$T_{f_3} = wT_{\text{cp}}\frac{r^{n-1}(r-1)}{r^n - 1} \tag{2.51}$$

and the maximum throughput(γ) is

$$\gamma = \frac{1}{T_{f_3}}. \tag{2.52}$$

III. Tree Network

A. Introduction

Consider a tree network of communicating processors as depicted in Fig. 5. In the tree we have three types of nodes (processors): root, intermediate and terminal nodes. Each tree has one root node that originates the load. An intermediate node can be viewed as a parent of lower level nodes with which it has a direct connection. Also it is a child of an upper level node with which it has a direct connection. The terminal nodes can only be children nodes. The kind and the number of levels in a particular tree determine its size, that is the total number of nodes in that tree. The kind of a tree is determined by the number of nodes that a parent node has. A parent in a "binary" tree would have two children. The root is assumed to be level 0 and its children would be in level 1 and so on. The lowest level is $N-1$. Every processor can only communicate with it's children processors and parent processor.

In this section, we will discuss two types of trees. One is where processors are equipped with front-end processors. Therefore, communication and computation can take place in each processor at the same time. In the second type of tree, processors do not have front-end processors. That is, processors can either communicate or compute but not do both at the same time.

In [2], a finite tree, where processors have different speeds, for the above two cases was discussed. However closed form solution for the minimum finish time were not presented.

In this paper, the processors in the tree are assumed to have the same computational speed, $\frac{1}{w}$. The communication speed between a parent processor and each of its children is also assumed to have the same value, $\frac{1}{Z}$. This assumption enables us to collapse the tree into one equivalent node that preserves the same characteristics as the original tree. This allows an easy examination of large tree networks. In addition, it becomes possible to find a closed form solution for the optimal amount of data that is to be assigned to each processor in order to achieve the minimum finish time and also to find a numerical solution to the minimum finish time..

In the following we will use the same definitions for T_{cp}, T_{cm} and w, as in the previous section; however, Z is defined as follows:

Z : A constant that is inversely proportional to the channel speed between a parent processor and it's child. The entire load can be transmitted over the channel in time ZT_{cm}.

B. The Tree Network With No Front-End Processors

To collapse the whole tree in Fig. 5 into one equivalent node we start from the terminal nodes (the last level in the tree, level $N-1$) and move up to the root processor(the first level in the tree, level 0). On our way up, every parent processor and its children will be replaced by one equivalent processor. The process will continue until the root processor and its children are replaced by one equivalent processor. In this aggregation process, only two cases are possible: the first case occurs at the last two levels level where all of the processors have the same speed as shown in Fig. 6; the second case occurs for the children at level k and their parents at level $k-1$, $k = 1, 2, \cdots, N-2$, where all processors, except the parent, have the same speed as depicted in Fig. 7. In the following, we will discuss analytically the two cases.

The timing diagram of the first case is the same as the bus network timing diagram discussed in Section II-B and depicted in Fig. 3 and so we can use the results obtained there to get an expression for w_{eqt} which is stated below. Here, w_{eqt} is a constant that is inversely proportional to the speed of an equivalent processor that replaces all the processors in Fig. 6 and preserves the same characteristics of the original system.

$$w_{\text{eqt}} = \frac{1}{T_{\text{cp}}}\left(ZT_{\text{cm}} + \left(\frac{r_t - 1}{r_t^{n-1} + r_t - 2}\right)(wT_{\text{cp}} - ZT_{\text{cm}})\right) \tag{3.1}$$

where

$$r_t = \frac{wT_{\text{cp}} + ZT_{\text{cm}}}{wT_{\text{cp}}}$$

This equation is obtained by equating (2.33) and $w_{\text{eqt}}T_{\text{cp}}$. Note that in order for load sharing to produce a net savings the right most parenthesis in (3.1) must be positive [3].

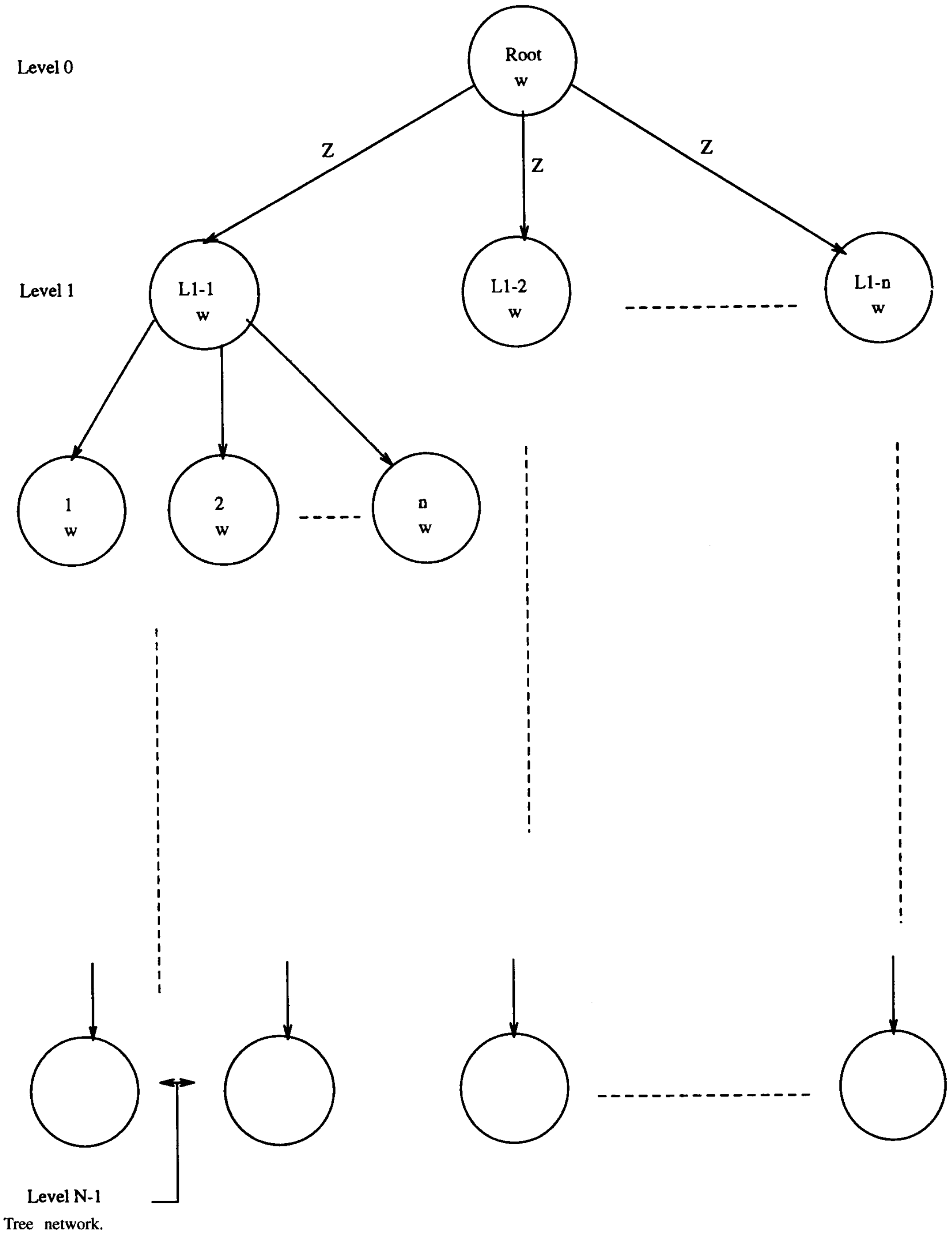

Fig. 5. Tree network.

The timing diagram of the second case, shown in Fig. 8, shows that this is the same as the bus network discussed in subsection (2.2) where all processors except the first have the same speed. The time that takes each processor to process its share is computed by the following set of equations:

$$T_1 = (1 - \alpha_1)ZT_{\rm cm} + \alpha_1 wT_{\rm cp} \quad (3.2)$$

$$T_2 = \alpha_2 ZT_{\rm cm} + \alpha_2 w_{\rm eq}T_{\rm cp} \quad (3.3)$$

$$T_3 = (\alpha_2 + \alpha_3)ZT_{\rm cm} + \alpha_3 w_{\rm eq}T_{\rm cp} \quad (3.4)$$

$$T_4 = (\alpha_2 + \alpha_3 + \alpha_4)ZT_{\rm cm} + \alpha_4 w_{\rm eq}T_{\rm cp} \quad (3.5)$$

$$\vdots$$

$$T_n = (1 - \alpha_1)ZT_{\rm cm} + \alpha_n w_{\rm eq}T_{\rm cp}. \quad (3.6)$$

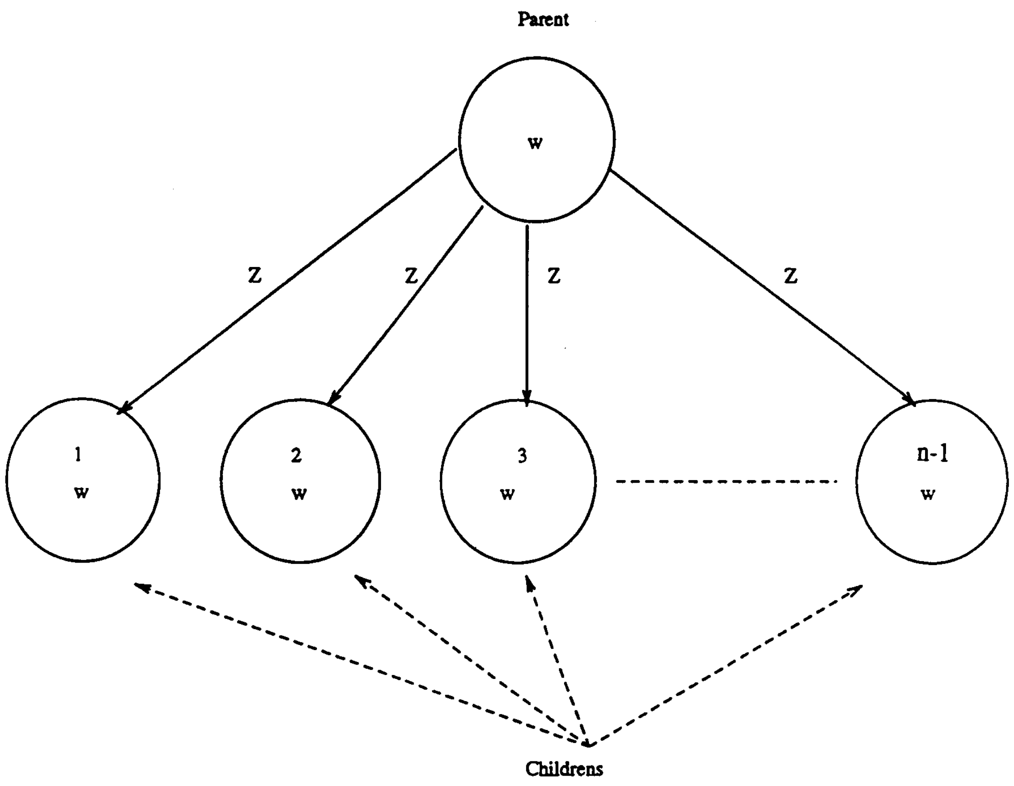

Fig. 6. A subtree where the children are all terminal nodes in the original tree.

The fractions of the total measurement load should sum to one

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1. \tag{3.7}$$

The optimal values of $\alpha's$ that has to be assigned to each processor in order to achieve the minimum finish time, based on all processors stopping at the same time, is given by the following set of equations:

$$\alpha_{n-1} = \alpha_n r_i \tag{3.8}$$
$$\alpha_{n-2} = \alpha_{n-1} r_i \tag{3.9}$$
$$\vdots$$
$$\alpha_3 = \alpha_4 r_i \tag{3.10}$$
$$\alpha_2 = \alpha_3 r_i \tag{3.11}$$
$$\alpha_1 = \alpha_n c \tag{3.12}$$

where $r_i = \frac{w^i_{\text{eq}} T_{\text{cp}} + Z T_{\text{cm}}}{w^i_{\text{eq}} T_{\text{cp}}}$ and $c^i = \frac{w^i_{\text{eq}}}{w}$.

Here i indicates the level of children nodes being considered. It should be noted that, to ac hieve the minimum finish time, α_i is solved for by equating T_i to T_{i+1}, and α_1 is solved for by equating T_1 to T_n [3], [4]. The equations can be written in terms of of α_n, r_i, and c^i as follows:

$$\alpha_j = \begin{cases} \alpha_n r^{n-j}, & \text{if } j = 2, 3, \cdots, n-1, \\ \alpha_n c^i, & \text{if } j = 1. \end{cases} \tag{3.13}$$

Using (3.7) and (3.13) α_n can be found as a function of r_i and c.

$$\alpha_n = \frac{r_i - 1}{c^i(r_i - 1) + r_i^{n-1} - 1}. \tag{3.14}$$

Now all other optimal values of $\alpha's$ can be computed using (3.13) Since $\alpha_1 = \alpha_n c^i$, α_1 can be expressed in terms of r_i and c^i as follows:

$$\alpha_1 = \frac{r_i - 1}{(r_i - 1) + \frac{1}{c^i}(r_i^{n-1} - 1)}. \tag{3.15}$$

We now equate (3.2) to $w_{\text{eqi}} T_{\text{cp}}$ in order to find w_{eqi}, a constant that is inversely proportional to the speed of an "equivalent" processor that will replace all processors in Fig. 7 and preserves the same characteristics as the original system. Note again that for load sharing to produce a net savings the parenthesis term in (3.16) must be positive.

$$w_{\text{eqi}} = Z\rho + \alpha_1(w - Z\rho), \tag{3.16}$$

where $\rho = \frac{T_{\text{cm}}}{T_{\text{cp}}}$.

Substituting the value obtained for α_1 in the above equation, we find that:

$$w_{\text{eqi}} = Z\rho + \frac{r_i - 1}{(r_i - 1) + \frac{1}{c^i}(r_i^{n-1} - 1)}(w - Z\rho). \tag{3.17}$$

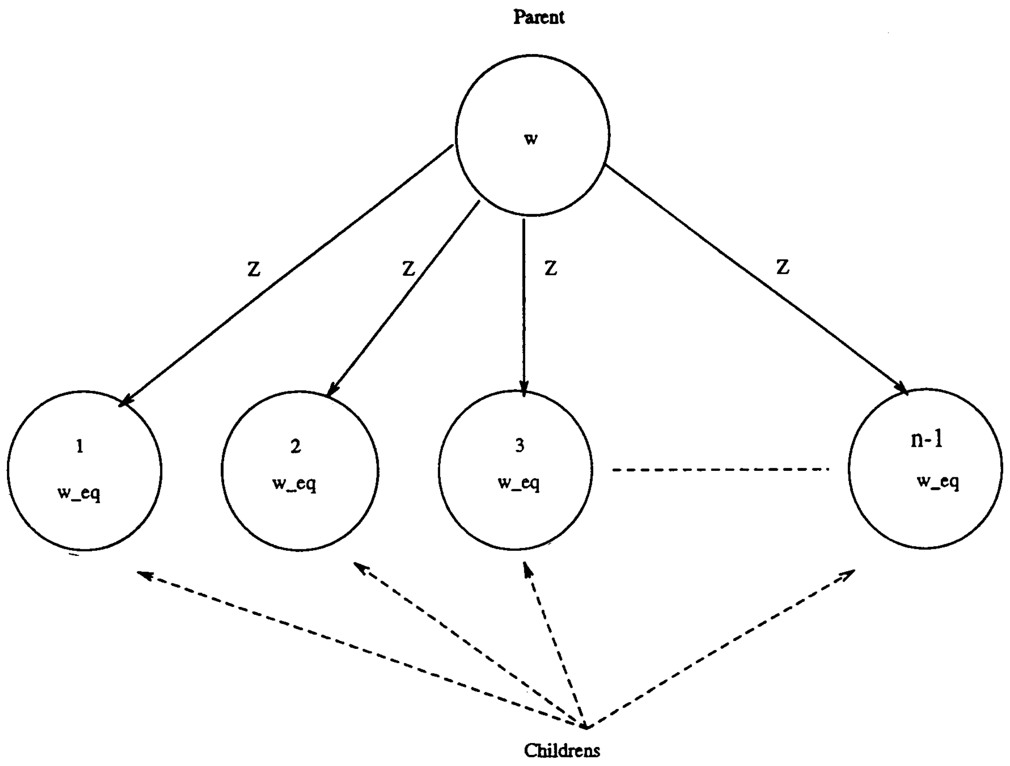

Fig. 7. A reduced case where the children are parents in the original tree.

Starting at level $N-1$, one can use equation (3.1) to reduce the tree in Fig. 5 by one level and then move up one level. Starting from the subtrees whose children are at level $N-2$ and up to the root processor one uses equation (3.17) to find $w_{\text{eq}_{\text{total}}}$. Here $w_{\text{eq}_{\text{total}}}$ is a constant that is inversely proportional to the speed of an "equivalent" processor that will replace the whole tree in Fig. 5 while preserving the same characteristics as the original system. Computing $w_{\text{eq}_{\text{total}}}$, the minimum finish time T_{ftnf} can be written as follows:

$$T_{\text{ftnf}} = T_{\text{cp}} w_{\text{eq}_{\text{total}}} \tag{3.18}$$

and the maximum throughput is

$$\gamma = \frac{1}{T_{\text{ftnf}}}. \tag{3.19}$$

C. The Tree Network With Front-End Processors

This subsection is similar to the previous one except for the fact that now all the processors in the tree possess front-end processors. That is, each processor can communicate and compute at the same time. This fact will help to reduce the finish time. We will proceed as in the previous subsection and collapse the whole tree in Fig. 5 into one equivalent node. We start from the terminal nodes(the last level in the tree, level $N-1$) and move up to the root processor (the first level in the tree, level 0). Similarly we will encounter two cases in our aggregation process: the first case occurs at the last two levels where all processors have the same speed as shown in Fig. 6; the second case occurs for the children at level k and their parents at level $k-1$, $k = 1, 2, \cdots, N-2$, where all processors, except the parent, have the same speed as depicted in Fig. 7. In the following, we will discuss analytically the two cases.

The timing diagram of case one is the same as the bus network timing diagram discussed in Section II-C and depicted in Fig. 4. The results there can be used to obtain an expression for w_{eqt} which is stated below. Here, w_{eqt} is a constant that is inversely proportional to the speed of an equivalent processor that replaces all the processors in Fig. 6 and preserves the characteristics of the original system.

$$w_{\text{eqt}} = w \frac{r_t^{n-1}(r_t - 1)}{r_t^n - 1} \tag{3.20}$$

where

$$r_t = \frac{wT_{\text{cp}} + ZT_{\text{cm}}}{wT_{\text{cp}}}.$$

This equation is obtained by equating (2.51) and $w_{\text{eqt}}T_{\text{cp}}$.

The timing diagram of the second case, shown in Fig. 9, shows that this is the same as the bus network discussed in subsection (2.3) where all processors except the the first have the same speed. The time that takes each processor to process

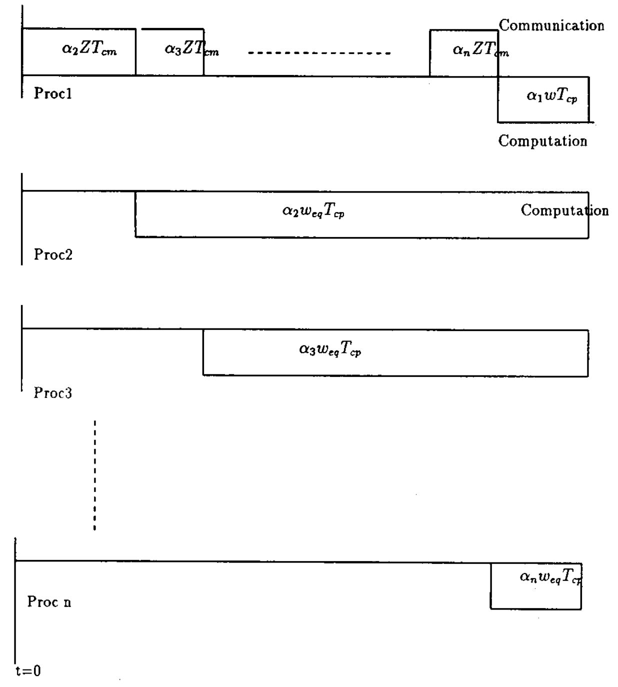

Fig. 8. Timing diagram for second case subtree of tree network with no front-end processors.

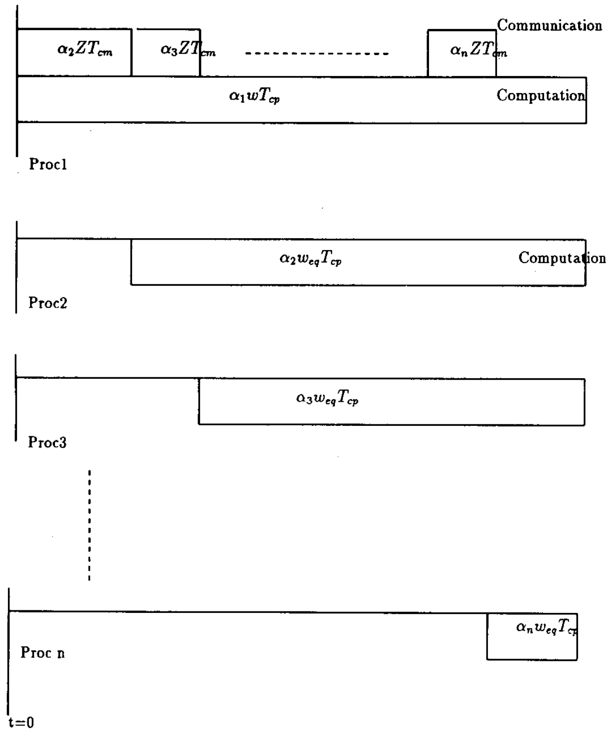

Fig. 9. Timing diagram for second case subtree of tree network with front-end processors.

its share is computed by the following set of equations:

$$T_1 = \alpha_1 wT_{\text{cp}} \tag{3.21}$$
$$T_2 = \alpha_2 ZT_{\text{cm}} + \alpha_2 w_{\text{eq}} T_{\text{cp}} \tag{3.22}$$
$$T_3 = (\alpha_2 + \alpha_3) ZT_{\text{cm}} + \alpha_3 w_{\text{eq}} T_{\text{cp}} \tag{3.23}$$
$$T_4 = (\alpha_2 + \alpha_3 + \alpha_4) ZT_{\text{cm}} + \alpha_4 w_{\text{eq}} T_{\text{cp}} \tag{3.24}$$
$$\vdots$$
$$T_n = (1 - \alpha_1) ZT_{\text{cm}} + \alpha_n w_{\text{eq}} T_{\text{cp}}. \tag{3.25}$$

The fractions of the total measurement load should sum to one

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1. \tag{3.26}$$

The optimal values of $\alpha's$ that has to be assigned to each processor in order to achieve the minimum finish time is given by the following set of equations:

$$\alpha_{n-1} = \alpha_n r_i \tag{3.27}$$
$$\alpha_{n-2} = \alpha_{n-1} r_i \tag{3.28}$$
$$\vdots$$
$$\alpha_3 = \alpha_4 r_i \tag{3.29}$$
$$\alpha_2 = \alpha_3 r_i \tag{3.30}$$
$$\alpha_1 = \alpha_2 c \tag{3.31}$$

where $r_i = \frac{w_{\text{eq}}^i T_{\text{cp}} + ZT_{\text{cm}}}{w_{\text{eq}}^i T_{\text{cp}}}$ and $c^i = \frac{w_{\text{eq}}^i T_{\text{cp}} + ZT_{\text{cm}}}{wT_{\text{cp}}}$.

Here i indicates the level of children nodes being considered. It should be noted that, to achieve the minimum finish time, α_i is solved for by equating T_i to T_{i+1} [3, 4]. The equations can be written in terms of of α_n, r_i, and c^i as follows:

$$\alpha_j = \begin{cases} \alpha_n r^{n-j}, & \text{if } j = 2, 3, \cdots, n-1, \\ \alpha_n r_i^{n-2} c^i, & \text{if } j = 1. \end{cases} \tag{3.32}$$

Using (3.26) and (3.32), α_n can be found as a function of r_i and c.

$$\alpha_n = \frac{r_i - 1}{(c^i + 1) r_i^{n-1} - c^i r_i^{n-2} - 1}. \tag{3.33}$$

Now all other optimal values of α's can be computed using (3.32) Since $\alpha_1 = \alpha_n r_i^{n-2} c$, α_1 can be expressed in terms of r_i and c as follows:

$$\alpha_1 = \frac{r_i - 1}{c^i (r_i^{n-1} - r_i^{n-2}) + r_i^{n-1} - 1} (r_i^{n-2} c^i) \tag{3.34}$$
$$= \frac{r_i^{n-1} - r_i^{n-2}}{r_i^{n-1} - r_i^{n-2} + \frac{1}{c^i}(r_i^{n-1} - 1)}.$$

In order to find $w_{\text{eq}i}$, we equate (3.21) to $w_{\text{eq}i} T_{\text{cp}}$. Here $w_{\text{eq}i}$ is a constant that is inversely proportional to the speed of an "equivalent" processor that will replace all processors in Fig. 7 and preserves the characteristics as the original system.

$$w_{\text{eq}i} = w\alpha_1. \tag{3.35}$$

Substituting the value obtained for α_1 in the above equation, we find that:

$$w_{\text{eq}i} = w \left(\frac{r_i^{n-1} - r_i^{n-2}}{r_i^{n-1} - r_i^{n-2} + \frac{1}{c^i}(r_i^{n-1} - 1)} \right). \tag{3.36}$$

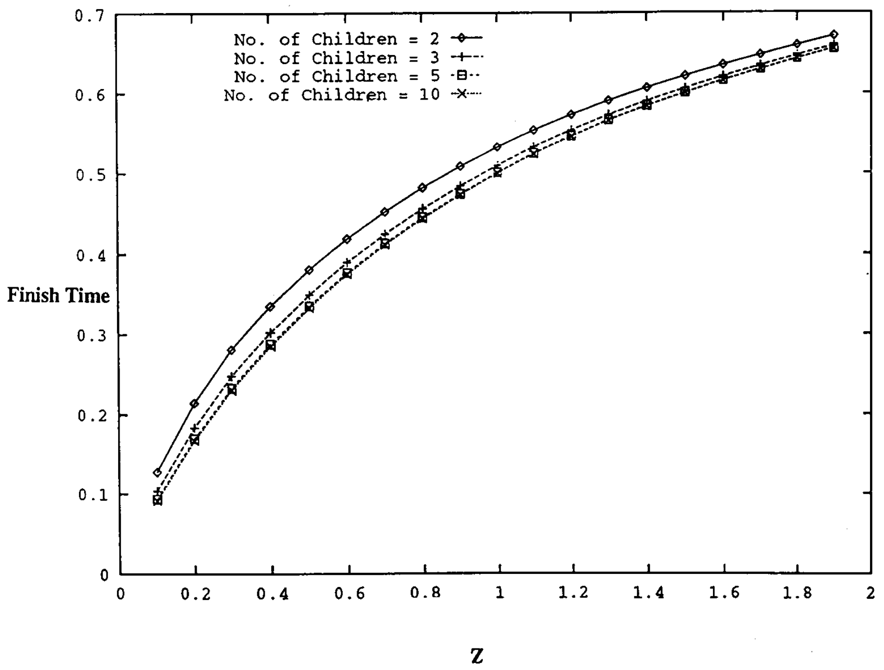

Fig. 10. Finish time versus inverse bus speed, Z, for 11 level symmetric trees with varying numbers of children and with front-end processors. Here, $w = T_{\text{cm}} = T_{\text{cp}} = 1.0$.

Starting at level $N-1$, one can use (3.20) to reduce the tree in Fig. 5 by one level and then move up to level $N-2$. Starting from the subtrees where children are at level $N-2$ and up to the root processor one uses (3.36) to find $w_{\text{eq}_{\text{total}}}$. Here, $w_{\text{eq}_{\text{total}}}$ is a constant that is inversely proportional to the speed of an "equivalent" processor that will replace the whole tree in Fig. 5 while preserving the same characteristics as the original system. Computing $w_{\text{eq}_{\text{total}}}$, the minimum finish time T_{ftnf} can be written as follows:

$$T_{\text{ftnf}} = T_{\text{cp}} w_{\text{eq}_{\text{total}}} \tag{3.37}$$

and the maximum throughput is

$$\gamma = \frac{1}{T_{\text{ftnf}}}. \tag{3.38}$$

IV. Performance Evaluation of Large Symmetric Trees

The minimum finish time expressions obtained in the previous sections and subsections will be used to study the effect of the speed of the processors and the channel speed on the minimum finish time for large symmetric trees. To do so, two sets of plots were obtained. In the first the minimum finish time is plotted against Z and in the second, which consists of only one plot, the ultimate minimum finish time is plotted against w. In both sets $T_{\text{cm}} = 1$ and $T_{\text{cp}} = 1$. In the first set $w = 1$ while in the second $Z = 1$. Note that for the n in Section III to be consistent with that in Section II a subtree must have $n-1$ children plus a root node for a total of n nodes.

- In Fig. 10 and 11, the finish time is plotted against Z for various types of trees which all have 11 levels. The tree network that is used to obtain Fig. 10 has all its' processors equipped with front-end processor while the processors used to obtain Fig. 11 do not have no front-end processors. The horizontal performance line in Fig. 11 is

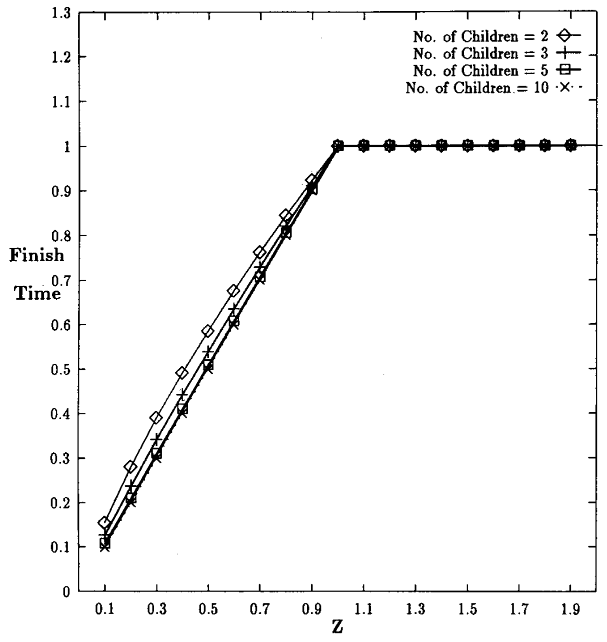

Fig. 11. Finish time versus inverse bus speed, Z, for 11 level symmetric trees with varying numbers of children and without front-end processors. Here, $w = T_{\text{cm}} = T_{\text{cp}} = 1.0$.

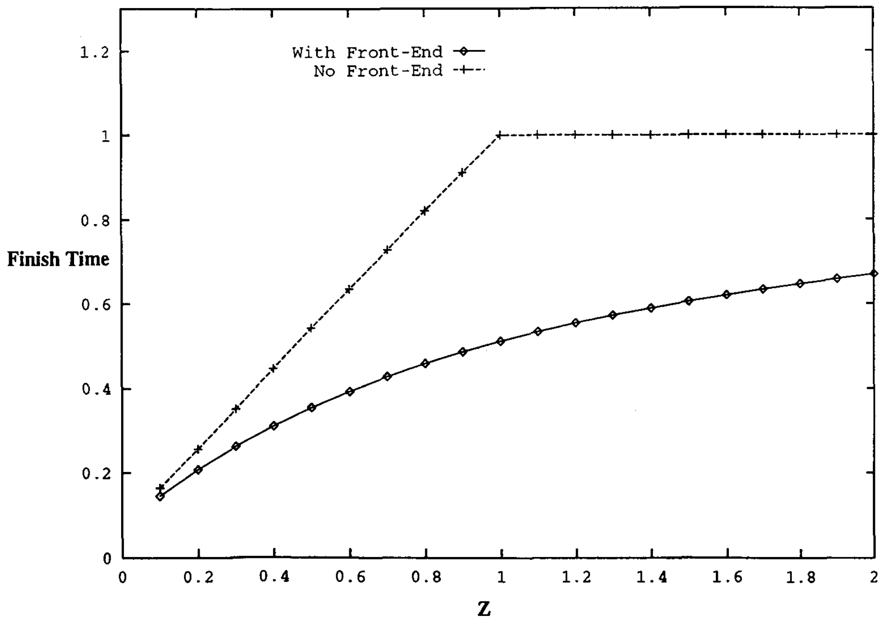

Fig. 12. Finish timing versus inverse bus speed, Z, for trinary symmetric tree with three levels and with and without front-end processors. Here, $w = T_{cm} = T_{cp} = 1.0$.

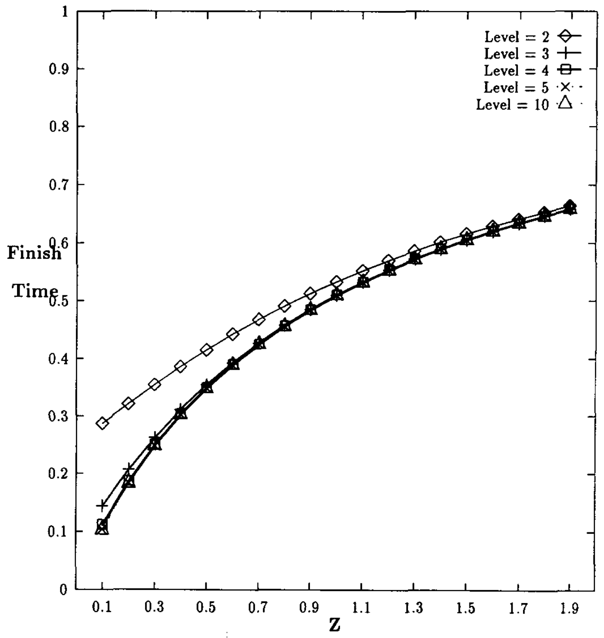

Fig. 13. Finish timing versus inverse bus speed, Z, for trinary symmetric tree with varying number of levels and with front-end processors. Here, $w = T_{cm} = T_{cp} = 1.0$. The levels listed are equal to $N - 1$.

due to the lack of a time saving in distributing the load when link speed is slow. The plot shows that a better finish time is obtained as the size of the trees gets larger. This is expected as more processors would have been involved in computation. It also shows that there is slight difference in the performance curves among trees where the parents have more than three children, especially when the links are slow. This is because the majority of the load will be delivered to the first few processors. The rest of the processors' share of the load tends to be small and so they will not contribute a significant improvement in performance.

- In Fig. 12, the finish time is plotted against Z for a trinary tree with only three levels (13 nodes). Fig. 12, shows a difference in performance between the network with front-end processors and the one with no front-end processors.
- Fig. 13 shows the effect of enlarging the size of a trinary tree with front-end processors by incrementing the number of levels (adding more processors). The plot shows that the performance is not significantly improved as the size of the tree increases. Again, this is because most of the load is distributed to the upper level processors. The levels listed in the figure are equal to N.
- Fig. 14 and Fig. 15 are meant to study the effect of the trade-off between the number of processors and the number of the levels in the tree network. The minimum finish time is plotted against Z and w in Fig. 14 and Fig. 15, respectively. Two types of trees were studied: a binary tree with 15 processors (4 levels with $N = 4$) and a trinary tree with 13 processors (3 levels with $N = 3$). Although the number of processors in the trinary tree is four less than that in the binary tree, it gives a slightly better performance results in Fig. 14. The gap in Fig. 15 between the performance curves increases as the processor speed decreases. This is because, as mentioned above, a large amount of the load will be allocated to the first few processors to overcome the overhead of

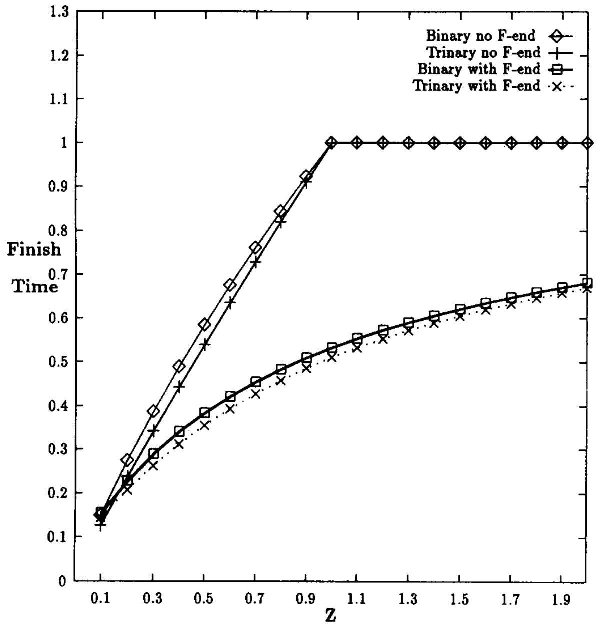

Fig. 14. Finish time versus inverse bus speed, Z for binary (15 processor) and trinary (13 processors) symmetric trees with and without front-end processors. Here, $w = T_{cm} = T_{cp} = 1.0$.

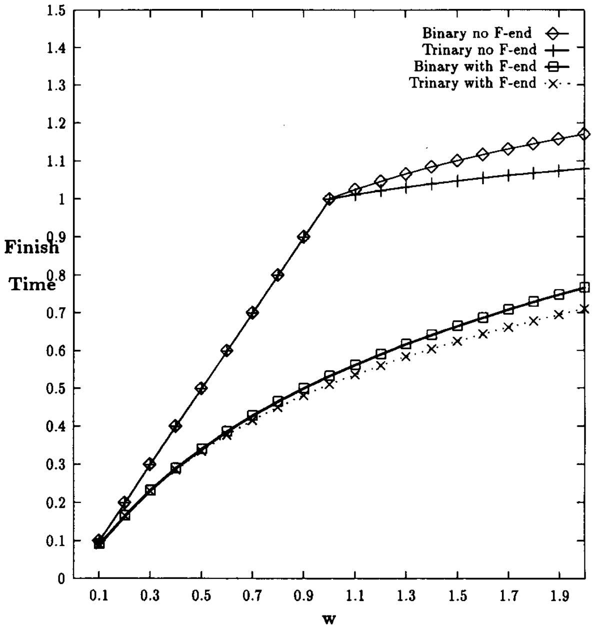

Fig. 15. Finish time versus inverse processor speed, w, for binary (15 processors) and trinary (13 processors) symmetric trees with and without front-end processors. Here, $Z = T_{cm} = T_{cp} = 1.0$.

communicating large fragments of data over the links. In Fig. 15, for the curves with no front-end processor in the region up to $Z = 1.0$ it is faster to process the load in a single processor rather than to distribute load to several processors [2].

V. Conclusion

In this paper, closed form solutions for minimum finish time are obtained for several types of bus architectures and tree network architectures. The performance of these architectures are examined and the effect of the link speed is studied. Processing time for tree networks is only slightly improved as the number of children per node increases, especially if the link speed is slow. Moreover, there is a point of diminishing returns for performance (finish time) as the size of a tree is increased.

Acknowledgment

We would like to thank E. Foo for plotting some of the results.

References

[1] Y. C. Cheng and T. G. Robertazzi, "Distributed computation with communication delays," *IEEE Trans. Aerospace and Electron. Syst.*, vol. 24, no. 6, pp. 700–712, Nov. 1988.

[2] ———, "Distributed computation for tree network with communication delays," *IEEE Trans. Aerospace and Syst.*, vol. 26, no. 3, pp. 511–516, May 1990.

[3] S. Bataineh and T. G. Robertazzi, "Distributed computation for a bus networks with communication delays," in *Proc. 1991 Conf. Inform. Sci. Syst.*, The Johns Hopkins Univ., Baltimore MD, Mar. 1991, pp. 709–714.

[4] ———, "Bus oriented load sharing for a network of sensor driven processors," *IEEE Trans. Syst., Man Cybern.*, vol. 21, no. 5, pp. 1202–1205, Sept. 1991.

[5] T. Hsiung and T. G. Robertazzi, "Performance evaluation for distributed communication systems for load balancing," Technical Report no. 612, SUNY at Stony Brook, College of Eng. and Appl. Sci., Dec. 17, 1991 (available from T. Robertazzi).

[6] K. M. Baumgartner and B. W. Wah, "GAMMON: A load balancing strategy for local computer systems with multiaccess networks," *IEEE Trans. Comput.*, vol. 38, no. 8, pp. 1098–1109, Aug. 1989.

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

[8] V. M. Lo, "Heuristic algorithms for task assignment in distributed systems," *IEEE Trans. Comput.*, vol. 37, no. 11, pp. 1384–1397, Nov. 1988.

[9] K. Ramamritham, J. A. Stankovic, and W. Zhao, "Distributed scheduling of tasks with deadlines and resources requirements," *IEEE Trans. Comput.*, vol. 38, no. 8, pp. 1110–1122, Aug. 1989.

[10] K. G. Shin and Y.-C. Chang, "Load sharing in distributed real-time systems with state change broadcasts," *IEEE Trans. Comput.*, vol. 38, no. 8, pp. 1124–1142, Aug. 1989.

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

[12] R. Mirchandaney, D. Towsley, and J. A. Stankovic, "Analysis of the effects of delays on the load sharing," *IEEE Trans. Comput.*, vol. 38, no. 11, pp. 1513–1525, Nov. 1989.

[13] L. M. Ni and K. Hwang, "Optimal load balancing in a multiple processor system with many job classes," *IEEE Trans. Software Eng.*, vol. SE-11, no. 5, pp. 491–496, May 1985.

[14] J. Sohn and T. G. Robertazzi, "Optimal load sharing for a divisible job on a bus network," in *Proc. 1993 Conf. Inform. Sci. Syst.*, The Johns Hopkins Univ., Baltimore MD, Mar. 1993.

[15] V. Bharadwaj, D. Ghose, and V. Mani, "Design and analysis of load distribution strategies for infinitely divisible loads in distributed processing networks with communication delays," Tech. Rep. 422-GC-01-92, Dept. of Aerospace Eng., Indian Inst. of Science, Bangalore, India, Oct. 1992.

[16] ———, "A study of optimality conditions for load distribution in tree networks with communication delays," Tech. Rep. 423-GI-02-92, Dept. of Aerospace Eng., Indian Inst. of Science, Bangalore, India, Dec. 1992.

[17] J. Du and J. Y.-T. Leung, "Complexity of scheduling parallel task systems," *SIAM J. Discrete Math.*, vol. 2, pp. 473–487, Nov. 1989.

[18] J. Blazewicz, M. Drabowski, and J. Weglarz, "Scheduling multiprocessor tasks to minimize schedule length," *IEEE Trans. Comput.*, vol. C-35, pp. 389–393, May 1986.
[19] W. Zhao, K. Ramamritham, and J. A. Stankovic, "Preemptive scheduling under time and resource constraints," *IEEE Trans. Comput.*, vol. C-36, pp. 949–960, Aug. 1987.
[20] S. Bataineh and T. Robertazzi, "Ultimate performance limits for networks of load sharing processors," in *Proc. 1992 Conf. Inform. Sci. Syst.*, Princeton Univ., Princeton NJ, Mar. 1992, pp. 794–799.
[21] H. J. Kim, G.-I. Jee and J. G. Lee, "Optimal load distribution for tree network processors," submitted for publication.

Sameer Bataineh received the B.S. degree in electrical engineering in 1985 from Syracuse University and the Ph.D. degree in electrical engineering from the State University of New York at Stony Brook in 1992.

He is currently on the faculty of the Electrical Engineering Department of the Jordan University of Science and Technology, Irbid, Jordan.

Te-Yu Hsiung receivd the B.S. degree in electrical engineering and applied mathematics and statistics and the M.S. degeee in electrical engineering from the State University of New York at Stony Brook, in 1990 and 1991, respectively.

Since then she has been with Corporate Computer System as a member of the technical staff in a group dealing with telemetry monitoring and control systems. Her areas of research interest include digital signal processing, computer communications and networks.

Thomas G. Robertazzi (S'75–M'77–S'78–M'81–SM'91) received the B.E.E. degree from Cooper Union in 1977 and the Ph.D. degree in electrical engineering from Princeton University in 1981.

He was an Assistant Professor of electrical engineering at Manhattan College, Riverdale, NY, during 1982–1983. Since 1983, he has been at the State University of New York at Stony Brook where he is presently an Associate Professor in the Electrical Enginerring Department. During the Fall of 1990, he was a visiting Research Scientist at Columbia University's Electrical Engineering Department. His research interests are in the performance evaluation of computer networks and computer systems.