

OKLAHOMA STATE UNIVERSITY

PROCESSOR TOPOLOGIES FOR IMAGE PROCESSING
APPLICATIONS

By

KIT SAI WONG

Bachelor of Engineering

National University of Singapore


Republic of Singapore

1985

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
May, 1997

PROCESSOR TOPOLOGIES FOR IMAGE PROCESSING
APPLICATIONS

Thesis Approved:



H. Lu

J. Chandler

Blayne E. Mayfield

Thomas C. Collins

Dean of the Graduate College

PREFACE

Arabnia's process-and-data-decomposition approach can be applied to many image processing jobs other than image rotation. I have developed a generalized topology for image processing. I investigate the efficiency, speed-up of the generalized topology and the factors which affect the performance of the generalized topology. But the generalized topology has a drawback, it is not load balanced in most cases. So I further develop a linked-trees topology with better load balance capability. And the performances such as efficiency and speed-up of the linked-trees topology are investigated. And I analyze the factors affecting these performances. In conclusion, the linked-trees topology is superior in time efficiency and speed-up than the generalized topology.

ACKNOWLEDGMENT

First and foremost, I would like to express my sincere appreciation to my M.S. thesis advisor, Dr. K. M. George for his consistent and continuous advice, guidance, kindness, and valuable instruction through my graduate work.

Special thanks are due to my advisory committee members, Dr. J. P. Chandler, Dr. H. Lu, for their advice, support, and valuable comments on my research.

I would like to take this opportunity to thank my parents, Wong Man Sum and Pong Yip Moi, who have provided consistent support and endless love in my whole life. Also, I would like to thank my sisters, Wong Sai Wah and Wong Sai Yuet for their encouragement and care.

TABLE OF CONTENTS

Chapter	Page
I. INTRODUCTION.....	1
1.1 Image Processing and Parallel Processing.....	1
1.2 Scope and Objectives.....	2
1.2.1 Objective 1.....	2
1.2.2 Objective 2.....	3
1.3 Organization of the Thesis.....	4
II. RELATED WORKS.....	5
2.1 Machine Vision Quality Inspection System for Textile Industries.....	5
2.2 Evaluation of Transputer-based Architecture for Image Compression and Reconstruction.....	6
2.3 Image Processing on a Transputer-based Perfect Shuffle Machine.....	6
2.4 Distributing Pixmap Images Among Parallel Disk Arrays.....	7
2.5 Arbitrary Rotation of Digitized Images Using Process-and-Data-Decomposition Approach.....	7
2.6 Image Processing Methods.....	9
III. TRANSPUTER NETWORK.....	11
3.1 Introduction to Transputer.....	11
3.2 Generalization of the Process-and-Data-Decomposition Approach.....	12
3.3 Factors Likely to Affect Speed-up and Time Efficiency.....	18
3.4 Experiment.....	20
3.4.1 A Simple Implementation of Farming or SIMD Topology.....	20
3.4.2 Execution Time of the Algorithms.....	21
3.4.3 Pipelined Implementation of Three Processes.....	22
3.5 The Types of Pipelining.....	22
3.6 Linked-Trees Topology.....	31
3.6.1 Data partitioning in linked-trees topology.....	31
3.6.2 Illustration by Examples.....	34
3.7 Results.....	35
3.7.1 A Simple Implementation of Farming or SIMD Topology.....	35
3.7.2 Pipelining of Images and Pipelining of Segments.....	41
3.7.3 Linked-Trees Topology Experiments.....	42
IV CONCLUSION.....	49
BIBLIOGRAPHY.....	51

LIST OF FIGURES

Figure	Page
1. The Transputer network for an image space divided into four blocks.....	15
2. Generalized topology for process-and-data decomposition approach using transputers for parallel image processing.....	16
3. Arabia's two rings network is a version of the generalized topology with two levels and four nodes in each level.....	17
4. Transputer chain network used for the speed-up and time efficiency experiments.....	24
5. The data and operation flow of the three nodes chain network..	24
6. A 3 x 3 mesh network using process-and-data decomposition approach.....	25
7. Data and operations flow of pipelining of images.....	26
8. Data and operations flow of pipelining of segments.....	29
9. (a) Image pipelining. (b) Segment pipeline timing.....	30
10. Linked-trees topology.....	33
11. Experiment results by executing various image tasks on a farming topology.....	38
12. The graphs for experiment results.....	39
13. Results and calculations for pipelining of images and pipelining of segments.....	40
14. Results and calculations for pipelining through the linked-trees topology.....	45
15. The results of the three linked-trees when they are processed individually.....	46

16. (a) The effect of number of nodes in the linked-tree for blurring on the efficiency of the whole linked-trees topology.....	47
(b) Effect of the number of nodes in the linked-tree for blurring on the speed-up ratio of the whole linked-trees topology.....	47
17. Comparison of network efficiencies and number of transputers used for generalized topology and linked-trees topology accomplishing the same speed-up ratio.....	47

CHAPTER I

INTRODUCTION

1.1 Image processing and parallel processing

Serial processing on a digital computer is probably the most common method of implementing pattern/picture processing operations even today. For many image processing applications pixel-by-pixel processing is necessary. For example, if we are processing an image of 512 x 512 pixels, the same operation will have to be repeated about 250000 times. So a serial computer is not the most time efficient processor considering the inherently iterative nature of many image processing operations; particularly, when time constraints are important. Furthermore, serial processing is certainly not like the biological systems of human beings, such as eyes and brain [Fairhurst88].

Depending on the type of image processing job, it may take minutes to process an image using serial processing, which is not suitable for many real-time applications. On the other hand, parallel processing provides a more effective correspondence with the inherent parallelism in many tasks of interest in image processing. Great improvement in computing power and shortening of processing time can be achieved by parallel processing.

The above mentioned observations have led to the exploration of parallel processing architecture for image processing [Uhr87],[Tomohisu93],[Morrow91],[Crookes89]. The

different computer architecture models used for parallel image data processing are SIMD and MIMD architectures. These models fall into Flynn's taxonomy of computer architectures [Dasgupta89]. This thesis proposes new topologies for transputer based networks suitable for image processing. The research presented in this thesis is based on transputer network.

1.2 Scope and objectives

1.2.1 Objective I

The first objective of this study is to develop a generalized topology for image processing based on Arabnia's process-and-data-decomposition approach [Arabnia90] for image transformation, which is described in chapter II. In the generalized topology (described later), there are parallel computation of different segments of the image and parallel execution of different subprocesses of a process, like Arabnia's network.

1.2.2 Objective 2

As outlined later in chapter II, Arabnia presents a network consisting of two rings of transputers [Arabnia90]. He also presents a parallel rotation algorithm. In Arabnia's rotation algorithm, the inner ring processes are less time consuming than the outer ring processes. That means the load on the two rings is not balanced. Arabnia suggests that it is not necessary to have transputers of the same processing power in both rings. For example, he suggests that 20 MIPS transputers could be used to execute outer ring processes and 10 MIPS transputers could be used to execute inner ring processes to achieve load balancing. That means the inner ring will not wait idle for the outer ring to complete its job.

But this is not an ideal solution to the network's load balancing problem, because a computer engineer may only have one type of transputers available. Also, balancing of load is intended to increase efficiency of a transputer network, but deliberate slow down of one component to 10 MIPS to accomplish load balancing is not ideal. Since the generalized topology is developed from Arabnia's topology, it also inherits the load balancing problem.

So, a second objective of this study is to search and propose an alternate solution for balancing the load of different processes and subprocesses in the generalized transputer network which uses the process-and-data-decomposition approach for image

processing. We use a sequence of processes on an image as an example to illustrate the effectiveness of the proposed network with load balance. In addition, the speed-up and time efficiency of the proposed network are investigated and they are compared with those of the unbalanced generalized topology to show the advantages of having a load balanced network in terms of speed-up and time efficiency.

1.3 Organization of the thesis

The reminder of this thesis is organized as follows. Chapter II describes the related works of Karkanis on machine vision quality inspection, image compression and reconstruction by Antola, process-and-data-decomposition approach by Arabnia, and other related works. Chapter III gives an introduction to transputers. Generalization of the process-and-data-decomposition approach of Arabnia is given in chapter III. It then mentions factors likely to affect the time efficiency and speed-up of the generalized topology. It investigates the efficiency and speed-up of the generalized topology by means of experiments. It introduces the linked-trees topology to solve load balancing problem of the generalized topology. It investigates the speed-up, time efficiency and load balance of the linked-trees topology. Then results of the experiments are given. Chapter IV summarizes the results of the investigations and concludes the thesis.

CHAPTER II

RELATED WORKS

In this chapter, we examine the works related to the research undertaken in this thesis.

2.1 Machine vision quality inspection system for textile industries

Karkanis presents a machine-vision based quality inspection system which can be applied for textile inspection. The objective is to increase the inspection speed and improve the quality assurance performance. It is implemented using a transputer network [Karkanis89].

The image processing algorithms were implemented on a parallel architecture allowing concurrent processing of different parts of an image at the same time with the same set of instructions. In other words, it is a SIMD architecture [Dasgupta89].

Various topologies containing four or nine transputers in chain network and tree network have been tried.

2.2 Evaluation of transputer-based architecture for image compression and reconstruction

Two different methods of obtaining parallelism are evaluated for the execution of the 2D-DCT (two-dimensional Discrete Cosine Transform). The first one, is the decomposition of the algorithm into a number of steps, each executed by different transputers simultaneously (algorithm parallelism). The second one is decomposition of data into blocks to be transformed in parallel. Such architecture is also called farming processing [Antola91].

2.3 Image Processing on a Transputer-based Perfect Shuffle Machine

A transputer-based parallel computer has been developed for image processing. This is a multiple instruction multiple data(MIMD) architecture. The computer is built using an one-dimensional array of processing nodes with nearest neighbor connections and perfect shuffle network. Several examples are used to demonstrate that this machine is capable of efficiently solving the typical computational tasks of low- and medium-level image processing [Schomberg89].

2.4 Distributing pixmap images among parallel disk arrays

Professionals in various scientific fields (for example, medical imaging and civil engineering) require rapid access to very large amounts of pixmap image data. Browsing through large pixmap images requires segmentation of the image into rectangular areas, which can be retrieved from disks or cache on demand. A server architecture that consists of four image-handling processors and eight disk nodes is proposed in [Hersch93]. It is reported that the server architecture is cost-effective.

2.5 Arbitrary rotation of digitized images using process-and-data-decomposition approach

Arabnia [Arabnia90] presents a process-and-data-decomposition approach using transputers for rotation of digitized images. This approach can be applied not only for rotation of image, but also for many other image processing jobs such as Laplace edge enhancement [Lindley91]. Arabnia's process-and-data-decomposition approach is outlined below:

(a) Process decomposition - the decomposition of a process into a number of subprocesses and the mapping of each subprocess to a processor for execution. Here, a set of concurrent processes operate simultaneously and cooperatively to solve a given problem.

This approach is equivalent to the MIMD machine architecture [Dasgupta89].

(b) Data decomposition - the decomposition of data into smaller portions (they are not necessarily equal) and the mapping of each portion of data to a processor for execution. This approach is used in both MIMD and SIMD machine architecture [Dasgupta89].

(c) Process and data decomposition - this approach can be regarded as the combination of decompositions (a) and (b).

Arabnia uses a transputer network partitioned into two rings as shown in Fig. 1(b). The image is divided into four blocks, each assigned to a transputer node in the outer ring for processing as shown in Fig.1(a), (b). The algorithm for rotation can be divided into two major processes, one for each of the two rings of transputers. These two processes are executed simultaneously.

Both the data portions processed and the two major processes are executed simultaneously at the same time.

It is interesting to note that Antola also uses the decomposition of processes and decomposition of data, but separately [Antola91]. But in the case of Arabnia, he uses a combination of both methods to achieve greater parallelism. This thesis is based on the process-and-data-decomposition approach of Arabnia.

2.6 Image processing methods

In this section, the author briefly introduces the image enhancement methods used in experiments conducted as part of this research. The methods described are Laplace edge enhancement, image smoothing, image thresholding, and image blurring.

Laplace Edge Enhancement- local, or neighborhood, equalization of image contrast produces an increase in local contrast at boundaries. This has the effect of making edges easier for the viewer to see, consequently making the image appear sharper [PhilipsD94].

Image Smoothing- in a 3 x 3 pixel area, all pixel grey scale values are averaged to produce the resulting value for the pixel at the center of the area. This is done for every pixel in the whole image [Russ92].

Image Thresholding- a range of brightness values is defined by means of a threshold value. Select the pixels within this range as belonging to the foreground, and reject all the other pixels to the background. Such an image is then usually displayed as binary or two-level image [Russ92].

Image Blurring- in a 3 x 3 pixel area, all pixel values are changed to the same value as the pixel at the center of the area. This is done for all the 3 x 3 pixel areas in the image [Russ92].

The image used in the experiments in this thesis is an image of a book cover.

CHAPTER III

TRANSPUTER NETWORK

3.1 Introduction to transputer

The transputer gets its name by combining the words - transistor and computer. This device was developed by INMOS Ltd of Britain, and it is based on the multiple-instruction multiple-data (MIMD) loosely coupled architecture [Dasgupta89]. This family of 16- and 32-bit microprocessors can be used in single processor applications or linked together in a network to form a multiprocessor system. Each transputer contains a CPU, on-chip static RAM, timers, an external memory interface, and four high-speed serial links which allow communications between processing nodes [CSA90c]. The T800 transputer series also has an on-chip floating point unit (math coprocessor)[Im90],[Hull94].

It has been demonstrated that the transputer is a powerful and flexible device for building large multiple-processor parallel systems [Leung90], [Mohan90], [Pachowicz89], [Kirland91], [Gupta93], [Gray91], [Phillips94], [Stallard93], [Lakshmi90], [Stalker91], [EET95]. The T800 transputer delivers a 4 Mega Whetstone benchmark performance and a capability of 1.5 sustained MFlops/sec, and includes four 10/20 Mbits/sec INMOS serial communication links. In addition to their 4 Kbytes of fast on-chip RAM, the T400s and T800s can directly access a linear address space

of up to 4 Gbytes of local memory and a data rate up to 50 Mbytes/sec [CSA90c].

There are a number of compilers and development systems available for the use of most high level programming languages in transputer programming. Those languages include Ada, C, FORTRAN, MODULA-2, OCCAM (the native language of transputers), PASCAL, CS_PROLOG, and T-CODE which is the transputer equivalent of assembly language [CSA90b],[CSA90a],[Kerridge94].

3.2 Generalization of the process-and-data-decomposition approach

To generalize the process-and-data-decomposition approach in image rotation [Arabnia90] for applications to other image processing jobs, the author proposes a generalized topology. The organization of transputers in the generalized topology is shown in Fig. 2. The generalized topology is based on the architecture taxonomies and pipeline processing. Close observation reveals that the inner ring and outer ring topology of Arabnia is a special case of the generalized topology. This will be explained later after introducing the generalized topology.

In the generalized topology, the transputers are connected in a mesh form. A row of transputers constitutes a level. In each

level, each transputer is connected to its left and right neighbors. And each transputer is connected to the corresponding transputer in the next level. The first transputer in the first level serves as a PC/Link connection to the host PC. The last transputer in each level is connected to the first transputer in the same level. The first transputer in the last level is connected to the first transputer in the first level. Double lines represent connections for both data communication and system services (e.g. program loading). Single lines represent connections for data communication only. So, except in the case of the leftmost nodes, level links between nodes are used for data transfer only.

In the generalized topology, image can be split into portions and fed to the first level of the processors. All of them perform the same processing instructions. Then the processed pixels are channeled to the second level of processors all of which execute the second set of processing instructions. In general, pass the pixels processed at the i th level to the $(i+1)$ th level of processors. All the levels together form a pipeline. Each level corresponds to a stage of the pipeline. Each level is a farming topology [Antola91].

This generalized topology is not only suitable for decomposition of a process into a number of subprocesses. It also is suitable for image processing jobs involving multiple processes, which corresponds to a MIMD structure [Dasgupta89]. That means each

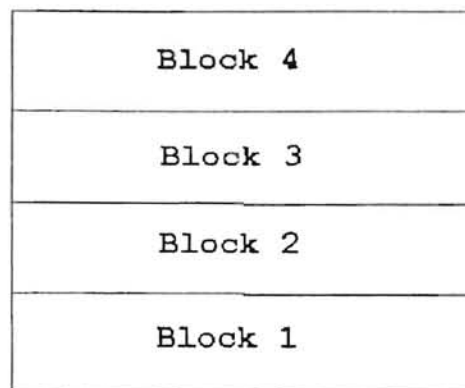
level of processors represents different processings necessary for the enhancement of an image.

Arabnia's topology in Fig.1 (a) is a special case of the generalized topology with two levels and each level has four nodes, the I/O function is performed by one of the nodes in the outer ring, as shown in Fig. 3.

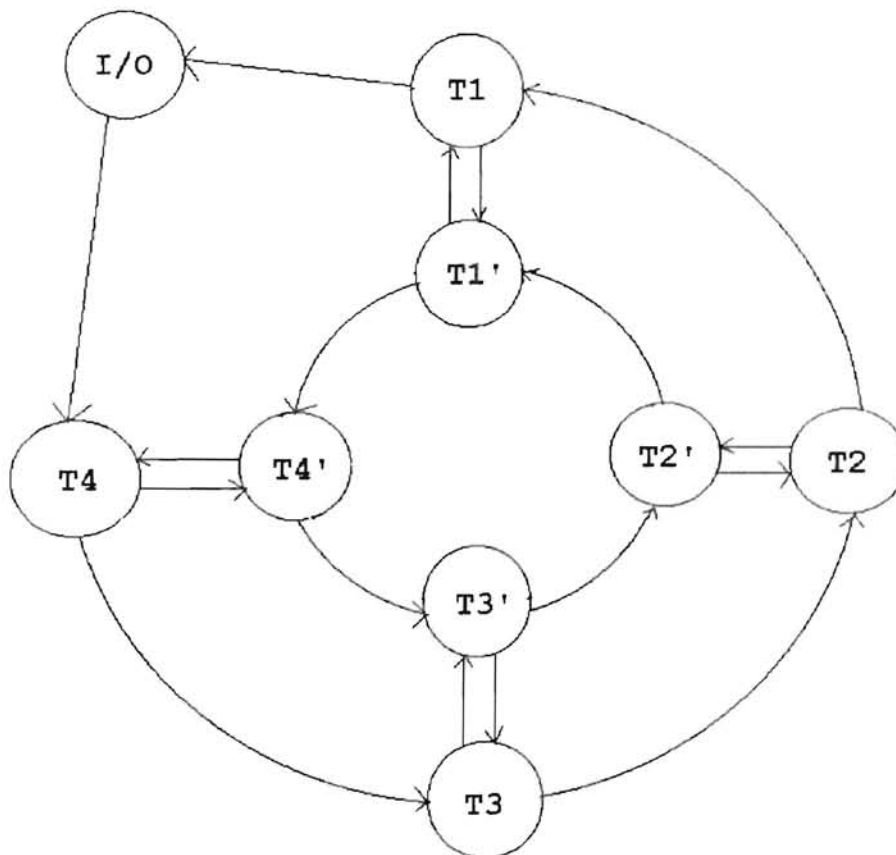
Every node in the network will receive the whole image before computation starts. This is because of the computation problem that occurs at section boundaries, where a neighborhood is physically split across two transputers. And also because the processes at different levels could be in different area operations.

This study reported is based on this generalized topology. But this generalized topology have drawbacks in load balancing. An improved topology for process-and-data-decomposition using linked-trees is proposed later.

In section 3.4 of this chapter we will describe the details of experiments to estimate the speed-up and time efficiency of proposed generalized transputer network used for low level image



(a)



(b)

Fig.1 The transputer network for an image space divided into four blocks. (a) An image space divided into four blocks. (b) The network of four blocks [adopted from Arabnia90].

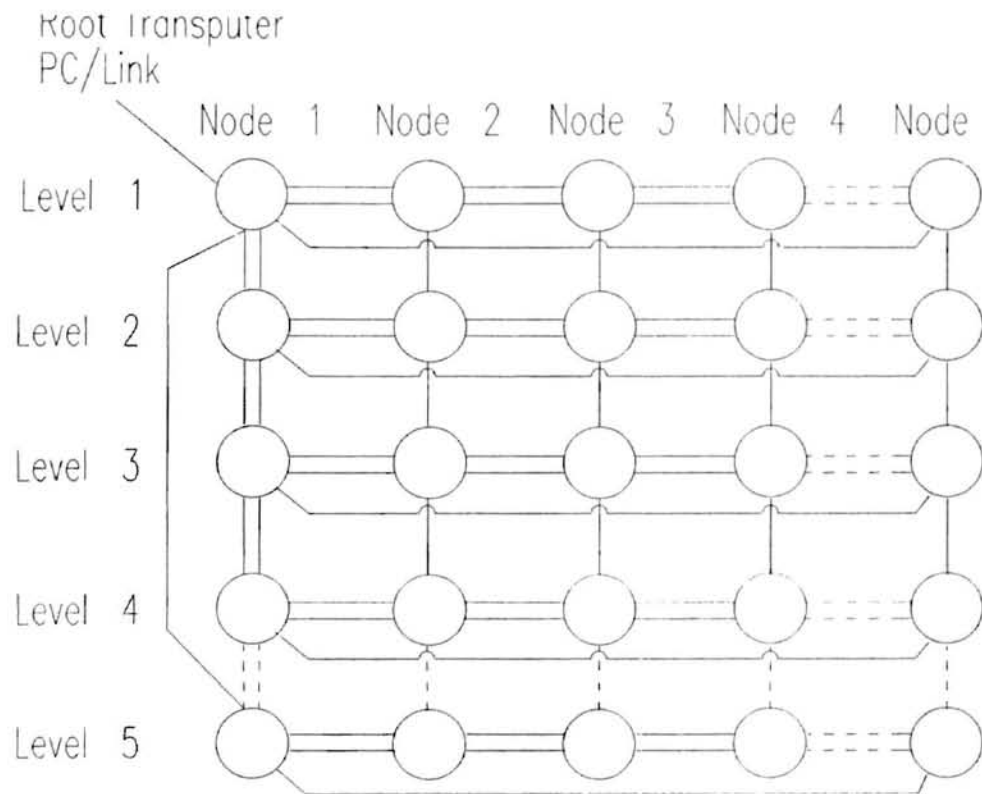


Fig. 2 Generalized topology for process-and-data decomposition approach using transputers for parallel image processing. (double lines are links for both communications and system services, single lines are for communications only.)

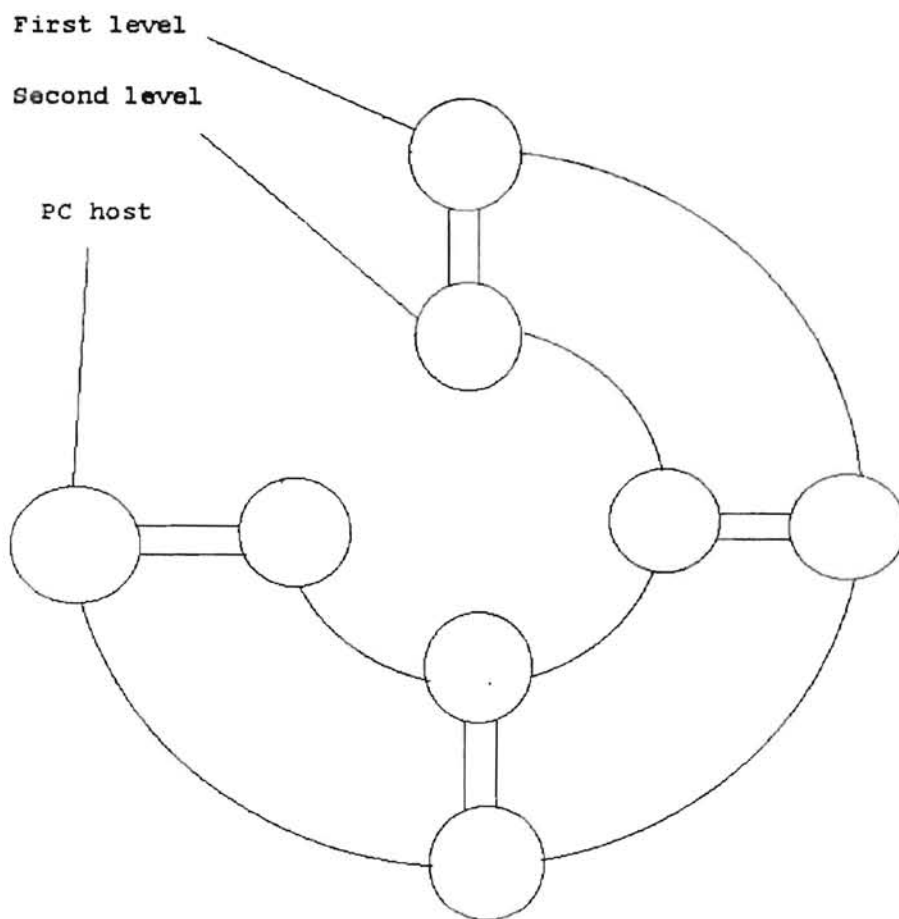


Fig.3 Arabia's two rings network. This is a version of the generalized topology with two levels and four nodes in each level. Arabia's image rotation algorithm can be implemented by the above topology.

processing jobs using process-and-data-decomposition approach. In section 3.6 of this chapter we describe experiments with a proposed new transputer topology, called linked-trees topology. We illustrate that it is a better load balanced topology for a transputer network used in image processing compared with the generalized process-and-data-decomposition topology described earlier. It is a better topology than the generalized topology if load balancing in a transputer network is critical to its performance. But it also has the disadvantages that it is good only for pipelining of images, not good for pipelining of segments. These two types of pipelining will be described later. Then the speed-up and time efficiency of the linked-trees topology will be investigated.

3.3 Factors likely to affect speed-up and time efficiency

In a parallel transputers network, the overall execution time of the network consists of the computation times and communication times among the transputers through their interconnections. In many applications such as robotic trajectory calculations [McKeever92],[Zomaya92], internode communications involve only a few bytes of data. But for image processing, communications of image pixels' value often are done data block by data block, thousands of bytes for each block at a time. A typical

block size is 100 x 500 pixels. Take a SIMD image processing network as example, first, the image needs to be split up and transmitted to each slave transputer of the SIMD or farming network [Antola91]. After transformation, the transformed image of each slave transputer needs to be transmitted back to the master transputer for storage or output. So the communication of data is very intensive in image processing job.

The purpose of parallel network for image processing is to reduce the computation load of each node so as to reduce the overall execution time. But if the communication times among nodes is large, they will form a bottleneck for the overall execution time. That means if the number of nodes is increased in a transputer network, up to certain number of nodes, the overall execution time can no more be reduced because of the bottleneck mentioned above. Thus the speed-up and time efficiency of the transputer network will be affected.

But if the computation is CPU intensive, the effect of communication time on the speed-up and time efficiency can be reduced. That is the reason why in our experiments, image operations of different computation intensiveness need to be chosen in order to have a better understanding of the speed-up and time efficiency problem in a transputer network for image processing.

The load balancing of the transputer network also affects the speed-up and time efficiency of the network. For example, in a

pipelining network, the workloads in the nodes are not balanced, some nodes will have idle times waiting for the nodes with slower processes to finish. A transputer network with well balanced load will not have such a problem. The experiment conducted in this study for performance analysis of the generalized topology is described in the following section. The algorithms are implemented in C language.

3.4 Experiment

Three processes are executed independently of each other using only one level of the generalized topology. Then these three process are executed in three levels of the generalized topology in a pipelining mode. The speed-up and time efficiency are found for both cases.

3.4.1 A simple implementation of farming or SIMD topology

The first part of experiment one is based on the generalized process-and-data-decomposition topology described earlier. The experiment only involves one image process at a time, it is not a multiprocess operation. So, only the first level of the generalized topology is used. The topology for the experiment is as

shown in Fig. 4, which is actually a farming topology . But the assumption and description of the generalized topology still apply here. The network in Fig. 4 can have different number of nodes. In our experiment, we execute three algorithms in a transputer chain network of one, two, and three transputers. The names of the algorithms are image thresholding, image smoothing, and image blurring. They are adopted from [Lindley91], [Duff86].

Fig. 5 shows the data and operations flow in the three nodes network.

3.4.2 Execution time of the algorithms

The algorithms in these three nodes can be split into three parts, the receiving and transmitting of the initial image, the computation, and the receiving and transmitting of the transformed image. The execution times for these three parts are represented by T_1 , T_2 and T_3 respectively. The file reading and writing times are ignored, because they vary according to the I/O hardware used. For example, different hard disks have different access times which in turn affect the file reading and writing time. Therefore, we define total execution time as:

$$\text{Total execution time} = T_1 + T_2 + T_3$$

Only the master node algorithm is timed. It is the master node which outputs the final transformed image to the user. So the overall job is not done, until the master has finished its job.

The resulting times will be used to calculate the speed-up and time efficiency of the transputer network.

3.4.3 Pipelined implementation of three processes

In this part of the experiment, the three processes, namely image thresholding, image smoothing and image blurring [Lindley91] are executed consecutively in three levels of the generalized topology in the order above. Three nodes are used for each level. That means the topology used is a 3×3 mesh network which is shown in Fig. 6. The three levels of nodes form a pipelined network of depth three. So, data decomposition and process pipelining are included in the network.

3.5 The types of pipelining

There are two different types of pipelining possible for the 3×3 mesh generalized topology. First type is the pipelining of the whole image. This type of pipelining only gets parallel

processing in pipelining if more than one image is channeled into the network for processing. Pipelining of image is suitable in many applications where large numbers of image need to go through same transformation processes in the same order. An example is continuous quality inspection of manufactured goods by computer vision [Karkanis89]. But in applications where only a single image needs processing, there will not be parallel processing. Fig. 7 shows the data and operation flow of pipelining of images in the 3 x 3 mesh network.

First, the first image is channeled to the first node in the first level of the network. The first node will communicate the image to each node in the first level. Then computations of one third of the image begin in all the three nodes. After that the transformed images are communicated among the three nodes in the first level. Then, the three nodes will communicate the whole image to the corresponding nodes in the second level. Nodes in the second level receive image and begin computation on one third of the image each. Then communications of the transformed images among the three nodes in the second level start. After that, they transmit the whole image to the corresponding nodes in the third level. Same operations as in the second level repeat here. The

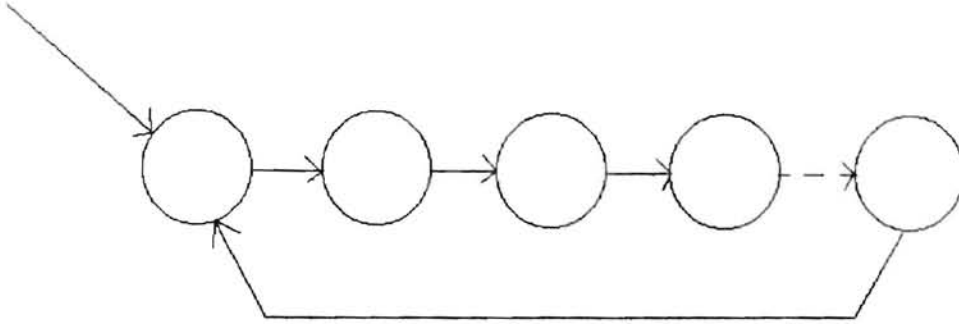


Fig. 4 Transputer chain network used for the speed-up and time efficiency experiments.

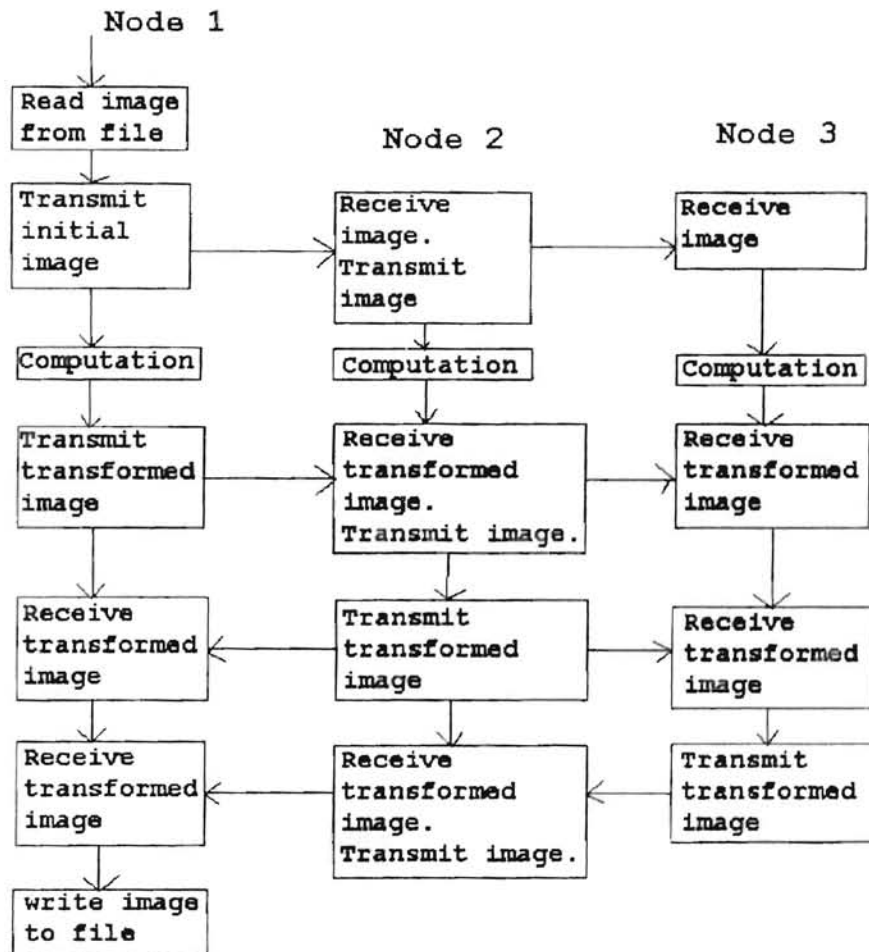


Fig.5 The data and operation flow of the three nodes chain network.

Root transputer PC
link

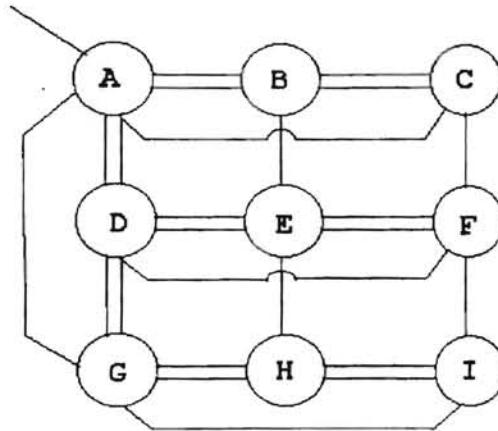


Fig. 6 A 3 x 3 mesh network using
process-and-data-decomposition approach.

final image will be channeled back to the root for I/O. All communications and computation are done pixel by pixel, so image is not partitioned into blocks.

The second type of pipelining is called segment pipelining. The image is divided into segments. For example each segment contains 3 columns of pixels. In this case, once the nodes in the first level finished transformation of a segment of the image, they will channel the transformed segments to the corresponding nodes in the second level for further processing. Then the first level nodes will continue transforming the second segment. The second level nodes begin processing almost as soon as the first level nodes begin processing the first segment. In the same way, the nodes in the second level will channel segments to the corresponding nodes in the third level as soon as they finish processing a segment. The computation will stop at each node when it is finished processing 1/3 of the whole image in segments. Then communications among the nodes in each of the three levels occur. The nodes in the first level will send the other two thirds of the image to the corresponding nodes in the second level. Because they may need that to finish computations of their last segments, which need pixel values in the border sections. Nodes in the second level will do the same to the third level. The final image is channeled back to the root from the first node in the third level for I/O. In the case of pipelining of the three processes, the three processes will be executed simultaneously and therefore the three execution times will overlap with each other. Even if only a single image

needs to be processed, parallel processing occurs and there is reduction in processing time by pipelining, unlike pipelining of images. In addition, if large number of different images need processing, they will also benefit from pipelining of segments because of parallel processing.

Fig. 8 shows the data and operations flow of the pipelining of segments.

Fig. 9 (a) shows the execution times of the pipelining of images. Four images are used as illustration. The total execution time for processing four images is equal to $(T_1 + T_2 + 4 * T_3)$ where T_1, T_2, T_3 are execution times for processes one, two and three respectively when they are executed individually on an image ($T_1 < T_2 < T_3$). Fig.9(b) shows the execution times of the pipelining of segments. The total execution time for four images is equal to $4 * T_3$. The four different images are channeled to the 3×3 mesh network for processing. As shown, the pipelining of segments gives slightly shorter total execution time because there is overlapping of the execution times of the three processes.

Here, we are interested in investigating the speed-up and time efficiency of the 3×3 mesh network using decomposition of data and pipelining of processes. The investigation includes both

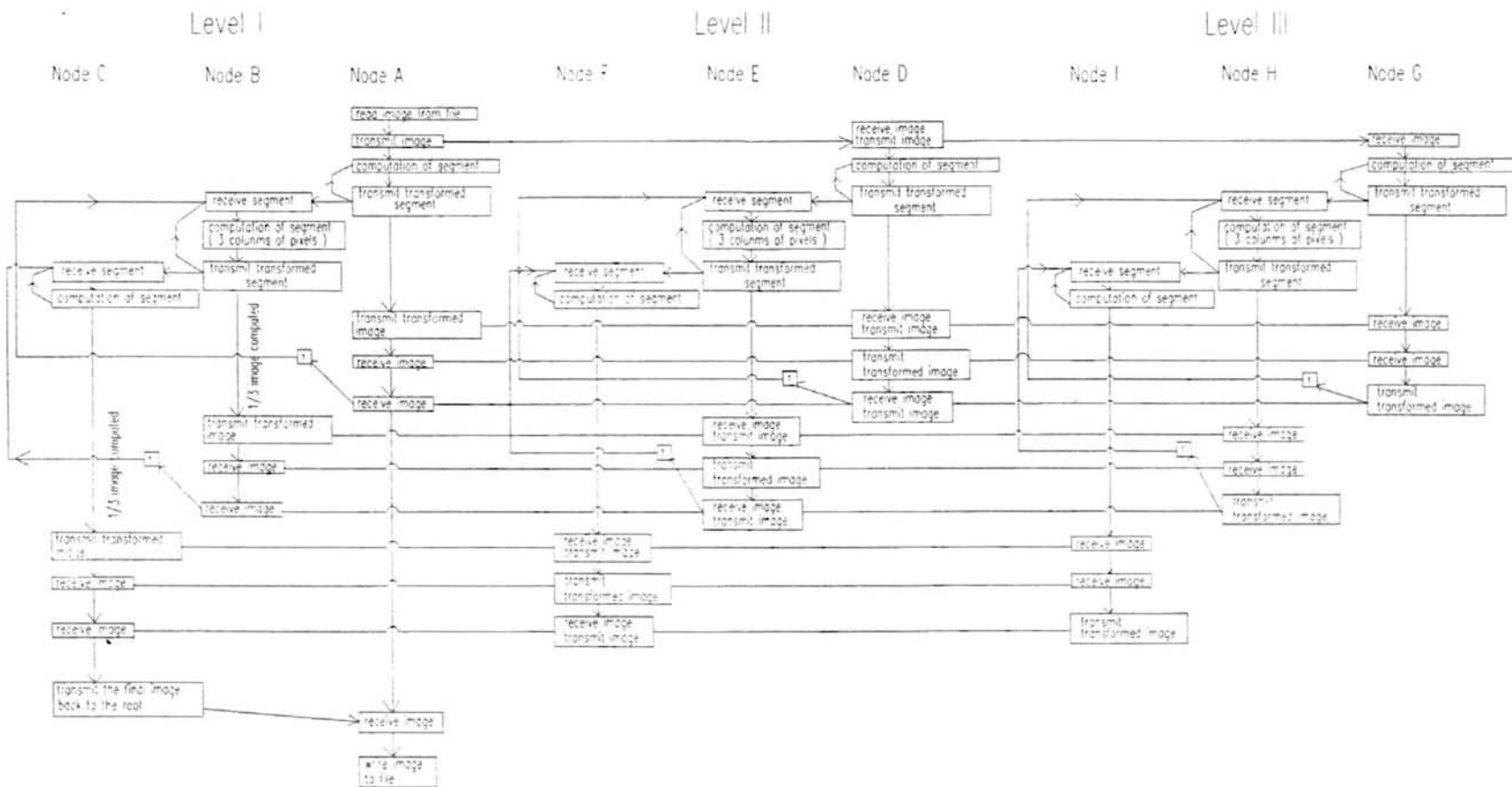


Fig. 8 Data and operations flow of pipelining of segments.

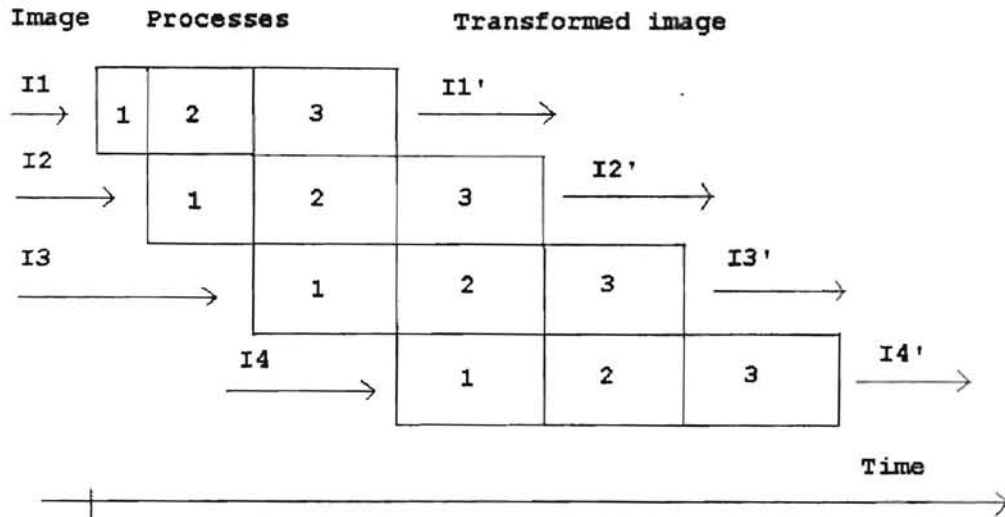


Fig. 9a Image pipelining. Process execution times satisfy $T_3 > T_2 > T_1$, T_1 becomes larger and larger because of idle times to wait for other processes to finish.

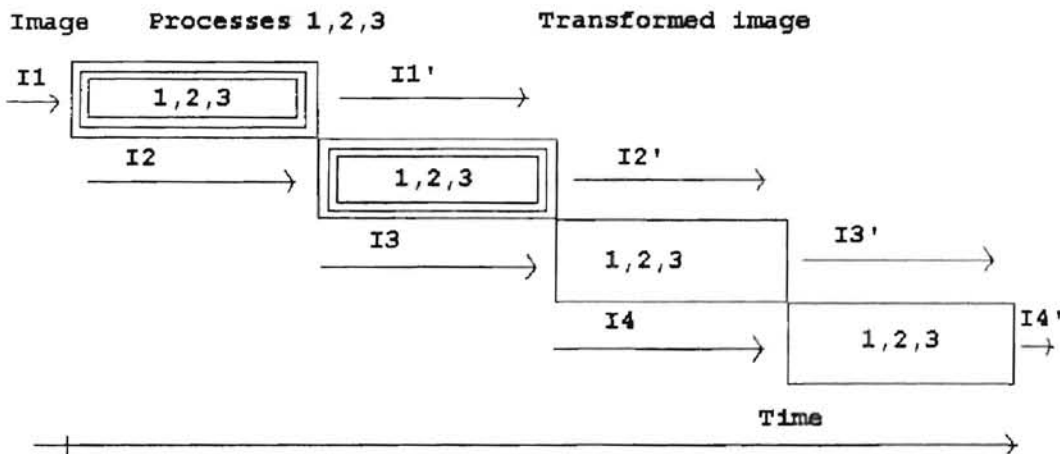


Fig. 9b Segment pipeline timing. Segment pipelining. Process execution times satisfy $T_3 > T_2 > T_1$, but T_1 , T_2 are lengthened to T_3 , because of idle times to wait for process three to finish. And these execution times overlap with each other.

pipelining of image and pipelining of segments. For illustration, four different images are channeled to the network for processing.

3.6 Linked-trees topology

In the generalized topology, high speed-up is accomplished, but load balance is still a problem which causes low time efficiency of the generalized topology. A linked-trees topology is recommended here in order to balance the load of the nodes in the transputer network. Fig. 10 shows a typical linked-trees topology. It is a better topology for process-and-data-decomposition approach [Arabnia90] for image processing. All nodes in the same tree execute same set of image processing instructions. Each tree is in itself a farming topology [Antola91].

3.6.1 Data partitioning in linked-trees topology

To illustrate how image processing is done in a linked-trees topology, consider a linked-tree with eight nodes. The root receives image pixel values from the PC host. The whole image is divided into eight blocks. The root channels the pixel values of the blocks to other nodes in the tree directly, or indirectly through some of the nodes. So each node receives one of the eight blocks of image. Each node including the root will process one eighth of the image. After computation, the values of the processed

pixels are channeled back to the root of the tree. The root then sends the processed image to the root of the second tree for the second image operation. After processing by the second tree, root of the second tree then channels the processed pixels to the root of the third tree, and so on. The root of the last tree will channel the final processed image back to the root of the whole network which is the root of the first tree, for output/storage. Each tree executes one process, so the linked-trees topology is a multiprocessing topology.

There is no need to transmit the whole image to each node of a tree. Only the portion of the image need to be processed by that node is transmitted to it, together with the border section of the adjacent portions of the image. This eliminates unnecessary communication times.

Unlike the generalized topology of process-and-data-decomposition approach, there is no link between individual nodes of different levels. So each tree can have different number of nodes from the other trees.

The rationale of such linked-trees topology is that each tree represents different image operations/processes. If one process is more time consuming than the other processes, there will be no waiting time for other trees or nodes to finish their jobs. The time consuming process will simply be given more nodes to

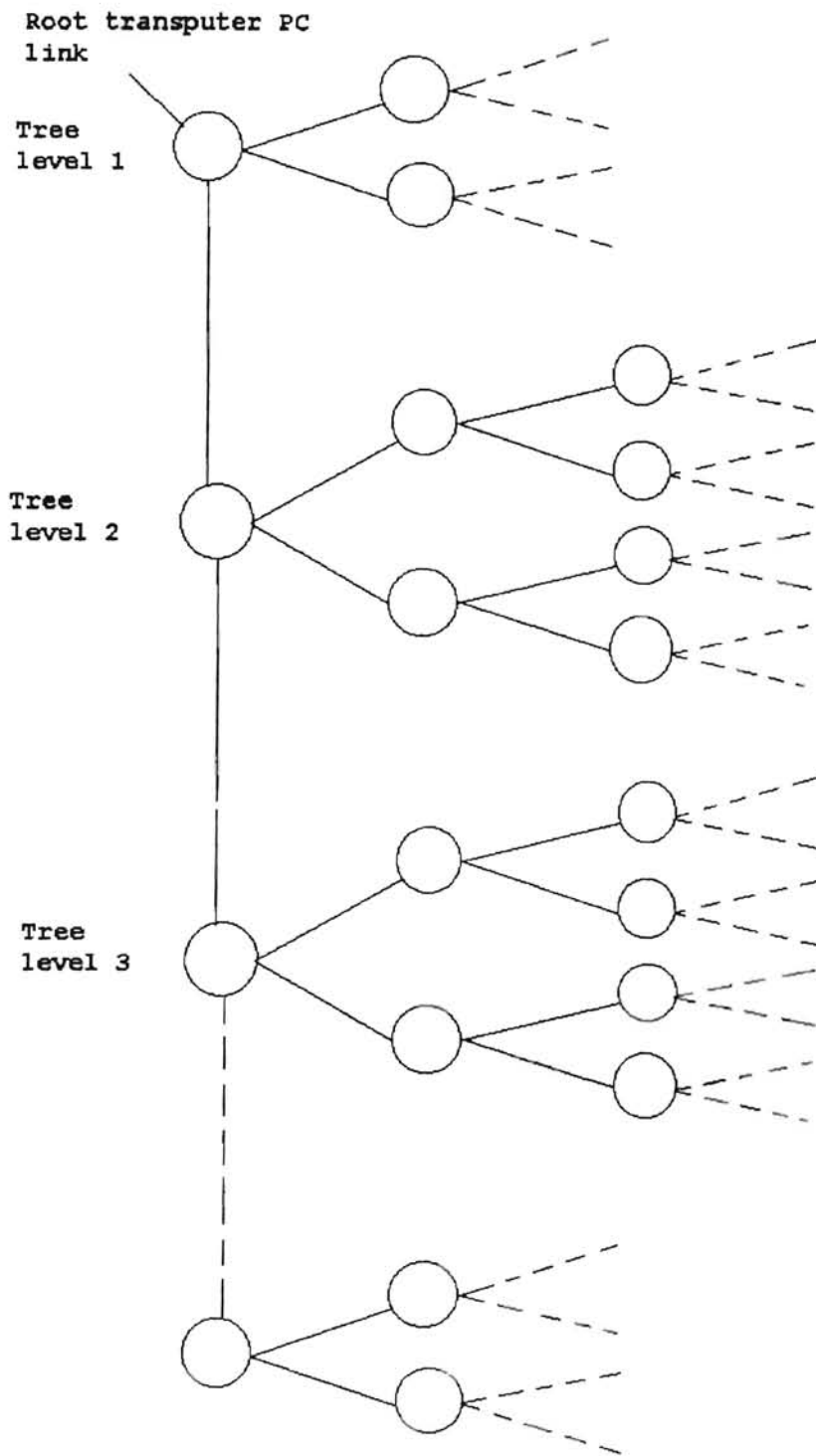


Fig. 10 Linked-trees topology (all lines represent links for both communications and system services. And each tree can have different number of nodes so as to balance load.)

finish its job faster. And in each topology, there is still decomposition of data and decomposition/pipelining of processes [Arabnia90]. So all portions of the decomposed image and all decomposed/pipelined processes will be executed simultaneously.

3.6.2 Illustration by examples

To illustrate the effectiveness of the linked-trees topology in load balancing, and its effects on the speed-up and time efficiency of a transputer network, three tasks of different computation intensiveness are used. The three tasks form a sequence of three processes and are executed on a linked-trees topology. The three tasks in sequence are: image smoothing, image thresholding, and blurring (10×10 pixels area) [Lindley91],[Duff86]. These three levels of trees form a pipeline of processes. The number of nodes in each individual tree is chosen such that each tree or process should have approximately same execution time, so one process in a pipeline does not have to wait for the other processes to finish. In this illustration, the execution time of each tree is timed and compared to see if loads in a linked-trees topology are well balanced. Also the speed-up and time efficiency of the linked-trees topology are investigated.

3.7 Results

This section will present the results of the experiments described in the previous section. The results are presented in three parts: results for implementing the farming or SIMD topology; results for pipelining of images and pipelining of segments; results for implementing linked-trees topology.

The results for experiments using network of four or more transputers are obtained by simulations. The simulations use the execution times and communication times of three nodes chain or tree networks.

3.7.1 A simple implementation of farming or SIMD topology

The topology used in this experiment is as shown in Fig 4. Results are shown in Fig.11 (only the figures for image thresholding are presented in table form) and Fig. 12. Figures in the table and all calculations are in $64 \mu\text{s}$ per unit. Fig. 12(a) shows the percentages of total communication time to the total execution time against the number of transputers used in the network. Fig.12 (b) shows the speed-up ratios against the number of transputers used in the network. Fig.12 (c) shows the time efficiencies of the topology against the number of transputers used in the network. Eight different image processes are experimented with the farming topology, namely, image thresholding, Laplace edge

enhancement, image smoothing, blurring(5 x 5 pixels area operation), blurring(10 x 10 pixels area operation), blurring(20 x 20 pixels area operation), image mirroring, and image enlargement [Lindley91],[Duff86]. Their computation intensiveness is linearly related to their size of operation.

From the table in Fig. 11, image thresholding have speed-up ratios < 1 for the farming topology using 2 or more transputers (speed-up ratios range from 0.7 to 0.9 for two to twenty transputers used in the topology). Using parallel computation for image thresholding costs more in total execution time compared with serial computation. Image mirroring and image enlargement also show speed-up ratios of less than one. Therefore, this farming topology may not be appropriate for these three processes and other processes with similar computation intensiveness in the transputer network. The time efficiencies for these three processes are not plotted, because they have no meaning if speed-up is less than 1. From the graphs in Fig.12 (b) and (c), it can be seen that the more computation intensive processes have better speed-up ratios and time efficiencies than the less computation intensive processes. It can also be observed that as the number of transputers increased, the total communication time became a dominant portion of the total execution time, the speed-up ratio slows down and efficiency decreases. The computation intensive processes are less affected, they have milder slow down in speed-up and milder decrease in efficiency. Actually, the computation intensive processes also have their slow down as the number of transputers used in the topology

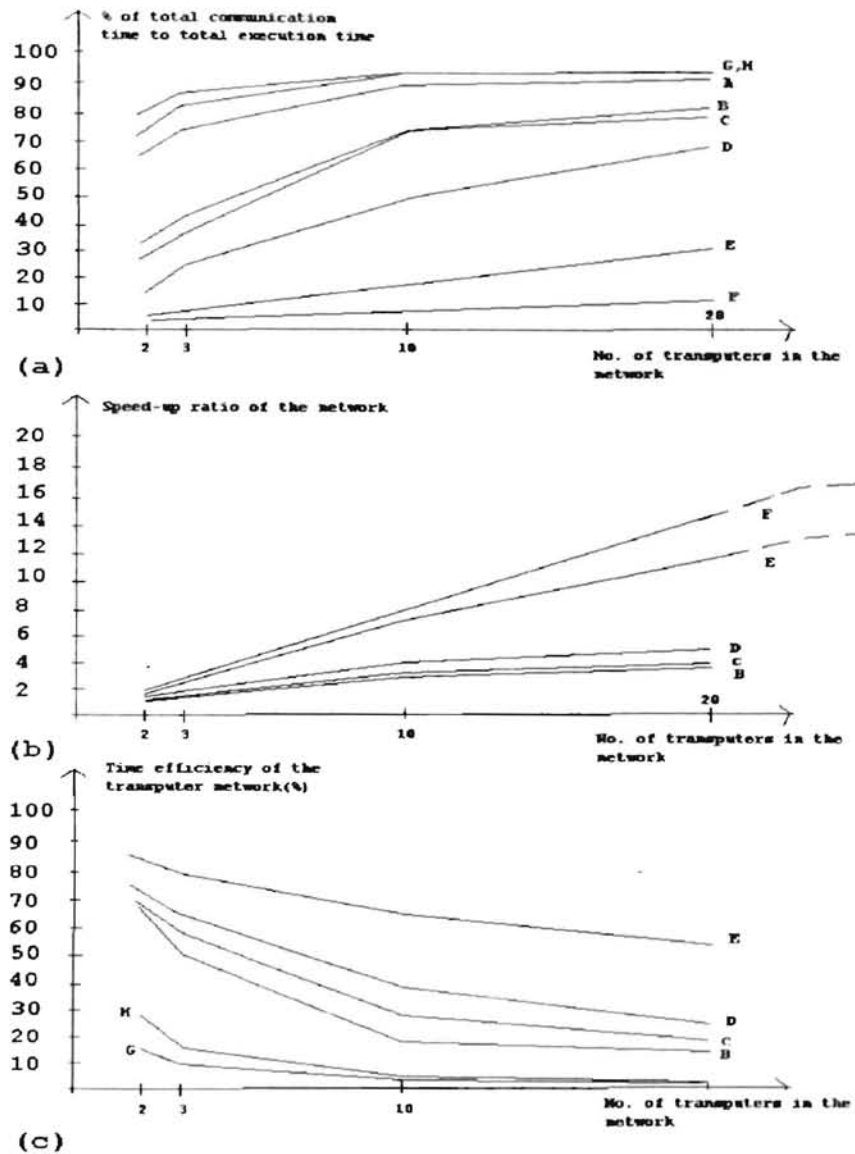
increases beyond 20 transputers. So for all image processes, the speed-up curves consist of a linear portion and a slow down portion. The dotted lines in Fig.12(b) are the slow down portion for some of the computation intensive processes. The slow down portions for processes B, C, and D are shown in the solid lines. The communication time became a dominant portion of the total execution time. This is not due to the total communication time increase. The computation time in each transputer reduces as more transputers are used in the network and therefore the communication time becomes a dominant factor.

The fact that the total communication time becomes a dominant portion of the total execution time as number of transputers in the network increases is shown by the percentages of communication graph. But again, the total communication time is less dominant for the more computation intensive processes compared with the less computation intensive processes.

It is up to the users of the farming topology [Antola91] to decide whether an image process is suitable to be parallelized with cost effectiveness based on the graph in Fig.12. But point operations are certainly not suitable for parallel processing using transputers because their speed-up is less than 1. The 1-bit array-processors may be the best alternative.

Results of applying SIMD topology in image thresholding						
No. of transputers used in the SIMD topology	Total execution time for the SIMD topology	Communication time part 1 in the algorithm	Communication time part 2 in the algorithm	Speed-up ratio of the network	Time efficiency of the network	Percentage of total communication time against the total execution time
1	17100					
2	24000	7900	7600	<1	N/A	64%
3	23900	9000	9100	<1	N/A	76%
10	19900	9100	9000	<1	N/A	91%
20	19100	9200	9000	<1	N/A	95%

Fig. 11 Experiment results by executing various image tasks on a farming topology.



- (A) Image thresholding
- (B) Image smoothing
- (C) Laplace edge enhancement
- (D) Blurring (5 x 5 pixels area)
- (E) Blurring (10 x 10 pixels area)
- (F) Blurring (20 x 20 pixels area)
- (G) Image mirroring
- (H) Image enlargement

Fig. 12 The graphs for experiment results

Total execution time for pipelining of 4 images through the network	Total execution time for pipelining of 4 images using pipelining of segments through the network	Total execution time if the 4 images are processed by serial computer	Speed-up, pipelining of images	Efficiency, pipelining of images	Speed-up, pipelining of segments	Efficiency, pipelining of segments
T'	T''	T'''	S'	E'	S''	E''
$T_1 \cdot T_2 \cdot T_3 \cdot 4$	$4 \cdot T_3$	$(t_1 + t_2 + t_3) \cdot 4$	T'''/T'	$S'/9$	T''/T''	$S''/9$
1241500	1162000	3405300	2.7	30%	3.0	33%

(a) Results for pipelining 4 images through smoothing, Laplace edge enhancement and blurring (10 x 10 pixels area) using the generalized topology.

Total execution time for pipelining of 4 images through the network	Total execution time for pipelining of 4 images using pipelining of segments through the network	Total execution time if the 4 images are processed by serial computer	Speed-up, pipelining of images	Efficiency, pipelining of images	Speed-up, pipelining of segments	Efficiency, pipelining of segments
T'	T''	T'''	S'	E'	S''	E''
$T_1 \cdot T_2 \cdot T_3 \cdot 4$	$4 \cdot T_3$	$(t_1 + t_2 + t_3) \cdot 4$	T'''/T'	$S'/9$	T''/T''	$S''/9$
271200	184400	870800	3.21	34%	4.7	52%

(b) Results for pipelining 4 images through smoothing, smoothing (2nd round) and Laplace edge enhancement using the generalized topology.

Total execution time for pipelining of 4 images through the network	Total execution time for pipelining of 4 images using pipelining of segments through the network	Total execution time if the 4 images are processed by serial computer	Speed-up, pipelining of images	Efficiency, pipelining of images	Speed-up, pipelining of segments	Efficiency, pipelining of segments
T'	T''	T'''	S'	E'	S''	E''
$T_1 \cdot T_2 \cdot T_3 \cdot 4$	$4 \cdot T_3$	$(t_1 + t_2 + t_3) \cdot 4$	T'''/T'	$S'/9$	T''/T''	$S''/9$
547900	461000	2177000	4.0	44%	4.7	52%

(c) Results for pipelining 10 images through smoothing, smoothing (2nd round) and Laplace edge enhancement using the generalized topology.

Fig. 13 Results and calculations for pipelining of images and pipelining of segments. $T_3 > T_2 > T_1$, where T_1 , T_2 , T_3 are the execution times for each level when they are processed individually. t_1 , t_2 , t_3 are the execution times for each process using serial computer.

3.7.2 Pipelining of images and pipelining of segments

The schemes shown in Fig. 9 (a) and Fig. 9 (b) are used for calculation of the total execution times. Four images are pipelined through image smoothing, blurring (10 x 10 pixels area) and Laplace edge enhancement [Lindley91], [Duff86] in order. The calculations and results are shown in Fig. 13 (a).

In a load unbalanced parallel network, communication times and idle times cannot be avoided. Both contribute to low efficiency of the parallel network. In an ideal network, every second of the processors' time should be used on productive work of computation, not on idling and communications. An ideal parallel transputer network should have 100% efficiency.

If the three processes have computation times close to each other, in other words, load in the network is well balanced, the idle time will be less. The efficiencies will be higher, because unproductive processors' idling time is less significant. Load is the execution time required for each node or level in the network. So here we explain a fact that load balancing of the generalized topology will affect the overall speed-up ratio and efficiency of the network.

If for the sake of balancing, a part of the network has very low efficiency, the gain in efficiency due to load balance will be offset by this low efficiency. This is illustrated in the

linked-trees topology later. So it is not always true that a well balanced network will increase overall efficiency.

In another experiment, the three pipelining we used are smoothing, smoothing (2nd round) and Laplace edge enhancement [Lindley91]. These three processes have close computation times. The calculations and results are shown in Fig. 13 (b).

The efficiency of this better load balanced network is better than the efficiency of the unbalanced network implementing smoothing, blurring and Laplace edge enhancement because of less idle time.

We also conducted experiments with 10 images going through the pipeline. The calculations and results are shown in Fig. 13 (c). Fig.13 (b) and Fig. 13 (c) show that the number of images pipelined will affect the overall efficiency for pipelining of image. But the overall efficiency is still limited by the percentage of communication times and amount of idle time.

3.7.3 Linked-trees topology experiments

Three processes, namely, blurring (10 x 10 pixels area), smoothing and Laplace edge enhancement [Lindley91] are executed in order on 4 images using the linked-trees topology. The topology

consists of three trees linked together for pipelining. The three trees are, tree of 50 nodes for blurring, tree of 3 nodes for smoothing, tree of 3 nodes for Laplace edge enhancement. The calculations and results for individual trees are shown in Fig.15. The calculations and results for the whole linked-trees are shown in Fig.14 (a). The results show low efficiency.

One may ask why the efficiency for a balanced network is low. The efficiency of the overall network does not only depend on the amount of idle time (which we have successfully reduced by load balancing), but it also depends on the efficiency of individual trees. In order to have execution times of the three trees close to each other, the tree with 50 nodes for blurring (10×10) is forced to have poor efficiency. So it affects the overall efficiency of the network.

If we allow some imbalance in the linked-trees topology by using only a 10 nodes tree for the blurring job, using data for a 10 nodes tree from Fig. 15 (d), the calculations and results are shown in Fig. 14 (b).

By increasing the efficiency of the tree for blurring by using less nodes, the overall efficiency has increased. The speed-up is half of the speed-up when 50 nodes are used. This network will again have better efficiency when compared to the generalized topology accomplishing speed-up of 7.6, which required a network of 30 nodes instead of 16 nodes.

We provide summarized results of Fig. 14 (a), Fig. 14 (b) and other simulations into two graphs, in Fig. 16 (a) and Fig. 16 (b). And a comparison of network efficiencies and number of transputers used for the generalized topology and the linked-trees topology accomplishing the same speed-up ratio is done in Fig. 17. One can observe that the linked-trees topology is superior in process-and-data-decomposition approach by using less nodes and having higher network efficiencies in accomplishing the same speed-up when compared to the generalized topology. The disadvantage of the linked-trees topology is: it is not capable of pipelining of segment.

One can observe that by increasing the number of nodes in the tree for blurring by 40, the speed-up is only doubled. It may not be cost effective. So it depends on the user of the topology, who needs to decide whether the speed-up or the efficiency is more important. Then he/she can choose between a perfectly balanced linked-tree and a slightly unbalanced one.

Among the linked-trees topology, the generalized topology and the farming chain topology, linked-trees topology gives better efficiency than the generalized topology while accomplishing the same speed-up ratio. That means users can use less number of nodes for linked-trees topology while accomplishing the same speed-up

Total execution time for pipelining of 4 images through the network	Total execution time if the 4 images are processed by serial computer	Speed-up, pipelining of images	Efficiency, pipelining of images
T'	T'''	S'	E'
$T_1+T_2+T_3*4$	$(t_1+t_2+t_3)*4$	T'''/T'	$S'/56$
216580	3405380	15.7	28%

(a) Results for pipelining 4 images through three linked-trees (3 nodes tree for smoothing, 3 nodes tree for Laplace edge enhancement and 50 nodes tree for blurring (10 x 10 pixels area)).

Total execution time for pipelining of 4 images through the network	Total execution time if the 4 images are processed by serial computer	Speed-up, pipelining of images	Efficiency, pipelining of images
T'	T'''	S'	E'
$T_1+T_2+T_3*4$	$(t_1+t_2+t_3)*4$	T'''/T'	$S'/56$
447639	3405380	7.6	48%

(b) Results for pipelining 4 images through three linked-trees (3 nodes tree for smoothing, 3 nodes tree for Laplace edge enhancement and 10 nodes tree for blurring (10 x 10 pixels area)).

Fig. 14 Results and calculations for pipelining through the linked-trees topology. $T_3 > T_2 > T_1$, where T_1, T_2, T_3 are the execution times for each level when they are processed individually. t_1, t_2, t_3 are the execution times for each process using serial computer.

Total execution time	Communication time part 1 in the algorithm	Communication time part 2 in the algorithm	Speed-up ratio of the network. (t is the execution time using serial computer)	Time efficiency of the network	Percentage of total communication time against the total execution time
T	Tc1	Tc2	t/T	t/T/no. of nodes	(Tc1+Tc2)/T
35591	5500	4900	1.9	63%	29%

(a) Results for 3 nodes tree for smoothing when processed individually.

Total execution time	Communication time part 1 in the algorithm	Communication time part 2 in the algorithm	Speed-up ratio of the network. (t is the execution time using serial computer)	Time efficiency of the network	Percentage of total communication time against the total execution time
T	Tc1	Tc2	t/T	t/T/no. of nodes	(Tc1+Tc2)/T
37700	5200	4700	2.1	72%	26%

(b) Results for 3 nodes tree for Laplace edge enhancement when processed individually.

Total execution time	Communication time part 1 in the algorithm	Communication time part 2 in the algorithm	Speed-up ratio of the network. (t is the execution time using serial computer)	Time efficiency of the network	Percentage of total communication time against the total execution time
T	Tc1	Tc2	t/T	t/T/no. of nodes	(Tc1+Tc2)/T
30200	6000	8000	2.3	46%	53%

(c) Results for 50 nodes tree for blurring (10 x 10 pixels area) when processed individually.

Total execution time	Communication time part 1 in the algorithm	Communication time part 2 in the algorithm	Speed-up ratio of the network. (t is the execution time using serial computer)	Time efficiency of the network	Percentage of total communication time against the total execution time
T	Tc1	Tc2	t/T	t/T/no. of nodes	(Tc1+Tc2)/T
93578	8000	8000	7.4	75%	17%

(d) Results for 10 nodes tree for blurring (10 x 10 pixels area) when processed individually.

Fig. 15 The results of the three linked-trees when they are processed individually.

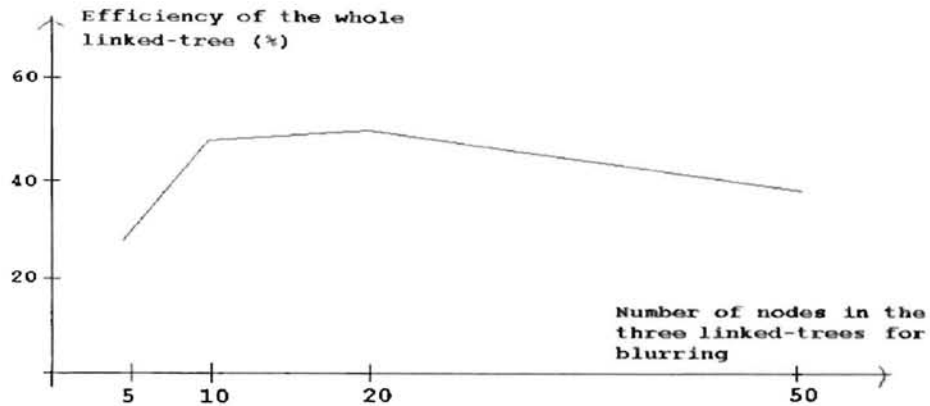


Fig. 16(a) The effect of number of nodes in the linked-tree for blurring on the efficiency of the whole linked-trees topology.

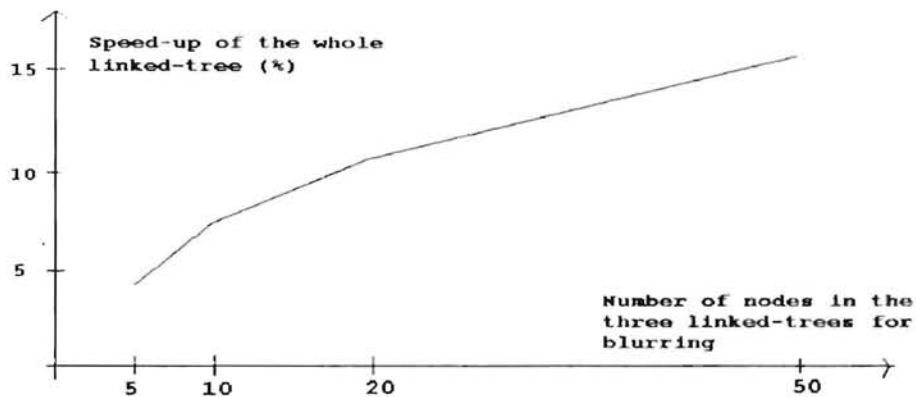


Fig. 16(b) Effect of the number of nodes in the linked-tree for blurring on the speed-up ratio of the whole linked-trees topology.

Total number of nodes in the generalized topology	Total number of nodes in the linked-trees topology accomplishing the same speed-up ratio	Speed-up ratio for both network	Efficiency for the generalized topology	Efficiency for the linked-trees topology
150	56	15.7	12%	28%
60	26	11.6	19%	44%
30	16	7.6	25%	48%
15	11	4.5	28%	41%

Fig. 17 Comparison of network efficiencies and number of transputers used for generalized topology and linked-trees topology accomplishing the same speed-up ratio.

ratio as the generalized topology. Communication times among nodes in the linked-trees topology is less than those of the generalized topology, because only the pixel values that need to be processed by each individual node will be transmitted. The farming chain topology is a special case of the generalized topology, it is similar in speed-up and efficiency to a linked-trees topology which consists of one tree.

CHAPTER IV

CONCLUSION

In this thesis, a generalized topology for image processing is proposed and its effectiveness illustrated. The effect of process computation intensiveness, percentage of communication time, load balance of the network, and the number of nodes used in the network on the efficiency and speed-up is illustrated. In order to achieve better resource utilization, a linked-trees topology is proposed. The effectiveness of the linked-trees topology in load balancing is illustrated. Load balancing is intended to improve efficiency of the network. The fact that perfectly load balanced linked-trees network may in some cases have low efficiency is illustrated. This happens due to other factors such as individual efficiencies of trees that affect the overall efficiency.

The nature of point operation and small area operation makes them very suitable for 1-bit VLSI processors array. Transputer network for image processing despite its great processing power, is not competitive and suitable for these jobs. But for high level image processing jobs and large area operations, where the 1-bit processors array is not capable or not efficient, transputer network have great potential for development. More image processing jobs with more complex algorithms need to be tried out to further illustrate the usefulness of the generalized topology and the linked-trees topology.

There is no significant difference in processing time between pipelining of images and pipelining of segments if the number of images pipelined is large. The linked-trees topology is good only for pipelining of images. But if an algorithm requires pipelining of segments and at the same time, load balance is desired to increase network efficiency, linked-trees topology is not a good choice. Exploration of topologies suitable for the above mentioned task is suggested as future work.

Bibliography

[Arabnia90]

Arabnia, H.R., "A parallel algorithm for the arbitrary rotation of digitized images using process-and-data-decomposition approach", Journal of parallel and distributed computing, No. 10, pp. 188-192, 1990.

[Antola91]

Antola, A., Tellarini, M., "Definition and evaluation of a transputer-based architecture for image compression and reconstruction", Microprocessing and Microprogramming, Vol.31, pp. 127-132, 1991.

[Ashford92]

Ashford, R.W., Connard, P., Daniel, R., "Experiments in solving mixed integer programming problems on a small array of transputers", Journal of operation Research Society, Vol. 43, No. 5, pp. 519-531, 1992.

[Crookes89]

Crookes, D., Morrow, P.J., Sharif, B., McClatchey, I., "An environment for developing concurrent software for transputer-based image processing", Microprocessing and Microprogramming, Vol. 27, pp. 417-422, 1989.

[CSA90a]

Logical Systems C for the transputer: Version 89.1 User Manual, Computer System Architects press, 1990.

[CSA90b]

Transputer Education Kits User Guild, Computer System Architects Press, 1990.

[CSA90c]

Transputer Architecture and Overview, Computer System Architects Press, 1990.

[Dasgupta89]

Dasgupta, S., Computer Architecture, a modern synthesis Vol. 2, John Wiley & Sons, 1989.

[Duff86]

Duff, M.J.B., Intermediate-Level Image Processing, Academic Press, 1986.

[EET95]

"Transputer is the core of control", Electronic Engineering Times, March 20, 1995.

[Fairhurst88]

Fairhurst, M.C., Computer Vision for Robotic Systems, Prentice Hall, 1988.

[Gupta93]

Gupta, A., Kumar, U., "Performance properties of large scale parallel systems", Journal of parallel and distributed computing, 19(3), November, 1993.

[Gray91]

Gray, J.P., Poble, F., "Object-oriented approach for transputer-based database system", Information and Software Technology, Vol.33, No.1, February 1991.

[Hersch93]

Hersch, Roger D., "Distributing Pixel Images Among Parallel Disk Arrays", Microprocessing & Microprogramming, pp. 33-36, Jan 1993.

[Hull94]

Hull, M.E.C., Crookes, D., Sweeney, P.J., Parallel Processing, The Transputer and its applications, Addison-Wesley, 1994.

[Im90]

"Parallel world of a new superpower", International Management, pp. 58-60, November 1990.

[Karkanis89]

Karkanis, S., Metaxaki-Kossionides, C., Dimitriadis, B., "A machine-vision quality inspection system for texture industries supported by parallel multitransputer architecture", Microprocessing and Microprogramming, Vol. 28, pp. 247-252, 1989.

[Kerridge94]

Kerridge, J., "Dynamic allocation of processes and channels in T9000/C164 networks using OCCAM 3", Progress in Transputer and OCCAM Research, IOS Press, pp. 1-17, April 1994.

[Kirland91]

Kirland, C., "Transputers: controllers for the 1990's", Plastics World, pp. 62-64, November, 1991.

[Lakshmi90]

Lakshmiarahan, S., Dhall, S.K., Analysis and Design of Parallel Algorithms: Arithmetic and Matrix problems, McGraw-Hill, 1990.

[Leung90]

Leung, C.H.C., Ghogomu, H.T., Mannock, K.L., "High Performance Relational Database Systems on Transputers", Application of Transputers, vol. 1, pp. 430-436, 1990.

[Lindley91]

Lindley, C.A., Practical Image Processing in C, John Wiley & Sons, 1991.

[Morrow91]

Morrow, P.J., Crookes, D., "Parallelising an image segmentation and analysis system for infra-red images", Applications of Transputers 3, vol 2, pp. 327-332, 1991.

[Mckeever92]

Mckeever, J.D.M., Holton, D.R.W., Mckeag, R.M., "Using transputers in a robotic programming and control system", Microprocessing and Microprogramming, Vol. 34, pp. 117-120, 1992.

[Mohan90]

Mohan Kumar, J., Patnaik, L.M., Prasad, D.K., "A transputer-based extended hypercube", Microprocessing and Microprogramming, Vol. 29, pp. 225-236, 1990.

[Pachowicz89]

Pachowicz, P.M., "Image processing by software parallel computation", Image Vision Computing, Vol. 7, No. 2, 1989.

[PhilipsD94]

Philips, D., Image Processing in C, R & D Publications Inc., 1994

[Phillips94]

Phillips, I., Parish, D., "On the use of transputers in Multimedia teleconferencing system", Progress in Transputer and OCCAM Research, IOS Press, pp. 148-154, April 1994.

[Russ92]

Russ, J., The Image Processing Handbook, CRC Press, 1992

[Schomberg89]

Schomberg, H., "Image Processing on a Transputer-based Perfect Shuffle Machine", Microprocessing & Microprogramming, pp. 277-280, Jan 1989.

[Stalling90]

Stalling, W., Computer Organization and Architecture, MacMillan, 1990.

[Stalker91]

Stalker, M.D., "Simplifying parallel programming on the transputer network", Computer Technology Review, pp. 15-22, summer 1991.

[Stallard93]

Stallard, P.W.A., Duno, R.W., Daniels, A.R., "Dynamic real-time scheduling for a parallel production system on an enhanced transputer array", Transputer and OCCAM Research: New Direction, IOS Press, pp. 218-231, March 1993.

[Tomohisu93]

Tomohisu, k., Motok, O., Yoshio, S., Atuo, K., "Parallel image processing for defect inspection by layered structure of transputers", Transputer/OCCAM Japan 5, pp. 161-170, June 1993.

[Uhr87]

Uhr, L., Parallel Computer Vision, Academic Press, 1987.

[Zomaya92]

Zomaya, A.Y., "On the fast simulation of direct and inverse Jacobians for robotic manipulators", Robotics and Autonomous Systems, Vol.10, pp. 43-61, 1992.

2
UITA

Kit Sai Wong

Candidate for the Degree of

Master of Science

Thesis: PROCESSOR TOPOLOGIES FOR IMAGE PROCESSING
 APPLICATIONS

Major Field: Computer Science

Biographical:

Personal Data: Born in Hong Kong, on June 24, 1959, the son of
 Wong Man Sum and Pong Yip Moi.

Education: Graduated from Pentecostal School, Hong Kong in
 1977; received the Bachelor of Engineering degree in
 Mechanical Engineering from National University of
 Singapore, Singapore in 1985; completed the requirements
 for Master of Science degree at Oklahoma State
 University in May, 1997.

Professional Experience: Engineer, Seagate Technology Ltd.,
 Singapore, March 1986, to March 1987; Engineer,
 Micropolis Ltd., Singapore, March 1987, to October 1989;
 Engineer, Hong Kong Polytechnic, Hong Kong, October
 1989, to July 1991.