SRC TR 88-19

The Kyklos Multicomputer Network
Interconnection Strategies
Topological Properties
Applications

by

B.L. Menezes

THE KYKLOS MULTICOMPUTER NETWORK:

INTERCONNECTION STRATEGIES

TOPOLOGICAL PROPERTIES

APPLICATIONS

APPROVED BY SUPERVISORY COMMITTEE:

Carter F. Womack

Ray Mercer

J C Brown

Dedicated to my parents

THE KYKLOS MULTICOMPUTER NETWORK:
INTERCONNECTION STRATEGIES
TOPOLOGICAL PROPERTIES
APPLICATIONS

by

BERNARD LOURDES MENEZES, M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May, 1988

ACKNOWLEDGEMENTS

I am especially indebted to my advisor Dr. R.M.Jenevein for his guidance and encouragement and for the many fruitful discussions related to this dissertation. I wish to thank my co-advisor Dr. G.J.Lipovski for providing some of the motivation for this work through the Lookahead Network. I am thankful for the financial support provided by Dr. J.C.Browne and Dr. A.G.Dale and also for the opportunity to interact with them. Finally, I am grateful to Drs. R.M.Mercer, B.F.Womack and C-L Wu for serving on my committee.

I wish to acknowledge the help of my friends and colleagues in the course of my study. In particular, Ashok Adiga, Sanjay Deshpande, Brian Garvey and Jane Garvey were helpful in the preparation of this document. Lastly, part of my work was supported by the MCD Fellowship awarded through the Department of Electrical and Computer Engineering.

The task of formatting this dissertation was greatly eased by use of automated document processing. The Scribe document formatter was conceived of and created by Brian Reid. The current version has been maintained and enhanced by Unilogic, Ltd. The Scribe format definitions for proper dissertation format for The University of Texas at Austin were developed by Richard Cohen.

Bernard Lourdes Menezes

The University of Texas at Austin
May, 1988

The KYKLOS Multicomputer Network:
Interconnection Strategies
Topological Properties
Applications

Publication No. _____

Bernard Lourdes Menezes, Ph.D.
The University of Texas at Austin, 1988

Supervising Professors: Roy M. Jenevein and G. Jack Lipovski

KYKLOS, a new multiple tree-based interconnection network for multi/parallel processing systems, is proposed. This architecture provides fault tolerance and asymptotic improvements in performance while retaining the simplicity and low fanout of the binary tree unlike other augmented tree architectures proposed in the literature.

Different topological variations of KYKLOS are presented. In particular, the interconnection for a dual tree KYKLOS involves a shuffle of the links in one of the two trees. Simple distributed routing strategies are defined for KYKLOS and it is shown that the cross product: {Topology × Routing Strategy} maps to different distance and traffic characteristics. Maximum link traffic grows subquadratically with network size ($O(N^{1.5})$), while the cost of the network increases only linearly with the number of processor resources. Also, normalized communication latencies are shown to be superior to competing tree topologies and to the Hypercube.

Applications of this network topology to facilitate *parallel* access to I/O and *parallel* processing of relational join operations is explored. Finally, it is shown that KYKLOS has excellent potential in minimizing network diameter under the constraints of maximum node degree for a given network size.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Future applications in fields such as fluid dynamics and geophysical modeling will require processing in rates far in excess of what today's supercomputers can deliver. Single processor computers like the CRAY-1 and Fujitsu VP200 are already within an order of magnitude of their technological limit of 3 gigaflops imposed by the speed of light. What is required for the next generation of computers is a detour of the von Neumann bottleneck through the use of parallelism.

1.1 Parallel Processing and Interconnection Networks

Efforts at exploiting parallelism to achieve speedup necessitate the partitioning of a problem into tasks and subtasks to be executed on multiple processing units. The introduction of a medium to facilitate communication and synchronization between these processing units is an essential aspect of recent and future computing systems. It is this medium of communication that is referred to as the *interconnection network (ICN)*.

Parallel or distributed systems are generally classified as either *multiprocessors* or *multicomputers*. Multiprocessors permit all processors to share a common memory, hence they are sometimes referred to as shared memory systems. Multicomputers, on the other hand, operate by passing messages. In the former, the ICN is used as a communication medium between processors and memory, in the latter case it is used for interprocessor communication.

The performance of a parallel system is dependent on the ability of the processing elements (PE's) to communicate data and synchronization primitives in a smooth and efficient manner. This means that the ICN should be capable of providing adequate bandwidth with low communication latencies. In addition, the ICN should have fault tolerance built in and provide acceptable levels of performance even in the presence of faults. Finally, concerns of cost, simplicity, etc. can hardly be emphasized. Because the ICN is such a vital component of a parallel processing system, a thorough understanding of the attributes of the ICN in relation to its impact on system performance is an active area of research. This dissertation will propose a novel way of interconnecting processors called KYKLOS and will study the properties of this network.

The next section introduces a taxonomy for ICN's. In Section 1.3, important issues that relate to ICN design are presented. Tree topologies are surveyed in Section 1.4 and the dissertation outline concludes this chapter.

1.2 Interconnection Networks: A Taxonomy

Feng has classified ICN's on the basis of the following four design decisions[FENG81]

- **Network Topology:** Network topologies are classified as either *static* or *dynamic*. Static topologies have passive links while dynamic topologies are characterized by having links that can be reconfigured by setting the network's active elements.

- **Switching Methodology:** Data may be transferred using either *circuit* or *packet* switching. In circuit switching, once a device is granted a path in the ICN, it will occupy that path for the duration of the data transfer. In packet switching, the information is broken into small packets that individually compete for a path in the ICN.

- **Control Strategy:** The control strategy may be either *centralized* or *decentralized.* In the former, all requests are processed by a central controller. In decentralized systems, requests are handled independently by different devices in the ICN.
- **Operation Mode:** Systems can operate in either *synchronous* or *asynchronous* mode. In the former, a global clock broadcasts the clock signal to all devices so that they operate in lockstep fashion. Asynchronous systems, on the other hand, support independent operation of the different elements in the network without a global clock.

Since network topology is the main focus of the dissertation, we take a close look at the classification of ICN topologies. in the remainder of this section.

1.2.1 Dynamic Topologies

There are three types of dynamic topologies: crossbar, single stage and multistage. The classes and subclasses are shown in Fig 1-1.

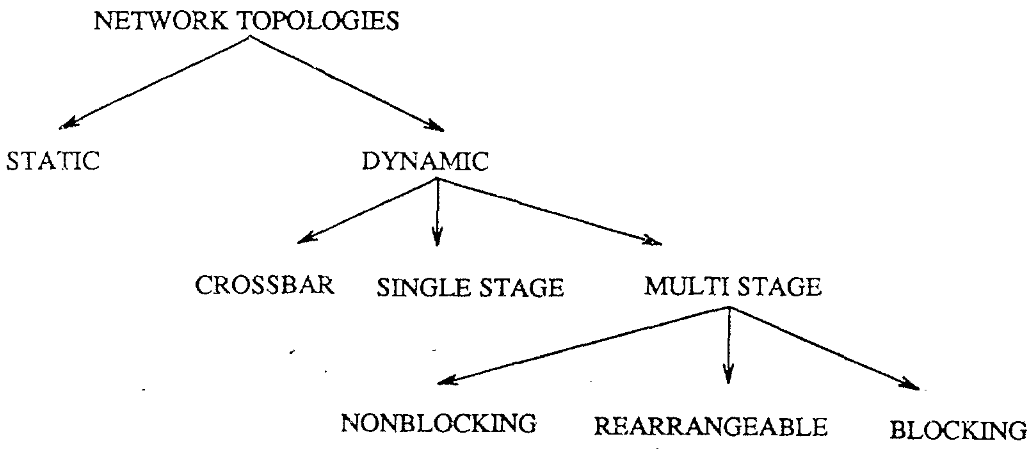


Figure 1-1: Taxonomy of Network Topologies

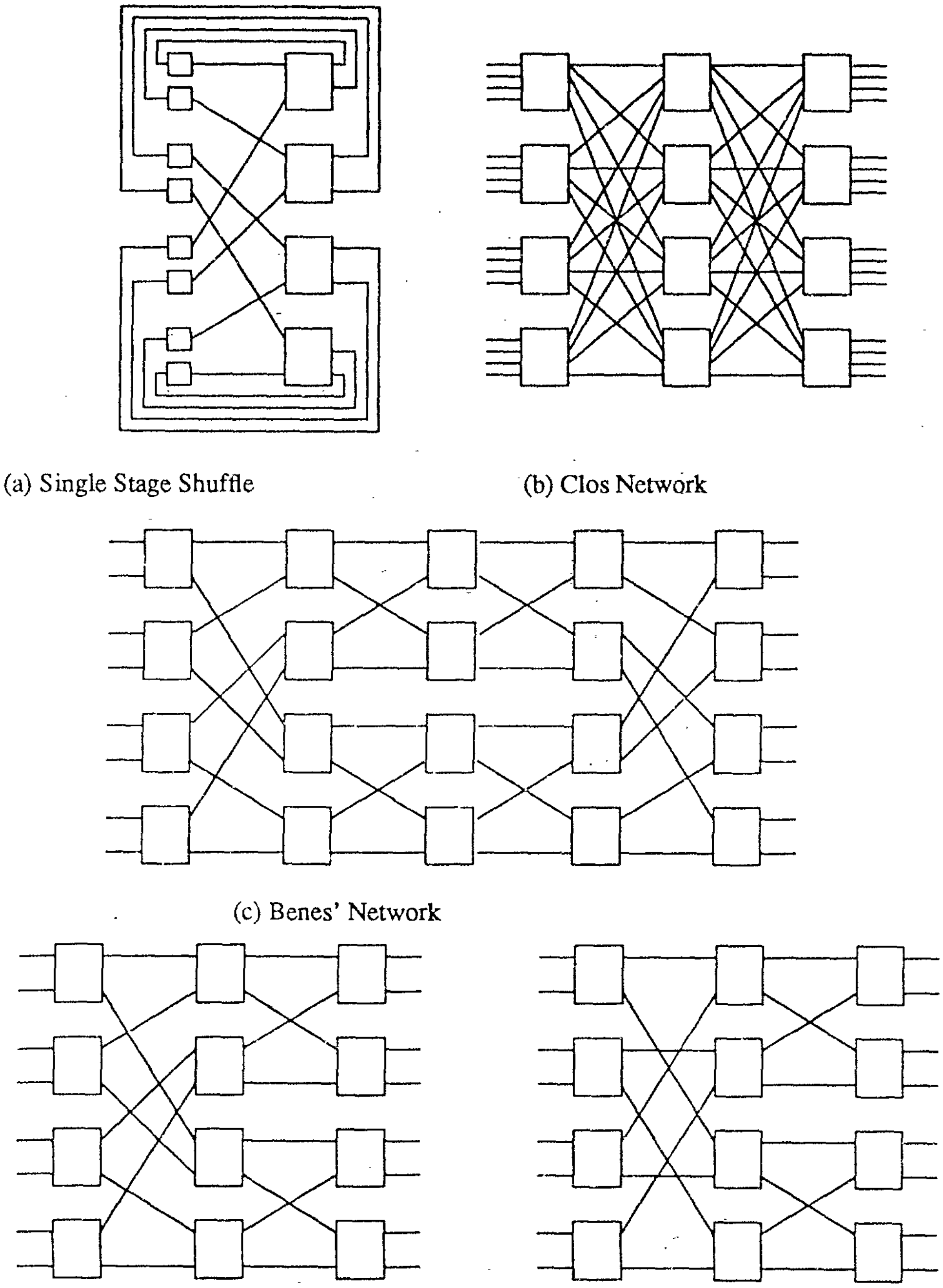
Crossbar Networks are those in which every input is connected to every output.

Single Stage Networks, as the name suggests have a single stage of switching elements (switches). One of the best known examples in this class is the *perfect shuffle* network (Fig. 1-2(a)) first proposed by Stone [STON71]. Here, the inputs at the leftmost end are permuted before being fed to the array of switches while the outputs are wrapped around to the input end as shown.

Multistage Networks are made up of several stages of switches. These networks are further subdivided into three subclasses depending upon their ability to support simultaneous connections of more than one input-output pair without conflicts. A *nonblocking network* [CLOS53] can handle all possible connections without conflict (Fig. 1-2(b)). A *rearrangeable nonblocking network* [BENE62] can perform all possible connections between inputs and outputs by rearranging its existing connections (if necessary) so that a connection path for a new input-output pair can always be established (Fig. 1-2(c)). Finally, in *blocking networks*, simultaneous connection between two or more input-output pairs may cause conflicts in the use of communication resources.

All the above categories of multistage networks have been researched in connection with telephone switching [BENE65]. The last decade, however, saw the emergence of the blocking multistage networks for a new application viz. multiprocessing systems. These networks (henceforth called MIN's[1]) are characterized by the fact that they have N inputs and N outputs with $\log_m N$ stages of switches. Each switch is an $m \times m$ crossbar which can realize any permutation between its m inputs and m outputs. Goke

[1] acronym for Multistage Interconnection Network.



(a) Single Stage Shuffle

(b) Clos Network

(c) Benes' Network

(d) Baseline MIN

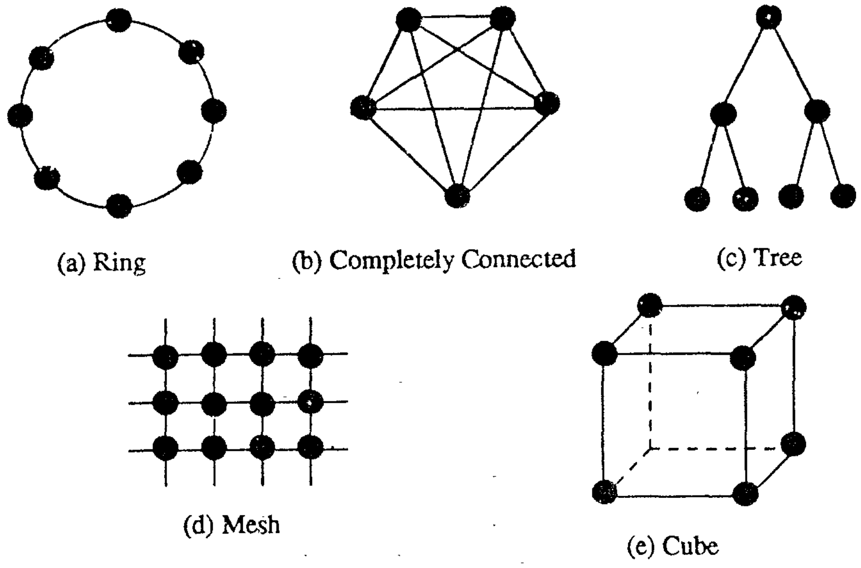
(e) Banyan MIN

Figure 1-2: Dynamic Topologies

and Lipovski proposed the banyan network [GOKE73] (Fig. 1-2(d)) for partitioning multiprocessor systems. Other MIN networks that have been proposed include the Omega[LAWR75], Baseline [WU80a], Flip [BATC76], multistage Cube [SIEG81], etc. Wu and Feng [WU80b] showed that these networks were all isomorphic i.e. topologically equivalent. Much work has been reported in the literature related to their performance, both circuit and packet, fault tolerance, VLSI layout, mapping, etc. A survey of this work in tutorial form appears in [WU84].

1.2.2 Static Topologies

These networks are often characterized by the fact that each node is a small computer with its own local memory. Static network topologies have been classified on the basis of node degree. On one end of the spectrum are simple ring topologies (Fig. 1-3(a)). At the other extreme, one finds the completely connected network (Fig. 1-3(b)). Important topologies of networks in this class include the full binary tree, rectangular mesh, n-cube (Fig. 1-3(c)-(e)), multiple buses and the cube-connected cycles network proposed by Preparata [PREP81].



(a) Ring

(b) Completely Connected

(c) Tree

(d) Mesh

(e) Cube

Figure 1-3: Static Topologies

1.3 Issues in ICN Design

Multicomputer Network topologies are judged on several criteria, some of the more common being

1. Message Latency
2. Fanout
3. Cost
4. Traffic Load
5. Ease of Routing
6. Fault Tolerance
7. Scalability

Message Latency: There are several evaluative measures for communication latency. The maximum internode distance, called **diameter**, denoted k, is a useful upper bound on communication delay.

A measure that better captures the communication delay is **expected delay**. This is the average number of links a message would traverse in order to reach its destination. This depends on the algorithm being implemented and also on the mapping of the algorithm onto the architecture. For example, if the algorithm and mapping are such that each node has need to communicate only with its nearest neighbors, then the average communication latency is simply 1.

On the other hand, in the absence of any a priori knowledge of an algorithm and/or the goodness of a mapping technique, it behooves the network designer to attempt to minimize the **average communication latency**. This assumes that every node may send a message to every other node with equal probability. Besides the uniform message distribution, other models have been proposed [REED87]. These assume that the probability of a message exchange between two nodes is a function of distance between them. For example, the probability may be an inverse function of distance.

Fanout: This is the number of communication ports supported by each node. The design of a communication port will depend *inter alia* upon whether circuit or packet switching is being used. In the latter, the port should have the circuitry to buffer in an incoming packet, read the destination tag and place it in the appropriate output queue. In a circuit switched environment, the logic to handle requests for virtual circuit reservation with the additional demands on guaranteeing fairness, preventing deadlock, etc. need to be addressed.

Cost: We define cost as being related to the number of links that are used to connect all the computers in a multicomputer network or the processing elements in a multiprocessing network. For networks with uniform degree, the cost of the links is simply the number of nodes times the degree. Also, the cost of the communication hardware at the ports in the nodes of the network needs to be considered. In MIN's, the additional cost of the switches should be factored in. Finally, in a VLSI environment, the cost associated with layout is relevant.

Traffic Load: Network traffic is a function of several parameters related to the topology such as average internode path length and the number of links. In symmetric networks, i.e. those networks in which each node has the same view of the network as any other, the traffic through each link would be the same under the *uniform message distribution* assumption. The traffic density would be proportional to the time each message spends in transmission (ignoring queueing delays). This in turn is related to the average path length. Hence **message traffic density** is

$$T_{avg} = \frac{No.\ of\ messages\ \times\ Avg.\ path\ length}{No.\ of\ links}$$

In non-symmetric networks, the message density would, in general, be nonuniform. In the network of Fig. 1-4, for example, suppose that every pendant node (nodes on the extreme right and extreme left) needs to send a message to every other pendant node. Clearly, the east-west artery between node_a and node_b would carry a total of $N^2/4$ messages in both directions and be the bottleneck. Note that the other links would not carry as much traffic. While links on the same longitude would see the same number of messages, link traffic density would increase monotonically from left to right until node_a and then taper off beyond node_b. Traffic through this network is therefore asymmetric (nonuniform) unlike networks like the ring or the hypercube that are inherently symmetric.

The network of Fig. 1-4 is really a binary tree with the two halves rotated 90^o away from each other. The congestion between node_a and node_b is commonly referred to as the *root bottleneck problem.*

Fault Tolerance: A network is said to be k-fault tolerant (k-ft) if it can *survive* the failure of any set of k arbitrary node faults. The *survivability criterion* may be interpreted as connectivity of the network or of a given part thereof [GREY84]. Alternatively, the fault tolerance criterion may be the survival of a completely defined subnetwork [HAYE76] (for example, a 32 node binary cube within a 64 node Hypercube.

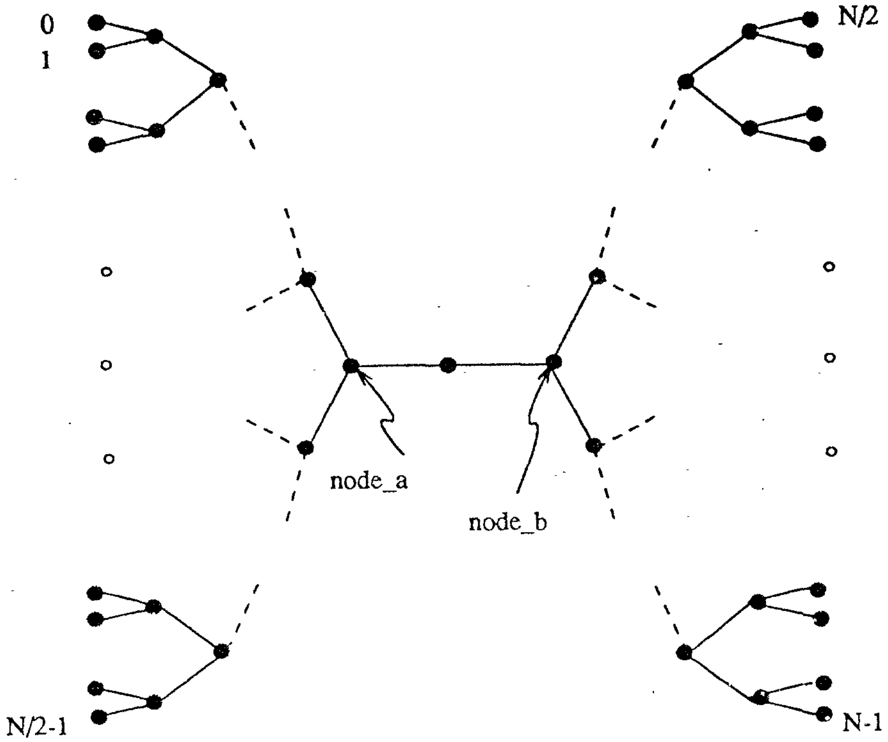


Figure 1-4: Root bottleneck Problem in Binary Trees

Ease of Routing: It is essential that every network have algorithms which enable routing of messages from every node to every other node. For some topologies, there may be several routing algorithms, each with different merit. For example, shortest path routing in a network may not be the simplest but it will guarantee the least communication delay [GOOD81]. A common form of routing called distributed tag routing encapsulates the destination address or some function of source and destination address (such as their exclusive or) in the data packet. Every intermediate node on the path between source and destination checks one or more bits of the tag which determines which link it should be sent through. This technique is common for hypercube, tree and banyan networks.

There are other related issues in ICN design. These include such concerns as **extensibility** or **scalability** which is the ability of a large system to scale up with a minimum of disruption to the existing setup. Another issue relates to the power of a given network to **embed** useful topologies. For example, an algorithm well mapped to an architecture x may be easily implementable on a given network, y, if y embeds x. For example, an n-cube embeds a full binary tree of height n-1 [DESH86].

1.3.1 Design Tradeoffs

As in other areas of science and engineering, design of networks inevitably warrants making tradeoffs in the midst of differing and often conflicting requirements. For example, the demand of decreased communication latency could be accomplished at the cost of increasing the number of links. This could, in addition, increase node fanout. On the other hand, reduced communication latencies would decrease the average link traffic and could, under certain circumstances, alleviate traffic bottlenecks.

A metric that takes both, average distance and fanout, into consideration is simply their product, called **normalized distance** [AGRA86], and denoted L'. Another metric that relates diameter and degree appears as an upper bound on the number of nodes that can be packed into a graph. Formally referred to as the **(d,k) Graph Problem** it may be stated thus:

Given a maximum degree d and a maximal distance k, find a graph of maximal order.

A variation of the problem is

Given a number of nodes, interconnect them minimizing both the number of edges between nodes (degree) and the distance between nodes.

This problem has been popular both, from the viewpoint of computer networks as well as a theoretical problem in its own right. There have been a plethora of papers on the subject in recent years [ELSP64, AKER65, ERDO66, STOR70, ARDE78, IMAS81, MEMM82, DOTY84]. An upper bound on the number of nodes, n, has been expressed in terms of the diameter (k) and the maximum degree (d) and is referred to as the Moore bound.

$$n(d,k) \leq \frac{d(d-1)^k-2}{d-2}$$

Clearly, networks designed to optimize n will have more computing resources for as few connections as possible (certainly no more than d per node) without at the same time exceeding a maximum communication latency of k between any two nodes of the network.

1.3.2 Designing in multi-dimensional Space

Concerns about cost and performance have traditionally dominated the design of computer systems. As computer applications proliferated and the demand for reliable, high-performance systems grew, it became increasingly clear that fault tolerance had to be factored in during the design phase, not as an afterthought. The repercussions of this new dimension to thinking and designing have produced many new architectural enhancements for fault-tolerant MIN's, which are surveyed in [ADAM87]. This chapter will conclude with a brief look at how the seemingly unrelated concerns of high performance and high reliability[2] may be integrated in the area of computer networks with a focus on KYKLOS.

Apropos of the discussion of designing systems that combine performance and fault tolerance: it is often necessary to maintain a working system in the presence of faults

[2] This concept, sometimes called integrity management, was the focus of three workshops held this year at the University of Texas

that can guarantee certain performance levels. The ability of a system to perform at or above certain specified levels in the presence of faults characterizes the system as **gracefully degradable** [CHER85].

The combined provision for fault tolerance and high performance: There are two types of networks in general - those that are inherently fault-tolerant and those that are not. The former possess multiple redundant paths which can be used to bypass faults in the network. Such examples are the mesh and the binary cube. On the other hand, a tree network is not intrinsically fault-tolerant. Could the injection of redundancy to provide fault tolerance be used, *in addition*, to enhance the performance under the *non-fault condition*? Could the effects of this technique on fault tolerance and performance be somehow quantified? What are the implications of this strategy to cost? This dissertation will attempt to answer some of these questions by first proposing the KYKLOS augmented binary tree topology and then understanding its properties.

A brief survey of tree architectures in multiprocessing systems is undertaken next.

1.4 Tree Topologies in Multicomputer Systems

Trees play an important role in algorithmic studies. For example, divide and conquer algorithms lend themselves naturally to tree structures. It is not surprising that tree interconnection schemes perform well on this class of problems.

Algorithms on tree machines for sorting, matrix multiplication and for solving several NP-complete problems are described in [BROW78]. Here, a binary tree of n processors can be used to sort n numbers in O(n) time it takes to load and remove the numbers. For multiplying two n×n matrices on tree machines, the basic divide-and-conquer step splits the multiplicand into rows and the multiplier into columns. The multiplication can be perfomed in O(n^2) time and requires $2n^2$-1 processors. If the data paths between nodes can handle an entire column at once, the multiplication can be performed in O(n) time.

Tree architectures have been used in a variety of database machines. The Non-Von Database Machine[SHAW79] used a hierarchical associative architecture to execute relational algebra primitives more efficiently than the single level associative processor designs. In this design, the primary associative memory(PAM) is organized as a tree machine with a large number of processors implementable in VLSI.

The binary tree has also been used in such commercially available machines as Teradata Corporation's Database Computer [EHRE84]. Highly Concurrent Tree Machines [SONG80] have been used to solve problems in DBMS. For example, two complete binary trees connected in a mirror image fashion have been employed to perform sorting in O(log N) time. One of the trees broadcasts streams of data and instructions and the other tree is used to combine and route outputs (Fig. 1-5).

Many database specific algorithms executed on tree-based architectures involve some sort of a merge operation on sorted streams of data. This technique has been widely used for performing joins, duplicate elimination, global sorts, intersections, unions, etc. The effect of this technique is to meet the need to compare one leaf node's data with that of many other leaf nodes.

A problem with these operations is traffic congestion near the root which saturates the bandwidth of links at or near the root. Thus the advantages of logarithmic delay and the recursive, scalable structure of a tree are offset by the traffic bottleneck problem.

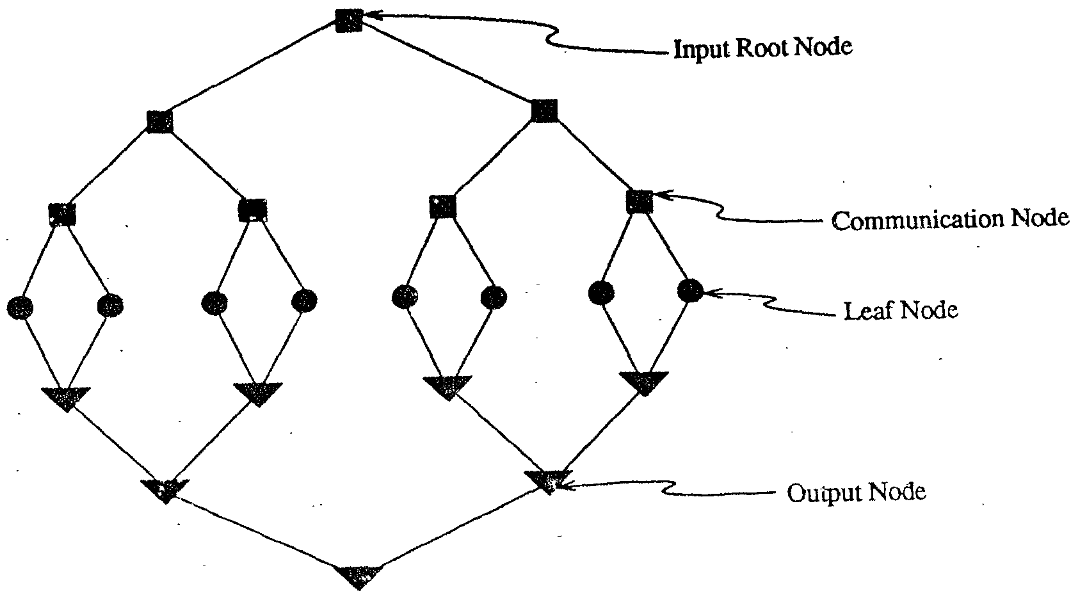


Figure 1-5: Double Tree for Broadcast/Merge

In addition, the tree structure is not tolerant of faults since a single link fault or a single non-leaf node fault can destroy its connectivity.

There have been many approaches to the rectification of these problems. These include the addition of extra nodes or links for redundancy or to alleviate the traffic bottleneck problem. A few of these are described next.

Fault tolerance was the major motivating factor in a scheme proposed by Hayes[HAYE76]. Here an extra node at each level of the binary tree was added and extra links were introduced between nodes at adjacent levels as shown in Fig. 1-6. It was shown that the redundant structure could sustain the loss of any node and still contain a subgraph in the resulting structure isomorphic to a full binary tree. The redundant graph was shown to be optimal with respect to the number of links added.

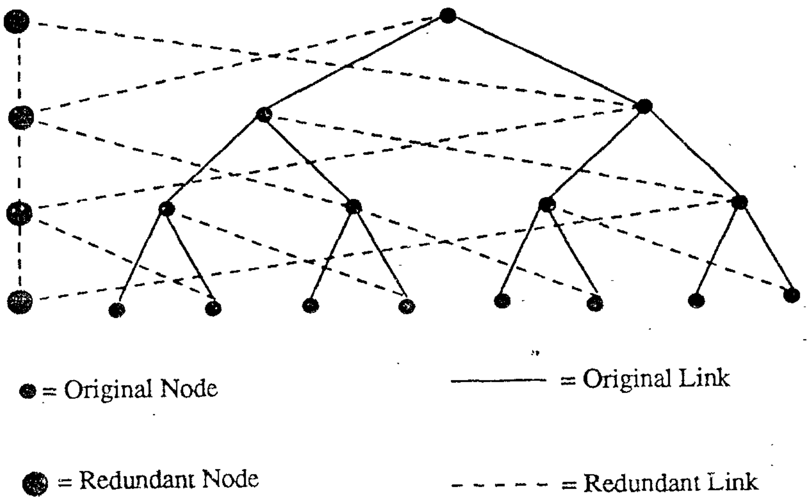
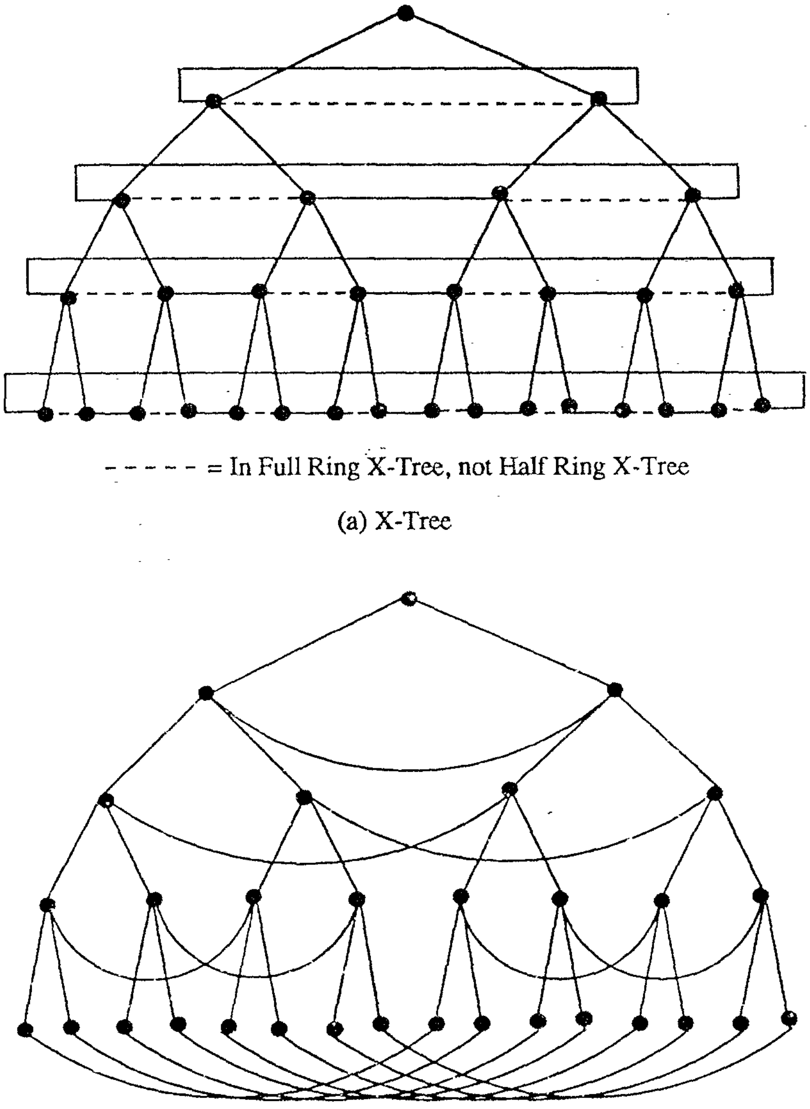


Figure 1-6: Scheme for fault tolerance proposed by Hayes

Half-ring and full-ring X-tree structures were proposed by Patterson and Despain [DESP78]. In these structures, extra links were added between nodes at the same horizontal level of the tree as shown in Fig. 1-7(a) and 1-7(b). Another interesting structure is the Hypertree [GOOD81], so called because extra links were added between nodes at the same level in the tree that differed by a single address bit as in the Hypercube (Fig. 1-7(b)). While both of these structures improve the traffic bottleneck problem, neither of them is fault-tolerant with respect to a single node failure in the sense that the non-faulty subgraph does not embed a full binary tree of the same size as the original irredundant tree.

Leiserson proposed Fat-trees [LEIS85] as a means of interconnecting processors as a binary tree in which the number of wires connecting a node with its parent increase as one goes up the tree. This provides higher communication bandwidth at the higher levels of the tree, thus ameliorating the traffic bottleneck problem.



------ = In Full Ring X-Tree, not Half Ring X-Tree

(a) X-Tree

(b) HyperTree

Figure 1-7: X-Tree and Hypertree

1.5 KYKLOS in Perspective

As mentioned earlier, KYKLOS is an attempt to integrate the concerns of fault tolerance and performance in binary tree architectures. A brief history of the network is presented and this is followed by an outline of this dissertation.

1.5.1 History

The primary motivation behind KYKLOS was to provide fault tolerance in the tree-structured Lookahead Local Area Network [LIPO82]. Perhaps the simplest way to do so, it was felt, would be to add a second tree so that the trees would share the same set of leaf nodes, in this case workstations. The Double Tree would, besides providing fault tolerance, double the potential bandwidth of the network (much like a dual bus system would under similar circumstances). The bottom tree was to be a mirror image of the top tree. It soon became apparent that other useful properties could also be reaped by the addition of the extra tree and by altering the interconnections in the bottom tree. This motivated the consideration of other interconnection structures in which a mere rearrangement of network resources (links or nodes) in the double tree would bring about an improvement in average communication latencies over and above the already low communication distances afforded by the single tree or the simple Double Tree. Moreover, the root bottleneck problem, a characteristic of the tree architecture, might also be ameliorated. Finally, the low fanout of the nodes and the linear cost of the network (with respect to leaf resources) were intentionally placed at a premium.

In a nutshell, the concerns of simplicity, fault tolerance and performance were to be integrated and a solution that addressed these was envisioned in the form of KYKLOS.

1.5.2 Dissertation Outline

In Chapter II, multiple tree networks are introduced. Both KYKLOS-I and KYKLOS-II are defined. A constructive definition of KYKLOS-II is presented. It is shown that, using permutations and labeling sequences, any KYKLOS topology may be defined. An isomorphism of KYKLOS-II called the W-Form, is constructed and it is shown that KYKLOS-III, which embeds a ring between a double tree, may be generated quite easily from KYKLOS-II. Finally, a theory of optimality for KYKLOS networks is enunciated and the different KYKLOS topologies are examined in the light of this optimality criterion.

Various routing strategies for KYKLOS-II are described in Chapter 3 and the interprocessor distance distribution for each is computed. It is shown that the distance distribution has a nice generalized Fibonacci form. The solution of the characteristic equations of the recurrences describing the distribution are useful in understanding the asymptotic properties of the network. Reach Factor and average interprocessor distance for each routing strategy are computed and compared with KYKLOS-I.

In Chapter 4, the traffic density through the different links in the network is examined under different routing strategies. The concept of a slice in the KYKLOS-II network is introduced. A new routing strategy which uses this concept to minimize traffic is defined and an algorithm for implementing it is outlined. Maximum traffic density and link utilization as a function of routing strategy and network size are investigated.

Fault tolerance of KYKLOS is investigated in Chapter 5. Two aspects of fault tolerance provided by KYKLOS-II are attacked. The ability of KYKLOS-II to withstand double node faults is compared with that of KYKLOS-I. Finally, the effect of a single node fault on performance degradation is investigated.

In Chapter 6, the use of KYKLOS-II as the ICN for an I/O Engine is proposed. Its suitability for implementing a distributed Join Processing Algorithm is examined in relation to KYKLOS-I. The strengths of the modified three-tree KYKLOS as a candidate solution for the d,k Graph Problem are explored. Finally, spinoffs of KYKLOS in the form of HyperKYKLOS and the SK-Banyan are described.

In Chapter 7, an account of the work performed on KYKLOS is summarized and the major accomplishments are highlighted. Also, suggestions for future work are made.

Chapter 2

Interconnection Strategies

In this chapter, we look at interconnection strategies for multiple tree networks. More specifically, three versions of the m-ary double tree network are presented. Section 2.1 introduces the basic KYKLOS-I multiple tree-structured topology. The KYKLOS-II network is defined in Section 2.2 and an alternative definition using labeling sequences (LS's) is presented. In Section 2.3, an isomorphism of KYKLOS-II is constructed which sets the stage for the introduction of KYKLOS-III. Finally, in Section 2.4, the concept of m-sense optimality in a class of KYKLOS networks is presented.

2.1 Multiple Tree Networks

The general form of KYKLOS is an interconnection network consisting of r sets of m-ary trees joined so that they share a common set of leaf nodes. A three-dimensional view of such a network is shown in Fig. 2-1. Here, the original ternary tree (shown in dark links), except for the 9 leaf nodes, is replicated three times.

2.1.1 KYKLOS Nomenclature

We employ the notation KYKLOS-x<m,r,n> to represent any member of the KYKLOS family of interconnection networks. Here

x is the *version number*[3] representing the connection strategy,

r is the *tree replication factor*,

m is the *tree branching factor*,

[3] Roman numerals used

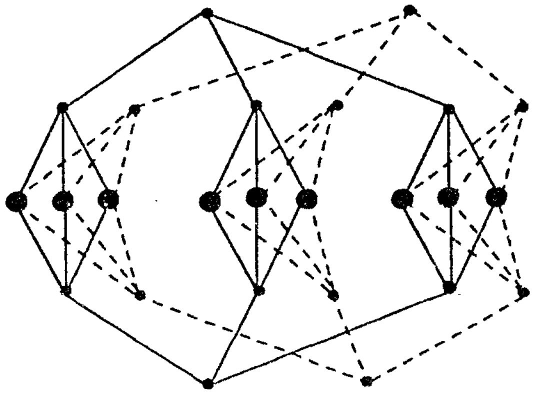


Figure 2-1: A 4-Tree Ternary KYKLOS

and

n=$\log_m$N is the *height of a single tree*[4] of N leaf nodes.

Unless otherwise stated, processors will be located at the leaf nodes and switches at the non-leaf nodes. Both, switches and processors perform routing and buffering of data. However, it is assumed that the switches have far less processing capability and no access to I/O except in the case of the switches located at the root nodes.

2.1.1.1. Node Labels

The addressing scheme for nodes is hierarchical with each node being represented by a triple.

The i^{th} node at the j^{th} level of the k^{th} tree is referred to as node <k,j,i>

$1 \leq k \leq r$,

$0 \leq j \leq n$,

$0 \leq i \leq m^{n-j}-1$.

[4] full m-ary tree

Nodes within a level are numbered left to right, hence node 0 is the leftmost node at a given level.

Most of this document will be concerned with the properties of Double Trees (r=2). Accordingly, the triple representation for nodes will be replaced by a compact tuple representation. The first element of the tuple will represent level number and will have a sign associated with it (positive for the upper tree and negative for the lower tree). The second element will represent node number. For example, node 2 at level 3 is represented as <3,2> in the top tree and as <-3,2> in the bottom tree. Finally, each link also has a level number associated with it. A link between a level ±i node and a level ±(i-1) node is referred to as a level ±i link.

2.1.2 KYKLOS-I

Perhaps the simplest version of KYKLOS is the *simple* Double Tree shown in Fig.2-2(a). This topology has been employed in DON [IMAI84]. Here, the bottom tree is a mirror image of the top tree.

Note however that the term "Double Tree" as used here is somewhat of a misnomer: the topology is neither a perfect duplicate of the original network (only the switches, not the processors have been duplicated), nor is the overall topology a tree any longer.

2.2 KYKLOS-II

The bottom tree in a binary Double Tree may be connected by combining leaf nodes in different ways. In the KYKLOS family of networks, this is permissible so long as the bottom tree of a configuration is isomorphic to a full binary tree of height n. This condition is sufficient (though not necessary) to preserve the uniform, bounded degree condition on the switches[5] (except for the switches at the root). One such topology termed KYKLOS-II is shown in Fig.2-2(b).

[5] From an implementation standpoint, low, uniform fanout is highly desirable

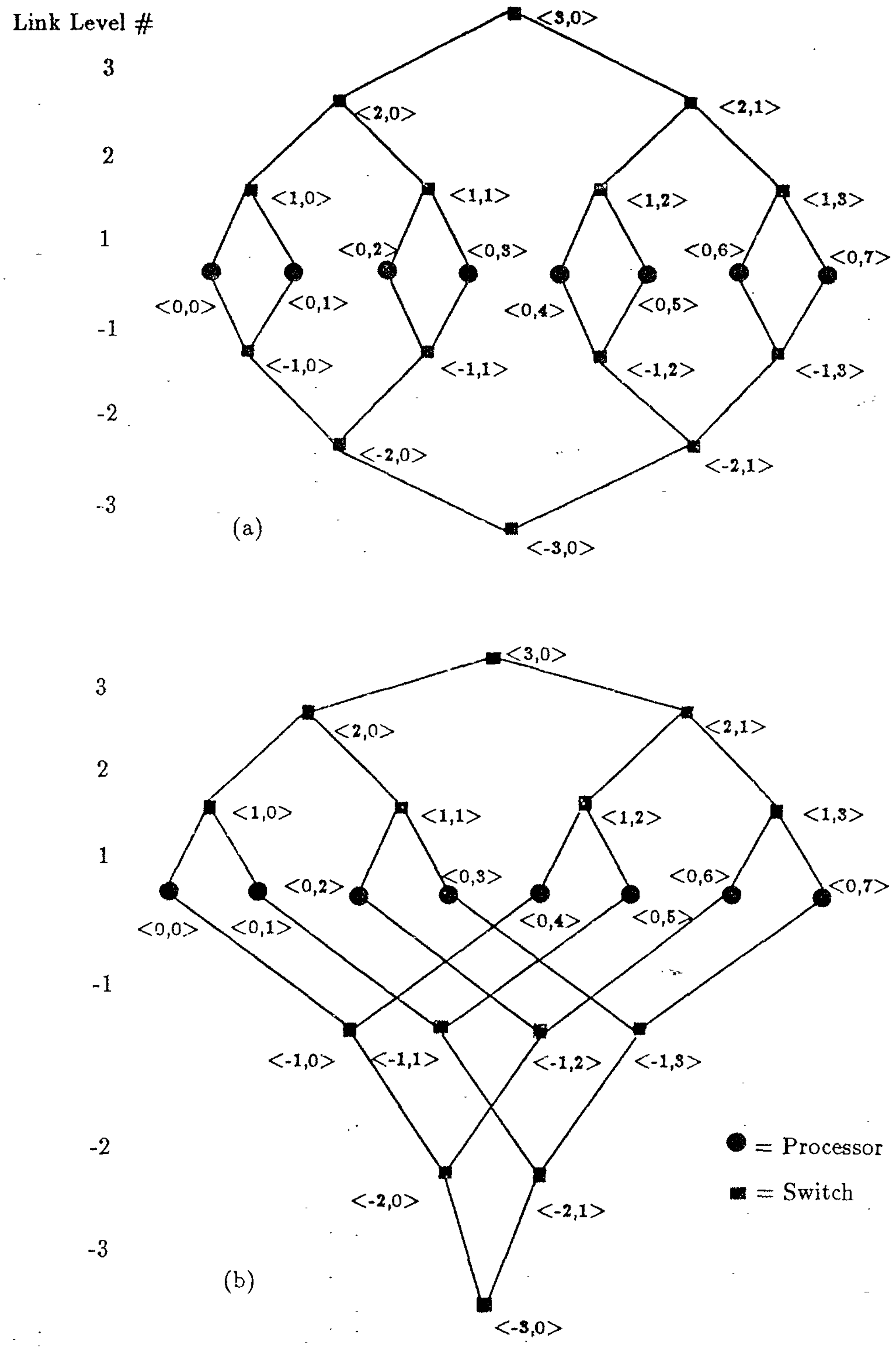


Figure 2-2: Illustrating notation and connection strategy for KYKLOS-I and KYKLOS-II

2.2.1 Shuffle Connected Bottom Tree

The connection between any two adjacent levels of the bottom tree (level -i and -(i+1)) may be visualized by considering level -i nodes, $0 \leq i < n$, split into two groups i.e. node 0 to node 2^{n-i-1} - 1 in one group and nodes 2^{n-i-1} to 2^{n-i} - 1 in the second group. The jth node at level -(i+1) is connected to the jth node in each of the two groups. Descendants of level -(i+1) nodes from left to right are thus ordered as a perfect shuffle (by way of analogy to the ordering in, for example, a shuffled deck of cards).

This idea can be generalized to a Double Tree with arbitrary branching factor where an m-way shuffle may be used to define the bottom tree of KYKLOS-II<m,2,n>. Thus

For 1 ≤ j < n;
level -j node <-j,i>, $0 \leq i < m^{(n-j)}$,
is connected to
level -(j-1) nodes:
<-j+1, i>, <-j+1, i+$m^{(n-j)}$>, . . . <-j+1, i+(m-1)$m^{(n-j)}$>

and to

level -(j+1) node:
<-j-1, i mod $m^{(n-j-1)}$>.

Root node <-n,0> is connected to nodes:
<-n+1, 0>, <-n+1, 1>, . . . <-n+1, m-1>.

The interconnection for the top tree is as in KYKLOS-I.

The above definition of KYKLOS-II is a constructive one. We next present a definition that uses sequences to describe each tree in KYKLOS. While this representation is terse, it is the primary vehicle used in unraveling the topological properties of KYKLOS-II as shown in Chapters 3-5.

2.2.2 Labeling Sequences and Permutations

Note that KYKLOS is, by definition, a multiple tree structure sharing the same set of leaf nodes. Further each of the r trees should have the same branching factor. Hence, the network in Fig. 2-3(a), for example, is disqualified from consideration. Also, the internal nodes of one tree are not directly connected to the internal nodes of any other tree. Hence, the network of Fig. 2-3(b) is not a KYKLOS. The fact being emphasized here is that, from a graph-theoretic point of view,

- each of the r trees are identical (height and branching factor) and
- there are no connections between the internal nodes of any two trees.

What then is the difference between the r trees, if any at all? Clearly, there is no difference between the two trees in KYKLOS-I<2,2,3> (Fig. 2-2(a)). For example, processors 0 and 1 in this topology are 2 links apart in both, top and bottom trees. However, the same pair of nodes are 6 link traversals apart in the bottom tree of KYKLOS-II<2,2,3> (Fig. 2-2(b)). It is obvious that processors that were x links apart in the upper tree of KYKLOS-II<2,2,3> will not, in general, be x links apart in the bottom tree of that topology.

To understand the interconnection scheme in KYKLOS-II more fully, the bottom tree of KYKLOS-II<2,2,3> (Fig. 2-4(a)) is redrawn *without edge crossings* as shown in Fig. 2-4(b). One possible ordering for the processors (left to right) is

0, 4, 2, 6, 1, 5, 3, 7.

It is precisely this sequence that captures the interconnection strategy of KYKLOS-II. Note that the corresponding sequence for the bottom tree of KYKLOS-I (Fig. 2-2(a)) is simply

0, 1, 2, . . ., 7

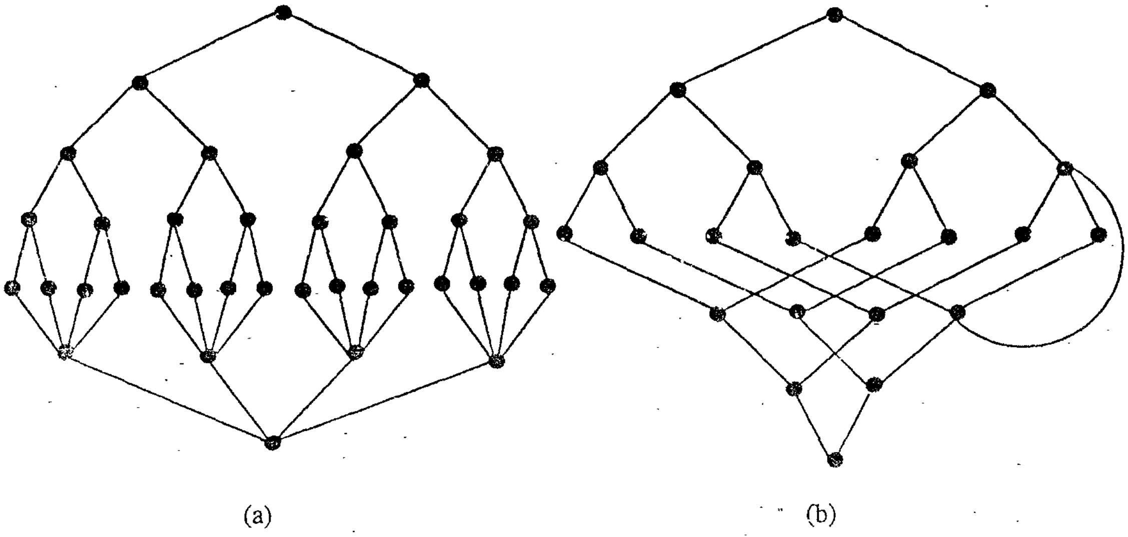


Figure 2-3: What KYKLOS is not

We henceforth refer to a sequence of processors in a given tree of a KYKLOS which has been redrawn without edge crossings as a *Labelling Sequence* (LS). In general, each tree in an r-replica KYKLOS may be isolated and redrawn without edge crossings. Further, each of the trees may correspond to a different LS. The LS's distinguish one tree from another and, indeed, one interconnection strategy from another. Hence the LS's for each tree are all that are needed to specify a given KYKLOS Interconnection Strategy.

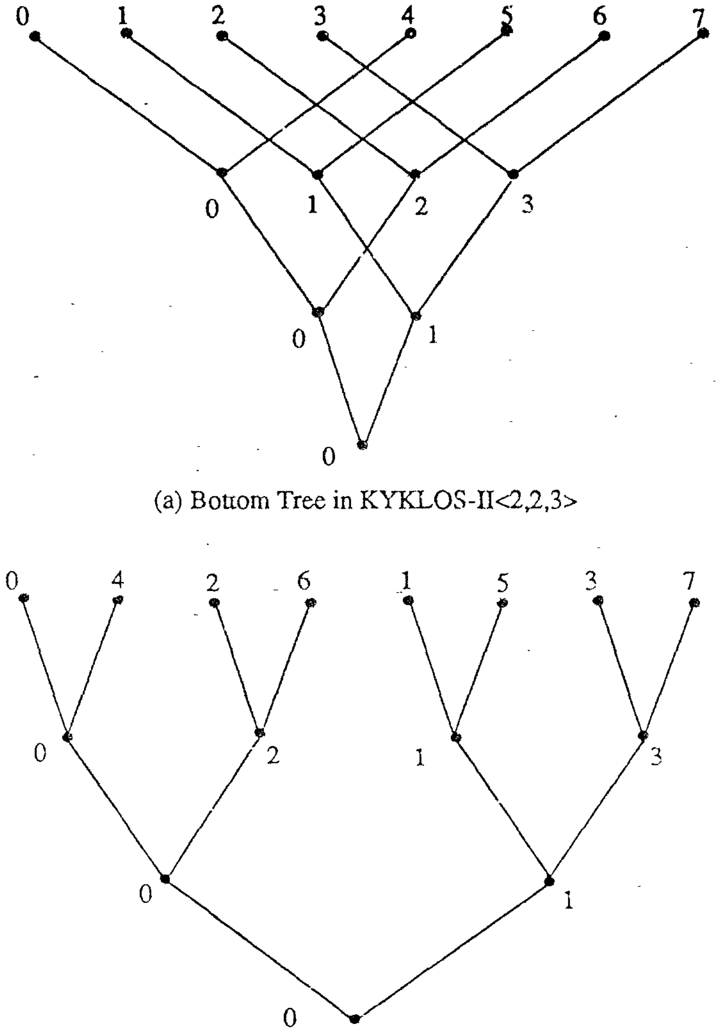
LS's will be denoted by uppercase Greek letters with an optional subscript denoting sequence length. For example,

$\Omega_N = <a_0, a_1, \ldots, a_{N-1}>$

In particular, the LS for the upper tree in KYKLOS will be represented by I_N (uppercase iota) i.e.

$I_N = <0,1, \ldots, N-1>$.

Further, note that the LS's for each of the other r-1 trees in an N-leaf node KYKLOS are permutations of I_N.



(a) Bottom Tree in KYKLOS-II<2,2,3>

(b) Bottom Tree redrawn without edge intersections

Figure 2-4: Redrawn bottom tree of KYKLOS-II<2,2,3>

Intimately related to KYKLOS-II is the Γ Sequence which is introduced next.

2.2.2.1. The Γ Sequence

The m-ary Γ sequence of length N=m^n (m>1, n>0) denoted Γ^m_N is defined as:

$$\Gamma^m_{mN} = <a_0, a_0+m^n, \ldots, a_0+(m-1)m^n,$$
$$a_1, a_1+m^n, \ldots, a_1+(m-1)m^n,$$
$$\cdot \quad \cdot \quad \cdot$$

$$\begin{aligned} &\quad . \quad . \quad . \\ &\quad . \quad . \quad . \\ &a_{N-1}, a_{N-1}+m^n, \ldots, a_{N-1}+(m-1)m^n > \end{aligned}$$

where $\Gamma_N^m = <a_0, a_1, \ldots, a_{N-1}>$ (2.1)

and

$\Gamma_1^m = <0>.$ (2.2)

Example 2.1

$\Gamma_1^3 = <0>$

$\Gamma_3^3 = <0,1,2>$

$\Gamma_9^3 = <0,3,6,1,4,7,2,5,8>$

$$\begin{aligned} \Gamma_{27}^3 = <&0,9,18,\\ &3,12,21,\\ &6,15,24,\\ &1,10,19,\\ &4,13,22,\\ &7,16,25,\\ &2,11,20,\\ &5,14,23,\\ &8,17,26> \end{aligned}$$

2.2.2.2. Permutations

A permutation, ρ_N, on the elements of the sequence, $\Omega=<a_0, a_1, \ldots, a_{N-1}>$, denoted $\rho_N(\Omega)$, is defined as

$\rho_N(\Omega) = <a_{\rho_N(0)}, a_{\rho_N(1)}, \ldots, a_{\rho_N(N-1)}>$. In general, the elements of the sequence (a_i's) may represent any object - numbers, tuples, etc. The permutation operator, ρ_N (typically lowercase Greek), operates on the indices 0 through N-1 and hence causes a rearrangement of the objects in the sequence.

Note that the Γ_N sequence as defined in Eqn. 2.1 is a permutation on I_N.

Two permutation operators that will be used in this chapter are next defined

Shuffle Permutation

$$\delta_N(i) = 2i, \qquad 0 \le i < N/2$$
$$= 2i + 1 - N, \qquad N/2 \le i < N \tag{2.3}$$

Digit Reversal Permutation

$\gamma_N^m(i) = i_0 i_1 \ldots i_{n-1}$

where $i_{n-1} \ldots i_1 i_0$ is the m-ary representation of i, $0 \le i < N=m^n$. (2.4)

(When the superscript is omitted, it is assumed to be 2).

Example 2.2

Let $\Omega_8 = <0, 1, 2, \ldots, 7>$

Then $\delta_8(\Omega) = <0, 2, 4, 6, 1, 3, 5, 7>$

and $\gamma_8(\Omega) = <0, 4, 2, 6, 1, 5, 3, 7>$.

2.2.3 Relationship between Γ Sequence and KYKLOS-II

To show the relationship between the Γ sequence and KYKLOS-II, consider redrawing the *bottom tree* of KYKLOS-II<2,2,4> (Fig. 2-5(a)) so that no two edges intersect. Starting with level -(n-1) nodes, rearrange the level -(n-2) nodes so that the descendants of <-n+1,i> precede those of <-n+1,j> if <-n+1,i> precedes <-n+1,j>. Note that by so doing, there are no crossovers between any two or more level -(n-1) links (Fig. 2-5(b)). Next, place the descendants of each level -(n-1) node in increasing order left to right. Repeat this procedure for level -(n-i) nodes, $2 \le i \le n-1$ (Fig. 2-5(c)). The complete picture is shown in Fig. 2-5(d).

Consider the nodes at an arbitrary level, -(n-i) in the bottom tree. Let the ordering of the -(n-i) nodes in the redrawn bottom tree be

$a_0, a_1, \ldots, a_{N'-1}$ where $N' = m^i$

From the definition of KYKLOS-II (Section 2.2.1) and the above procedure, the ordering of the level -(n-i-1) nodes should be

$a_0, a_0+m^i, \ldots, a_0+(m-1)m^i,$

$a_1, a_1+m^i, \ldots, a_1+(m-1)m^i,$

... ...

$a_{N'-1}, a_{N'-1}+m^i, \ldots, a_{N'-1}+(m-1)m^i.$

Since this holds inductively[6] for all i, $0 \leq i < n$, we conclude that the above ordering of level -(n-i-1) nodes is $\Gamma^m_{mN'} = \Gamma^m_{m^{i+1}}$. We thus have

Lemma 2.1 The bottom tree in KYKLOS-II<m,2,n> can be redrawn without link crossovers in which case the ordering of level -(n-i) nodes is $\Gamma^m_{m^i}$, $0 \leq i \leq n$. □

A special case of this lemma is when i=n. In that case, the LS for the bottom tree in KYKLOS-II<m,2,n> is Γ^m_N.

Theorem 2.1 The sequence Γ^m_N, as defined recursively in Eqns. (2.1)-(2.2), is equivalent to that obtained by reversing the digits in the n-digit m-ary representation, $n=\log_m N$, of each element of the sequence <0, 1, . . ., N-1>.

Proof: We use induction on n, the logarithm of the sequence length. The theorem holds trivially for n=1 i.e. for $\Gamma^m_m = <0,1, \ldots m-1>$.

$$\text{Let } I_{N'} = <0, 1, \ldots, N'-1> \qquad (2.5)$$

We hypothesize that the theorem is true for $n=n'=\log_m N'$, so that

$$\Gamma^m_{N'} = <a_0, a_1, \ldots, a_{N'-1}> = \gamma^n_{N'}(I) \qquad (2.6)$$

[6] It holds for i=0 i.e. for level -n nodes. Also, if it holds for level -(n-i) nodes, then it holds for level -(n-(i+1)) nodes.

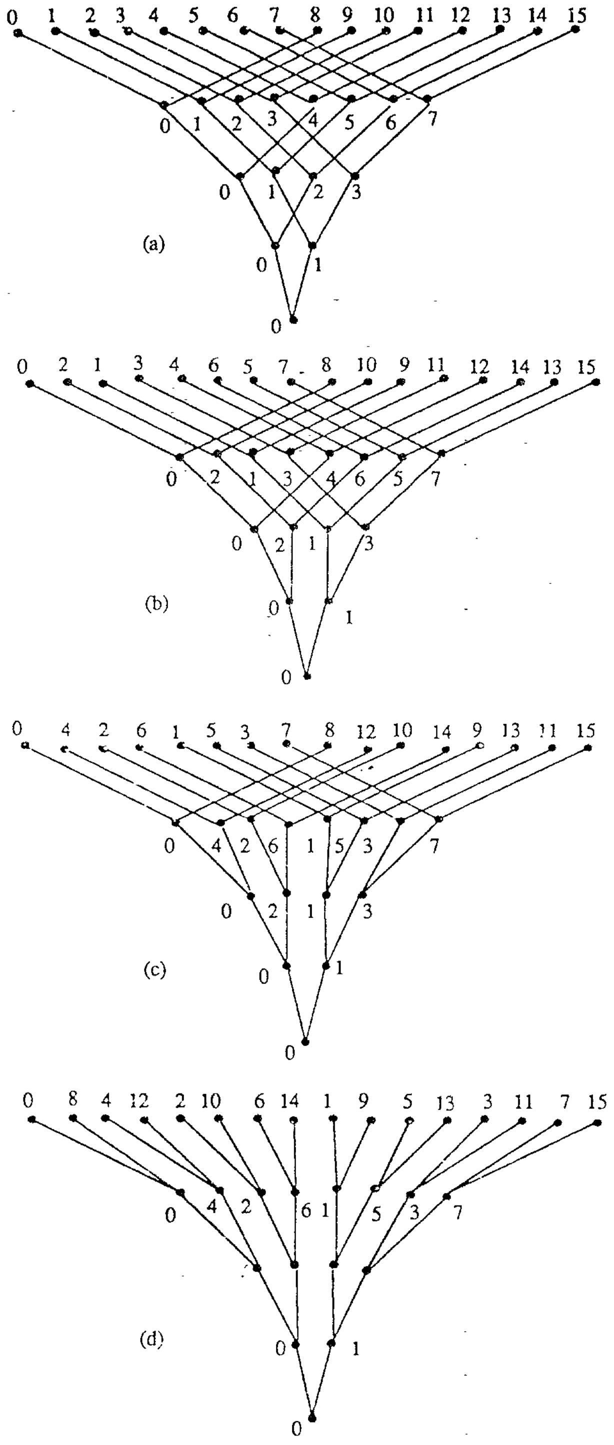


Figure 2-5: Unshuffling the bottom tree in KYKLOS-II<2,2,4>

To prove that the theorem is true for n=n'+1, consider an arbitrary, say t^{th}, term of sequences $\Gamma^{m}_{mN'}$ and $\gamma^{m}_{mN'}(I)$, $0 \le t < mN'$. Let

$$t = km + l, \quad 0 \le l < m \text{ and } 0 \le k < m^{n'}. \tag{2.7}$$

Pictorially k and l may be thought of as the row number and column number respectively of an element in $\Gamma^{m}_{mN'}$ as defined in (2.1). So the t^{th} term of $\Gamma^{m}_{mN'}$ is $a_k + lm^{n'}$. Also, the t^{th} term of $\gamma^{m}_{mN'}(I)$ is $\sum_{j=0}^{n'} q_j m^j$ where

$$t = \sum_{j=0}^{n'} q_{n'-j} m^j, \quad 0 \le q_i < m, 0 \le i \le n' \tag{2.8}$$

From (2.7),

$$l = t \bmod m \tag{2.9}$$

From (2.8),

$$q_{n'} = t \bmod m \tag{2.10}$$

From (2.7) and (2.8) $0 \le l, q_{n'} < m$. This, in conjunction with (2.9) and (2.10), gives

$$l = q_{n'} \tag{2.11}$$

Using (2.7), (2.8) and (2.11),

$$\begin{aligned} k &= \sum_{j=1}^{n'} q_{n'-j} m^{j-1} \\ &= \sum_{j=0}^{n'-1} q_{n'-1-j} m^j \end{aligned} \tag{2.12}$$

Hence the k^{th} term of $\gamma^{m}_{N'}(I)$ is

$$\gamma^{m}_{N'}(k) = \sum_{j=0}^{n'-1} q_j m^j \tag{2.13}$$

By the induction hypothesis of Eqn. (2.6), this is a_k. Hence

$$a_k = \sum_{j=0}^{n'-1} q_j m^j \tag{2.14}$$

and using (2.11),

$$a_k + lm^{n'} = \sum_{j=0}^{n'} q_j m^j \tag{2.15}$$

i.e. the t^{th} elements of $\Gamma^{m}_{mN'}$ and $\gamma^{m}_{mN'}(I)$ are equal for all t, $0 \le t < mN'$. □

Using Lemma 2.1 and the above theorem, it follows that

Corollary 2.1.1 The LS for the bottom tree of KYKLOS-II<m,2,n> is $\gamma^{m}_{N'}(I_{N'})$. □

We say that the processors in Fig. 2-5(d) are γ-connected.

Example 2.3 The LS for the bottom tree in KYKLOS-II<2,2,3> (see Fig. 2-4(b)) is <0, 4, 2, 6, 1, 5, 3, 7>. Note that each element here is a bit reversal of the corresponding element of I_8.

2.2.4 Physical Interpretation of the δ Permutation

Consider the interconnection of processors as shown in Fig. 2-6(a). Here the single tree of n levels may be thought of as being composed of two trees, an odd tree and an even tree. Each of these trees is of height n-1 with the leaf nodes of the odd tree having odd labels and the leaves of the even tree with even labels. The composite tree of height n is formed by interleaving the leaf nodes of the odd and even trees so that the sequence of labels of the leaf nodes appears as the monotonic sequence I_8. Finally, the roots of the odd and even trees are coupled.

Now the LS for the composite tree is shown in Fig. 2-6(b). Note that the LS is easily recognizable as $\delta_N(I_N)$. In analogy to the γ-connected processors of the previous section, we say that the leaf nodes in Fig 2-6(b) are δ-connected. Both these structures will be made use of in the next section in connection with the construction of KYKLOS-III.

2.2.5 Equivalent Labelling Sequences

The bottom tree of KYKLOS-II<2,2,3> (Fig.2-2(b)) has been redrawn without link crossovers in Fig. 2-7(a). The sequence of processors here is
<2 6 0 4 5 1 3 7>.

Another LS for the bottom tree reproduced from Fig. 2-4(b) is placed alongside for comparison. The two LS's are *distinct but equivalent*. Equivalence of two LS's means that the pathlength through the tree between any pair of processors is the same using either LS.

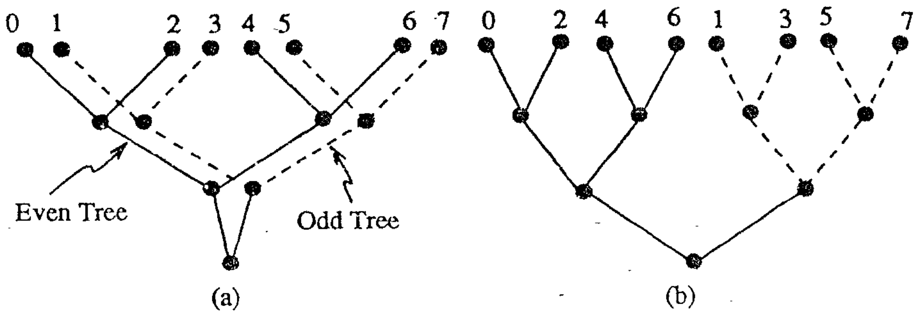


Figure 2-6: A δ-connected bottom tree

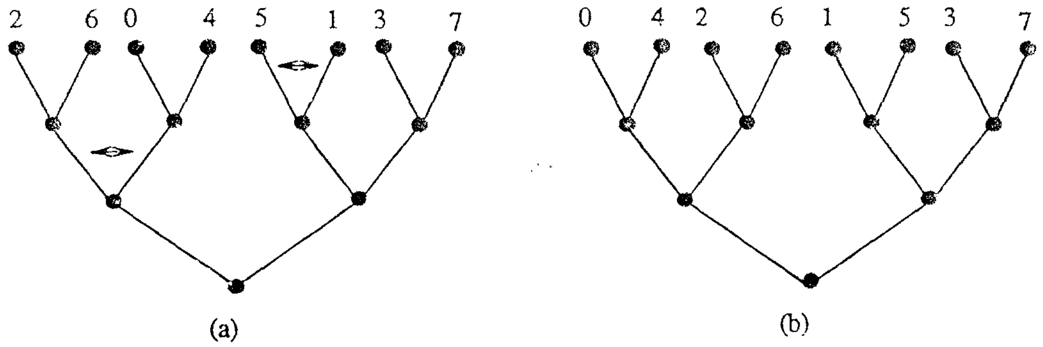


Figure 2-7: LS's for the bottom tree in KYKLOS-II<2,2,3>

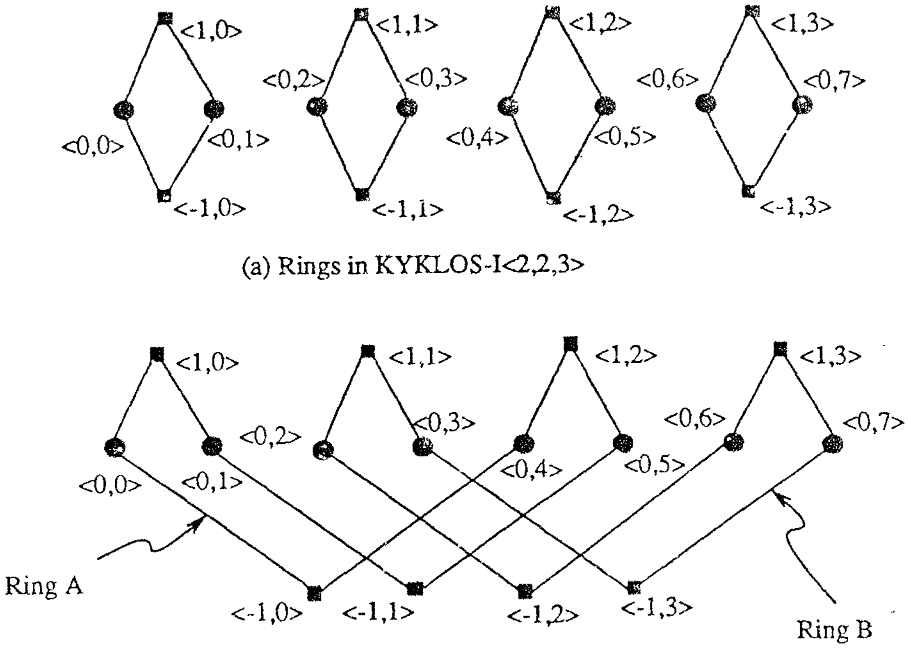
The concept of equivalent LS's induces a partitioning (into equivalence classes) on the N! possible LS's for the bottom tree of a double tree KYKLOS. For example, both LS's in the above example are members of the same equivalence class. Appendix A includes a procedure to determine a (unique) equivalence class leader. Using that procedure, the LS for the bottom tree of KYKLOS-II<m,2,n> would be Γ_N^m.

Since each tree may be characterized by an LS, any KYKLOS-x<m,r,n> may be represented by an r-tuple <LS_1, LS_2, . . ., LS_r> where LS_i is the LS for the ith tree. Once again, we use the convention that $LS_1 = I_N$.

2.3 Obtaining KYKLOS-III from KYKLOS-II

2.3.1 Rings in KYKLOS

Consider the subgraph of KYKLOS-I<2,2,3> (Fig. 2-2(a)) made up of nodes at levels 0, +1 and -1 (Fig. 2-8(a)). There are four disjoint rings each made up of 2 processors and 2 switches. For the N processor node case, there will be a total of N/2 rings in a similar subgraph of KYKLOS-I<2,2,n>. Fig. 2-8(b) shows the corresponding subgraph (level 0, +1 and -1 nodes) for KYKLOS-II<2,2,3> (Fig. 2-2(b)). Note that there are two complete and disjoint rings: Ring A comprises processors 0, 1, 4 and 5; Ring B comprises processors 2, 3, 6 and 7. For the general N-processor node case, there will be N/4 such rings, each linking 4 processors.



(a) Rings in KYKLOS-I<2,2,3>

(b) Rings in KYKOS-II<2,2,3>

Figure 2-8: Showing rings as subgraphs of KYKLOS-I and KYKLOS-II

The rings in KYKLOS-II<2,2,n> may be useful in a multiprocessing environment where a set of processes requiring frequent communication may be assigned to the four processors forming a given ring thus exploiting network locality. Perhaps of greater importance is the construction of a topology with a single ring threading through the processors and the level ±1 switches in a two-replica KYKLOS. Of course, the goal here is not merely to obtain the ring but to further improve the already good properties of KYKLOS-II that are explored in Chapters 3-5.

In [MENE85b], a synthetic approach to building this ring structure is employed. In this section, we use an analytic procedure to obtain the same result. Our modus operandi is to build an isomorphic version of KYKLOS-II. A subtle modification to this structure then provides the desired ring in a topology christened KYKLOS-III.

2.3.2 W-Form of KYKLOS-II

An interesting isomorphic version of KYKLOS-II<2,2,n> where the processors constituting a ring are in close proximity is now introduced. This will be referred to as the W-Form because of its appearance.

2.3.2.1. Construction of KYKLOS-II W-Form

Step 1 Order the processors in the following sequence:

$$\begin{aligned} A = \langle a_0, \quad & a_0-1, \quad a_0-1+N/2, \quad a_0+N/2, \\ a_1, \quad & a_1-1, \quad a_1-1+N/2, \quad a_1+N/2, \\ & \ldots \\ a_{N/4-1}, \quad & a_{N/4-1}-1, \quad a_{N/4-1}-1+N/2, \quad a_{N/4-1}+N/2 \rangle \end{aligned} \tag{2.16}$$

$$\text{where } a_i = \gamma_N(N/2 + 2i), \quad 0 \leq i < N/4. \tag{2.17}$$

Step 2 Top Tree

Level 1 Link Connections: Connect a_i to a_i-1 and a_i-1+N/2 to a_i+N/2, $0 \leq i < N/4$.

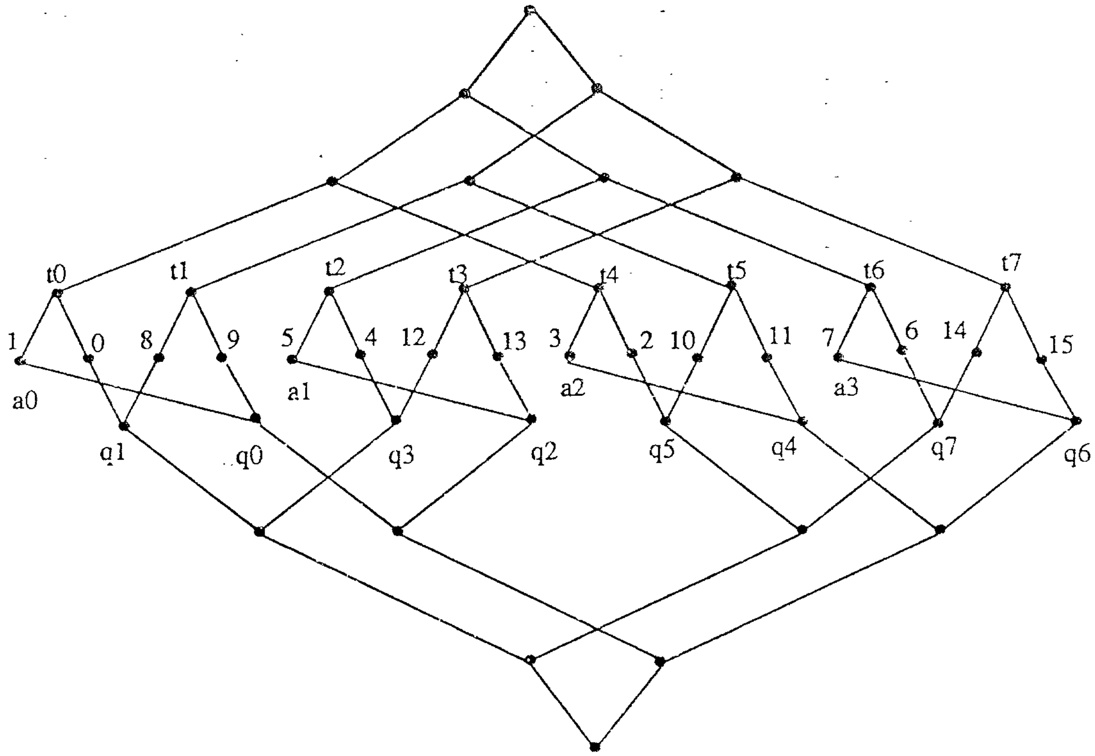
Level i Connections, $1 < i \leq n$: A γ-connected binary tree of height n-1 on level 1 nodes.

Step 3 Bottom Tree

Level -1 Link Connections: Connect a_i to a_i+N/2 and a_i-1 to a_i-1+N/2, $0 \leq i < N/4$.

Level -i Connections, $1 < i \leq n$: A δ-connected binary tree of height n-1 on level -1 nodes.

Fig. 2-9 is a W-Form representation of KYKLOS-II.



Note: ao is followed by ao-1, ao-1+N/2 and ao+N/2

Figure 2-9: W-Form Representation of KYKLOS-II

Our next task is to show that this representation preserves the interprocessor distances of Form-I[7], KYKLOS-II<2,2,n> i.e. if processors x and y are 2d links apart in Tree 1(2), Form-I, then they are 2d links apart in Tree 2(1), W-Form. We will first show this for the top tree and then for the bottom tree.

2.3.2.2. Generating the LS for the Top Tree

Construct the sequence made up of *processor pairs*,

$$A_{N/2} = <t_0, t_1, \quad . \quad . \quad . \quad , \quad t_{N/2-1}> \tag{2.18}$$

$$\text{where } t_{2i} = <a_i, a_i\text{-}1>$$

$$\text{and } t_{2i+1} = <a_i\text{-}1+N/2, a_i+N/2>, \qquad 0 \leq i < N/4 \tag{2.19}$$

This sequence of t_i's corresponds to the sequence of level +1 nodes (Fig. 2-9); the descendants of each level 1 node correspond to the processor pair defined by Eqn. (2.19). The level 1 nodes are γ-connected. So the subgraph between levels 1 and n can be redrawn without intersections, the LS for the level 1 nodes being

$$\gamma_{N/2}(A) = <t_{\gamma_{N/2}(0)}, \quad t_{\gamma_{N/2}(1)}, \quad \ldots, \quad t_{\gamma_{N/2}(N/4-1)}, \quad t_{\gamma_{N/2}(N/4)}, \quad \ldots, \quad t_{\gamma_{N/2}(N/2-1)}>$$

The first N/4 tuples above are of the form $t_{\gamma_{N/2}(i)}$, $0 \leq i < N/4$; hence they each have even indices. Similarly, the next N/4 tuples have odd indices. Using (2.19) to expand each tuple gives[8]

$$\begin{array}{llll}
\gamma_N(A) = <a_{\gamma_{N/2}(0)/2}, & a_{\gamma_{N/2}(0)/2}\text{-}1, & a_{\gamma_{N/2}(1)/2}, & a_{\gamma_{N/2}(1)/2}\text{-}1, \; \ldots \\
\quad \ldots & & a_{\gamma_{N/2}(N/4-1)/2}, & a_{\gamma_{N/2}(N/4-1)/2}\text{-}1, \\
\quad a_{(\gamma_{N/2}(N/4)-1)/2}\text{-}1+N/2, & & a_{(\gamma_{N/2}(N/4)-1)/2}+N/2, \; \ldots & \\
\quad \ldots & & a_{(\gamma_{N/2}(N/2-1)-1)/2}\text{-}1+N/2, & a_{(\gamma_{N/2}(N/2-1)-1)/2}+N/2>
\end{array}$$

[7] This is the shuffle-connected form of Fig.2-2(b).

[8] The γ-operator takes precedence over the arithmetic operators (eg. division, etc.)

Each of the first N/2 elements of the above sequence have terms of the form $a_{\gamma_{N/2}(i)/2}$, $0 \leq i < N/4$. Further, note that

$$\begin{aligned} a_{\gamma_{N/2}(i)/2} &= \gamma_N(N/2+\gamma_{N/2}(i)) && \text{from Eqn. (2.17)} \\ &= 2i+1 && \text{from Result A.3(a)} \end{aligned} \qquad (2.20)$$

Also, each of the N/2 elements in the latter half of the sequence have terms with indices of the form $(\gamma_{N/2}(N/4+i)-1)/2$, $0 \leq i < N/4$. Using Result A.3(b), $(\gamma_{N/2}(N/4+i)-1)/2 = \gamma_{N/4}(i)$ so that the corresponding term of the elements in the second half of the sequence are of the form

$$\begin{aligned} a_{\gamma_{N/4}(i)} &= \gamma_N(N/2 + 2\gamma_{N/4}(i)) && \text{from Eqn. (2.17)} \\ &= \gamma_N(N/2+\gamma_{N/2}(i)) && \text{from Result A.2} \\ &= 2i+1 && \text{from Result A.3(a)} \end{aligned}$$

Consequently, the terms in the second half of the sequence are of the form

$(2i+1)-1+N/2, (2i+1)+N/2, \ldots \qquad 0 \leq i < N/4.$

Hence the LS for the top tree is

<1, 0, 3, 2, . . ., N/2-1, N/2-2,
N/2, N/2+1, . . ., N-2, N-1>

which is *equivalent* to the LS <0, 1, 2, 3, . . ., N-2, N-1> = I_N, as it should.

2.3.2.3. Generating the LS for the Bottom Tree

For the bottom tree, we define the following sequence

$$B_{N/2} \doteq \langle q_0, q_1, \ldots, q_{N/2-1} \rangle \qquad (2.21)$$

$$\begin{aligned} &\text{where } q_{2i} = \langle a_i, a_i+N/2 \rangle \\ &\text{and } q_{2i+1} = \langle a_i-1, a_i-1+N/2 \rangle, \qquad 0 \leq i < N/4 \end{aligned} \qquad (2.22)$$

This corresponds to the sequence of level -1 nodes (Fig. 2-9). If the subgraph made up of the bottom tree between levels -1 and -n is redrawn without crossovers (see Section 2.2.4), the sequence of level -1 nodes is

$$\delta_{N/2}(B) = <q_{\delta(0)}, \quad q_{\delta(1)}, \quad \ldots, \quad q_{\delta(N/4-1)}, \quad q_{\delta(N/4)} \quad \ldots, \quad q_{\delta(N/2-1)}>$$

From (2.3)

$$\delta_{N/2}(B) = <q_0, \quad q_2, \quad \ldots, \quad q_{N/2-2}, \quad q_1, \quad q_3, \quad \ldots, \quad q_{N/2-1}>$$

From this sequence of level -1 nodes, the LS for the bottom tree may be obtained using (2.22) as

$$\begin{aligned} \delta_N(B) = <a_0, \quad a_0+N/2, \quad a_1, \quad a_1+N/2, \ldots \\ \ldots, a_{N/4-1}, \quad a_{N/4-1}+N/2, \\ a_0-1, \quad a_0-1+N/2, \quad a_1-1, \quad a_1-1+N/2, \quad \ldots \\ \ldots, \quad a_{N/4-1}-1, \quad a_{N/4-1}-1+N/2> \end{aligned}$$

Using (2.17),

$$\begin{aligned} \delta_N(B)=<\gamma_N(N/2+0), \quad \gamma_N(N/2+0)+N/2, \quad \gamma_N(N/2+2), \quad \gamma_N(N/2+2)+N/2, \ldots \\ \ldots \quad ,\gamma_N(N-2), \quad \gamma_N(N-2)+N/2, \\ \gamma_N(N/2+0)-1, \quad \gamma(N/2+0)-1+N/2, \quad \gamma_N(N/2+2)-1, \quad \gamma(N/2+2)-1+N/2, \ldots \\ \ldots \quad ,\gamma_N(N-2)-1, \quad \gamma_N(N-2)-1+N/2> \end{aligned}$$

Each element of the above sequence has a term of the form $\gamma_N(N/2+2k)$, $0 \leq k < N/4$. From Result A.3(b),

$$\begin{aligned} \gamma_N(N/2+2k) &= 2\gamma_{N/2}(2k)+1 \\ &= \gamma_N(2k)+1 \qquad \text{from Result A.2} \end{aligned} \tag{2.23}$$

Permuting the two halves of the sequence and using (2.23) on the first half, we obtain the LS for the bottom tree as

$$\begin{aligned} \gamma_N(B)=<\gamma_N(0), \quad \gamma_N(0)+N/2, \quad \gamma_N(2), \quad \gamma_N(2)+N/2, \ldots \\ \ldots \gamma_N(N/2-2), \quad \gamma_N(N/2-2)+N/2, \\ \gamma_N(N/2), \quad \gamma_N(N/2)+N/2, \quad \gamma_N(N/2+2), \quad \gamma_N(N/2+2)+N/2, \quad \ldots \\ \ldots, \quad \gamma_N(N-2), \quad \gamma_N(N-2)+N/2> \end{aligned}$$

This is Γ_N as it should be for the bottom tree of KYKLOS-II<2,2,n>.

2.3.3 KYKLOS-III

KYKLOS-III may be derived quite easily from the W-form of KYKLOS-II. The procedure to obtain KYKLOS-III is

```
For $0 \leq i < N/4$,
  {
        Remove the link between $q_{2i}$ and $a_i$
        Connect $q_{2i}$ to $a_{(i+1)\ mod\ N/4}$
  }
```

This is shown in Fig. 2-10. The original links are shown in dashed lines while the new links are in dark lines. Observe that there is a complete ring threading through the processors and the level $\pm$i nodes. It is from this fact that the KYKLOS[9] Network derives its name. Finally, note that the degree of each node remains unchanged.

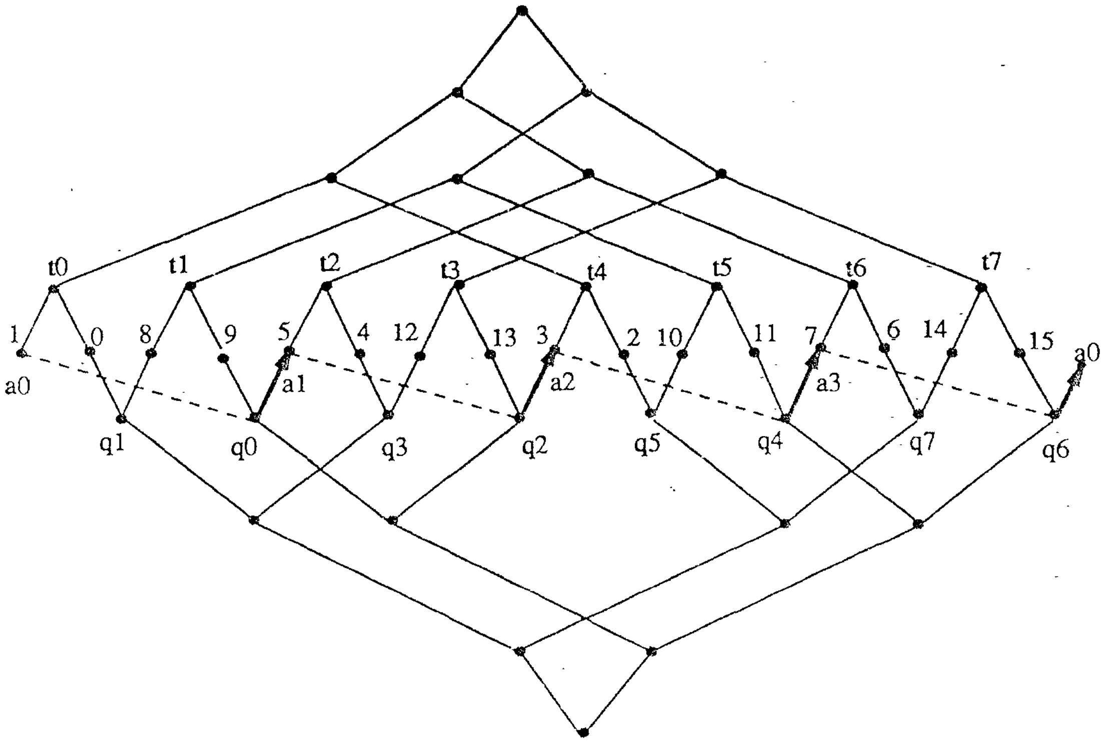
2.3.3.1. LS for the Bottom Tree of KYKLOS-III

The LS for the bottom tree in KYKLOS-II<2,2,n> is Γ_N, from Lemma 2.1. From Theorem 2.1, the LS may be expressed as

$$\begin{aligned}\Gamma_N = <&\gamma_N(0), \ldots\ldots\ldots, \gamma_N(N/2-1),\\ &\gamma_N(N/2), \gamma_N(N/2+1), \gamma_N(N/2+2), \ldots \gamma_N(N-2), \gamma_N(N-1)>\end{aligned} \tag{2.24}$$

Using Eqn. (2.17) on each of the even terms of the latter half of (2.24), we get

[9] KYKLOS is Greek for ring



Note: ao is followed by ao-1, ao-1+N/2 and ao+N/2

- - - - - = Links removed in KYKLOS-III

⟶ = Links added to form KYKLOS-III

Figure 2-10: KYKLOS-III

$$\Gamma_N = <\gamma_N(0), \ldots\ldots\ldots, \gamma_N(N/2-1), \\ a_0, \gamma_N(N/2+1), a_1, \ldots, a_{N/4-1}, \gamma_N(N-1)> \qquad (2.25)$$

The procedure for building KYKLOS-III from KYKLOS-II involves a readjustment of link connections between levels 0 and -1 (i.e. in the bottom tree alone). From the point of view of the bottom tree, the link adjustment is equivalent to the cyclic shift of processors

$a_{N/4-1} \rightarrow a_{N/4-2}$,

$a_{N/4-2} \rightarrow a_{N/4-3}$,

. . .

$a_1 \rightarrow a_0$,

$a_0 \rightarrow a_{N/4-1}$.

In terms of the sequence of (2.25), this corresponds to a left circular shift of the elements in boldface. Hence, the LS for the bottom tree of KYKLOS-III is the Γ_N Sequence in which every even element of the latter half of the sequence is shifted two notches to the left as under

$$\Gamma_N' = <\gamma_N(0), \ldots\ldots\ldots, \gamma_N(N/2-1),$$
$$\mathbf{a}_1, \gamma_N(N/2+1), \mathbf{a}_2, \ldots \mathbf{a}_0, \gamma_N(N-1)> \qquad (2.26)$$

Hence, the LS for the bottom tree in KYKLOS-III is the **Quarter-shifted Γ Sequence.**[10]

2.4 KYKLOS Interconnection Philosophy

If a certain subset of processors are clustered in one tree, KYKLOS-II attempts to spread them out in each of the other r-1 trees. Thus, link resources in these other trees may be used to cluster a different subset of processors thereby reducing average ensemble distances. As a specific example, consider KYKLOS-II<2,2,4> (Fig. 2-11). Here, processors 4 and 5 are in the same subtree of height 1 in the upper tree. However, they are in the two *different* subtrees of height 4 in the bottom tree. Further, processors 4, 5, 6 and 7 (in the same subtree of height 2 in the top tree) are dispersed among four *different* subtrees, each of height 2, rooted at the four nonleaf nodes at level -2 (the paths from each of these processors to the roots of the subtrees in the bottom tree is shown in dark lines in Fig. 2-11).

[10] In [MENE85b], the Quarter-shifted Γ Sequence is obtained by a right circular shift of the odd elements in the latter half of the Γ Sequence. That topology is isomorphic to the one generated here.

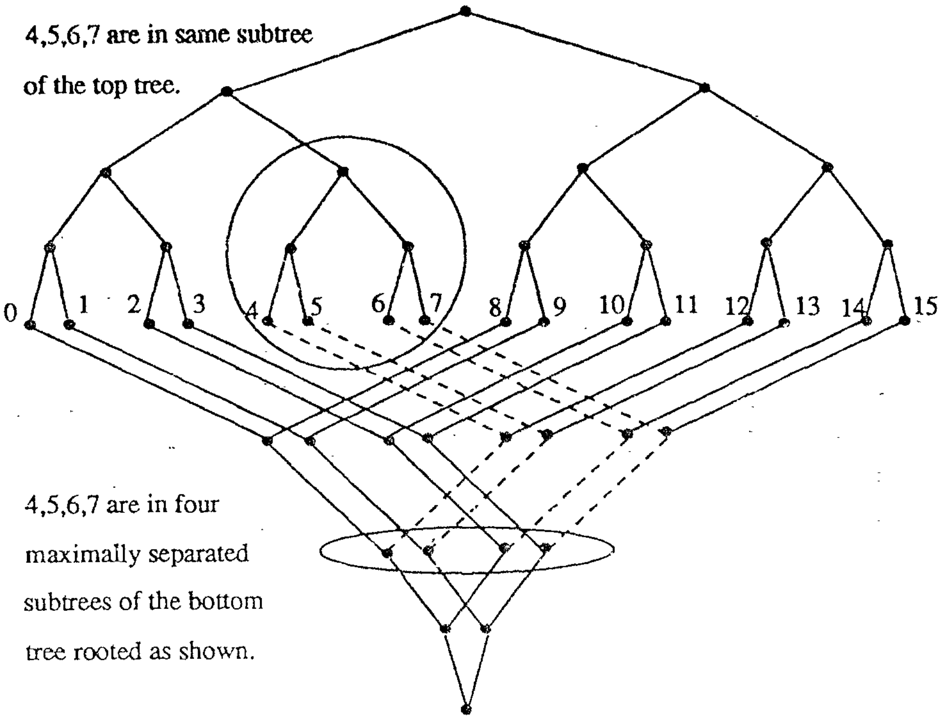


Figure 2-11: Illustrating m-sense optimality in KYKLOS-II<2,2,4>

Definition 2.1 A KYKLOS-x<m,2,n> is said to satisfy Property P* if any set of m^i leaf nodes,

$k.m^i$, $k.m^i+1$, . . ., $(k+1).m^i$ -1,

$0 < i < n$,

$0 \leq k < m^{n-i}$,

in the same subtree of height i in the top tree are each contained in m^i *distinct* subtrees of height n-i rooted at each of the nonleaf nodes at level -(n-i) in the bottom tree.

In our quest for KYKLOS topologies that try to reduce interprocessor communication distances (message latencies), this property is especially significant. Note that we are implicitly considering the use of paths between two processors that are wholly contained within one of the r trees of a general KYKLOS-x<m,2,n>. As such, we estimate the pathlength between 2 processors y and z using each of the r trees. (The distance using the ith tree is designated $d_{ix}(y,z)$ where x is the KYKLOS version number.) We then select the one (or ones) that yield(s) the minimum pathlength. We refer to this distance, $d_{mx}(y,z)$, as the m-sense distance i.e.

$d_{mx}(y,z) = \underline{\min}_{i=1,r} \{ d_{ix}(y,z) \}$

2.4.1 M-sense Optimality

Property P* can be expressed in terms of m-sense distances in the general case of r trees:

Definition 2.2 If $\forall$ y,z $\in$ {leaf nodes}, $y \neq z$

$d_{ix}(y,z) = 2l \quad => \quad d_{jx}(y,z) > 2(n-l)$,

$1 \leq i,j \leq r, \quad i \neq j$,

then the corresponding KYKLOS-x is said to be optimal with respect to the m-sense distance characteristics (abbreviated m.s.o.).

Example 2.4 Consider a KYKLOS-I<2,2,3> (Fig. 2-2(a)).

Here $d_{11}(2,3) = 2$, where Tree 1 is the top tree.

However $d_{21}(2,3) = 2 < 2n - d_{11}(2,3)$ where Tree 2 is the bottom tree.

Hence the optimality criterion fails and KYKLOS-I is not m.s.o..

We next turn our attention to constructing m.s.o. KYKLOS topologies.

2.4.2 Procedure to construct m.s.o. KYKLOS-x<m,2,n>

We begin with an unlabeled bottom tree of height n and a list of labels <0, 1, . . ., N-1>. The list may be thought of as a pack of cards that has to be dealt out by the root to each of the level -(n-1) nodes such that at no instant in the dealing process (or at the end of it) is the number of cards dealt out to any two nodes greater than 1. For example, the deal shown in Fig. 2-12 is illegal. This is so since a snapshot of the deal, represented by the dashed line of Fig. 2-12, shows that at the instant of the snapshot two more cards were dealt out to node 1 than to node 0 thus violating the invariant above.

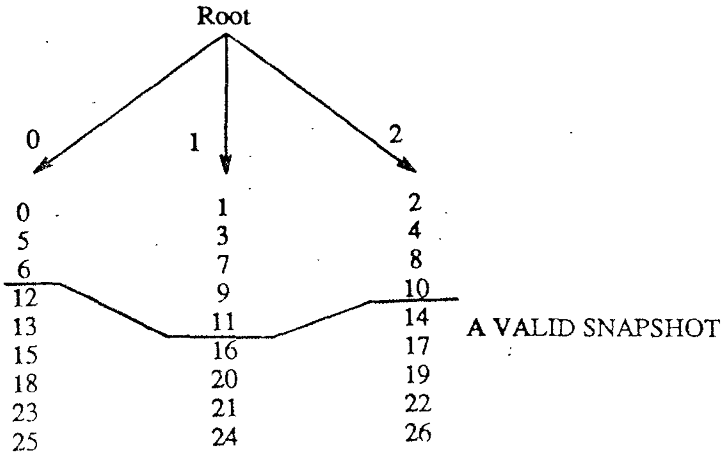


Figure 2-12: An illegal Deal by the root of a ternary KYKLOS

When all cards have been dealt out by the root, each level -(n-1) node deals out its received deck of cards to its descendants, again taking care to satisfy the above invariant. The procedure is repeated until the labels have propagated to the processors, one per processor.

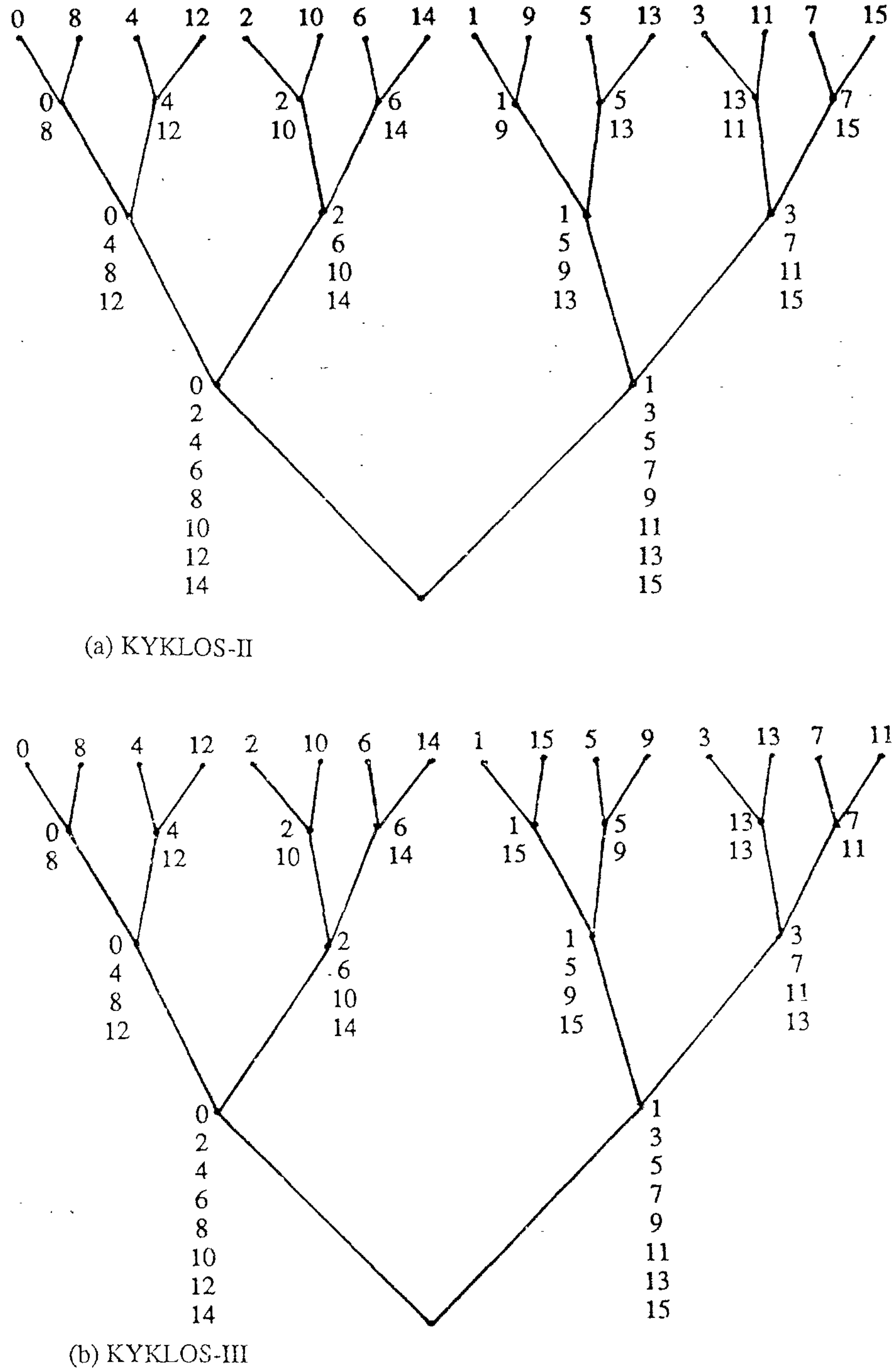
We state without proof[11] that

- The LS that results for the bottom tree using the above procedure generates an m.s.o. KYKLOS.
- Further, if each node deals out labels in a round-robin fashion starting with its leftmost descendant, the procedure above will generate an LS for the bottom tree corresponding to Γ_N^m. This is shown in Fig. 2-13(a). The list of labels that each node receives is shown alongside the node.

[11] Proof of these results are being compiled into a report

- Finally, in a binary Double Tree, let each node on the right hand side of the bottom tree (except for level -1 nodes and the root) deal out its labels in a round-robin fashion up to and including the penultimate round of the deal. If in the final round, the cards are dealt out in reverse order, then the LS so obtained (Fig. 2-13(b)) is the Quarter-shifted Γ sequence presented in [MENE85(b)] which is *equivalent* to the LS discussed in Section 2.3.3. Thus, in this case KYKLOS-III is generated and the topology is m.s.o.

We will return to the subject of m.s.o. KYKLOS topologies in Chapter 4. Before that we will look at strategies for routing and associated distance properties for KYKLOS-II in Chapter 3.



(a) KYKLOS-II

(b) KYKLOS-III

Figure 2-13: Dealing labels to obtain KYKLOS-II/III

Chapter 3

Basic Routing Strategies and Associated Distance Properties

It is reasonable to expect that different topological variations of KYKLOS will employ different routing techniques. In addition, the same topology may have a spectrum of routing strategies used for interprocessor communication. Thus, the cross-product: {Topology × Routing Strategy} will, in general, map to different distance and traffic characteristics as shown in Fig. (2.4).

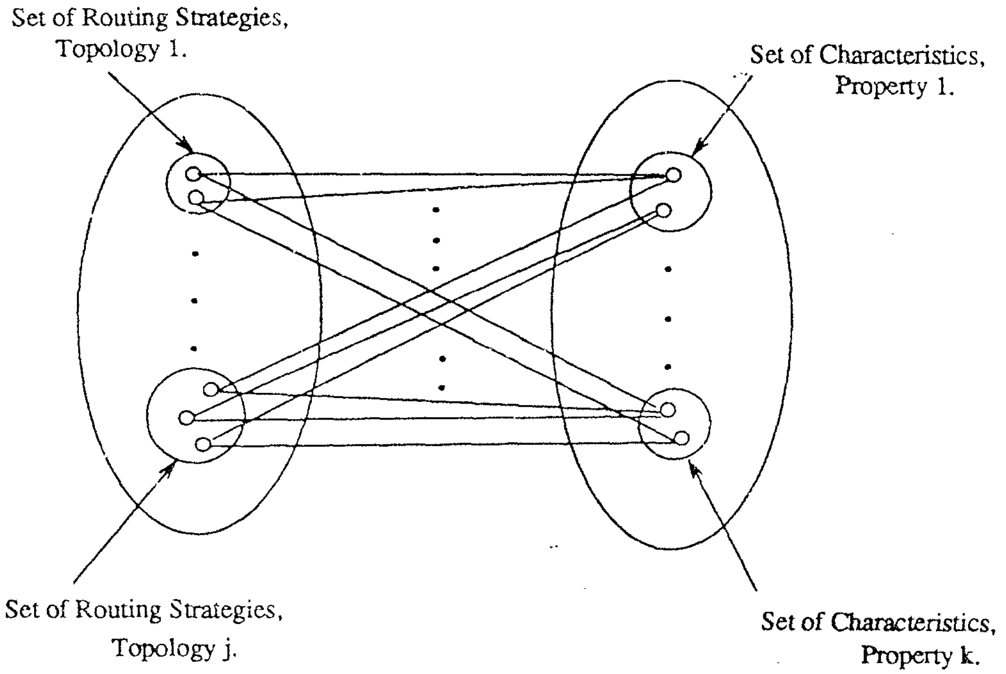


Figure 3-1: Mapping {Topology × Routing Strategy} to Properties

In the first two sections of this chapter, two different routing strategies for KYKLOS-II<m,2,n> are presented and their distance characteristics are explored. The last section concludes by comparing the distance characteristics as a function of topology and routing strategy.

3.1 M-II Routing and Distance

3.1.1 Motivation

In Section 2.4 it was shown that both, KYKLOS-II and KYKLOS-III were m-sense optimal. The optimality clause for a KYKLOS-x<m,2,n> may be rewritten as:

$$d_{1x}(y,z) + d_{2x}(y,z) > 2n.$$

In words, if the distance between 2 processors in one tree is *small* (< n), the corresponding distance in the other tree is *somewhat larger* (> n). For example, in KYKLOS-II<2,2,3> (Fig. 3-2), Processors 0 and 1 are separated by 2 link traversals in the top[12] tree. In the bottom tree however, there are 6 links separating them. This means that some other processor(s) (eg. 4) has(ve) been brought closer to 0 in the bottom tree. The implications of this property to reducing interprocessor communication latency suggest that we consider the option of using either the top tree or bottom tree (whichever yields a shorter path) to route messages.

[12] Top Tree is used synonymously for Upper Tree or Tree 1. Bottom Tree is used synonymously for Lower Tree or Tree 2

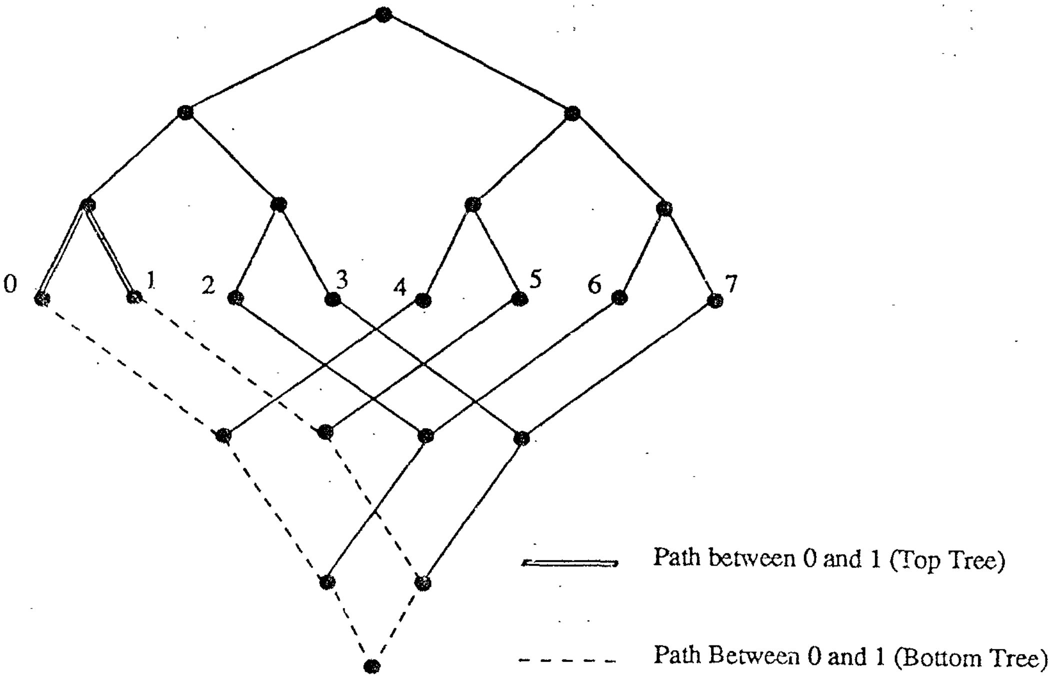

0
1
2
3
4
5
6
7
Path between 0 and 1 (Top Tree)
Path Between 0 and 1 (Bottom Tree)

Figure 3-2: Illustrating Optimality in KYKLOS-II<2,2,3>

3.1.2 Basic Definitions and Results

Let S be the source address of a message and D be the destination address. For a KYKLOS-x<m,2,n>, these may be expressed as n-digit m-ary strings.

$S = s_{n-1}\, s_{n-2} \ldots s_0$ $\quad 0 \le s_i < m$
$D = d_{n-1}\, d_{n-2} \ldots d_0$ $\quad 0 \le d_i < m$
$0 \le i < n.$

Definition 3.1 Let Σ_m be the finite alphabet $\{0,1, \ldots, m\text{-}1\}$. Then $s_i, d_i \in \Sigma_m$ and $S, D \in \Sigma_m^n$.

It is clear from Fig. 3-3(a) (top tree of Fig. 3.2) that S and D will be in the same half (subtree of height n-1) of the upper tree if the most significant bits (MSB)[13] of their

[13] By convention the MSB is bit 0 from the left or bit n-1 from the right

addresses are the same. Further, they will be in the same subtree of height n-2 if *in addition* bit n-2 of their addresses are the same. We could continue to check successive bits until we find a mismatch, say in bit t from the left i.e. bit n-t-1 in the example of fig. 3-3(a). This would imply that S and D are in the same subtree of height n-t but in different subtrees of height n-t-1, $0 \leq t < n$, of the top tree. In general, the equivalence of the following propositions holds:

P1.1 $(s_{n-1} = d_{n-1}) \wedge (s_{n-2} = d_{n-2}) \ldots \wedge (s_{n-t} = d_{n-t}) \wedge (s_{n-t-1} \neq d_{n-t-1})$

P1.2 S and D are in the same subtree of height n-t but in different subtrees of height n-t-1 of the top tree.

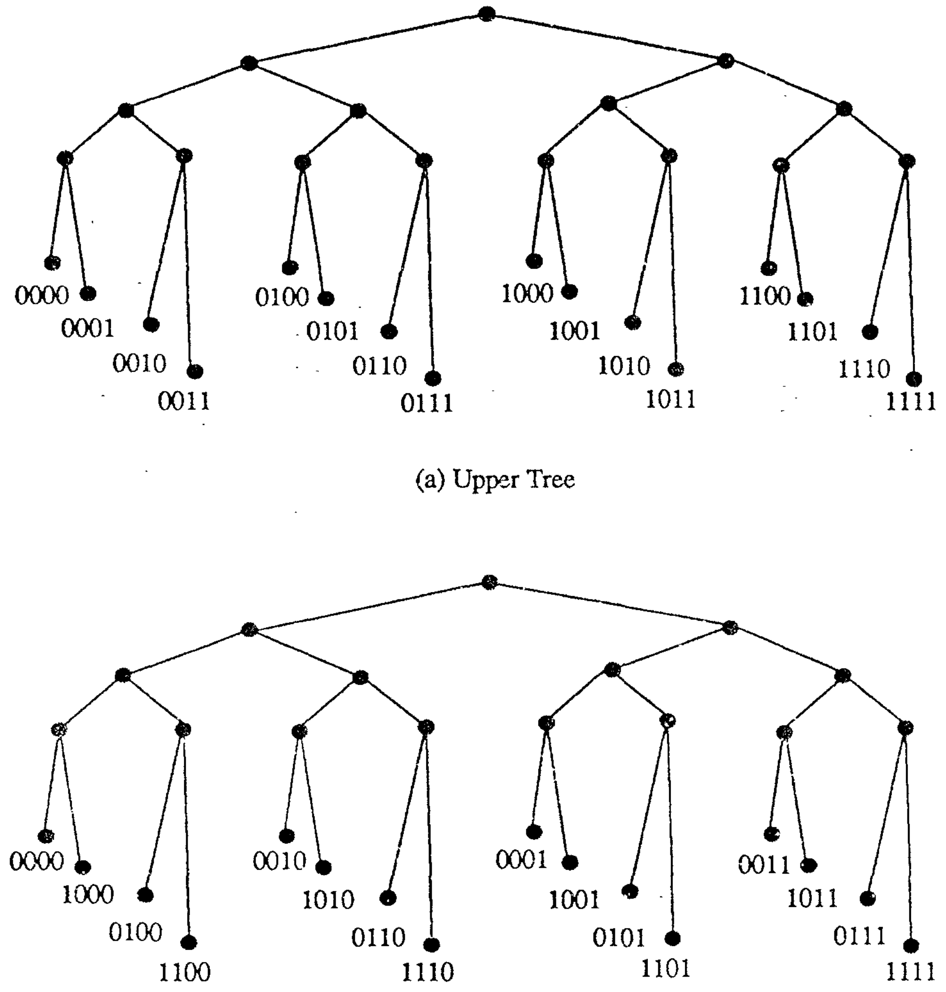
P1.3 $d_{12}(S,D) = 2(n-t)$

In Chapter II, it was shown that the LS for the bottom tree of KYKLOS-II<m,2,n> was a *digit reversal* of each element of the sequence $<0,1,\ldots,m^n-1>$ used to label processors in the top tree. Hence, the process of searching for a match between successive digits in S and D to ascertain their distance in the bottom tree would have to begin from the least significant digit. For example, S and D are in different subtrees of height n-1 of the bottom tree in Fig.3-3(b) if S and D differed in the least significant position, etc. Let us suppose that the first mismatch occurred in digit position b. Then corresponding to P1.1-P1.3 for the top tree, we have for the bottom tree

P2.1 $(s_0 = d_0) \wedge (s_1 = d_1) \ldots \wedge (s_{b-1} = d_{b-1}) \wedge (s_b \neq d_b)$

P2.2 S and D are in the same subtree of height n-b but in different subtrees of height n-b-1 of the bottom tree.

P2.3 $d_{22}(S,D) = 2(n-b)$.



(a) Upper Tree

(b) Lower Tree

Figure 3-3: Illustrating P1.1-P1.3 and P2.1-P2.3 in KYKLOS-II<2,2,4>

Definition 3.2 The "distance vector" between S and D, denoted X(S,D), is an n-bit binary string (i.e. X(S,D) $\in \Sigma_2^n$) defined as

$X = x_{n-1}\, x_{n-2} \cdots x_0$

where

$x_i = 0$ if $s_i = d_i$
$= 1$ otherwise.

We will also use u_i or v_i to represent a string of digits or bits. The length of a string u will be denoted |u|. y_i will represent a single digit or bit. Finally "<" will represent the relation "a substring of" and "$\not<$" will represent "not a substring of". Thus 011 < 10111 but 011 $\not<$ 10101.

Remark 3.1: Except in the trivial case where S=D, X(S,D) may be represented as $0^t u 0^b$ where u ≡ 1 or $1u_1 1$. In English, t and b are respectively the maximum number of 0's beginning at the MSB and LSB end of X(S,D). Since |X(S,D)|=n, b+t < n.

Remark 3.2: If u_1 and u_2 can respectively take k_1 and k_2 different values, u_1u_2 (the concatenation of u_1 and u_2) can take k_1k_2 different values.

Using the definition of X(S,D) and the equivalence of P1.1 and P1.3, and P2.1 and P2.3 we obtain:

Lemma 3.1 (Tree Distance (TD) Lemma) The distance between S and D in KYKLOS-II<m,2,n> is

$$\boxed{\begin{array}{c}\text{2n-2t using the top tree where } X(S,D) = 0^t1u_1 \\ \text{and} \\ \text{2n-2b using the bottom tree where } X(S,D) = u_2 10^b.\end{array}}$$

An immediate application is to the m-sense distances in KYKLOS-II<m,2,n> introduced in Section 2.4. These may now be expressed in terms of t and b viz.

$$\begin{aligned} d_{m2}(S,D) &= \min\{d_{12}(S,D), d_{22}(S,D)\} \\ &= \min\{2(n-t), 2(n-b)\} \end{aligned}$$

whence

Theorem 3.1 In KYKLOS-II<m,2,n>,

$$\boxed{\begin{aligned} d_{m2}(S,D) &= 2(n\text{-}t) \text{ if } t \geq b \text{ (top tree)} \\ &= 2(n\text{-}b) \text{ otherwise (bottom tree)} \end{aligned}}$$

3.1.3 M-II Routing Algorithm

The strategy that uses one of two trees *exclusively*, whichever yields Minimum distance in KYKLOS-II<m,2,n> is termed M-II Routing Strategy.
Let S be the source and D the destination for a message transfer.

At source Processor:

Compute X(S,D), t and b.

```
Then, if t > b, use the top tree
        else if b > t, use the bottom tree
        else either tree may be used.
```

To route through the top (bottom) tree, send message toward the root through n-t (n-b) switch levels. Then reflect message back toward leaf nodes using
At level i(-i) switch, if digit i-1 (n-i) of D = j, route to j^{th} child.

Example 3.1: Let S = 31(11111), D = 13(01101) in KYKLOS-II<2,2,5> (Fig. 3-4).
So X(S,D)=10010, t=0 and b=1.
In this case, the M-II Routing Strategy will prescribe the use of the bottom tree and a message between S and D will traverse four levels towards the root before being reflected towards its destination.

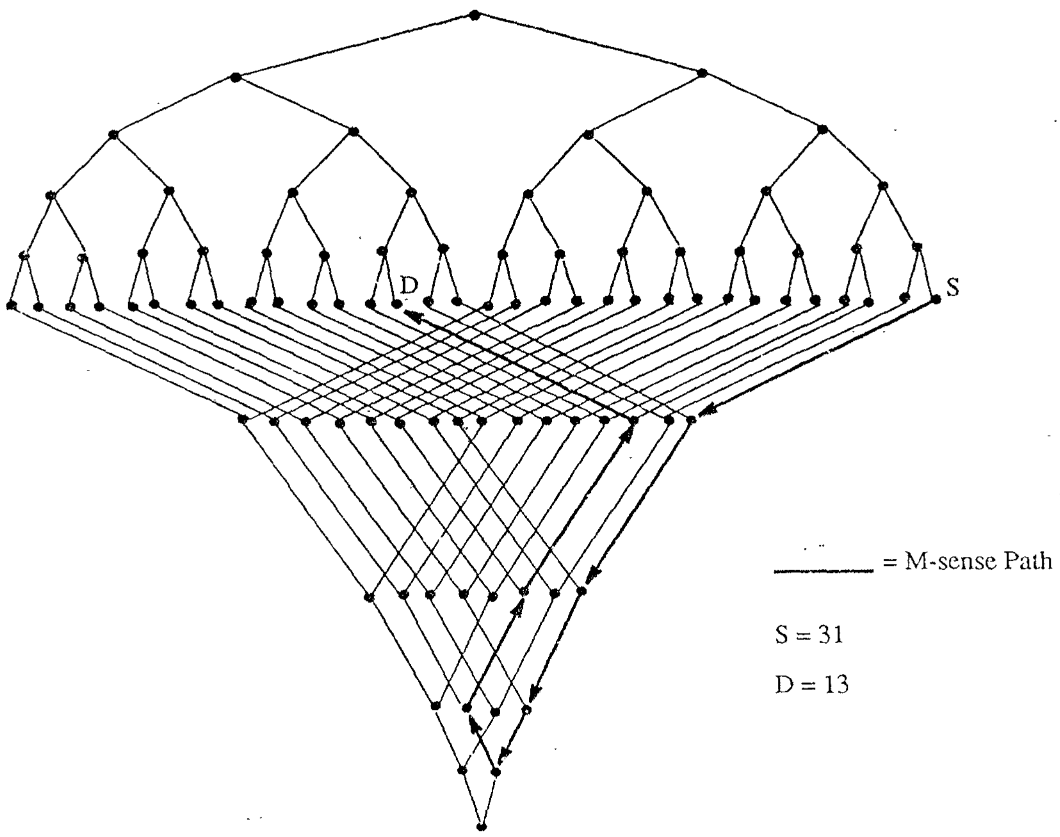


Figure 3-4: M-II Routing in KYKLOS-II<2,2,5>

3.1.4 Distance Matrices

Interprocessor distances will be represented by N × N matrices. Each such matrix has two subscripts: the first qualifies distance and the second subscript, x, denotes the KYKLOS version number.

Definition 3.3

- D_{1x} ≡ Tree 1 (Upper Tree) Distance Matrix defined by
  $D_{1x}[i,j] = d_{1x}(i,j)$, $i,j \in$ {leaf nodes}.

- D_{2x} ≡ Tree 2 (Lower Tree) Distance Matrix defined by
  $D_{2x}[i,j] = d_{2x}(i,j)$, $i,j \in$ {leaf nodes}.

- D_{mx} ≡ M-sense Distance Matrix defined by

  $D_{mx}[i,j] = d_{mx}(i,j)$, $i,j \in$ {leaf nodes}.

These matrices for KYKLOS-II<2,2,3> are shown in Fig.3-5.

Comments

Row Permutatibility Property: Any row is a permutation of any other row. This implies that the distance characteristics as viewed from one processor are identical as viewed from any other. For purposes of analysis we need consider any one row (say row 0 corresponding to Processor 0).

Worst Case Distance: This is characterized by the following definition.

Definition 3.4 *M-sense processor diameter*, denoted k_{m2}, is the maximum m-sense distance between any two processors. By analogy, processor diameters for Tree 1 and Tree 2 are denoted k_{12} and k_{22} respectively. In the case of KYKLOS-II<2,2,3>, for example, $k_{12}=k_{22}=k_{m2}=6$.

Average Distance: The average m-sense interprocessor distance in KYKLOS-II<2,2,3> is 4.00 (Fig. 3-5) while the corresponding value for KYKLOS-I<2,2,3> is 4.86. Average distance values are computed in Section 3.3. Before that, however, the m-sense interprocessor distance distribution is derived.

$$\begin{array}{cccc}
\begin{matrix}0&2&4&4&6&6&6&6\\2&0&4&4&6&6&6&6\\4&4&0&2&6&6&6&6\\4&4&2&0&6&6&6&6\\6&6&6&6&0&2&4&4\\6&6&6&6&2&0&4&4\\6&6&6&6&4&4&0&2\\6&6&6&6&4&4&2&0\end{matrix} &
\begin{matrix}0&6&4&6&2&6&4&6\\6&0&6&4&6&2&6&4\\4&6&0&6&4&6&2&6\\6&4&6&0&6&4&6&2\\2&6&4&6&0&6&4&6\\6&2&6&4&6&0&6&4\\4&6&2&6&4&6&0&6\\6&4&6&2&6&4&6&0\end{matrix} &
\begin{matrix}0&2&4&4&2&6&4&6\\2&0&4&4&6&2&6&4\\4&4&0&2&4&6&2&6\\4&4&2&0&6&4&6&2\\2&6&4&6&0&2&4&4\\6&2&6&4&2&0&4&4\\4&6&2&6&4&4&0&2\\6&4&6&2&4&4&2&0\end{matrix} &
\begin{matrix}0&2&4&4&2&4&4&6\\2&0&4&4&4&2&6&4\\4&4&0&2&4&6&2&4\\4&4&2&0&6&4&4&2\\2&4&4&6&0&2&4&4\\4&2&6&4&2&0&4&4\\4&6&2&4&4&4&0&2\\6&4&4&2&4&4&2&0\end{matrix} \\
\\
D_{12} & D_{22} & D_{m2} & D_{p2}
\end{array}$$

Figure 3-5: Distance Matrices for KYKLOS-II<2,2,3>

3.1.5 Interprocessor Distance Distribution

To make concrete our comparison of various KYKLOS networks, we attempt to estimate the number of processors at distance 2d from processor 0.

Definition 3.5 *M-sense Reach Number*, denoted $p_{mx}(d,n)$, is the number of processors that are at a distance 2d from Processor 0 using the M-II Routing Strategy in KYKLOS-x<r,m,n>. In particular, $p_{m2}(d,n) = |\{u|\ d_{m2}(0,u) = 2d\}|$.

Example 3.2 For KYKLOS-II<2,2,3> (Fig. 3-5),

$p_{m2}(1,3)=2$,

$p_{m2}(2,3)=3$,

$p_{m2}(3,3)=2$.

Theorem 3.2 In KYKLOS-II<m,2,n>,

$$\begin{aligned} p_{m2}(d,n) &= 2(m-1)m^{d-1}, & 1 \le d \le \lfloor n/2 \rfloor \\ &= 2(m-1)m^{d-1} - \lfloor (1-1/m^2)m^{2d-n} \rfloor, & \lceil n/2 \rceil \le d \le n \end{aligned}$$

Proof **Case (i): $\lceil n/2 \rceil \le i \le n$**

Define a set of distance vectors,

$$A_i = \{u | u \equiv 0^i 1 u_1\} \tag{3.1}$$

Since b+t<n from Remark 3.1 and t=i >⌈n/2⌉,

$$t > b \tag{3.2}$$

From Theorem 3.1

$$d_{m2}(0,u)=d_{12}(0,u)=2(n-i) \tag{3.3}$$

Define

$$B_i = \{u | u \equiv u_2 1 0^i\} \tag{3.4}$$

In a similar fashion,

$$b > t \tag{3.5}$$

and

$$d_{m2}(0,u)=d_{22}(0,u)=2(n-i) \tag{3.6}$$

From (3.3) and (3.6),

$$\{u | d_{m2}(0,u)=2(n-i)\} = A_i \cup B_i \tag{3.7}$$

Taking cardinality of both sides and using Defn. 3.5,

$$p_{m2}(n-i,n) = |A_i| + |B_i| \tag{3.8}$$

since from (3.2) and (3.5) $A_i \cap B_i = \Phi$

To find $|A_i|$, note that $u \in A_i \Rightarrow u \equiv 0^i 1 u_1$. In terms of the destination address, substrings 1 and u_1 may respectively be defined as elements of $S_1 = \{x | x \in \Sigma_m, x \neq 0\}$ and $S_2 = \{u | u \in \Sigma_m^{n-i-1}\}$ so that $|S_1| = m-1$ and $|S_2| = m^{n-i-1}$. From Remark 3.2, $|A_i| = (m-1)m^{n-i-1}$.

Since a similar analysis also holds for $|B_i|$, it follows that (3.8) may be expressed as

$$p_{m2}(n-i,n)=2(m-1)m^{n-i-1}, \qquad 0 \leq i < \lceil n/2 \rceil \tag{3.9}$$

Case(ii): $0 \leq i < \lfloor n/2 \rfloor$

Define A_i as

$$A_i = \{u|u \equiv 0^i 1 u_1 u_2,\ |u_2|=i, 1 \leq u_2\} \tag{3.10}$$

$\forall u \in A_i$, $t=i$ and $b<i$ since $1<u_2$ and $|u_2|=i$

So

$$b<t \tag{3.11}$$

From Theorem 3.1

$$d_{m2}(0,u)=d_{12}(0,u)=2(n\text{-}i)<d_{22}(0,u) \tag{3.12}$$

Define

$$B_i=\{u|u \equiv v_2 v_1 1 0^i,\ |v_2|=i,\ 1 \leq v_2\} \tag{3.13}$$

By similar reasoning,

$$t<b \tag{3.14}$$

and

$$d_{m2}(0,u)=d_{22}(0,u)=2(n\text{-}i)<d_{12}(0,u) \tag{3.15}$$

Define

$$C_i=\{u|u \equiv 0^i 1 w 1 0^i\} \tag{3.16}$$

So

$$b=t=i \tag{3.17}$$

From Theorem 3.1,

$$d_{m2}(0,u)=d_{12}(0,u)=d_{22}(0,u)=2(n\text{-}i) \tag{3.18}$$

So

$$\{u|d_{m2}(0,u)=2n\text{-}2i\}=A_i \cup B_i \cup C_i \tag{3.19}$$

Using (3.11),(3.14) and (3.17)

$$A_i \cap B_i = B_i \cap C_i = C_i \cap A_i = \Phi \tag{3.20}$$

Taking cardinality of both sides of (3.19) and using the above fact,

$$p_{m2}(n\text{-}i,n) = |A_i| + |B_i| + |C_i| \tag{3.21}$$

To find $|A_i|$ note that $u \in A_i \Rightarrow u \equiv 0^i 1 u_1 u_2$. In terms of the destination address, the substrings 1, u_1 and u_2 may respectively be defined as elements of the sets $S_1 = \{x|\ x \in \Sigma_m,\ x \neq 0\}$, $S_2 = \{u|\ u \in \Sigma_m^{n\text{-}2i\text{-}1}\}$ and $S_3 = \{u|\ u \in \Sigma_m^i,\ u \neq 0\}$ so that $|S_1| = m\text{-}1$, $|S_2| = m^{n-2i-1}$ and $|S_3| = m^i\text{-}1$. From Remark 3.2, $|A_i| = (m\text{-}1)(m^i\text{-}1)m^{n-2i-1}$. Similarly $|B_i| = (m\text{-}1)(m^i\text{-}1)m^{n-2i-1}$ and $|C_i| = (m\text{-}1)^2 m^{n-2i-2}$. On substitution into (3.21)

$$p_{m2}(n\text{-}i,n)=2(m\text{-}1)m^{n-i-1}\text{-}[(m^2\text{-}1)/m^2]m^{n-2i} \tag{3.22}$$

Case(iii): $i = \lfloor n/2 \rfloor$, n odd.

The definitions of A_i and B_i are the same as in Case (ii). However C_i is defined as

$$C_{\lfloor n/2 \rfloor} = \{u|u=0^i10^i\} \tag{3.23}$$

Here $|C_{\lfloor n/2 \rfloor}| = m\text{-}1$

so that

$$p_{m2}(n\text{-}\lfloor n/2 \rfloor,n) = 2(m\text{-}1)m^{n-i-1} \text{ - } (m\text{-}1) \tag{3.24}$$

Combining (3.9), (3.22) and (3.24) and substituting n-i for d, we obtain

$$\boxed{\begin{array}{ll} p_{m2}(d,n) = 2(m\text{-}1)m^{d-1}, & 1 \leq d \leq \lfloor n/2 \rfloor \\ \quad = 2(m\text{-}1)m^{d-1} \text{ - } \lfloor (1\text{-}1/m^2)m^{2d-n} \rfloor, & \lceil n/2 \rceil \leq d \leq n \end{array}} \tag{3.25}$$

Example 3.3 Substituting m=2, we verify the values for $p_{m2}(d,n)$, $1 \leq d \leq n$ for KYKLOS-II<2,2,3> that were obtained by inspection in Example 3.2.

3.2 P-II Routing and Distance

3.2.1 Concept of passthrough

Consider a message transfer where

$S = u1\ y1\ u2\ y2\ u3$

$D = u4\ y3\ u2\ y4\ u5$

where $|u4|=|u1|$, $|u5|=|u3|$, $y1 \neq y3$, $y2 \neq y4$, $|u2|=i \geq 0$.

So $X(S,D) = u6\ 1\ 0^{i}\ 1\ u7$

where $|u6|=|u1|$, $|u7|=|u3|$.

In the first phase, let the message be routed from S to an intermediate processor, A, through the top tree where

$A = u1\ y1\ u2\ y4\ u5$ (See Fig. 3-6)

So $X(S,A) = 0^{|u1|+1+i}\ 1\ u8$

where $|u8|=|u3|$

From the TD Lemma (3.1),

$d_{12}(S,A) = 2(n-|u1|-i-1)$

In the second phase, let the message be routed from A to D through the bottom tree.

Since $X(A,D) = u9\ 1\ 0^{i+1+|u5|}$, $|u9|=|u1|$,

it follows from the TD Lemma that

$d_{22}(A,D) = 2(n-|u5|-i-1) = 2(n-|u3|-i-1)$

The total pathlength between S and D is therefore

$2(n-|u1|-i-1) + 2(n-|u3|-i-1)$

$= 2(n-i) + 2(n-|u1|-|u3|-i-2)$

$=2(n-i)$

This establishes

Theorem 3.3 There exists a path using passthrough of length 2(n-i) between S and D where $X(S,D)=u_1 10^i 1 u_2$ and $|u_1|+|u_2|=n-i-2$. □

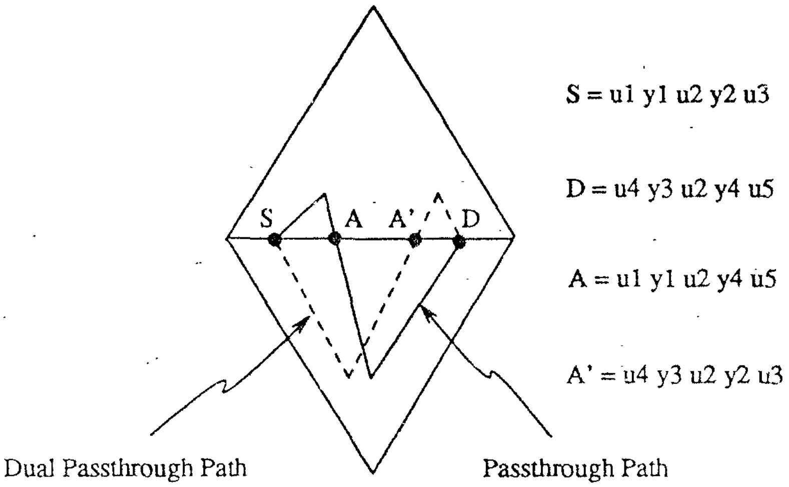


Figure 3-6: Illustrating Passthrough

Note that the two phases above could be interchanged i.e. the message could be sent from S to A' through the bottom tree first and from A' to D through the top tree in Phase 2 where A' = u4 y3 u2 y2 u3. It is easy to verify that the pathlength will remain 2(n-i). Also, the depth traversed in the top (or bottom) tree using either option is the same. We say that these two passthrough paths are duals of each other.

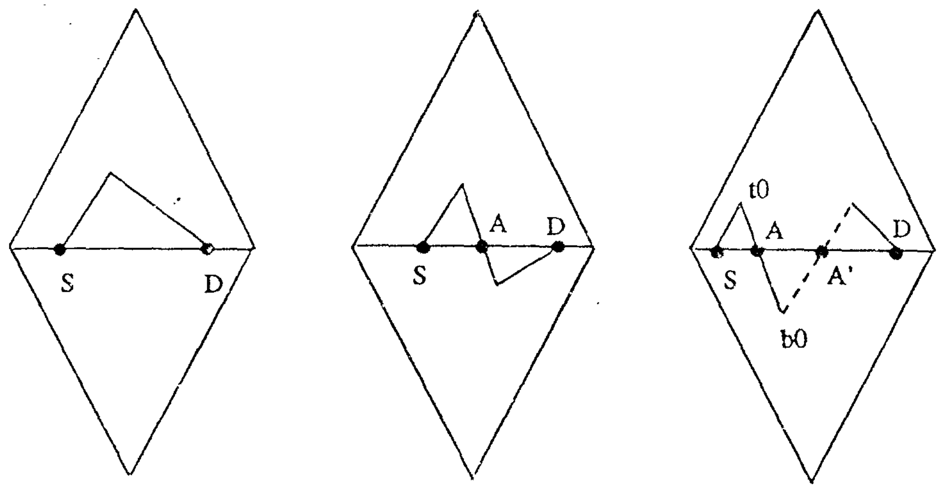
The strategy just described will, in general, entail identifying a cluster of consecutive zeros flanked by "1's" on either side. The larger the cluster of zeros, the shorter the pathlength by Theorem 3.3. Let p be the size of the maximum cluster of zeros flanked by two "1's" in X(S,D). Then it follows that

Corollary 3.3.1 The length of the shortest path between S and D using precisely one passthrough is 2(n-p) where $10^p1 < X(S,D)$ and $10^{p+j}1 \not< X(S,D)$, j>0. □

3.2.2 Shortest Path Routing

We are ready to investigate the shortest path between any two processors, S and D. The following three possibilities are exhaustive:

- The shortest path may involve a single tree i.e. tree 1 only or tree 2 only (Fig. 3-7(a)) or
- The shortest path may involve both trees with a single passthrough at some processor A, A≠S and A≠D (Fig. 3-7(b)) or
- The shortest path may involve both trees with multiple passthroughs at $A_1, A_2, \ldots, A_k$ (Fig. 3-7(c)).



(a) Non Passthrough

(b) Single Passthrough

(c) Multiple Passthrough

Figure 3-7: Three Possibilities for Shortest Path

Let us investigate the third possibility i.e. suppose more than one passthrough resulted in a shorter path between S and D not possible with either one passthrough or no passthrough at all. Let k>1 be the number of passthroughs that will guarantee the shortest path between S and D.

Without loss of generality, let t_0, $t_1,...,t_{\lfloor k/2 \rfloor}$ be the excursions (link traversals toward root) into the top tree and $b_0,b_1,...,b_{\lfloor (k-1)/2 \rfloor}$ be the excursions into the bottom tree (Fig. 3-7(c)). The total pathlength, L is given by

$$L = 2\{\sum_{i=0}^{\lfloor (k/2) \rfloor} t_i + \sum_{i=0}^{\lfloor (k-1)/2 \rfloor} b_i\} \tag{3.26}$$

Then

$$\sum_{i=0}^{\lfloor k/2 \rfloor} t_i + \sum_{i=0}^{\lfloor (k-1)/2 \rfloor} b_i < n \tag{3.27}$$

or else a simple path through the root of either of the two trees could be used between S and D.

Let

$$t_{max} = \max_{i=0,\lfloor k/2 \rfloor}\{t_i\} \tag{3.28}$$

and

$$b_{max} = \max_{i=0,\lfloor (k-1)/2 \rfloor}\{b_i\} \tag{3.29}$$

From (3.27),

$$t_{max} + b_{max} < n \tag{3.30}$$

This implies that each of the one or more bits between the b_{max} leftmost bits and the t_{max} rightmost bits of X(S,D) are zero i.e. digits t_{max} through n-b_{max}-1 of S and D are identical (Fig. 3-8).

From Theorem 3.3, there exists a path of length $2(b_{max} + t_{max})$ between S and D using a single passthrough.

From (3.26), (3.28) and (3.29), the length of this path is < L. We thus have a shorter path which uses a single passthrough, a contradiction of our earlier assumption that the shortest path between a given S-D pair may necessitate multiple passthroughs.

This eliminates the third possibility so that

Theorem 3.4 The shortest path between two processors need never involve more than one passthrough. □

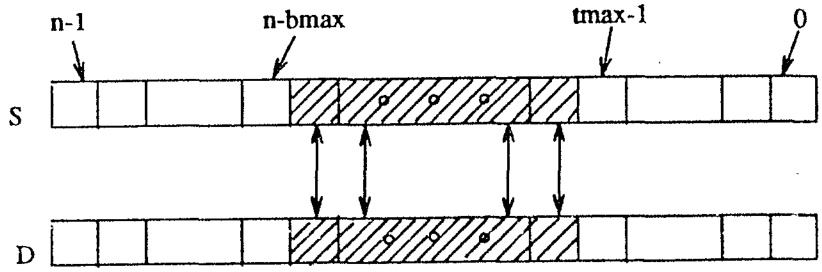


Figure 3-8: Showing b_{max} and t_{max} in X(S,D)

This means that the shortest path between two processors is wholly within 1 tree or involves both trees with only a single passthrough. If the former is the case, the pathlength is related to the size of the larger of the two clusters of zeros beginning at the MSB or LSB end of X(S,D) by Theorem 3.1. If the latter were the case, the length of the shortest path would be related to the size of the largest cluster of consecutive zeros flanked by 1's on either side from Corollary 3.3.1. These two cases may be combined to obtain

Theorem 3.5 The length of the shortest path between S and D, $d_{p2}(S,D)$, is related to the width of the maximum cluster of consecutive zeros *anywhere* in X(S,D) or, in symbols,

$d_{p2}(S,D) = 2(n-i)$ where $0^i < X(S,D)$ and $0^{i+j} \not< X(S,D), j>0$. □

The symbol $d_{m2}(S,D)$ was used to define the m-sense distance between S and D. By analogy $d_{p2}(S,D)$ has been used to define the p-sense distance between S and D. P-sense means that, unlike the m-sense case, there is no restriction on passing between trees i.e. confining the path to one of two trees. Strictly speaking, the qualification p-sense is superfluous since distance, by definition, is the length of the shortest path between two nodes. We, however, retain it for the sole purpose of differentiating it from the m-sense case. As a further clarification, note that the p-sense path is merely the shortest path between two processors which may not even involve passthrough.

We had defined t and b as the size of the maximum cluster of consecutive zeros beginning at the MSB and LSB end of X(S,D) respectively. Also, p was defined as the size of the maximum cluster of zeros flanked by "1's". Using these symbols, Theorem 3.5 may be expressed as

Corollary 3.5.1 $d_{p2}(S,D) = 2(n-i)$ where $i = \max(t,b,p)$. □

This leads to the P-II Routing Strategy.

3.2.3 P-II Routing Strategy

Let S be the source and D the destination of a message.

Compute X(S,D), t, b and p.

If $t > b$ and $t \geq p$, use the top tree

else if $b > t$ and $b \geq p$, use the bottom tree

else if $b = t$ and $b \geq p$, use either top or bottom tree

else use passthrough in the manner described in subsection 3.2.1. where u2 is chosen so that $|u2| = p$.

Example 3.4 Consider routing a message from S=31(11111) to D=13(01101) using the above routing strategy in KYKLOS-II<2,2,5>. Here X(S,D)=10010, t=0, b=1 and p=2. So the distance between S and D is 6 link traversals. The path prescribed by the P-II Routing Strategy is shown in thick lines in Fig. 3-9. Also shown is the dual path in dashed lines. Finally, note that the corresponding pathlength between S and D using M-II routing is 8 links as shown in Fig. 3-4.

We note some salient points about the P-II Routing Strategy

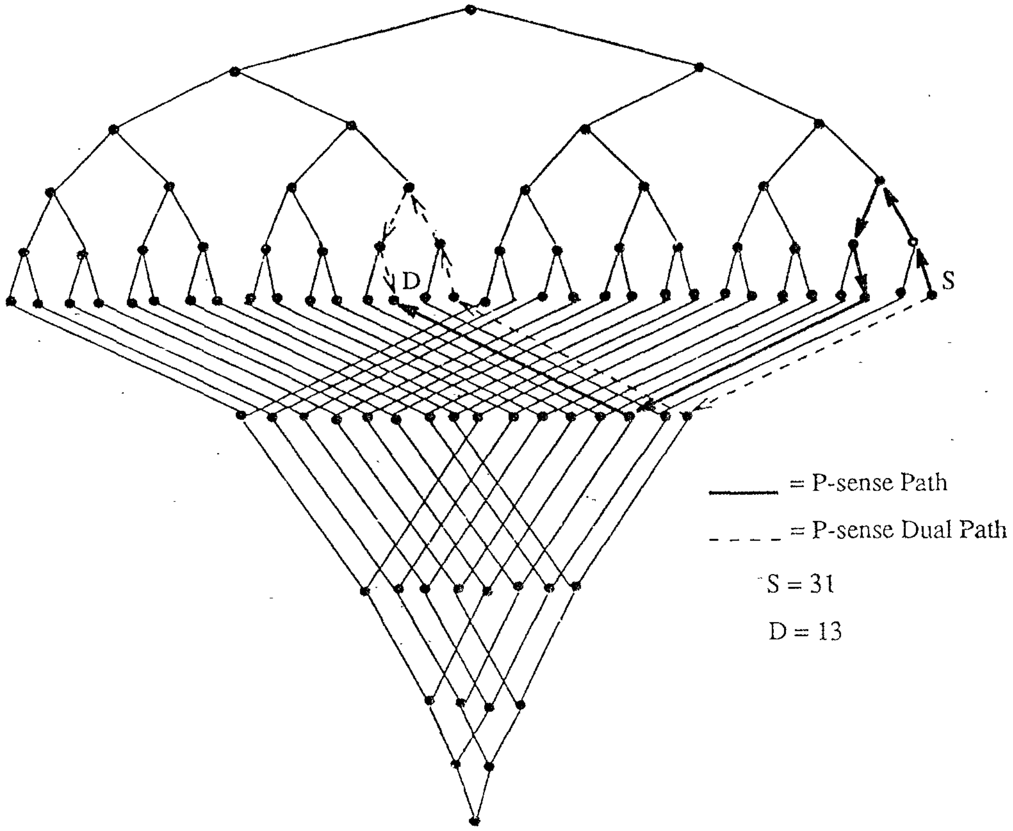


Figure 3-9: P-II Routing in KYKLOS-II<2,2,5>

Non Passthrough Case If there are two shortest paths between S and D, one using passthrough and another without passthrough, the latter will be used.

Passthrough Case As noted earlier, every passthrough path has a corresponding dual. Hence it is possible to use the bottom tree first before passing into the top tree. To guarantee fairness, both options should be exercised with the same frequency.

It may also happen that there are several clusters of p consecutive zeros i.e. there may be several candidate values for the prefix u1 in the source address string of Section 3.2.1. A discussion of this is postponed until Chapter 4.

The following definitions and properties for the p-sense distances are direct analogies of those in the m-sense case.

Definition 3.6 *P-sense Distance Matrix*, D_{Px}, is defined as $D_{Px}[i,j] = d_{Px}(i,j)$, $i,j \in$ {leaf nodes}. As an example, D_{p2} for KYKLOS-II<2,2,3> is shown in Fig. 3-5.

Row Permutatibility: As in the m-sense case, every row is a permutation of every other row.

Definition 3.7 *Processor Diameter*, denoted k_{p2}, is the maximum distance between two processors. From Fig. 3-5, k_{p2} of KYKLOS-II<2,2,3> is 6.

Definition 3.8 *P-sense Reach Number*, denoted $p_{p2}(d,n)$, is the number of processors at a distance 2d from Processor 0 using the P-II Routing Strategy in KYKLOS-II<m,2,n>. i.e.

$$p_{p2}(d,n) = |\{D \mid d_{p2}(0,D) = 2d\}|. \qquad (3.31)$$

From Theorem 3.5

$$p_{p2}(d,n) = |\{D \mid |X(0,D)|=n \wedge 0^{n-d}<X(0,D) \wedge 0^{n-d+j} \nless X(0,D), j>0\}| \qquad (3.32)$$

Example 3.5 For KYKLOS-II<2,2,3> (Fig.3-5),

$p_{p2}(1,3)=2$,

$p_{p2}(2,3)=4$,

$p_{p2}(3,3)=1$.

3.2.4 P-sense Distance Distribution

Definition 3.9 *Cumulative P-sense Reach Number*, denoted p_{px}'(d,n), is the number of processors at a distance 2d or greater from Processor 0 using the P-x (Shortest Path) Routing Strategy in KYKLOS-x<r,m,n> or[14]

$$p_{px}'(d,n) = \sum_{i=d}^{n} p_{px}(i,n) \tag{3.33}$$

From (3.32),

$$p_{p2}'(d,n) = |\{D \mid |X(0,D)|=n \wedge 0^{n-d+j} \not\prec X(0,D), j>0\}| \tag{3.34}$$

Extending Example 3.5,

$p_{p2}'(0,3)=8$

$p_{p2}'(1,3)=7$

$p_{p2}'(2,3)=5$

$p_{p2}'(3,3)=1$.

Remark 3.3: $p_{p2}'(j,n) = m^n$, $j = 0, -1, -2, \ldots$
$p_{p2}'(n+k,n) = 0$, $k > 0$.

Theorem 3.6 In KYKLOS-II<m,2,n>,

$p_{p2}'(d,n) = (m-1)\sum_{k=1}^{n-d+1} p_{p2}'(d-k,n-k)$, $0<d\leq n$, $n\geq 2$.

Proof

$$\text{Let } E_{k,i,n} \equiv \{0^k 1 u_i \mid |u_i|=n\text{-}k\text{-}1 \;\wedge\; 0^{i+j} \not\prec u_i, j>0\} \tag{3.35}$$

In words, $E_{k,i,n}$ is the set of all distance vectors (n-bit strings) prefixed by 0^k1 and with suffixes of length n-k-1 that do not have i (or more) consecutive zeros.

We next consider

[14] Actually N-p_{p_x}'(d+1,n) represents the cumulative reach number. However, the definition as is results in some interesting mathematical properties

$$\cup_{k=0,i}\mathrm{E}_{k,i,n} = \{0^k 1 \mathrm{u}_i \mid 0 \leq \mathrm{k} \leq \mathrm{i} \ \wedge \ |\mathrm{u}_i| = \mathrm{n-k-1} \ \wedge \ 0^{i+j} \nprec \mathrm{u}_i, \mathrm{j}>0\} \tag{3.36}$$

The above union results in a set of all n-bit strings that have no more than i consecutive zeros or

$$\cup_{k=0,i}\mathrm{E}_{k,i,n} = \{\mathrm{u} \mid |\mathrm{u}| = \mathrm{n} \ \wedge \ 0^{i+j} \nprec \mathrm{u}, \mathrm{j}>0\} \tag{3.37}$$

Taking cardinalities and using (3.34) we get

$$\mathrm{p}_{p2}'(\mathrm{n-i,n}) = |\{\mathrm{D} \mid \mathrm{X(0,D)} \in \cup_{k=0,i}\mathrm{E}_{k,i,n}\}| \tag{3.38}$$

Since the prefixes $0^{k_1}1$ and $0^{k_2}1$, $\mathrm{k}_1 \neq \mathrm{k}_2$, for every pair of strings, one from $\mathrm{E}_{k_1,i,n}$ and the other from $\mathrm{E}_{k_2,i,n}$ are mutually exclusive

$$\mathrm{E}_{k_1,i,n} \cap \mathrm{E}_{k_2,i,n} = \Phi, \quad \mathrm{k}_1 \neq \mathrm{k}_2 \tag{3.39}$$

So

$$p_{p2}'(n-i,n) \ = \ \sum_{k=0}^{i} |E_{k,i,n}| \tag{3.40}$$

Consider the cardinality of the set $\mathrm{E}_{k,i,n}$ defined in Eqn. (3.35).

In terms of the destination addresses, substrings 1 and u_i may be respectively defined as elements of $\mathrm{S}_1 = \{\mathrm{x} \mid \mathrm{x} \in \Sigma_m, \mathrm{x} \neq 0\}$ and $\mathrm{S}_2 = \{\mathrm{u} \mid |\mathrm{u}| = \mathrm{n-k-1} \wedge 0^{i+j} \nprec \mathrm{u}, \mathrm{j}>0\}$ so that $|\mathrm{S}_1| = \mathrm{m-1}$. Also, from Eqn. (3.34), $|\mathrm{S}_2| = \mathrm{p}_{p2}'(\mathrm{n-k-i-1,n-k-1})$ so that Eqn. (3.40) may be rewritten as

$$p_{p2}'(n-i,n) \ = \ \sum_{k=0}^{i} (m-1) p_{p2}'(n-k-i-1, n-k-1) \tag{3.41}$$

Letting d=n-i and assigning k-1 to k

$$\boxed{p_{p2}'(d,n) \ = \ (m-1) \sum_{k=1}^{n-d+1} p_{p2}'(d-k, n-k), \quad 0 < d \leq n} \tag{3.42}$$

3.2.5 Computing $p_{p2}'(d,n)$

$p_{p2}'(d,n)$ is shown graphed on the d,n plane for m=2 (Table 3-1) and m=3 (Table 3-2). Remark 3.3 defines the quadrant $d \leq 0$, $n \geq 0$. We next show that Remark 3.3 supplies the initial conditions for a set of recurrences defined by Theorem 3.6 which is used to determine $p_{p2}'(d,n)$, $d=1,2,\ldots,n$; $n=1,2,\ldots$

n->	0	1	2	3	4	5	6	7	8
d = -5	1	2	4	8	16	32	64	128	256
d = -4	1	2	4	8	16	32	64	128	256
d = -3	1	2	4	8	16	32	64	128	256
d = -2	1	2	4	8	16	32	64	128	256
d = -1	1	2	4	8	16	32	64	128	256
d = 0	1	2	4	8	16	32	64	128	256
d = 1	-	1	3	7	15	31	63	127	255
d = 2	-	-	1	5	13	29	61	125	253
d = 3	-	-	-	1	8	24	56	120	248
d = 4	-	-	-	-	1	13	44	108	236
d = 5	-	-	-	-	-	1	21	81	208
d = 6	-	-	-	-	-	-	1	34	149
d = 7	-	-	-	-	-	-	-	1	55
d = 8	-	-	-	-	-	-	-	-	1

Table 3-1: Cumulative Reach Number Values for KYKLOS-II<2,2,n> (p-sense)

Expanding Theorem 3.6

$$p_{p2}'(n-k,n) = (m-1)[p_{p2}'(n-k-1,n-1) + p_{p2}'(n-k-2,n-2)\ldots k+1 \text{ terms}]. \qquad (3.43)$$

From Remark 3.3,

$$p_{p2}'(-k,0) = p_{p2}'(-k+1,1) = \ldots = p_{p2}'(0,k) = m^k, \quad k \geq 0 \qquad (3.44)$$

Equation (3.43) defines a $k+1^{th}$ order recurrence while (3.44) provides the initial conditions. Together, they determine the sequence

$p_{p2}'(-k,0)$, $p_{p2}'(-k+1,1)$, $p_{p2}'(-k+2,2)$, ...

n->	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
d = -4	1	3	9	27	81	243	729	2187	6561
d = -3	1	3	9	27	81	243	729	2187	6561
d = -2	1	3	9	27	81	243	729	2187	6561
d = -1	1	3	9	27	81	243	729	2187	6561
d = 0	1	3	9	27	81	243	729	2187	6561
d = 1	-	2	8	26	80	242	728	2186	6560
d = 2	-	-	4	22	76	238	724	2182	6556
d = 3	-	-	-	8	60	222	708	2166	6540
d = 4	-	-	-	-	16	164	648	2106	6480
d = 5	-	-	-	-	-	32	448	1892	6264
d = 6	-	-	-	-	-	-	64	1224	5525
d = 7	-	-	-	-	-	-	-	128	3344
d = 8	-	-	-	-	-	-	-	-	256

Table 3-2: Cumulative Reach Number Values for KYKLOS-II<3,2,n> (p-sense)

This sequence may be thought of as being mapped on to discrete (integral) coordinates on the semi-infinite line d=n-k originating at d=-k, n=0. We refer to the line d=n as the *principal diagonal* and to the family of lines d=n-k as *paradiagonals* (lines parallel to the principal diagonal).

The following observations are in the nature of corollaries of Theorem 3.6

i) For k=0[15], the above sequence maps a geometric sequence with factor m-1 onto the principal diagonal. In the binary case (Table 3-1), this is a sequence of 1's. This means that there is precisely one processor at a maximum distance of 2n from any given processor in KYKLOS-II<2,2,n>.

[15] Note that k+1 is the order of the recurrence.

ii) For m=2, k=1, the sequence mapped onto the paradiagonal d=n-1 is the well known Fibonacci sequence in which any term is the sum of the previous two terms. For m=2, k>1, the sequence is the sum of the previous k+1 terms referred to as a Tribonacci sequence in [GODS83].

iii) For m>2, the above sequence is a further generalization of the Fibonacci sequence in that any given term is the sum of the preceding k+1 terms multiplied by m-1.

In Appendix B, an attempt is made at studying the roots of the characteristic polynomial of (3.43) insofar as the results provide insight into certain asymptotic properties of KYKLOS.

3.3 Comparison of m-sense and p-sense Distance Characteristics

3.3.1 Reach Factor

Definition 3.10 *Reach Factor*, denoted $p_{zx}''(d,n)$, is defined as the number of processors within 2d link traversals of node 0 in KYKLOS-x<m,2,n> using the z-x Routing Strategy, expressed as a fraction of the total number of processors.

From Definition 3.5, it follows that

$$p_{m2}''(d,n) = \frac{\sum_{i=0}^{d} p_{m2}(i,n)}{N} \qquad (3.45)$$

Using Definition 3.9,

$$\begin{aligned} p_{p2}''(d,n) &= \frac{N - p_{p2}'(d+1,n)}{N} \\ &= 1 - \frac{p_{p2}'(d+1,n)}{N} \qquad (3.46) \end{aligned}$$

For purposes of comparing the reach factor using different KYKLOS topologies and routing strategies, consider a 4096 processor KYKLOS (m=2, n=12). Now, in a single tree or in KYKLOS-I, no more than 50% of the processors can be reached within a distance 2n-2 (or 22 link traversals in this case). Using the M-II Routing Strategy, 75% of the processors are within 22 link traversals of a given processor as shown in Table 3-4. Reach Factor values in the p-sense case for KYKLOS-II<2,2,n> are also tabulated in Table 3-3. It is for this case that the reach factor values are truly impressive with close to 100% of the nodes being within a distance of 22 link traversals.

This is a special case of a more general observation that the reach factor values along a paradiagonal in the M-II case (or in KYKLOS-I) remain constant beyond a certain value of n. For example, at and beyond n=4, $p_{m2}''(n-2,n)$ is 44%. The corresponding figure in KYKLOS-I is 25%. However, $p_{p2}''(n-2,n)$ increases monotonically. How long will it increase? Does it have an asymptote?

This may be answered by substituting Eqn. B.26 (Appendix B) into (3.46) to obtain

$$p_{p2}''(d,n) = 1 - \frac{c_{1,n-d+1,m} r^{n}_{1,n-d+1,m} + c_{2,n-d+1,m} r^{n}_{2,n-d+1,m} + \ldots n-d+1\,terms}{m^n} \tag{3.47}$$

Since $|r_i| < m$, i=1,2, ... n-d+1 (Eqn. B.25), it follows that, for a given value of d

$$\lim_{n \to \infty} p_{p2}''(d,n) = 1 \tag{3.48}$$

This implies that the reach factor values along a paradiagonal in the P-II case (Table 3-3) asymptotically approach unity. This does not, however, hold for a simple binary tree (and hence KYKLOS-I) or for the m-sense case as we had observed earlier. In fact, regardless of the value of n, $p_{p1}''(n-1,n) = .5$ (reach factor for KYKLOS-I<2,2,n>) and $p_{m2}''(n-1,n) = .75$ (for KYKLOS-II<2,2,n> using the M-II Routing Strategy).

n->	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
d=0	0.06	0.03	0.02	0.01	0.00	0.00	0.00	0.00	0.00
d=1	0.19	0.09	0.05	0.02	0.01	0.01	0.00	0.00	0.00
d=2	0.50	0.25	0.13	0.06	0.03	0.02	0.01	0.00	0.00
d=3	0.94	0.59	0.31	0.16	0.08	0.04	0.02	0.01	0.00
d=4	1.00	0.97	0.67	0.37	0.19	0.09	0.05	0.02	0.01
d=5	-	1.00	0.98	0.73	0.42	0.22	0.11	0.05	0.03
d=6	-	-	1.00	0.99	0.79	0.46	0.25	0.12	0.06
d=7	-	-	-	1.00	1.00	0.83	0.51	0.27	0.14
d=8	-	-	-	-	1.00	1.00	0.86	0.55	0.30
d=9	-	-	-	-	-	1.00	1.00	0.89	0.58
d=10	-	-	-	-	-	-	1.00	1.00	0.91
d=11	-	-	-	-	-	-	-	1.00	1.00
d=12	-	-	-	-	-	-	-	-	1.00

Table 3-3: Reach Factor Values in p-sense case for KYKLOS-II<2,2,n>

n->	4	5	6	7	8	9	10	11	12
d=0	0.06	0.03	0.02	0.01	0.00	0.00	0.00	0.00	0.00
d=1	0.19	0.09	0.05	0.02	0.01	0.01	0.00	0.00	0.00
d=2	0.44	0.22	0.11	0.05	0.03	0.01	0.01	0.00	0.00
d=3	0.75	0.44	0.23	0.12	0.06	0.03	0.01	0.01	0.00
d=4	1.00	0.75	0.44	0.23	0.12	0.06	0.03	0.02	0.01
d=5	-	1.00	0.75	0.44	0.23	0.12	0.06	0.03	0.02
d=6	-	-	1.00	0.75	0.44	0.23	0.12	0.06	0.03
d=7	-	-	-	1.00	0.75	0.44	0.23	0.12	0.06
d=8	-	-	-	-	1.00	0.75	0.44	0.23	0.12
d=9	-	-	-	-	-	1.00	0.75	0.44	0.23
d=10	-	-	-	-	-	-	1.00	0.75	0.44
d=11	-	-	-	-	-	-	-	1.00	0.75
d=12	-	-	-	-	-	-	-	-	1.00

Table 3-4: Reach Factor Values in m-sense case for KYKLOS-II<2,2,n>

3.3.2 Average Distance Estimation

Average m-sense and p-sense interprocessor distances in KYKLOS-II<m,2,n> are respectively given by

$$d_{m2}(n) = \frac{\sum_{d=1}^{n} 2dp_{m2}(d,n)}{N} \tag{3.49}$$

and

$$\begin{aligned} d_{p2}(n) &= \frac{\sum_{d=1}^{n} 2dp_{p2}(d,n)}{N} \\ &= 2\frac{\{[p_{p2}(n,n)]+[p_{p2}(n-1,n)+p_{p2}(n,n)]\ldots+[p_{p2}(1,n)+p_{p2}(2,n)\ldots+p_{p2}(n,n)]\}}{N} \\ &= \frac{2\sum_{d=1}^{n} p_{p2}'(d,n)}{N} \end{aligned} \tag{3.50}$$

d_{m2}(n) and d_{p2}(n) are tabulated (Table 3-5) for the binary tree and KYKLOS-II<2,2,n> using both M-II and P-II Routing Strategies. Observe that up to 1024 leaf nodes, P-II Routing improves the average interprocessor distance by 20-25% over the simple binary tree (and hence KYKLOS-I) though the improvement in the m-sense case is not as marked.

In Chapter 7, the average distance values are compared with other augmented Binary Tree topologies and with the Hypercube.

n	BT	d_{m2}	d_{p2}
3	4.25	3.50	3.25
4	6.13	5.13	4.63
4	8.06	6.94	6.13
6	10.03	8.81	7.69
7	12.02	10.75	9.31
8	14.01	12.71	10.98
9	16.00	14.69	12.68
10	18.00	16.68	14.40
11	20.00	18.67	16.15
12	22.00	20.67	17.92
13	24.00	22.67	19.71
14	26.00	24.67	21.51

BT = Binary Tree

Table 3-5: Average interprocessor distances, Binary Tree and KYKLOS-II<2,2,n>, m-sense and p-sense

Chapter 4

Traffic Estimation

In this chapter, we look at the traffic through the links in KYKLOS-II<m,2,n>. We study both, the variation of network traffic as a function of link level as well as the performance-limiting traffic bottleneck problem.

As in the case of distance properties, KYKLOS-II will be compared with KYKLOS-I and with the single binary tree. Also, different routing strategies employed in KYKLOS-II will be compared with respect to traffic congestion, link utilization, etc.

4.1 Preliminaries

To analyze the traffic characteristics of KYKLOS-II, we assume that each link in the network is being monitored for messages in both directions. We use the "uniform message distribution" model to analyze network traffic. This model provides a useful upper bound on maximum link traffic by ignoring the effect of locality under which a real world system would operate. By so doing, the analysis is somewhat simplified. Moreover, we have a common standard by which to compare different network topologies. The following assumptions characterize our model

- A1: The probability of a message transfer between a pair of processors is the same for every pair (i,j), $i \neq j$. Further, the average length of a message exchanged by any pair of processors is the same.

- A2: If there is more than one shortest path between a given source-destination pair under a given routing scheme, each path will be used with equal probability.

- A3: Each processor has the same probability of generating a message at any given instant.

From A3, each processor generates the same number of messages in an "observation interval", τ. From A1, this "observation interval" may be thought of as being comprised of several rounds of message exchange interleaved, a round being a period during which each processor sends out a message to every other processor.

4.1.1 Symmetry within a given Link Level

From A3, each pair of level ±1 links incident on a processor would see the same number of messages as any other during τ. From the digit reversal property of the LS for the bottom tree of KYKLOS-II and A1 and A2, we conclude that both trees will be equally utilized. Hence, each link at level ±1 will see the same number of messages. We can extend this argument to encompass higher and higher levels (lower and lower levels in the bottom tree) until it is clear that each link at any given level i or -i will see the same number of messages in a *round* of τ.

In summary, given the KYKLOS-II topology, and A1-A3, we conclude that the traffic through a link is only a function of its level number[16] in the network and is independent of the tree (i.e. the sign of the level number).

[16] An exception to this rule occurs in KYKLOS-II using H-II routing with n odd. This routing is introduced in Section 4.3.

4.2 M-sense Traffic Characteristics

Definition 4.1 Let $T_{m2}(d,n)$ be the number of messages that traverse a level +d or -d link in either direction in a KYKLOS-II<m,2,n> using the M-II Routing Strategy under A1-A3 in a *round* of message exchange.

For a given source, S, define

$$Q_{in} = \{D \mid d_{m2}(S,D) = 2i\} \tag{4.1}$$

In words, Q_{in} is the set of destinations at an m-sense distance of 2i from S. This means that a message from S to D', D' $\in Q_{in}$, will traverse links between levels 1 to i in either top or the bottom tree as shown in Fig. 4-1.. Since a link at level +i or -i must see a message between processor pairs at an m-sense distance 2i from each other, the total number of messages through levels ±i due to messages originating at S is $\sum_{k=i}^{n} |Q_{kn}|$. Since there are N source processors and $2m^{n-i+1}$ links at level ±i, $T_{m2}(i,n)$ is given by:

$$T_{m2}(i,n) = \frac{2N\sum_{k=i}^{n} |Q_{kn}|}{2m^{n-i+1}} \tag{4.2}$$

Using Defn. 3.5 for $p_{m2}(d,n)$, we have

Theorem 4.1 In KYKLOS-II<m,2,n>, the link traffic distribution is given by

$$\boxed{T_{m2}(i,n) = m^{i-1}\sum_{k=i}^{n} p_{m2}(k,n)}$$

$T_{m2}(i,n)$ is tabulated as a function of level number for KYKLOS-II<2,2,6> (Table 4-1). Also, the link utilization (Util), defined as the ratio of link traffic to the traffic in the maximally congested link in the network, expressed as a percentage, is tabulated. For comparison, the link traffic densities and utilization for the single binary tree are included. Note that the maximum traffic density in KYKLOS-II using M-II Routing is

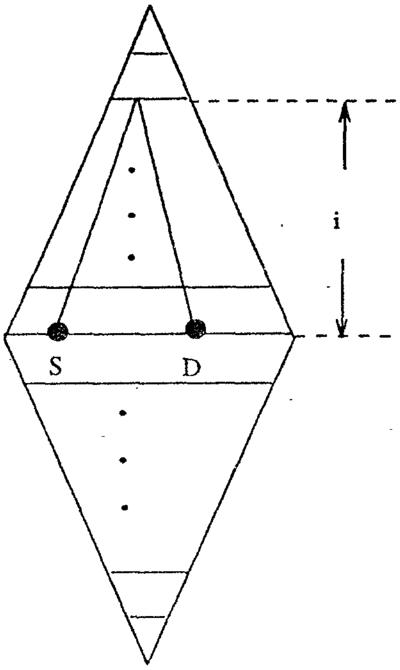


Figure 4-1: Link Utilization of message between processors at an m-sense distance = 2i

only about 25% that in the single tree case. Finally, it is clear that the relative utilization of link bandwidth is more uniform in the M-II Routing case as compared with that of the single tree.

Level #	B-Tree	Util	M-II	Util
6	2048	100.00	512	88.89
5	1536	75.00	576	100.00
4	896	43.75	392	68.05
3	480	23.44	228	39.58
2	248	12.11	122	21.18
1	126	6.15	63	10.94

B-Tree = Single Binary Tree

Util = Link Utilization

Table 4-1: Link Traffic Density as function of Level Number in KYKLOS-II<2,2,6> using M-II Routing and in a single tree (height 6)

Our next task is to determine the maximum link traffic density and identify the level number where this occurs.

In Appendix C , it is shown that for m≥3 (Eqn. C.4),

$$p_{m2}(d,n) \;>\; p_{m2}(d-1,n) \qquad 2\leq d\leq n,\;\; n\geq 2$$

$$So\;\; \sum_{i=d}^{n} p_{m2}(i,n)+p_{m2}(d,n) \;>\; \sum_{i=d-1}^{n} p_{m2}(i,n)$$

$$So\;\; m^{d-1}\sum_{i=d}^{n} p_{m2}(i,n) \;>\; m^{d-2}\sum_{i=d-1}^{n} p_{m2}(i,n)$$

$$or\;\; T_{m2}(d,n) \;>\; T_{m2}(d-1,n), \qquad 1<d\leq n,\;\; n>1. \tag{4.3}$$

Using reasoning similar to the above, it follows from Appendix C (Eqn. C.5) that for m=2, n>2,

$$T_{m2}(d,n) \;>\; T_{m2}(d-1,n), \qquad 1\leq d\leq n-1. \tag{4.4}$$

Hence the maximum link traffic should occur at level n or n+1. Using Theorem 3.2 for expressions for $p_{m2}(n-1,n)$ and $p_{m2}(n,n)$, n≥4, and substituting into Theorem 4.1,

$$T_{m2}(n,n) = N^2/8 \tag{4.5}$$

and

$$T_{m2}(n-1,n) = 9N^2/64 \tag{4.6}$$

Using (4.3)-(4.6) and Theorem 4.1, we have

Theorem 4.2

Under A1-A3, the maximum traffic density in KYKLOS-II<2,2,n>, n≥4, using M-II Routing is

$$9N^2/64 \quad messages/link\;\; and\;\; occurs\;\; in\;\; level\;\; n-1\;\; links.$$

In KYKLOS-II<m,2,n>, m≥3, n≥2, the maximum traffic density is

$$\frac{(m-1)^2N^2}{m^3} \quad messages/link\;\; and\;\; occurs\;\; in\;\; level\;\; n\;\; links.$$

4.3 H-II Routing and Traffic

4.3.1 Concept of Slice

Definition 4.2: A slice of KYKLOS-x<m,2,n> is a subgraph composed of:

1. nodes from levels i to -(n-i) inclusive,

2. links between levels i to -(n-i), $0 \leq i \leq n$.

A slice in KYKLOS-II<m,2,n> will be denoted by the symbol SL(i,n), where i is the level number which constitutes the upper (top tree) boundary of the slice (Fig. 4-2). The slice SL(n,n) is the subgraph from level n to level 0 which, of course, corresponds to the upper tree and SL(0,n) corresponds to the lower tree.

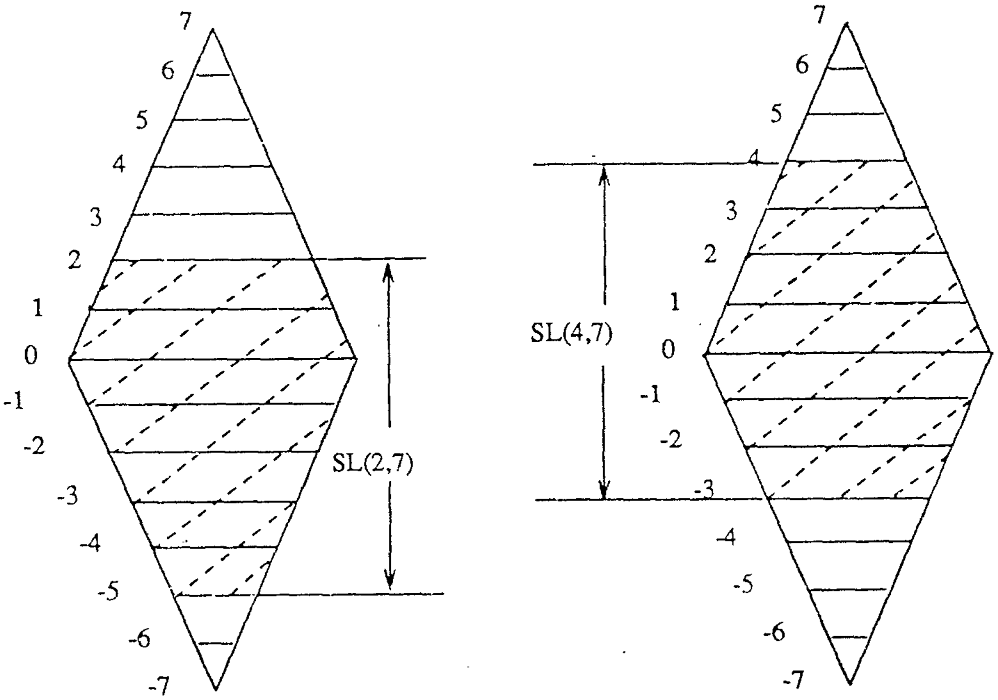


Figure 4-2: Slices in KYKLOS-II<m,2,n>

Theorem 4.3 There exist(s) one (or more) path(s) between every pair of processors in SL(l,n), $n \leq l \leq 0$, of an m.s.o. KYKLOS-x<m,2,n>.

Proof Define the set of processors within a subtree as follows:

$S_{t,l,i}$ = {k| k is a processor with ancestral ties to switch <t,l,i>}.

Sets $S_{1,l,i}$, $0 \le i < m^{n-l}$ and $S_{2,n-l,j}$, $0 \le j < m^l$ are shown in Fig. 4-3(a). Without loss of generality assume $l \le n-l$. Consider two processors x,y ∈ $S_{1,l,0}$ (see Fig. 4-3(a)). So

$$d_{1z}(x,y) \le 2l \tag{4.7}$$

From the definition of m-sense optimality (Section 2.4.1)

$$d_{2z}(x,y) > 2(n-l) \tag{4.8}$$

So x ∈ $S_{2,n-l,j}$ => y \∈ $S_{2,n-l,j}$. In words, no two processors in the same $S_{1,l,i}$ can exist in the same $S_{2,n-l,j}$. Since $|S_{1,l,i}| = m^l$ and $0 \le j < m^l$, we conclude that there exists a bijection ξ_i[17] s.t.

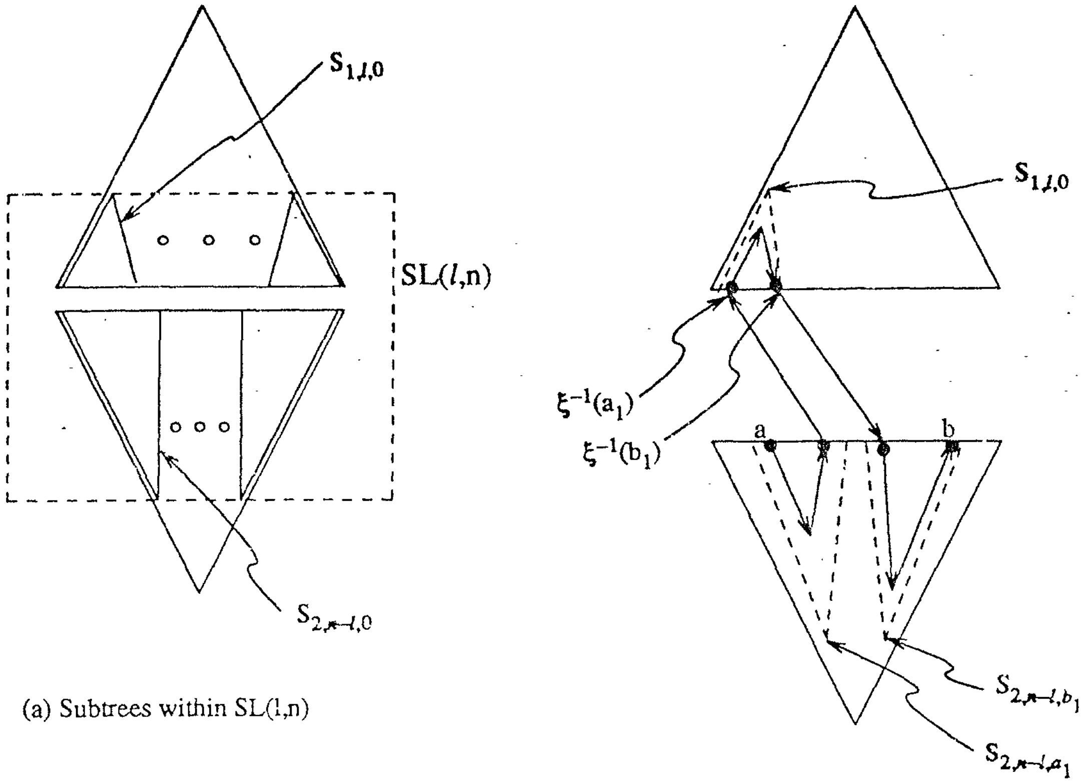
$$\xi_i(x)=y \quad \text{where } x \in S_{1,l,i} \wedge x \in S_{2,n-l,y}, \quad 0 \le y < m^l \tag{4.9}$$

To prove that every pair of processors within SL(l,n) is connected, consider two arbitrary processors, a ∈ $S_{2,n-l,a_1}$ and b ∈ $S_{2,n-l,b_1}$. If $a_1 = b_1$, they are obviously connected. If not, from (4.9), there exist processors $\xi_0^{-1}(a_1) \in S_{2,n-l,a_1}$ and $\xi_0^{-1}(b_1) \in S_{2,n-l,b_1}$ s.t. $\xi_0^{-1}(a_1), \xi_0^{-1}(b_1) \in S_{1,n,0}$. Hence a path between a and b would be a -> $\xi_0^{-1}(a_1)$ -> $\xi_0^{-1}(b_1)$ -> b. This is represented pictorially in Fig. 4-3(b). □

To each slice, there corresponds a different routing strategy and hence a different set of distance and traffic characteristics. A slice of special interest is the mid-slice defined as SL($\lceil n/2 \rceil$). While the mid-slice could be defined using the floor function, we will use the ceiling function unless otherwise specified.

We next present a routing algorithm which utilizes a mid slice to route a message between an arbitrary source-destination pair in KYKLOS-II<m,2,n>. Because this

[17] The subscript for ξ should, more appropriately, involve tree and level number besides node number. We have, however, dropped the former two since it is clear that we are referring to level l of Tree 1



(a) Subtrees within SL(l,n)

(b) Path from a to b.

Figure 4-3: SL(l,n) with subtrees $S_{1,l,i}$ and $S_{2,n-l,j}$ and path between a and b

strategy ensures that a message need travel at most half-way (or less) between the leaves and roots of either tree, it is referred to as the H-II Routing Algorithm.

4.3.2 H-II Routing Algorithm

Let the source and destination for a message transfer be respectively represented as

S=u1 u2, and

D=u3 u4, $|u2|=|u4|=\lceil n/2 \rceil$, $|u1|=|u3|=\lfloor n/2 \rfloor$.

Let A=u1 u4

Step 1 If u2 ≠ u4, use upper tree to route message from S to A.

Step 2 If u1 $\neq$ u3 use lower tree to route message from A to D.

Note that X(S,A) = $0^{\lfloor n/2 \rfloor}$u6, $|u6| = \lceil n/2 \rceil$. From the TD Lemma (3.1), $d_{12}(S,A) \leq 2(n - \lfloor n/2 \rfloor) = 2\lceil n/2 \rceil$ or alternately, no more than $\lceil n/2 \rceil$ levels of the upper tree would be utilized in this step. By similarly considering X(A,D), we conclude that at most $2\lfloor n/2 \rfloor$ link traversals of the bottom tree need be utilized in Step 2. Clearly, this strategy has confined a path between S and D within a mid slice.[18] In analogy to P-II and M-II Routing, we will refer to the path carved out by application of the H-II Routing Algorithm between two processors, S and D, as an *h-sense path* and denote the pathlength as $d_{h2}(S,D)$.

Finally, as in P-II Routing, the order of the above steps could be reversed i.e. the message could be sent from S to u3 u2 via the lower tree and then forwarded to D through the upper tree. This path is the h-sense dual of the path prescribed by the H-II Routing Algorithm (in analogy to the p-sense dual introduced in Section 3.2.1).

Example 4.1 Consider routing a message from S=00110 to D=01011 in KYKLOS-II<2,2,5>. Here u1=00, u2=110, u3=01, u4=011 and A=00011. So X(S,A)=00101 and u6=101. The entire h-sense path is sketched in dark lines as shown in Fig. 4-4. Note that three levels in the top tree and two levels in the bottom tree are used. The dual h-sense path is shown in the dashed lines of Fig. 4-4.

[18] Note that Theorem 4.3 established the existence of a path between every pair of processors in a slice of any m.s.o. KYKLOS. The above algorithm explains how to route a message between any two processors in a mid slice of KYKLOS-II<m,2,n>.

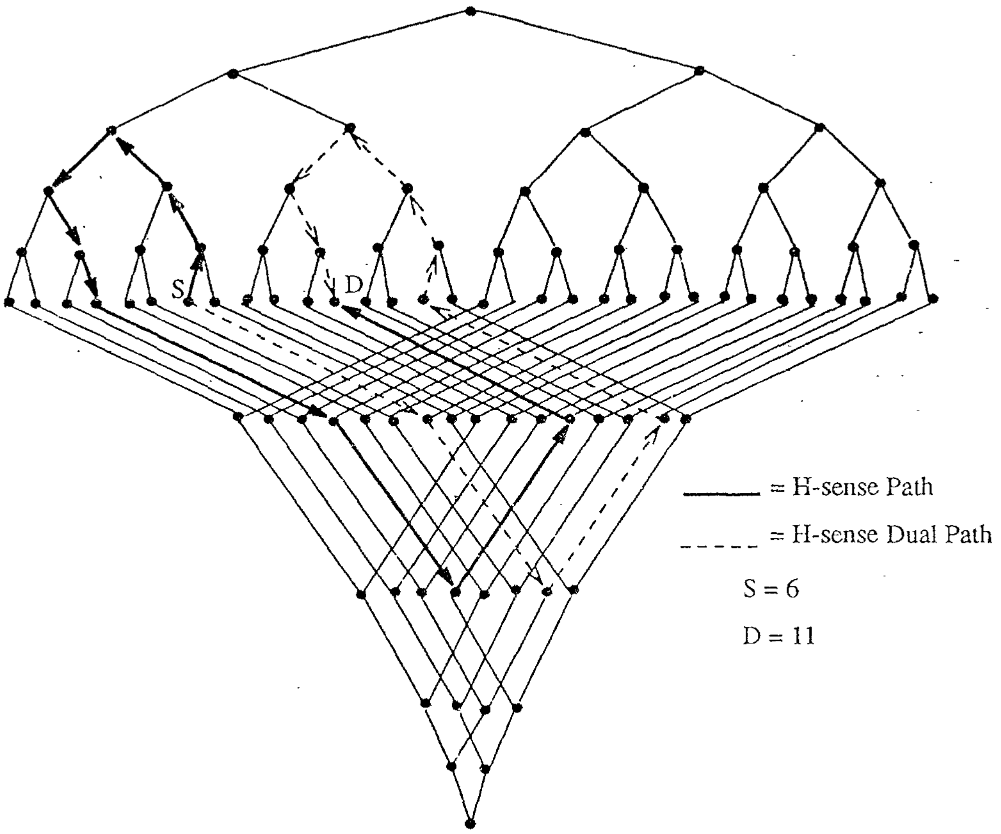


Figure 4-4: Example of H-II Routing in KYKLOS-II<2,2,5>

4.3.3 Traffic Analysis of H-II Routing in KYKLOS-II<m,2,n>

Definition 4.3 Let $T_{h2}(d,n)$ be the number of messages that traverse a level +d or -d link in either direction in a KYKLOS-II<m,2,n> using the H-II Routing Strategy under A1-A3 in a *round* of message exchange.

For a given source, S, define a set of processors R_{in}, $1 \leq i \leq \lceil n/2 \rceil$ as

$$R_{in} = \{D \mid X(S,D) \equiv u_1 0^{\lceil n/2 \rceil - i}\, 1\, u_2, \quad |u_1| = \lfloor n/2 \rfloor,\ |u_2| = i-1\} \tag{4.10}$$

From Step 1 of the H-II Routing Strategy, a message from S to D', D' $\in R_{in}$ would *passthrough* an intermediate processor A'[19] such that $X(S,A')=0^{n-i}1u_2$. From the TD Lemma, $d_{12}(S,A')=2i$ or equivalently two links[20] at each level j, $1 \leq j \leq i$, would be used for the message transfer. The total number of messages originating at S to every possible destination flowing through level i links in either direction is $2\sum_{k=i}^{\lceil n/2 \rceil} |R_{kn}|$. There are N possible source processors and a total of m^{n-i+1} links at level i. So the average traffic per level i link $T_{h2}(i,n)$ in both directions is

$$T_{h2}(i,n) = 2N\frac{\sum\limits_{k=i}^{\lceil n/2 \rceil} |R_{kn}|}{m^{n-i+1}}$$

$$= 2m^{i-1}\sum_{k=i}^{\lceil n/2 \rceil} |R_{kn}| \qquad (4.11)$$

In terms of the destination addresses (m-ary strings), substrings u_1, 1 and u_2 in (4.10) may be respectively defined as elements of the sets $S_1 = \{u| \, u \in \Sigma_m^{\lfloor n/2 \rfloor}\}$, $S_2 = \{u| \, u \in \Sigma_m^1, u \neq 0>\}$ and $S_3 = \{u| \, u \in \Sigma_m^{i-1}\}$ so that from Remark 3.2, $|R_{kn}| = (m-1)m^{\lfloor n/2 \rfloor + k-1}$. Computing $\sum_{k=i}^{\lceil n/2 \rceil} |R_{kn}|$ and substituting in (4.11) gives

$$T_{h2}(i,n) = 2m^{i-1}(m^n - m^{\lfloor n/2 \rfloor + i-1}), \qquad 1 \leq i \leq \lceil n/2 \rceil \qquad (4.12)$$

From symmetry considerations, we get

Theorem 4.4 The link traffic density in KYKLOS-II<m,2,n>, using the H-II Routing Strategy is given by

$$\boxed{\begin{aligned} T_{h2}(i,n) &= 2m^{i-1}(m^n - m^{\lfloor n/2 \rfloor + |i| - 1}), && -\lfloor n/2 \rfloor \leq i \leq \lceil n/2 \rceil, i \neq 0 \\ &= 0, && i > \lceil n/2 \rceil \text{ or } i < -\lfloor n/2 \rfloor. \end{aligned}}$$

Table 4-2 is the counterpart of Table 4-1 showing $T_{h2}(i,n)$ as a function of level

[19] except if $u_1 \equiv 0$ i.e. A'=D

[20] one for the rootward, the other for the leafward traversal of the message

number in KYKLOS-II<2,2,6>. Comparison of link utilization using H-II Routing with that in the single tree offers some interesting contrasts. Whereas the most heavily used links in the single tree are at level n, these links are totally unutilized as a result of H-II Routing in KYKLOS-II. The luxury of keeping the links between levels n and $\lceil n/2 \rceil$ unutilized is, however, a small price to pay. This is so since the other levels, where most of the network resources (nodes and links) are concentrated, are considerably better (more uniformly) utilized whereas those same levels were severely underutilized in the case of the single tree. For example, the traffic through level 1 links is about 50% of the maximum link traffic density using H-II Routing. By comparison, level 1 links in a single tree carry only about 6% of the traffic carried by the maximally congested links. Finally, the maximum traffic density using H-II Routing is only about 12% that in the single tree, a significant improvement. We now proceed to estimate the maximum link traffic density as a function of N.

Level #	B-Tree	Util	H-II	Util
6	2048	100.00	0	0
5	1536	75.00	0	0
4	896	43.75	0	0
3	480	23.44	256	100
2	248	12.11	192	75.00
1	126	6.15	112	43.75

B-Tree = Single Binary Tree

Util = Link Utilization

Table 4-2: Link Traffic Density as function of Level Number in KYKLOS-II<2,2,6> using H-II Routing and in a single tree (height 6)

Corollary 4.4.1 The maximum link traffic density in KYKLOS-II<m,2,n>, n≥3, using the H-II Routing Strategy is

- $2(m-1)N^{1.5}/m^2$ and occurs in level ±n/2 links when n is even and
- $2(m-1)N^{1.5}/m^{3/2}$ and occurs in level (n+1)/2 links when n is odd.

Proof: In Appendix C (Result C.6), it is shown that in KYKLOS-II<m,2,n>, $n \geq 3$,

$T_{h2}(i,n) > T_{h2}(i-1,n), \qquad 1 < i \leq \lceil n/2 \rceil$

By symmetry,

$T_{h2}(i,n) > T_{h2}(i+1,n), \qquad -\lfloor n/2 \rfloor \leq i < -1,\ n \geq 4$

Using this fact and Theorem 4.4 leads to the result. □

4.4 P-sense Traffic Characteristics

4.4.1 Modified P-II Routing

4.4.1.1. Motivation

Consider routing a message from S=000010 to D=101000 in KYKLOS-II<2,2,6>. Here X(S,D)=101010, t=0 and b=p=1.

From Corollary 3.5.1, the length of the shortest path between S and D is 2(n-max{t,b,p}) = 10. There are five distinct shortest paths enumerated below (see Fig. 4-5).

S --(bottom tree)--->	D	
S --(top tree)------>	001000 ---(bottom tree)-->	D
S --(bottom tree)--->	100010 ---(top tree)----->	D
S --(top tree)------>	000000 ---(bottom tree)-->	D
S --(bottom tree)--->	101010 ---(top tree)----->	D

The P-II Routing Strategy of subsection 3.2.3 would prescribe the path in (1) above. The spirit of this strategy may be cast as follows:

"If there are multiple shortest paths between a given source-destination pair, use one that does not involve passthrough, if such a one exists." This would, in some measure, obviate the need for an intermediate processor to handle routing, leaving that task to the switches *wherever possible*.

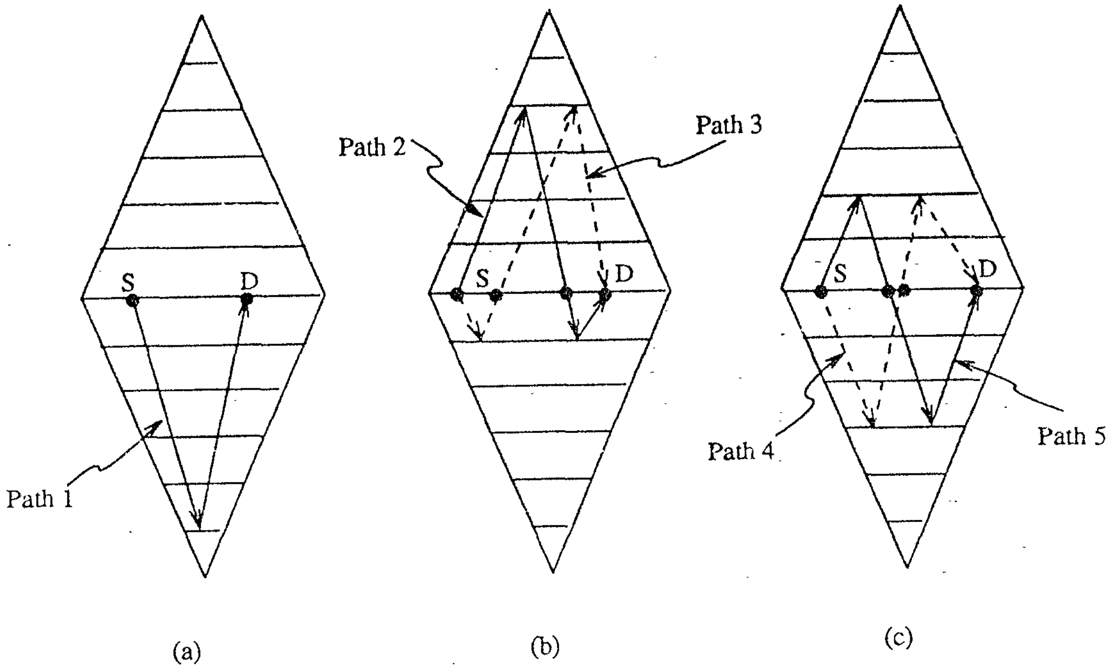


Figure 4-5: All shortest paths between S=000010 and D=101010

Paths 2 and 3 are dual paths; they involve levels 1-4 of the top tree and only level 1 of the bottom tree. Similarly, 4 and 5 are dual paths; they involve levels 1 and 2 of the top tree and levels 1-3 of the bottom tree. Note that the links involved in paths 4 and 5 are closest to the leaves. Experience with the traffic characteristics of a single tree, and with the M-II and H-II Routing Strategies in KYKLOS-II suggests that link saturation affects links near the roots first[21]. This is to be expected because the number of links decrease geometrically as one proceeds from the leaves to the root. If link traffic were a consideration, we should use links closest to the leaves, should such an option exist. For the example above, this *modified P-II Routing* would prescribe the use of paths 4 or 5.

[21] In the case of H-II Routing, links at the slice extremities are maximally congested

In general, the modified P-II Routing Strategy aims at selecting a shortest path which uses link levels closest to the leaf nodes.

4.4.1.2. Modified P-II Routing Algorithm

For a given X(S,D), let $k=\max\{t,b,p\}$ i.e. $0^k < X(S,D) \wedge 0^{k+j} \nless X(S,D), j>0$

Let $X(S,D)=u_1 0^k u_2$ s.t. $\forall v_1,v_2 (X(S,D)=v_1 0^k v_2 \Rightarrow ||u_1|-|u_2|| \leq ||v_1|-|v_2||)$.

Then,

if $(|u_1|>0 \wedge |u_2|>0)$,

use the strategy outlined in Section 3.2.1 with passthrough occurring at A where $X(S,A)=u_1 0^{k+|u_2|}$,

else use M-II Routing *a la* Section 3.1.2.

Note that $|u_2|$ and $|u_1|$ correspond respectively to the excursion into the top tree and bottom tree.

Example 4.2 Returning to the above example, Table 4-3 shows the different candidates for u_1 and u_2 satisfying $X(S,D)=u_1 0 u_2$. Of these, only the candidate in the third row of the table minimizes the value of $||u_1|-|u_2||$ and hence the modified Routing Strategy favors paths 4 and 5 with a view to minimizing the maximum traffic density.

Path #	u1	u2	||u1|-|u2||
1	10101	-	5
2,3	1	1010	3
4,5	101	10	1

Table 4-3: u_1 and u_2 for alternate shortest paths

4.4.1.3. Multiple Shortest Paths

As before, any source-destination pair which has a shortest path employing passthrough will have a dual shortest path. However, in some cases it may be possible that there exist v_1, v_2, s.t.

$X(S,D) = v_1 0^k v_2 = u_1 0^k u_2$ and $||v_1|-|v_2||=||u_1|-|u_2||$, $u_1 \neq v_1$

For example, if X(S,D)=101101,

u_1=1, u_2=1101

v_1=1011, v_2=1 so that $||u_1|-|u_2|| = ||v_1|-|v_2||=3$ These two possibilities leading to four distinct shortest paths are sketched in Figs. 4-6(a) and (b). Observe that the first alternative (u_1=1, u_2=1101) uses four levels of the top tree and one level of the bottom tree. The second alternative uses one level of the top tree and four levels of the bottom tree.

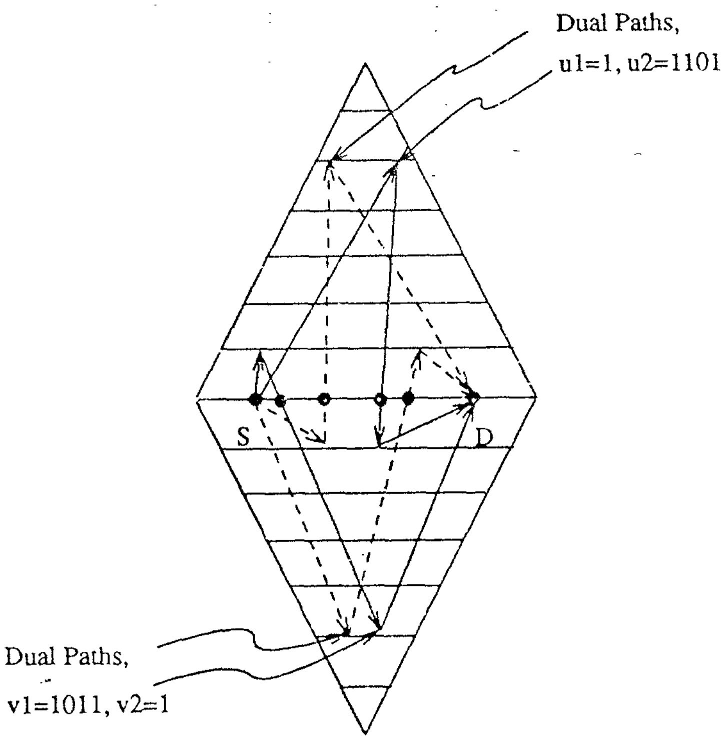


Figure 4-6: Example of multiple traffic-minimizing, shortest paths

A case that deserves special attention is when X(S,D)=1^n. In keeping with the spirit of the modified P-II Routing Strategy, an intermediate processor, A, could be used where

X(S,A) = $0^{\lceil n/2\rceil}1^{\lfloor n/2\rfloor}$ or $0^{\lfloor n/2\rfloor}1^{\lceil n/2\rceil}$

or $1^{\lceil n/2\rceil}0^{\lfloor n/2\rfloor}$ or $1^{\lfloor n/2\rfloor}0^{\lceil n/2\rceil}$

Of course these reduce to just two possibilities for even n.

As in the case for the other routing strategies, we define link traffic density for the modified P-II Routing Strategy.

Definition 4.4 Let $T_{p2}(d,n)$ be the number of messages that traverse a level +d or -d link in either direction in a KYKLOS-II<m,2,n> using the modified P-II Routing Strategy under A1-A3 in a *round* of message exchange.

4.4.2 Comparison of Link Traffic Characteristics

A program to implement the modified P-II Routing Algorithm was written. Both network resource utilization and maximum traffic density for varying network sizes were obtained. These are discussed next.

4.4.2.1. Link Utilization

Shown in Table 4-4 is the traffic density as a function of link level in KYKLOS-II<2,2,6> (p-sense) and the binary tree. Note that the traffic in the P-II case exhibits the best distribution over the different levels compared with the distribution in the single tree (or KYKLOS-I) or that in KYKLOS-II using the other routing strategies (see Tables 4-1 and 4-2 for comparison). For example, the plentiful links closer to the leaves are considerably better utilized with P-II routing in KYKLOS-II than they are in the single tree while the upper level links (toward the root) have better utilization than the corresponding links in the H-II case.

4.4.2.2. Maximum Traffic Density

Table 4-5 shows the maximum link traffic density as a function of n using each of the three routing strategies for KYKLOS-II<2,2,n> and the level where this occurs. Included for comparison is the maximum traffic density for KYKLOS-I (K-I). Note that the performance of P-II surpasses all other cases up to N=2048. Also, the improvement over KYKLOS-I in both, P-II as well as H-II increases with n.

Level #	B-Tree	Util	P-II	Util
6	2048	100.00	0	0.00
5	1536	75.00	48	24.49
4	896	43.75	152	77.55
3	480	23.44	196	100.00
2	248	12.11	154	78.57
1	126	6.15	98	50.00

B-Tree = Single Binary Tree

Util = Link Utilization

Table 4-4: Link Traffic Density as function of Level Number in KYKLOS-II<2,2,6> using P-II Routing and in a single tree (height 6)

Table 4-5 shows that the maximally congested links are descendants of the root in KYKLOS-I, two levels below the root in M-II and at the mid-slice extremities in H-II. Also, the maximally congested links in P-II are two link levels below the root up to N=16. Between N=32 and N=2048, the maximum congestion occurs at three link levels below the root. In fact, beyond N=8192, the maximally congested links descend yet another level. Of greater significance is the fact that beyond N=2048, H-II outperforms P-II. Could these observations be rationalized?

To gain a better understanding of the variation of maximum traffic density with N, T_{max} has been plotted vs. N on a log-log scale for each routing strategy (Fig. 4-7). Note that each curve is or approximates a straight line (or is piecewise linear). This is so, since T_{max} is $O(N^2)$ for K-I and M-II differing only by a constant of proportionality. The average slope for the H-II plot is less, reflecting the $O(N^{1.5})$ variation of T_{max} with N. (Note that the slope is proportional to the index of N). The P-II plot seems to have a slope that increases very gradually . Could it be that in this case $T_{max} = O(N^{f(n)})$ where f(n) is a slowly increasing function of n? A partial answer to the above question is attempted next.

n	K-I	LL#	M-II	LL#	H-II	LL#	P-II	LL#
3	16	3	10	2	16	2	9	1
4	64	4	36	3	32	2	26	2
5	256	5	144	4	128	3	66	2
6	1024	6	576	5	256	3	196	3
7	4096	7	2304	6	1024	4	568	4
8	16,384	8	9216	7	2048	4	1616	5
9	65,536	9	36,864	8	8192	5	4960	6
10	262,144	10	147,456	9	16,384	5	15,808	7
11	1,048,576	11	589,824	10	65,536	6	51,840	8
12	4,194,304	12	2,359,296	11	131,072	6	173,568	8

LL# = Link Level Number

K-I = KYKLOS-I<2,2,n>

M-II = KYKLOS-II<2,2,n> using M-II Routing

H-II = KYKLOS-II<2,2.n> using H-II Routing

P-II = KYKLOS-II<2,2.n> using P-II Routing

Table 4-5: Max. Traffic Density as function of Topology × Routing Strategy

4.4.3 Lower Asymptotic Bounds on Maximum P-sense Traffic Density

For a given source processor, S, consider the set of processors in KYKLOS-II<m,2,n> defined by

$$Z = \{D \mid X(S,D) = 0^2 1u \wedge |u| = n\text{-}3 \wedge 0^{1+j} \not\prec u, j>0\} \qquad (4.13)$$

From the P-II Routing Strategy (Section 3.2.3), a message from S to D, D $\in$ Z must traverse a level n-2 link. For N possible source processors, the total number of messages through level n-2 links is *at least* N|Z|. Since there are m^3 level n-2 links, the traffic at level n-2 is bounded as follows

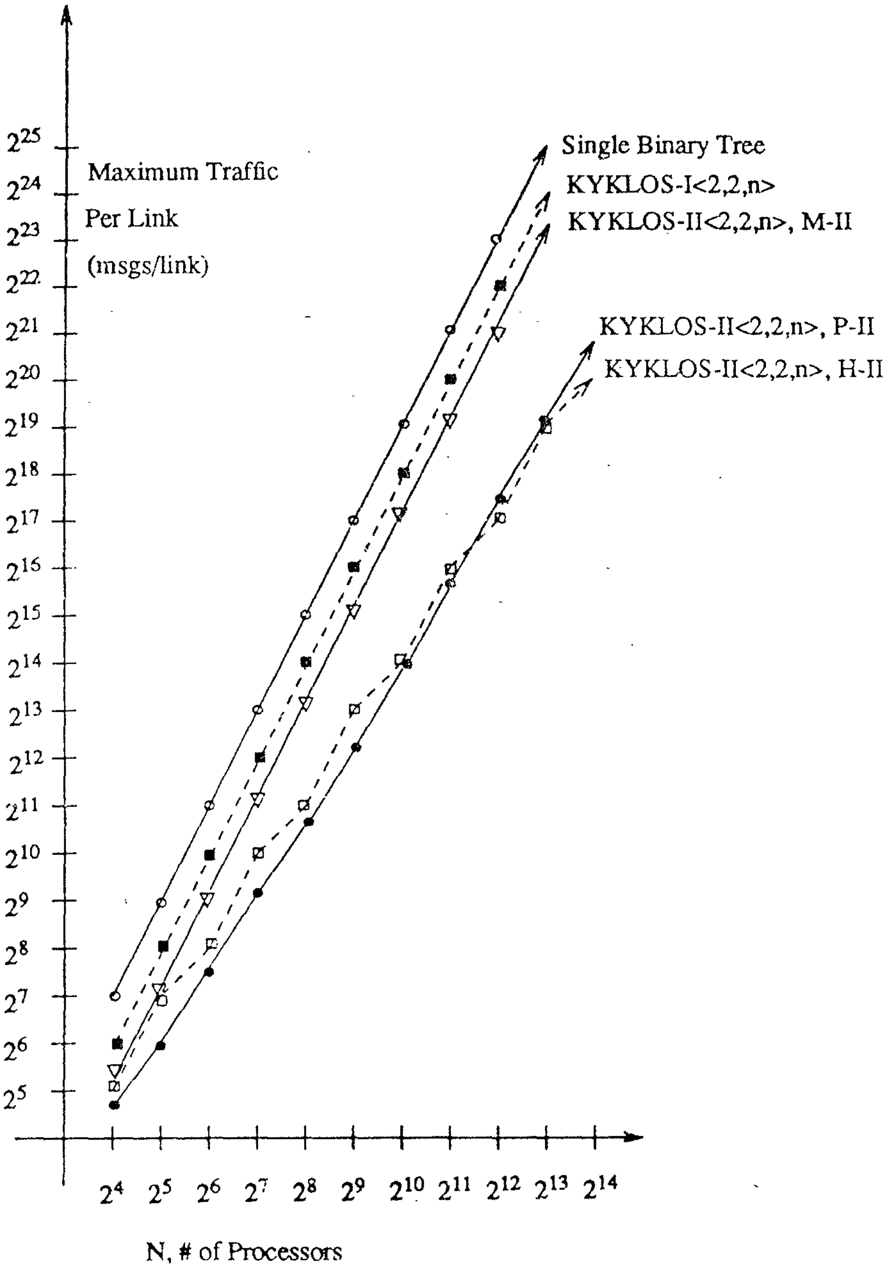


Figure 4-7: Plot of Maximum Traffic Density vs. N

$$T_{p2}(n-2,n) \; > \; \frac{2N|Z|}{m^3} \tag{4.14}$$

To obtain the cardinality of the set Z defined in (4.13), note that the substrings 1 and u in the definition of Z in (4.13) may be respectively defined in terms of the destination address as elements of the sets $\{u \mid u \in \Sigma_m, u \neq 0\}$ and $\{u \mid u \in \Sigma_m^{n-3}, 0^{1+j} \not\prec u, j>0\}$. From Definition 3.8 and Remark 3.2, $|Z| = (m-1)p_{p2}'(n-4,n-3)$ which when substituted into (4.14) yields

$$T_{p2}(n-2,n) \; > \; \frac{2(m-1)Np_{p2}'(n-4,n-3)}{m^3} \tag{4.15}$$

From Appendix B (Equation B.26),

$$p_{p2}'(n-4,n-3) = c_1 r_{1,2,m}^{n} + c_2 r_{2,2,m}^{n}, \qquad c_1,c_2 \equiv \text{constant}. \tag{4.16}$$

Since $|r_{1,2,2}|<1$, $p_{p2}'(n-4,n-3)$ may be approximated to $c_2 r_{2,2,2}^{n}$ for large n. On substitution into (4.15),

$$T_{p2}(n-2,n) > kN^{1+\log_m r_{2,2,m}}, \qquad k \equiv \text{constant}$$

In the binary KYKLOS, the P-II maximum traffic density is thus bounded by $\Omega(N^{1+\log_2 r_{2,2,2}}) = \Omega(N^{\log(1+\sqrt{5})}) \sim \Omega(N^{1.69})$. An analysis similar to the above could be performed to show that the traffic density bounds at lower levels in the tree would involve the real root of higher degreed characteristic polynomials albeit with smaller constants. Since the roots increase with degree, the constants in the expression would dominate for small N and the function of N would dominate for large N. That is why the maximally congested links descend slowly from the root. Interestingly enough, this also explains why H-II (with $O(N^{1.5})$ maximum traffic density) outperforms P-II eventually.

Chapter 5

Fault Tolerance

One of the primary motivations for KYKLOS was the need to provide fault tolerance to the simple binary tree. In Appendix D, the fault tolerance of a multiple-tree interconnection network is investigated. The reliability of this structure with respect to preserving at least one of r trees in an r-replica network is studied. It is shown that, with respect to preservation of at least one tree, the Failure Probability, Mission Time (MT), and Mean Time to Failure (MTTF) are significantly better than in a single tree.

In this chapter, a comparison between KYKLOS-I and KYKLOS-II is made in respect to processor connectivity and performance degradation under switch node faults. For this purpose, specific examples showing the effect of network switch failure(s) on performance are examined. A thorough investigation of the effect of faults on network performance would itself constitute a separate dissertation proposal. No attempt is made to be exhaustive; instead special cases of the effect of switch faults on connectivity of processors and on degradation in communication latencies are explored.

5.1 Comparison of KYKLOS-I/KYKLOS-II with respect to Connectivity Preservation under Switch Faults

For a k-node failure, $k \geq r$, the probability that at least one of r complete trees survives in a KYKLOS-x<m,r,n> is a function of m, n and r but is independent of x, the version number. While maintenance of a tree structure is a reasonable requirement [HAYE76], there are applications that require, at the very least, that the processors be connected. Note that the former implies processor connectivity, though the converse is not necessarily true. Fig. 5-1 shows a case of four failed switches in KYKLOS-II<2,2,4>. Note that with the loss of the faulty switches, neither one of the two full binary trees has survived. However, the processors are connected and a multicomputer system based on the network topology of Fig. 5-1 may still function, albeit with degraded performance.

5.1.1 Conceptual View of Multiple Switch Node Failures in KYKLOS

In a general r-replica KYKLOS, each tree may be examined separately for switch faults. Switch faults in each of the r trees may cause the set of processors to be partitioned into two or more disjoint sets. Faults in the i^{th} tree, $1 \leq i \leq r$, will define a partitioning Π_i on the set of processors. For example, the fault in the upper tree of KYKLOS-I of Fig. 5-2(a) would cause the partitioning

$\Pi_1 = \{0,1; 2,3; 4,5,6,7\}$.

The fragmentation of processors due to the two faults in the bottom tree of KYKLOS-I results in the partitioning

$\Pi_2 = \{0; 1; 2,3,6,7; 4; 5\}$.

Now consider processors 2 and 6. They are in different blocks of Π_1. However, they are in the same block of Π_2 and are hence connected.

The corresponding partition for the bottom tree of KYKLOS-II is shown in the redrawn version of its bottom tree in Fig. 5-2(b) as:

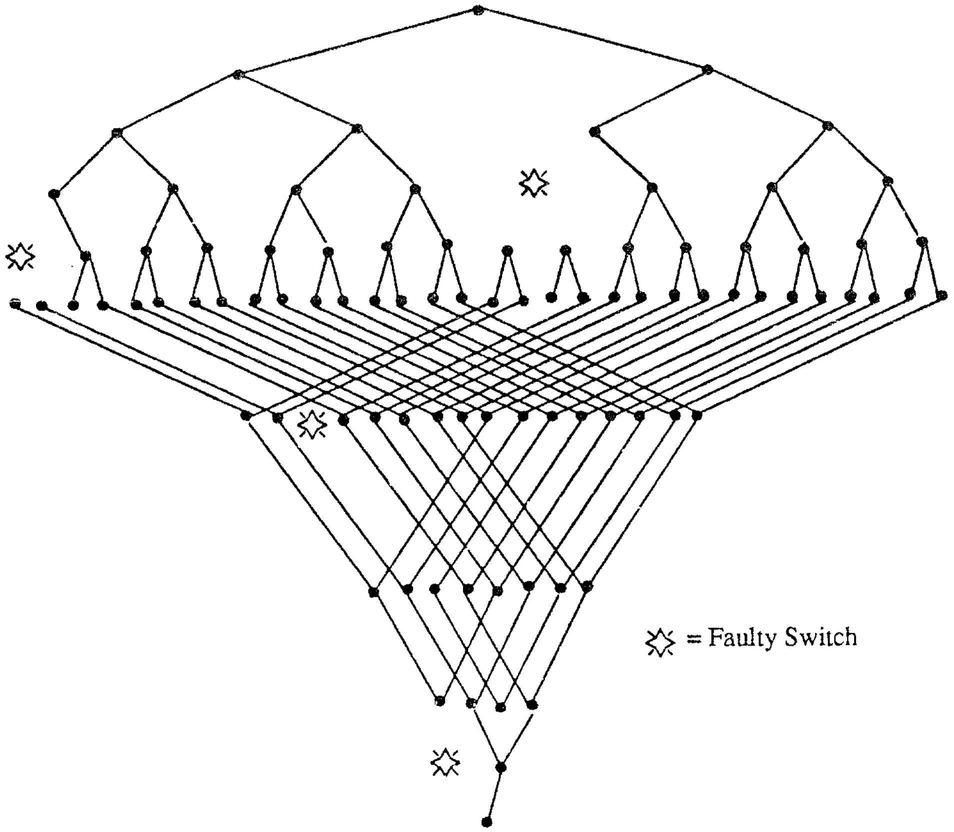


Figure 5-1: Effect of Switch Faults on Tree Preservation and Connectivity $\Pi_2 = \{0; 4; 2; 6; 1,5,3,7\}$.

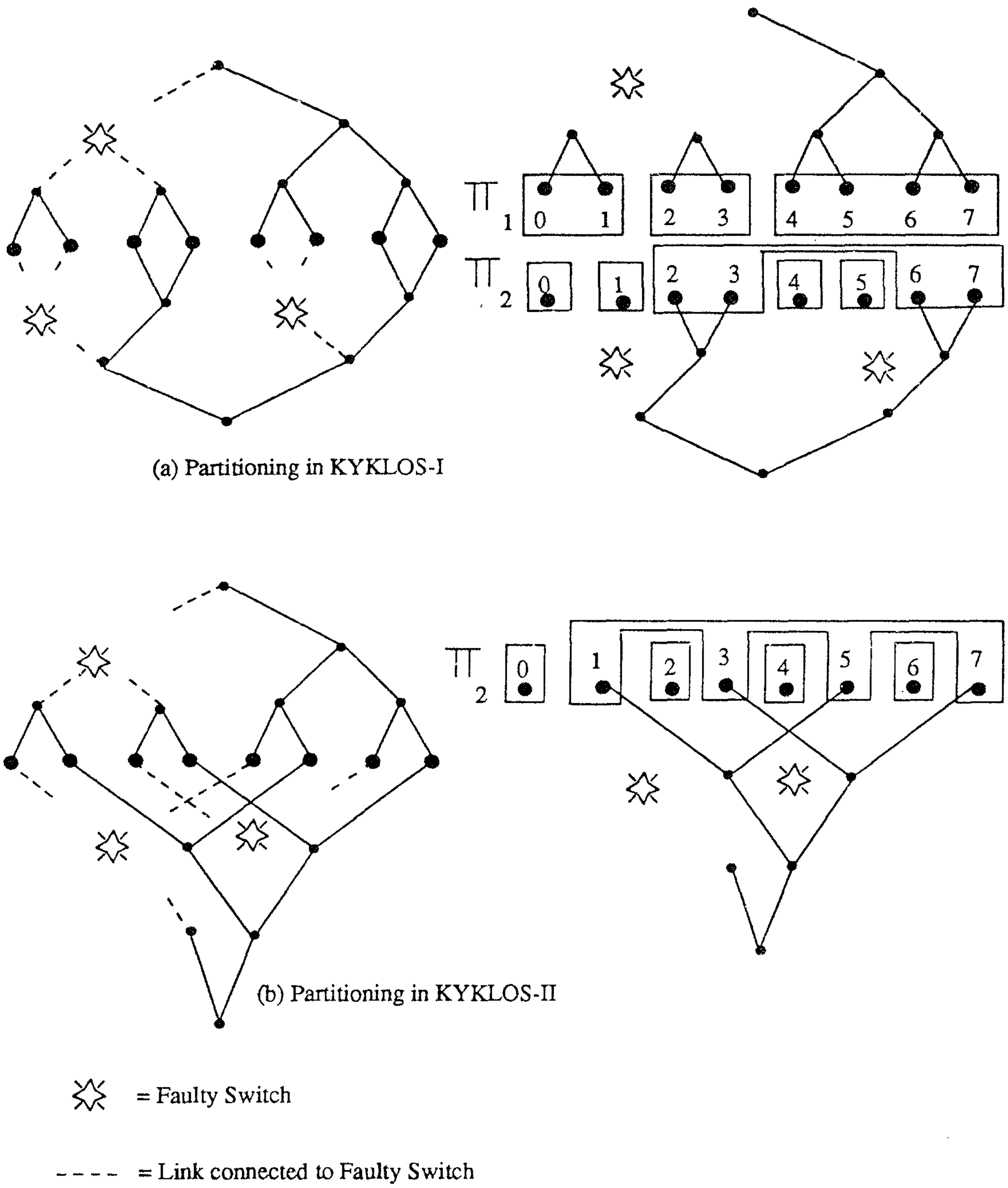
Now consider processors 0 and 2. In both, KYKLOS-I and KYKLOS-II, they are in different blocks of both, Π_1 and Π_2. Further, it may be verified that they are disconnected in KYKLOS-I. However, there exists a path between them in KYKLOS-II. More specifically, in the latter case,

0 and 1 are in the same block of Π_1

1 and 3 are in the same block of Π_2

3 and 2 are in the same block of Π_1.

Hence 0 and 2 are connected in the faulty KYKLOS-II network.


(a) Partitioning in KYKLOS-I

(b) Partitioning in KYKLOS-II

Figure 5-2: Network Partitioning in the Presence of Faults

In general, the problem of finding whether two processors, S and D, are connected in the presence of faults is equivalent to finding a string of processors,

$S, a_0, a_1, \ldots, a_k, D$

such that any two adjacent processors in the above string are in the same block of at least one partition. Also, finding whether all processors are connected is equivalent to performing the sum

$\Pi_0 = \sum_{i=1}^{r} \Pi_i.$

Now, two processors are connected iff they are in the same block of Π_0 so that $|\Pi_0|$ gives the number of components into which the processors are split. In particular, the set of processors are in a single connected component iff $|\Pi_0|=1$.

Processors in a two-replica KYKLOS cannot be disconnected by a single switch node failure. Hence, we focus attention on the effect of a double switch node failure as the next most probable cause of disconnecting the processors in KYKLOS.

5.1.2 Effect of Double Switch Node Failures on KYKLOS

The following theorem establishes, using induction, the number of double switch node failures, each of which will cause the processors to be disconnected into two or more components in KYKLOS-I<2,2,n>.

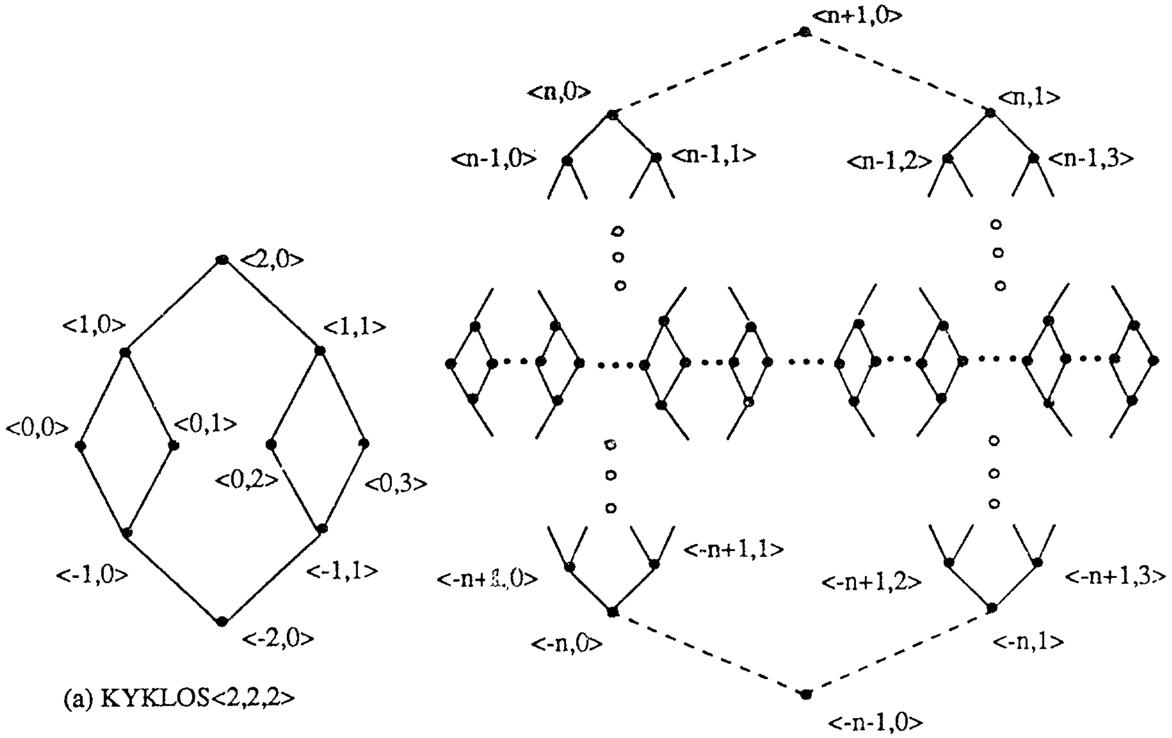
Theorem 5.1: The total number of pairs of switches, F_n, in a KYKLOS-I<2,2,n>, the failure of any one of which, causes the processors to be disconnected into two or more sets is

$F_n = 3(2^n - 1).$

Proof: We induct on n, the height of the tree. We consider the base case, n=2 (Fig. 5-3(a)). Fig. 5-4 lists a set of switch node pairs ; failure of any one (or more) of these pairs causes the processors to be disconnected. The cardinality of the above set is 9 which corroborates the expression in the theorem statement for n = 2, N = 4.

Hypothesize that the theorem holds in KYKLOS-I<2,2,n> i.e. there are a total of $F_n = 3(2^n-1)$ double switch node failures, each of which causes the processors to be disconnected.

We construct the next higher-order KYKLOS as shown in Fig. 5-3(b). (Dashed lines represent links added to obtain KYKLOS-I<2,2,n+1>).



(a) KYKLOS<2,2,2>

(b) Obtaining KYKLOS-II<2,2,n+1>

Figure 5-3: KYKLOS-I<2,2,2> and KYKLOS-I<2,2,n+1>

To see the effect of two node failures on KYKLOS-I<2,2,n+1>, consider the set of node pairs that caused processors to be disconnected in KYKLOS-I<2,2,n>. These node pairs except for <n-1,1>, <-n+1,0> and <n-1,0>, <-n+1,1> will also cause the processors in KYKLOS-I<2,2,n+1> to be disconnected. Further, KYKLOS-I<2,2,n+1> is composed of two KYKLOS-I<2,2,n>'s appropriately connected. Hence

<1,0> <-1,0>,
<1,1> <-1,1>,
<1,0> <-1,1>,
<1,1> <-1,0>,
<1,0> <-2,0>,
<1,1> <-2,0>,
<2,0> <-1,0>,
<2,0> <-1,1>,
<2,0> <-2,0>

Figure 5-4: Set of switch node pairs that disconnect processors in KYKLOS-II<2,2,2>

there are at least $2(F_n$-$2)$ pairs of switch nodes whose failure will disconnect the processors. *In addition*, any one of the pairs of switches listed in Fig. 5-5 will also disconnect the processors.

<n+1,0> <-n-1,0>
<n+1,0> <-n,0>
<n+1,0> <-n,1>
<n,0> <-n-1,0>
<n,1> <-n-1,0>
<n,0> <-n,1>
<n,1> <-n,0>

Figure 5-5: Subset of switch node pairs that disconnect processors in KYKLOS-I<2,2,n+1>

Hence the total number of switch node pairs that disconnect the processors in KYKLOS-I<2,2,n+1> is given by:

$$F_{n+1} = 2(F_n - 2) + 7$$

$= 2[3(2^n - 1) - 2] + 7$ (from the induction hypothesis)
$= 3(2^{n+1}-1).$ □

In KYKLOS-II<2,2,n>, the only pairs of switches that cause processors to be disconnected are the level ± 1 switch nodes adjacent to the same processor. Since there are N such switch pairs, we state:

Theorem 5.2: The total number of pairs of switches in KYKLOS-II<2,2,n> whose failure causes the processors to be disconnected is N, the number of processors. □

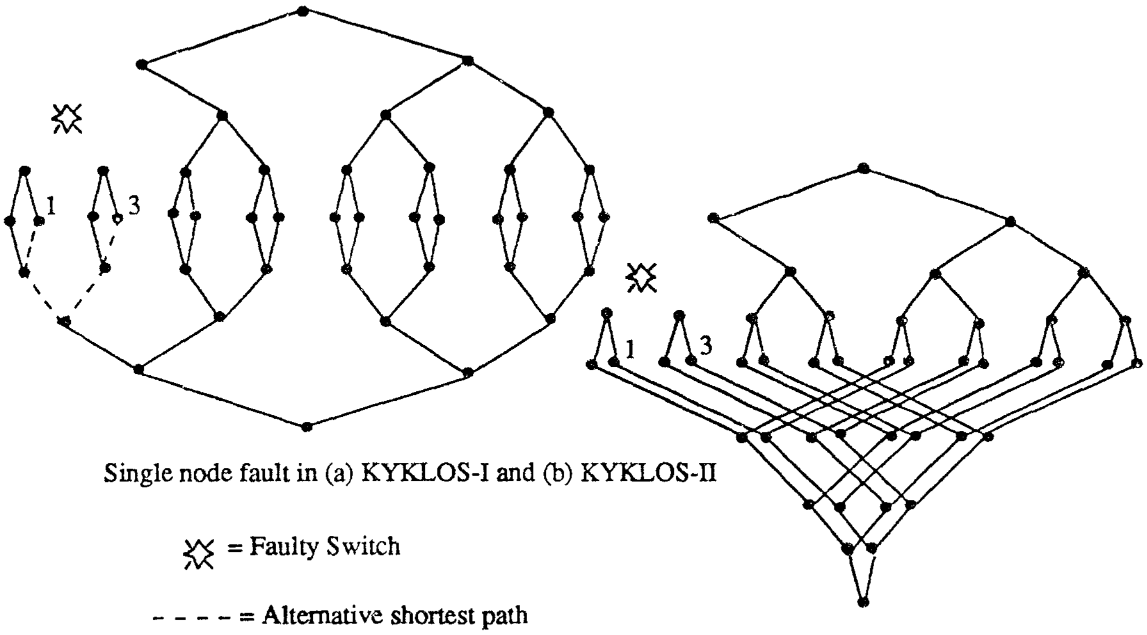
Obviously there are $3(N-1)/N \sim 3$ times as many double switch node failures that cause processors in KYKLOS-I<2,2,n> to be disconnected as compared with KYKLOS-II<2,2,n>. Thus KYKLOS-II<2,2,n> is three times less likely to be disconnected compared with KYKLOS-I<2,2,n> as a result of a double switch node failure. Further, this is obtained at no additional cost in hardware, only by rearranging the links in the bottom tree.

Finally, the probability of disconnecting the processors conditional on two switch node failures may be easily obtained by considering the ratio of double switch node failures that leave the processors disconnected to the total number of double switch node failures. For KYKLOS-II<2,2,n>, this ratio is $N/{}^{(2N-2)}C_2 = N/[(N-1)(2N-3)] \sim 1/2N$ which decreases with network size. Of course, the corresponding ratio for KYKLOS-I<2,2,n> is three times as high.

5.2 Degradation under single node failures

Consider a single node failure of switch <2,0> in KYKLOS-I<2,2,4> as shown in Fig. 5-6(a). The effect of this switch fault on the top tree alone is to split the set of processors into three sets represented by the partitioning
{0,1; 2,3; 4,5,6,7,8,9,10,11,12,13,14,15}.

Clearly, a shortest path between processors 1 and 3 will involve switch <2,0>. However, because the bottom tree is a mirror image of the top tree, any processor may communicate with any other processor *without any degradation in path length* by using the bottom tree. For example, the distance between processors 1 and 3 is 4 link traversals. With the switch node fault shown in Fig. 5-6(a), the shortest path between 1 and 3 in the upper tree is disrupted. However, there is an alternate shortest path through the bottom tree shown in dashed lines. Since this holds for each pair of processors in KYKLOS-I, we conclude that there is no degradation in average distance in this topology under single switch node failures.



Single node fault in (a) KYKLOS-I and (b) KYKLOS-II

Figure 5-6: Effect of single switch fault on degradation in pathlength.

On the other hand, the shortest path between processors 1 and 3 in KYKLOS-II<2,2,4> using the modified P-II Routing Strategy of Section 4.3.1 must pass through the faulty switch, <2,0> (see Fig. 5-6(b)). This is because the shortest path in this case is unique as will be explained in greater detail later. Hence, any alternate route between this pair employed to bypass the faulty switch will involve a greater path length and concomitant degradation in performance. It should be noted that this is not always true in KYKLOS-II i.e. there are processor pairs which have multiple shortest paths which may, in addition, be node-disjoint.

Our goal in the present section is to quantify the above observations in KYKLOS-II<2,2,n>. More specifically, we attempt to answer the questions:

- "How many processor pairs could *potentially* suffer increased path length due to a single switch node fault?"
- "To what extent is the average interprocessor distance increased due to a single switch node fault?"

The answer to the first question involves identifying those source-destination pairs connected by a unique shortest path. For this purpose, we define the following classification. By so doing, each possible destination, for a given source S, is assigned to one of four categories. The classification is based on the relative values of t, b, and p in X(S,D).

1. *Category 1:* {D | $p \geq t$, $p \geq b$ in X(S,D)}.
2. *Category 2:* {D | $t>p$, $b=t$ in X(S,D)}.
3. *Category 3:* {D | $t>p$, $t>b$ in X(S,D)}.
4. *Category 4:* {D | $b>p$, $b>t$ in X(S,D)}.

From the modified P-II Routing Strategy of Section 4.4.1, it follows that there is a shortest path that uses passthrough between S and a processor in Category 1. Because each passthrough path also has its corresponding dual, there are at least two shortest paths between S and any processor in Category 1. Also, there are two shortest paths between S and a processor in Category 2, one through the upper tree and the other through the lower tree. By contrast, destinations represented by categories 3 and 4 have unique shortest paths (USP) from S. We next take a closer look at the characteristics of multiple shortest paths in KYKLOS-II<m,2,n>.

5.2.1 Disjointness of multiple shortest paths in KYKLOS-II

We first prove an important result about dual shortest paths using paththrough.

Theorem 5.3 Let PP be a shortest path between S and D using passthrough. Let PP' be the dual of PP. Then PP and PP' are disjoint[22].

Proof Without loss of generality assume that there exists a node, x, in common between the shortest path, PP, and its dual, PP' in the bottom tree i.e. $x \in$ PP and $x \in$ PP' (see Fig 5-7). Also, let 2k be the length of the shortest path contained in the bottom tree[23]. The subpaths of PP and PP' in the bottom tree may be thought of as belonging to subtrees ST and ST', each of height k in the bottom tree. So $x \in$ ST and $x \in$ ST'. Now two *full* binary subtrees between the same levels of the parent tree are either identical or disjoint. Because ST and ST' have a common node x, they must be identical. Hence, S and D are in the same subtree of height k i.e. the distance between S and D is 2k or less. This contradicts the assumption that PP is the shortest path

[22] The implication here is to node disjointness which implies link disjointness.

[23] In Section 3.2, it was shown that a given passthrough path and its dual traverse the same number of links in the bottom tree, in this case 2k.

between S and D. Hence PP and PP' cannot have a node in common (except for S and D). □

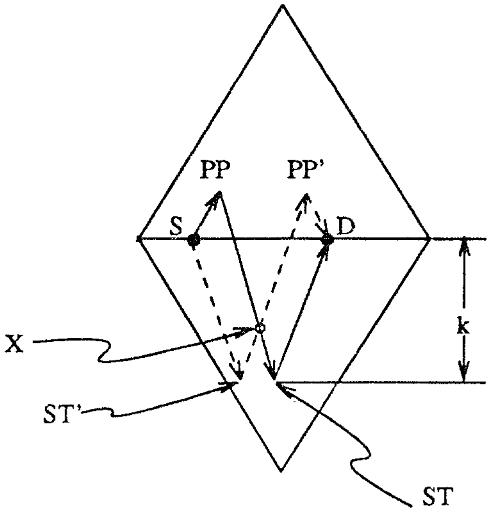


Figure 5-7: Node Disjointness of Dual Passthrough Paths

We conclude that a single switch node failure cannot affect the length of the shortest path between S and D if there exists a shortest path between them involving passthrough.

The complement of this case is that in which *no shortest path* between a given source-destination pair involves passthrough i.e. the shortest path is, in fact, the m-sense path for that pair. (See categories 2, 3 and 4). Of these, only Category 2 destinations have two shortest paths, one through the top tree, the other through the bottom tree. Here again, the two shortest paths are disjoint.

We could summarize the above discussion as follows

Theorem 5.4 The shortest path between a source, S, and destination, D, is unique iff $D \in \{D' \mid (t>p \wedge t>b) \vee (b>p \wedge b>t) \text{ in } X(S,D')\}$.

If not, there exist at least two shortest paths between S and D that are disjoint. □

Theorem 5.4 may be used to estimate the number of destinations from a given source that have a unique shortest path (USP). Table 5-1 shows the number of USP destinations together with the percentage of destinations that are USP. Note that for a given source, S, a respectable percentage of destinations have unique shortest paths. Also, the percentage drops with increasing N. This means that, as network size grows, a smaller fraction of the total number of destinations are potentially susceptible to a single node fault. By way of comparison, a destination in KYKLOS-I is never susceptible to a single node fault.

N	# of USP D's	% of USP D's
4	2	50
8	4	50
16	8	50
32	14	43.75
64	26	40.62
128	46	35.94
256	84	32.81
512	152	29.69
1024	278	27.15

# of USP D's = Number of Unique Shortest Path Destinations (from S)

% of USP D's = Percentage of Unique Shortest Path Destinations (from S)

Table 5-1: Number of destinations with USP from a given source

We proceed to estimate the actual degradation in the presence of faults in KYKLOS-II<2,2,n>.

5.2.2 Effect of Single Node Faults on Average Distance

Consider a source-destination pair, (S,D), as shown in Fig. 5-8. Let $d_{p2}(S,D) = 2d$. Given the presence of a single switch node fault, the conditional probability of it lying on the shortest path (the fault hit ratio) is

$$\frac{2d-1}{2(N-1)}$$

where $2(N-1)$ *is the number of switch nodes in* *KYKLOS–II*$<2,2,n>$

We have seen that, if there is a USP between S and D, then that path must be wholly in one tree. To simplify the calculation of degradation, we assume that, in the event of a fault lying on the USP of an S-D pair, that pair would use a path through the root of the other tree as illustrated in Fig. 5-8[24]. Using this alternate path, the increase of distance that this message would have to traverse = 2n-2d.

Hence the increase in pathlength that a message to a USP destination would have to traverse *on the average* is

$$\frac{2d-1}{2(N-1)}(2n-2d)$$

For a given source, S, the set of all possible destinations, $J_{d,n}$, that have a unique shortest path of length 2d from S is given by Theorem 5.4:

$$J_{d,n} = \{D \mid d_{p2}(S,D)=2d \wedge ((t>p \wedge t>b) \vee (b>p \wedge b>t) \text{ in } X(S,D)\}$$

[24] Using this assumption, we obtain an upper bound on degradation of average interprocessor distance.

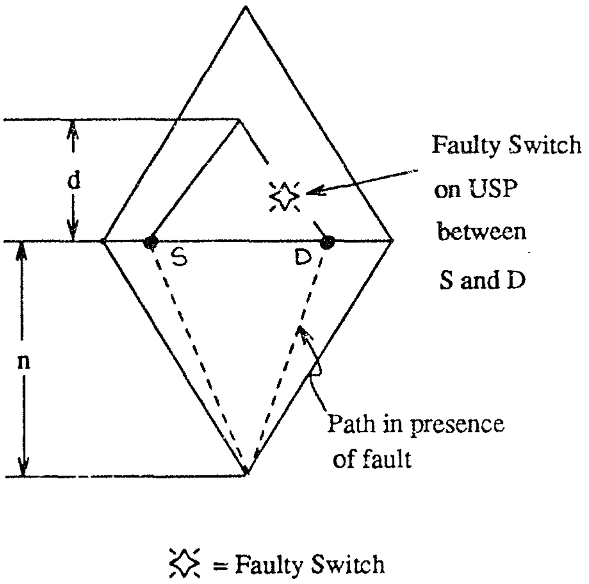

d
n
S
D
Faulty Switch on USP between S and D
Path in presence of fault
= Faulty Switch

Figure 5-8: Alternate path in presence of single node failure

Using the definition of t, b and p (Sections 3.1 and 3.2),

$$
\begin{aligned}
J_{d,n} = \{D| \quad & (X(S,D)=0^{n-d}1u \wedge 0^{n-d+j} \not< u, j \geq 0) \\
& \qquad \vee \\
& (X(S,D)=u10^{n-d} \wedge 0^{n-d+j} \not< u, j \geq 0)\}
\end{aligned}
$$

Note that the two disjuncts above generate mutually exclusive address vectors. Further, the substrings 1 and u may be thought of as members of sets $S_1 = \{u|\ u \in \Sigma_2, u \neq 0\}$ and $S_2 = \{u|\ |u|=d-1, 0^{n-d+j} \not< u, j \geq 0\}$. From Defn. 3.8 and Remark 3.2,

$$|J_{d,n}| = 2p_{p2}'(2d-n,d-1)$$

Recalling the expression for average distance (Eqn. 3.50) as

$$d_{p2}(n) = \frac{\sum_{d=1}^{n} 2dp_{p2}(d,n)}{N}$$

The increase in numerator will be the total of the increases in path length that S will see with each processor conditional on a single switch fault. This increase is simply

$$2\sum_{d=1}^{n-1}(2n-2d)\frac{2d-1}{2(N-1)}p_{p2}'(2d-n,d-1)$$

Using the above substitute for the numerator in the average distance formula, the average distance under a single node failure is computed. This has been tabulated (Table 5-2) together with the percentage degradation and the average distance in KYKLOS-I. Note that the percentage degradation starts at about 8% and decreases quite rapidly with increasing N until it is only few tenths of a percentage for N>128. Also, beyond N=64, the degradation, expressed as a percentage appears to decrease geometrically. Notwithstanding the fact that the above is a conservative upper bound on degradation, it is clear that regardless of N, KYKLOS-II is still vastly superior to KYKLOS-I even with a single node fault.

N	$\mathbf{d}_{p2}$	$\mathbf{d}_{p2}$**[deg]**	$\mathbf{d}_{p1}$	**% [deg]**
4	2.00	2.17	2.50	8.50
8	3.25	3.43	4.25	5.54
16	4.63	4.80	6.13	3.60
32	6.13	6.25	8.06	1.94
64	7.69	7.77	10.03	1.08
128	9.31	9.36	12.02	0.55
256	10.98	11.01	14.01	0.28
512	12.68	12.70	16.00	0.14
1024	14.40	14.41	18.00	0.07

$\mathbf{d}_{p2}$ = Average (p-sense) Distance in KYKLOS-II<2,2,n>

$\mathbf{d}_{p2}$**[deg]** = Upper bound on degraded avg. dist. in KYKLOS-II<2,2,n>

$\mathbf{d}_{p1}$ = Average Distance in KYKLOS-I<2,2,n>

%**[deg]** = Percentage degradation in avg. dist. in KYKLOS-II<2,2,n>

Table 5-2: Degradation in Average Distance in KYKLOS-II under single node fault

Chapter 6

Applications

In previous chapters, questions related to the properties of KYKLOS were posed and solved. Much of this chapter attempts to utilize some of those properties for specific applications. In Section 6.1, the motivation for KYKLOS as the ICN in an I/O Engine is investigated. A case study involving distributed joins over the KYKLOS-I and KYKLOS-II networks is presented in Section 6.2. A KYKLOS-based candidate for the d,k graph problem is proposed in Section 6.3. Finally, some spinoffs of KYKLOS are considered in Section 6.4.

6.1 The I/O Engine

Research related to the I/O Engine represents effort complementary to that for improving computational parallelism. In particular, the main objective of the I/O Engine is to develop an external memory system commensurate with the computational power of future generations of host machines [BROW85].

The main hallmarks of the I/O Engine as proposed involve

- *parallel access of databases*, and
- *parallel operations on data objects* that are being streamed from secondary storage toward the host.

6.1.1 I/O Engine Design

The gross architecture of the I/O Engine is shown in Fig. 6-1. The architecture is partitioned into four major levels:

- **Host processors**, which can be either general purpose or specialized processors.
- A set of "**Node Mappers**"[25] which make an associative translation from requested object names (ex. relation names, attribute names) to leaf (I/O) nodes where the objects are stored by generating a routing tag for the Interconnection Network.
- The **KYKLOS-II ICN** which couples host and leaf (I/O) processors, and also interconnects leaf processors. The non-leaf nodes in this network incorporate logic and buffering to support merge operations on data streams.
- **I/O nodes**, each consisting of a general purpose microprocessor, associative disk cache, a sort engine, and an intelligent controller for associated conventional moving-head disks.

6.1.2 Data Storage

To produce a machine capable of very high speed parallel access to data, the distribution of that data over the set of I/O nodes is crucial. To make effective use of the KYKLOS-II interconnection scheme, the data should be distributed to the I/O nodes, such that, given a sufficiently large quantity of data to be processed, the entire collection of network resources can be applied to that processing. The following data storage description is based on the relational model. The principal motivating factor in the design has been to provide a high degree of useful parallelism in processing

[25] These reside in the root node and are not shown in Fig. 6-1.

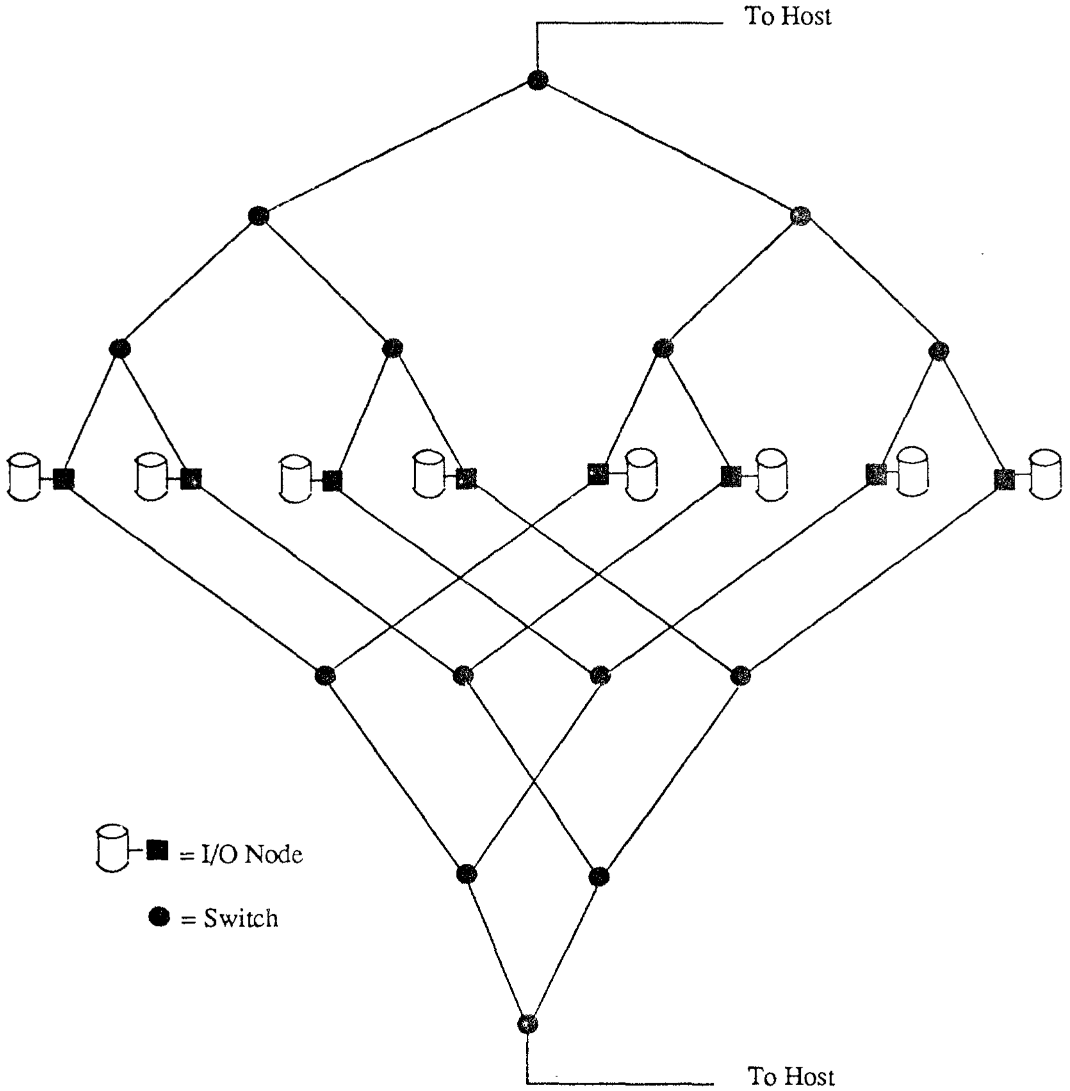


Figure 6-1: I/O Engine Overview

relational algebraic operations. Accordingly, each relation is partitioned and distributed among the leaf nodes of the architecture. There are two apparent methods for partitioning a given relation, R.

A *partitioning by rows* produces horizontal fragments r_i such that R is the union of all r_i i.e. $R = \cup_i r_i$.

A *partitioning by columns* produces vertical fragments ρ_j such that R is equal to the natural join of all ρ_j i.e. $R = \bowtie_j \rho_j$.
These vertical fragments may then be subject to horizontal fragmentation.

The model chosen for the I/O Engine makes use of both of these types of partitioning. The horizontal fragmentation, or *tuple-based schema* (TBS), is the primary storage model and we will be only concerned with it here. Finally, each fragment of the TBS is mapped to a particular leaf node of KYKLOS by the Node Mapper. Further details of the partitioning may be found in [MENE87b].

In the next section, we describe an algorithm[26] for efficient parallel processing of an operation that is a performance-limiting bottleneck for relational database systems. We show how KYKLOS-II may be used to obtain an appreciable improvement in performance over the single binary tree or KYKLOS-I.

6.2 Distributed Join Processing in KYKLOS

6.2.1 Semi-Join Algorithm

This discussion assumes that a natural join is to be computed on relations R and S, each of which is evenly distributed (horizontally) across the N leaf nodes of the I/O Engine using the TBS. The semi-join algorithm discussed below is only one of several parallel join algorithms that can be implemented on the architecture. It is used here as an example of how the architecture can support operations that require operands from different I/O nodes.

Let r_i and s_i denote the fragments of relations R and S at node i.

[26] Discussion with members of the I/O Project Group under the direction of Dr. A.G.Dale and Dr. R.M.Jenevein motivated the consideration of this algorithm for KYKLOS.

Let C denote the set of common attributes between R and S i.e. $C = R \cap S$. The semi-join algorithm may be thought of as a two-step process

- Phase 1: Each leaf node, i, broadcasts its lists of common attribute values $c_{ri} = \pi_C r_i$ and $c_{si} = \pi_C s_i$ to every other node[27]. On receipt of c_{rj} and c_{sj}, $j \neq i$, each leaf node, i, computes the semi-joins
  $r_{ij}' = r_i \bowtie c_{sj} = r_i \ltimes s_j$
  $s_{ij}' = s_i \bowtie c_{rj} = s_i \ltimes r_j$
  Also it computes the join between its local fragments i.e.
  $r_i \bowtie s_i$
- Phase 2: Each pair of leaf nodes, i,j, $i \neq j$ ships the results of the above semi-joins to a rendezvous determined by mutual consent, at which point a partial join
  $(r_{ij}' \bowtie s_{ji}') \cup (r_{ji}' \bowtie s_{ij}')$ is performed.

The first phase of the algorithm is particularly straightforward: The two trees in KYKLOS may be used to broadcast the c_{si}'s and c_{ri}'s, $0 \leq i < N$. These lists are used to compute that subset r_{ij}' of r_i which will participate in the partial join between the fragments of i and j. It is the second phase of the algorithm that uses the interconnection structure of KYKLOS-II to achieve a significant improvement in load balancing and network traffic over a single binary tree or KYKLOS-I. This is discussed in the next two subsections.

6.2.1.1. The Mid-point Strategy for performing Partial Joins

Consider a 64-node KYKLOS-II (Fig. 6-2) and consider the partial join between the semi-join fragments of nodes 11(001011) and 46(101110). The shortest paths between these two nodes as prescribed by the P-II Routing Strategy of Section 3.2 is sketched

[27] Alternatively, each I/O node may perform a hash on join attribute values to the set of I/O nodes in lieu of performing global broadcast.

in Fig. 6-2. To minimize network traffic, it makes sense to select a rendezvous for the partial join between these node fragments *somewhere* on the shortest path. If the results of the join were to be sent to the host (located at the root), the partial join should be done at the node closest to the root as shown in Fig. 6-2. However, it is often the case that the results of the partial join are an intermediate result which needs to be returned to the leaf nodes as input for the next phase of a computation. As such, a reasonable compromise between these conflicting requirements would be to perform the join at the mid-point of the path connecting them.

For the example of Fig. 6-2, the point on the path between leaf nodes 11 and 46 that is closest to the (top) root is a level 3 node. However, the strategy being considered would select the mid-point for performing the partial join. Thus, a level 2 node is selected which is closer to the leaf nodes. Intuitively, an added advantage of performing the partial join at the mid-point and hence at sites closer to the leaves is that as the level number in the top tree decreases, the number of nodes available for processing the partial joins increases. As a result, the workload gets more evenly distributed.

6.2.1.2. Case Study and Analysis

For the purpose of analyzing and comparing traffic using the Mid-point Strategy, a case study using KYKLOS-I and KYKLOS-II is presented, with each network composed of two binary trees.

6.2.1.3. Distribution of Partial Joins

In the KYKLOS-I case, the Mid-point Strategy is equivalent to performing the partial join at the root of the smallest subtree containing both i and j, an approach commonly employed for a single binary tree. Of course, half of the partial joins can be performed in each tree of KYKLOS-I. Fig. 6-3 is an example of this approach with N=8. The

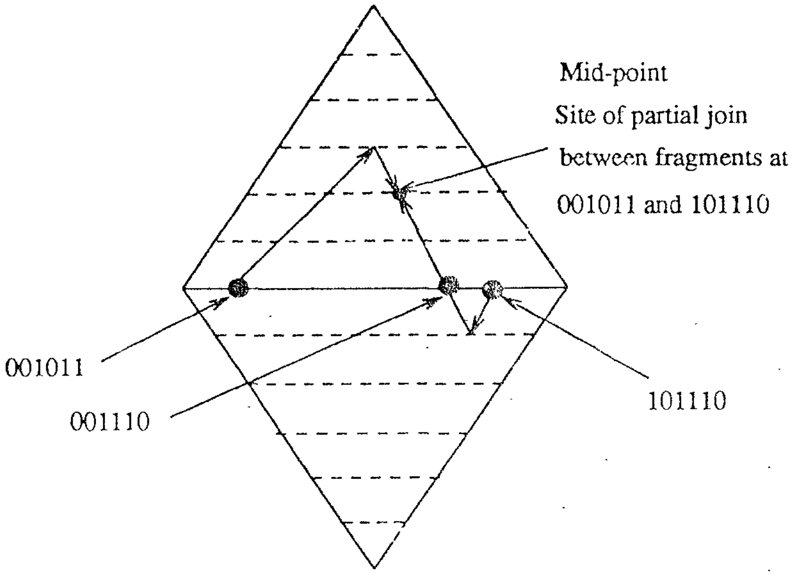


Figure 6-2: Midpoint Strategy in KYKLOS-II with 64 nodes

notation i,j at the nodes indicates the joining of the corresponding fragments of the relations at nodes i and j. Note that there are a total of 8 pairs of leaf nodes whose fragments join at a root which is the busiest node in the network.

Joining the partial fragments at the mid-point of the path using P-II routing in KYKLOS-II produces the partial joins shown in Fig. 6-4. Employing the same resources but a different interconnection strategy, the KYKLOS-II Network reduces the maximal load to 3.

Table 6-1 shows the maximum workload i.e. the number of partial joins performed at the busiest site. (Recall that for each pair of nodes i,j, i≠j, the Semi-join Algorithm requires 2 partial joins to be computed). For KYKLOS-I, it is easy to see that the roots are the sites for the partial joins of every pair of nodes separated by a distance of 2logN. Since there are $N^2/4$ such pairs of nodes and 2 partial joins per pair, there are $N^2/2$ partial joins to be computed at the roots or $N^2/4$ partial joins per root. This means that over half of the total number of partial joins are done at the root nodes. By

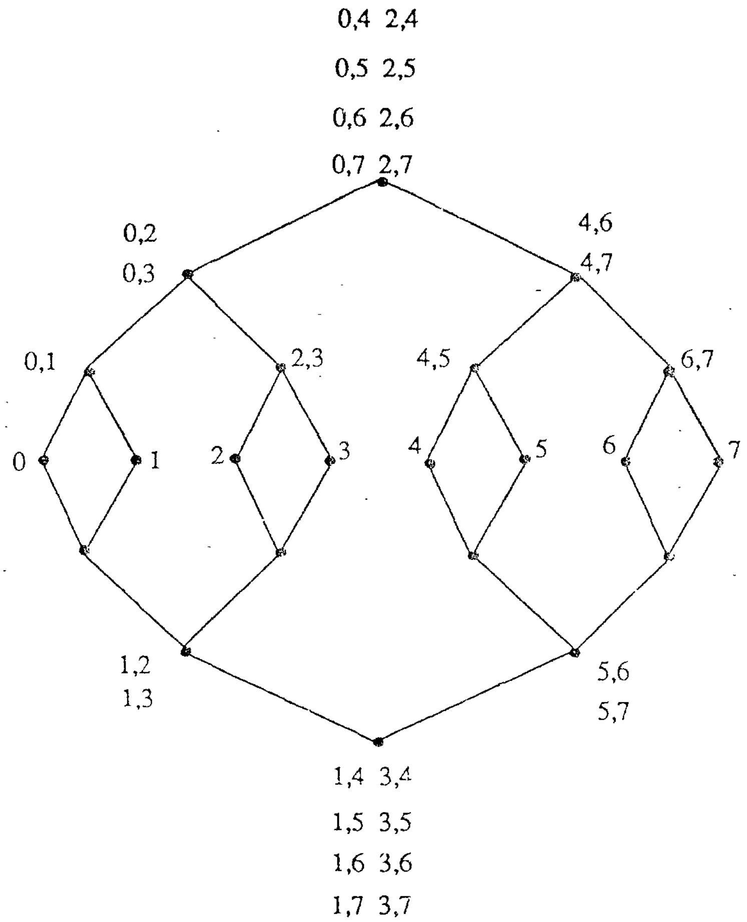


Figure 6-3: Distribution of partial joins in KYKLOS-I

comparison, the maximum workload for KYKLOS-II is at least an order of magnitude less for N>32. Also, the workload in KYKLOS-II is spread.out more evenly over the whole network.

6.2.1.4. Response Time

Another benefit from the KYKLOS-II interconnection strategy is the reduction in the average distance that the semi-join inputs must travel, thereby improving response time. For instance, for N=8, the average distance is about 1.8 links traversed as compared with 2.4 in the case of KYKLOS-I, an improvement of about 25%.

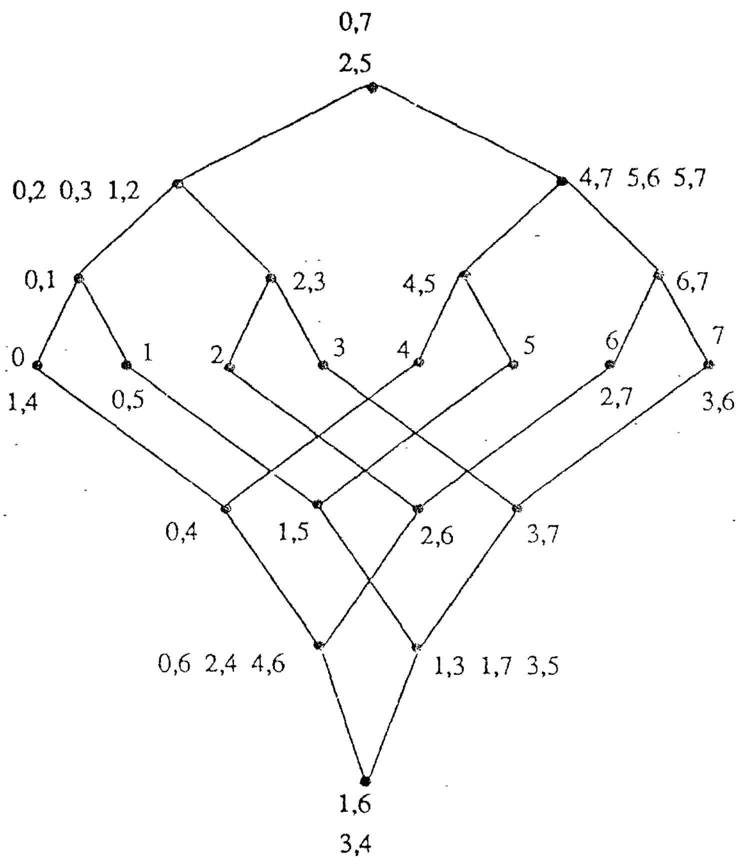


Figure 6-4: Distribution of partial joins in KYKLOS-II

N	KYKLOS-I	KYKLOS-II
8	16	6
16	64	12
32	256	28
64	1024	88
128	4096	288
256	16,384	928
512	65,536	3008
1024	262,144	9728

Table 6-1: Maximum Number of partial joins at a node site

6.2.1.5. Network Traffic

Three categories of communication requirements may be identified depending on the task at hand. The tasks requiring communication in the semi-join algorithm are

- Global broadcast of attribute values to all other leaf nodes
- Broadcast of semi-join inputs to the predetermined site for computing the partial joins and
- Transfer of semi-join output values to the host or to the leaf nodes.

The global broadcast is well suited to a tree-structured network. Both KYKLOS-I and KYKLOS-II will exhibit the same performance characteristics for this operation.

The second task necessitates transmission of tuples along the shortest path between every pair of leaf nodes. This task is similar to a transmission of data packets from one leaf node to every other leaf node. In Chapter 4, it was shown that the maximum traffic density for this latter task is $O(N^{1.5})$ if the H-II Routing Strategy is used. Though the P-II Routing Strategy has a higher asymptotic bound on maximum traffic density of $\sim \Omega(N^{1.7})$, as shown in Section 4.4, the constants in the expressions for traffic density are such that P-II actually outperforms H-II Routing up to N=2048. By comparison, the maximum traffic in KYKLOS-I is $O(N^2)$.

Finally, the third task could involve transmission of results to the leaf nodes. Table 6-2 shows the maximum traffic densities for both, KYKLOS-I and KYKLOS-II as a function of N. Note that as in the case of the distribution of partial joins, the maximum traffic in KYKLOS-II is only a tiny fraction of that in KYKLOS-I. This shows that congestion is a far more serious problem with KYKLOS-I. Note that the improvement in traffic characteristics in KYKLOS-II has been brought about by no addition of hardware resources, only by altering the interconnection strategy.

N	KYKLOS-I	KYKLOS-II
8	8	4
16	32	8
32	128	22
64	512	60
128	2048	176
256	8192	528
512	32,768	1696
1024	131,072	5664

Table 6-2: Maximum Link Traffic of partial joins at a node site

6.3 KYKLOS and the d,k Graph Problem: The Chef's Recommendation

In KYKLOS-II<2,2,n>, the leaf nodes and the root have degree=2 while the other nodes in the network have degree=3. Clearly, adding a third tree to share the common set of leaf nodes and coupling the roots of the three trees to a **Father Root** would not *cost* any more in terms of node degree. In Section 3.2.5, it was shown that that there is precisely one processor at a maximum distance of 2n from any other in KYKLOS-II<2,2,n>. It also follows from the P-II Routing Strategy that two processors are maximally separated iff their addresses are the 1's complements of each other. It makes sense for the switch nodes at level 1 of the third tree to couple every processor with its 1's complement. This immediately leads to the following theorem

Theorem 6.1 The *processor diameter* of the 3-tree modified KYKLOS as defined above is 2n-2 or less. □

The connections of the level 2 links in the third tree are dictated by the requirement that every pair of processors, x and x⊕0101..., should exist in the same subtree of height 2. The LS for the third tree is generated by the pseudo code in Fig.6-5. A is a boolean array of length N and LS is the array which holds the LS for the third tree as it is generated. Comp(x) returns the 1's complement of x and alt(x) returns $x \oplus 0101...$. The program works by initializing each element of A to false. The variable, i, is used to index the array A. A[i]=true implies that the LS array has already been loaded with i. Note that the LS so generated forces x and x' to be in the same subtree of height 1. Also, x and $x \oplus 0101...$ are guaranteed to be in the same subtree of height 2. The program forces the output of a deterministic sequence. In reality, connections for the rest of the third tree (levels 3 to n) are arbitrary so long as a full binary tree is obtained.

```
for(k=0; k<N; A[k++]=false)
j=0;
i=0;
do
{
        if(A[i]==false)
        {
             LS[j++]=i; A[i]=true;
             LS[j++]=comp(i); A[comp(i)]=true;
             LS[j++]=alt(i); A[alt(i)]=true;
             LS[j++]=comp(alt(i)); A[comp(alt(i))]=true;
        }
        i++;
}
while(j<N)
```

Figure 6-5: Procedure to generate the LS for the third tree

An example of this 3-tree KYKLOS with 16 leaf nodes is shown in Fig. 6-6.

The following [MENE85c] is beyond the scope of this dissertation.

Theorem 6.2: The modified KYKLOS-II<2,3,n> as defined above has a diameter $\leq$ 2n-2. □

The total number of nodes in the modified KYKLOS-II<2,3,n>

= # of leaf nodes + # of non-leaf nodes + 1(Father Root)

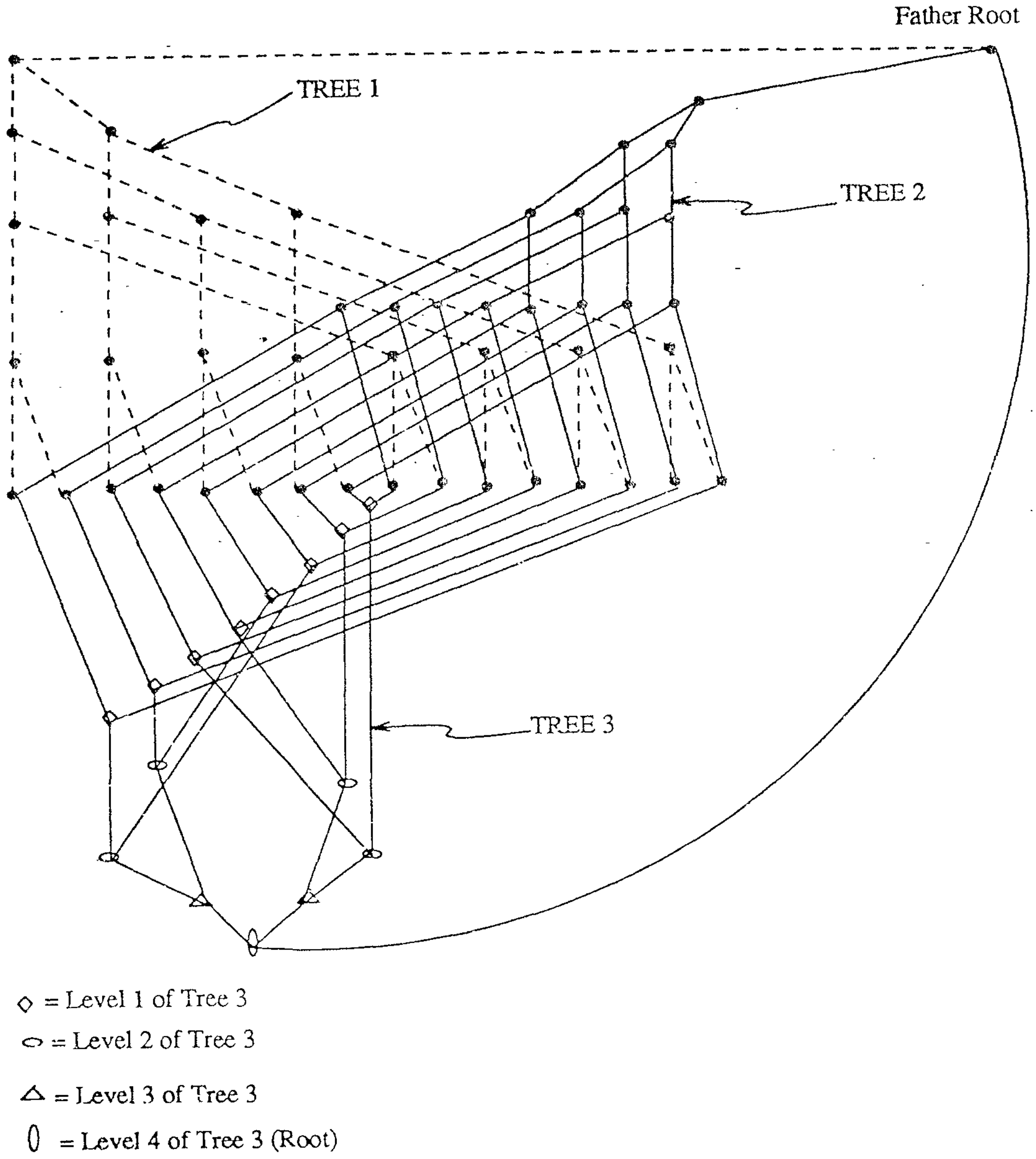


Figure 6-6: 3-tree KYKLOS with 16 leaf nodes

= 4N-2.

This is tabulated as a function of network diameter in Table 6-3. Included for comparison is the Moore bound which represents the upper limit on the number of nodes in the network for degree=3. In addition, results reported in [MEMM81][28] enable comparison of KYKLOS with other analytically obtained graphs [ARDE78, WILK70, TOUE79].

k	MB	MR	K-II
4	46	30	30
6	190	72	62
8	766	124	126
10	3070	230	254

k = diameter,

MB = Moore Bound,

MR = Results reported in [MEMM82] from [ARDE78], [WILK70] and [TOUE79].

K-II = Modified KYKLOS-II

Table 6-3: Network Size as a function of diameter, degree=3

The results reported herein are only meant to illustrate the power of the KYKLOS idea. Note that the construction of the third tree was arbitrary beyond level 2. If more caution were exercised in building the third tree, it is reasonable to expect a further decrease in diameter over that attained by Theorem 6.2. In addition, KYKLOS-III is known to have superior distance properties compared to KYKLOS-II. By judicious choice of connections for the third tree using the KYKLOS-III topology, the author feels that further improvements could result. Finally, note that as N (and hence k) gets larger, KYKLOS seems to fare better compared to the results reported in [MEMM81] which include some of the best networks constructed in the context of the (d,k) graph problem.

[28] This contains some of the best known results on diameters of graphs constructed in the context of the d,k graph problem.

6.4 Topological spinoffs of KYKLOS

6.4.1 HyperKYKLOS: The augmented Hypercube

HyperKYKLOS is a special case of the KYKLOS Network in which r=n=logN.

The simplest version of KYKLOS is KYKLOS-I. Analogously, the simplest version of HyperKYKLOS is HyperKYKLOS-I defined by the n-tuple

$<L_0, L_1, ..., L_{n-1}>$,

where L_i is the LS for the i^{th} tree.

Let $L_i(j)$ be the j^{th} term of L_i. Then

$L_i(j) = \rho_i(j)$

where $\rho_i(j)$ is the number obtained by rotating j, represented as a binary number, a total of i bits to the left.

Fig. 6-7(a) shows a HyperKYKLOS with N=8, r=3. The LS's for the three trees of this structure are

$L_0 = 0\ 1\ 2\ 3\ 4\ 5\ 6\ 7$

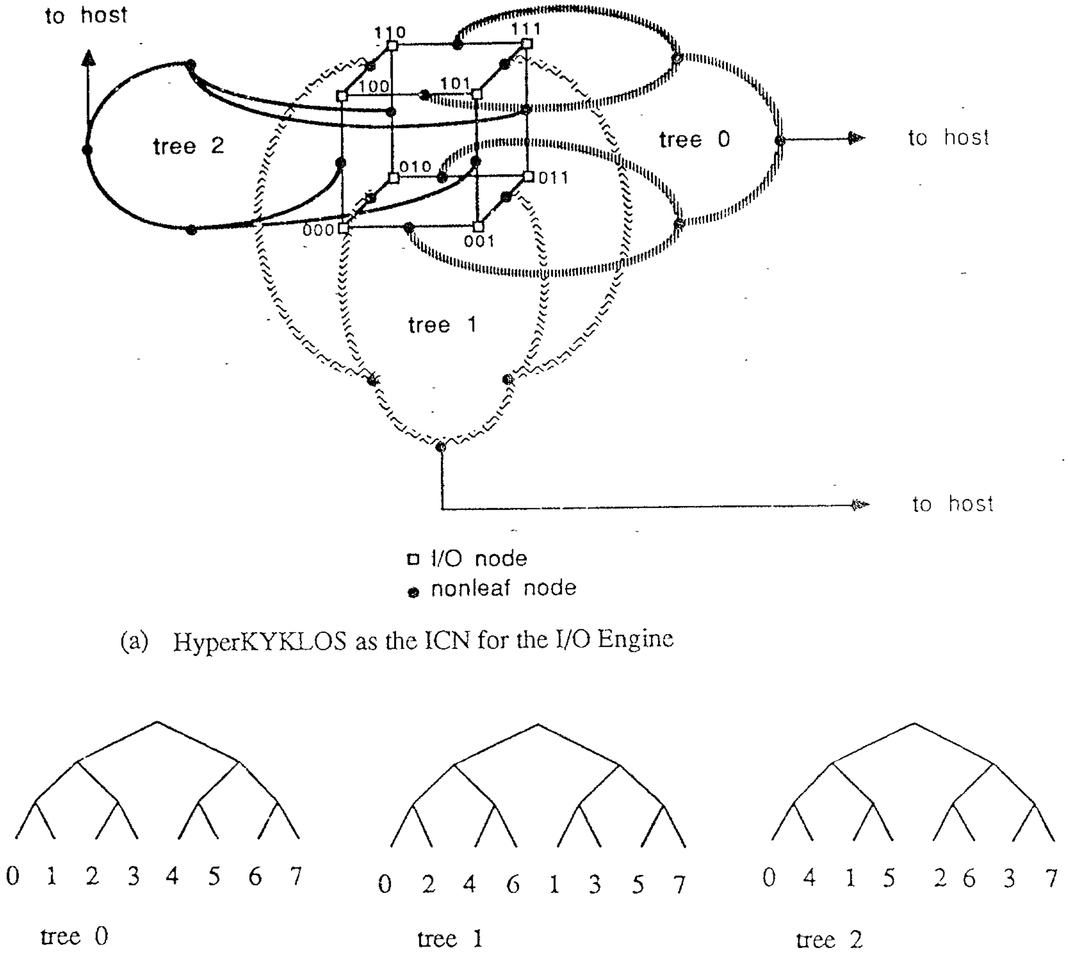
$L_1 = 0\ 2\ 4\ 6\ 1\ 3\ 5\ 7$

$L_2 = 0\ 4\ 1\ 5\ 2\ 6\ 3\ 7$

These are shown in Fig. 6-7(b).

It should be noted that the subgraph made up of the processors and level 1 nodes in each tree constitutes a *modified Hypercube* i.e. an n-cube with a processor at each vertex and a level 1 node along each cube edge. As a result, HyperKYKLOS is not merely an r-tree KYKLOS. It embeds a Hypercube between the level 1 nodes of the r trees,[29] hence the name HyperKYKLOS.

[29] A proof of this for HyperKYKLOS-I will be included in a forthcoming report on the graph-theoretic properties of HyperKYKLOS.



(a) HyperKYKLOS as the ICN for the I/O Engine

(b) LS's of the three trees in HyperKYKLOS

Figure 6-7: 3-Tree HyperKYKLOS

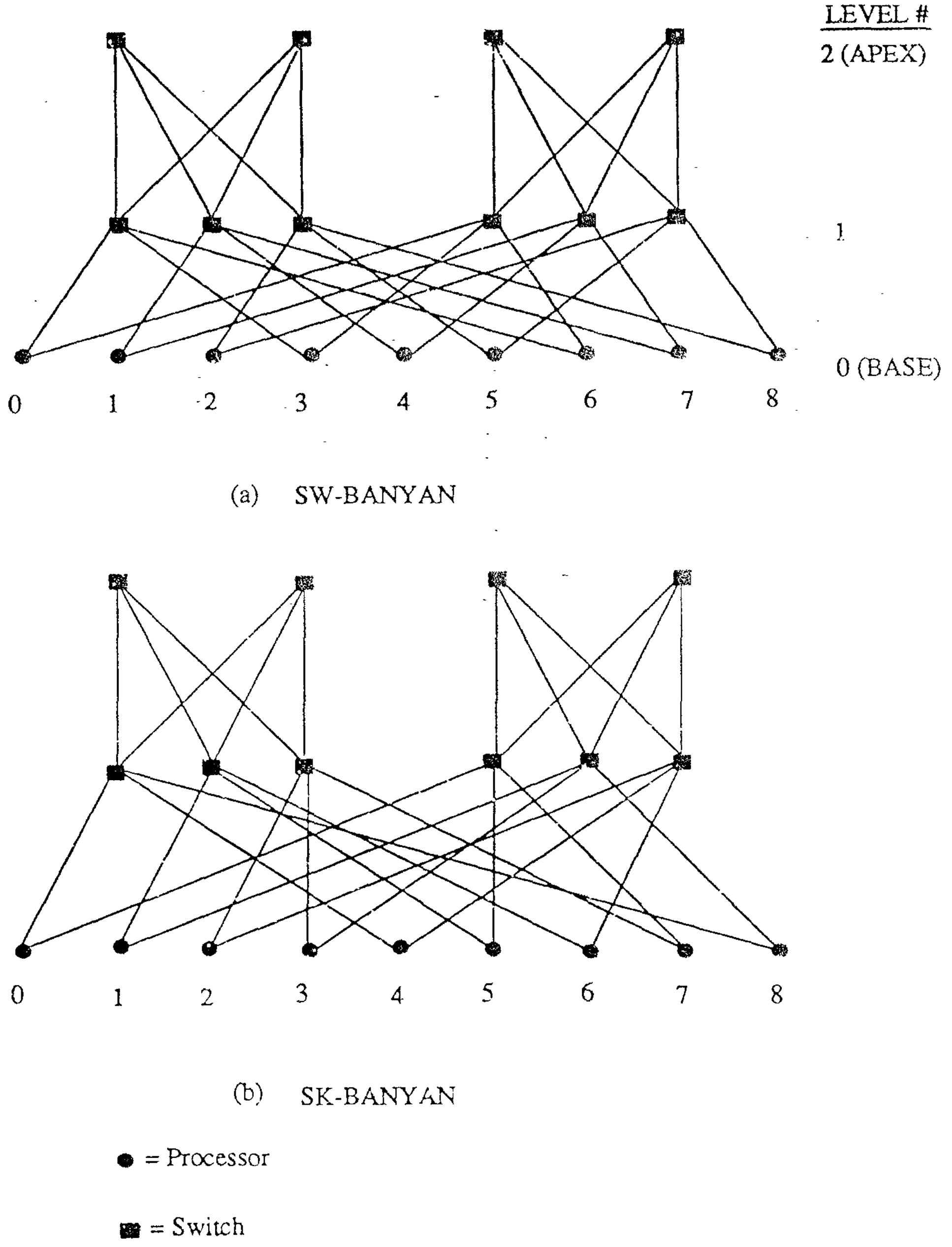
HyperKYKLOS was motivated by performance studies of distributed database algorithms on a Hypercube as reported in [MENE87a]. Current work includes an investigation of the topological properties of HyperKYKLOS and enhanced versions of this topology.

6.4.2 The SK-Banyan

We have seen that altering the interconnection of links in the bottom tree of KYKLOS resulted in a significant improvement in the topological properties of the network. This has motivated a similar study with the SW-Banyan [DeME86]. Fig. 6-8(a) shows a non-rectangular banyan with three levels of nodes(0-2). Level 0 represents the base nodes and level 2 represent the apex nodes. With processor-memory pairs at the base nodes and switches elsewhere, it would be desirable to reduce the inter-base node communication latency.

Consider processors 0 and 3 in Fig. 6-8(a). There are two shortest paths of length=2 between these two nodes. In fact, it is easy to see that there are multiple shortest paths between every pair of processors as in KYKLOS-I. The SK-Banyan (Fig. 6-8(b)) is an attempt to reduce interprocessor distances by breaking redundant connections and by supplanting these with a more intelligent connection scheme without at the same time compromising cost i.e. note that the number of links is the same in both, Fig. 6-8(a) and 6-8(b). This involves skewing the connections between every pair of adjacent node levels (hence the name SK-Banyan). Thus, processor 0, at a distance=2 from processors 3 and 6 in the SW-Banyan is at this minimal distance from four processors in the SK-Banyan (i.e. 4, 5, 7, and 8).

This description in this subsection represents an adumbration of the key idea behind the SK-Banyan. The details of the interconnection strategies for this topology may be found in [DeME86]. Finally, research on network properties such as distance, traffic and fault tolerance are being actively pursued.



(a) SW-BANYAN

(b) SK-BANYAN

Figure 6-8: SW-Banyan and SK-Banyan

Chapter 7

Conclusions

7.1 Accomplishments

Both, the goals and accomplishments in the study of KYKLOS and of this document, in particular, divide into the following broad categories:

- Interconnection Strategies
- Network Properties
- Application Areas

Interconnection Strategies: Two major alternative schemes for connecting the bottom tree were defined in addition to the well-known simple Double Tree. *Labeling sequences* and *permutations* were employed to define each tree in KYKLOS. This has provided a representation which, while being simple and compact, has proved very powerful in practice. It has facilitated precise mathematical analysis of the topological characteristics of the KYKLOS-II Network. Also, the derivation of KYKLOS-III from KYKLOS-II, while painful, was rendered manageable, thanks to the use of labeling sequences.

The desire to embed a ring structure within the Double Tree KYKLOS motivated the construction of KYKLOS-III. For this purpose, an isomorphic version of KYKLOS-II called the *W-Form* was built. A subtle change in the W-Form structure generated KYKLOS-III.

The concept of *m-sense optimality* was defined. This concept is at the heart of the KYKLOS design philosophy. A procedure to construct m-sense optimal KYKLOS topologies was presented. It was shown that, both, KYKLOS-II and KYKLOS-III, were m-sense optimal.

Network Properties: A study of such practical concerns in interconnection networks as routing, fault tolerance and traffic congestion have been more than adequately addressed. A noteworthy aspect of this work concerns the relationship between different strategies that have been proposed for routing in KYKLOS-II and the different performance characteristics obtained for each. For example, P-II is the "shortest path routing strategy" while H-II is the one that minimizes traffic bottlenecks (asymptotically). M-II Routing is particularly appealing in a circuit-switched environment because it is inherently deadlock-free. Further, it serves as a base with which to compare the performance of the other routing strategies. Finally, all three possess simplicity as a common virtue. Each of these three routing algorithms have been defined and their performance characteristics have been thoroughly investigated.

Expressions for interprocessor distance distribution were obtained. Given the interprocessor traffic matrix, this enables the estimation of average communication latency and link traffic. Traffic matrices are, in general, a function of application or algorithm. In the absence of any information about communication patterns for a specific task, the simplifying assumption of uniform message distribution was used to compute average distance as in other tree-based networks.

To standardize the comparison of various network topologies, *normalized distance*, L', is used here. This is the product of the average distance between processor pairs and the maximum number of ports per node of the network. For purposes of comparison, values for L' in KYKLOS-II are juxtaposed with corresponding values for various tree topologies and for the Hypercube as shown in Table 7-1[30]. Note that L' is least for the Hypercube up to about n=8 beyond which KYKLOS leads all the other topologies.

n	BT	HR	FR	HC	HT	K
2	7.50	6.38	5.00	2.00	6.00	6.00
1	12.75	10.50	10.00	4.50	10.00	9.75
4	18.38	16.56	17.50	8.00	15.00	13.89
5	24.19	23.59	26.25	12.50	19.50	18.39
6	30.09	31.11	35.63	18.00	24.75	23.07
7	36.05	38.87	45.31	24.50	29.38	27.93
8	42.02	46.75	55.16	32.00	34.69	32.94
9	48.01	54.69	65.08	40.50	39.34	38.04
10	54.01	62.66	75.04	50.00	44.67	43.20
11	60.00	70.64	85.02	60.50	49.34	48.45
12	66.00	78.63	95.01	72.00	54.67	53.76

n = Logarithm of number of leaf nodes

BT = Binary Tree

HR = Half Ring X-Tree

FR = Full Ring X-Tree

HC = Hypercube

HT = Hypertree

K = KYKLOS-II<2,2,n>, p-sense.

Table 7-1: Normalized Distances of different topologies as a function of Network Size (N)

[30] adapted from [GOOD81].

Both, *link utilization* and *maximum traffic density* in KYKLOS were studied. Again, routing strategy played a significant role in determining their characteristics. The maximum traffic congestion in KYKLOS-II was shown to be asymptotically less than that in a single tree or in the simple double tree. The maximum traffic density in the latter case is O(N^2), while it was shown to be O($N^{1.5}$) in the case of KYKLOS-II using the H-II Routing Strategy. So, for example, while the simple Double Tree would be expected to reduce the traffic bottleneck by a factor of 2 over a single tree, KYKLOS-II could easily achieve a $2\sqrt{N} \sim 60$-fold improvement in traffic density in a 1000 leaf node KYKLOS. That the cleverness of the interconnection scheme achieves an asymptotic improvement in traffic properties is no mean achievement.

The *slice concept*, inexorably linked with that of m-sense optimality, was instrumental in obtaining the low traffic density in KYKLOS-II. This concept can be generalized to the 3-tree KYKLOS in which case the maximum traffic density is decreased to O($N^{1.33}$), a further asymptotic improvement. Clearly, the communication latencies in this 3-tree KYKLOS will decline further. This development does hold promise when you consider the fact that the normalized distance, L', will shrink further. This is so, since the communication latencies will be further reduced while fanout remains unchanged. In addition, the O($N^{1.33}$) traffic density surpasses the performance of the other tree topologies.

Fault tolerance of the KYKLOS network was studied in respect to preserving connectivity of processing resources under switch node faults. This was exemplified by the special case of double switch node failures. Here, the probability of disconnecting processors was three times higher in KYKLOS-I than in KYKLOS-II--another vindication of the original claim that the shuffle-connected bottom tree of KYKLOS-II would ameliorate the properties of the simple Double Tree.

From a theoretical standpoint, KYKLOS-II represents a class of networks with *generalized Tribonacci distance distributions.* It is indeed heartening to see yet another application of Fibonacci sequences, this time in the realm of network topologies.

Application Areas: Rich application areas have been identified. In particular, the usefulness and versatility of KYKLOS as an interconnection network for the I/O subsystem of a high-performance host computer have been established. The network provides on-the-fly and pipelined data stream processing besides parallel access to data. Thus, both, I/O access and processing, are speeded up. Also, the workload in computation-intensive join operations may be distributed throughout the network. Finally, such communication paradigms as the N-broadcast, an integral part of several distributed join algorithms can be easily accommodated on the network for the very reason that the traffic characteristics in KYKLOS are superior to other tree-based architectures.

The work on KYKLOS has been directly applied to the d,k graph problem. A third tree addition to KYKLOS-II has been constructed, driven by the results on distance in KYKLOS-II. Its characteristics in terms of the d,k graph problem have been very encouraging. Also, given the results of some preliminary investigation on KYKLOS-III, it would not be presumptuous to expect even better properties of this structure (i.e. a three-tree KYKLOS built on top of a double tree KYKLOS-III).

In terms of spinoffs in the area of alternative network topologies, the KYKLOS interconnection concept has spawned similar ventures with the banyan to further improve some of its properties. Finally, the extension to log N trees in the garb of HyperKYKLOS and its use in the I/O Engine could have a significant impact on the performance of the I/O Engine.

In conclusion, KYKLOS retains the nice, simple properties of trees, yet does not inherit the obvious weaknesses of traffic congestion and poor fault tolerance. The linear cost, low fanout and logarithmic communication delays are attractive attributes of trees and KYKLOS shares those traits. Finally, the ability to embed a ring in KYKLOS-III enhances its utility besides suggesting a further improvement in its properties.

7.2 Future Work

Several areas of work directly related to KYKLOS may be identified.

- M-sense optimality needs more respectable treatment. Specifically, the following questions need to be addressed.
  "Are the m-sense distance characteristics obtained for KYKLOS-II (or KYKLOS-III) the ultimate, or is there anything better? If there is anything better, then what is it? If not, then prove it (that nothing better exists)." Also, these results need to be generalized to more than two trees. Next, are the m-sense distance expressions derived for KYKLOS-II, in fact, the *sine qua non* for m-sense optimality (as it appears)?

- Closely related to the above problem is the question "Are there any Γ-like Sequences lurking around? How can this be generalized to 3, 4, . . ., 100 trees?

- The d,k graph problem needs further investigation. Using KYKLOS or the KYKLOS idea, there is good reason to believe that we may get better results than any obtained so far. In particular, KYKLOS-III needs to be investigated both, in its own right as also from the perspective of the d,k graph problem. Because they are cousins, it is likely that the analysis in KYKLOS-III could be inspired by the results on KYKLOS-II.

- The Moore bound should be tightened. The insight provided by KYKLOS could be of use here. An upper bound for average distance seems to be an interesting extension worth pursuing.

- The m-Tribonacci sequence is still intriguing. Could the complex roots have an interpretation?
- It already appears that the distance properties in HyperKYKLOS may be obtained using similar though more complicated recurrences. This should be analyzed and other HyperKYKLOS topologies should be defined.
- Other areas of research include the study of algorithms that exploit the structure of KYKLOS, the study of metrics that measure the combined performance-fault-tolerance characteristic of a network topology and issues related to VLSI layout.

Appendix A.

Auxiliary Results on Permutations and LS's

Result A.1

Trivially

$\gamma_N(\gamma_N(x)) = x.$ (A.1)

A.1 Results Related to the γ-Permutation

The next three results will be used in Section 2.3

Result A.2

$\gamma_N(x) = 2\gamma_{N/2}(x), \quad 0 \leq x < N/2$ (A.2)

Let $x = 0i_{n-2}...i_1i_0$ (since $x < N/2$)

So $\gamma_N(x) = i_0i_1...i_{n-2}0$

Also $\gamma_{N/2}(x) = i_0i_1...i_{n-2}$

or $2\gamma_{N/2}(x) = i_0i_1...i_{n-2}0.$

Result A.3(a)

$\gamma_N[N/2 + \gamma_{N/2}(x)] = 2x + 1, \quad 0 \leq x < N/2$ (A.3)

Result A.3(b)

$\gamma_N[N/2 + y] = 2\gamma_{N/2}(y) + 1, \qquad 0 \leq y < N/2$ (A.4)

Let $x = 0i_{n-2}...i_1i_0$ (since $x < N/2$)

$\gamma_{N/2}(x) = i_0i_1...i_{n-2}$

$N/2 + \gamma_{N/2}(x) = 1i_0i_1...i_{n-2}$

$$\begin{aligned}\gamma_N[N/2 + \gamma_{N/2}(x)] &= i_{n-2}...i_1i_01 \\ &= i_{n-2}...i_00 + 1 \\ &= 2x + 1 \\ &= \text{RHS.}\end{aligned}$$

Since $y < N/2$, let $y = \gamma_{N/2}(x)$, so

$$\begin{aligned}\gamma_N[N/2 + y] &= \gamma_N[N/2 + \gamma_{N/2}(x)] \\ &= 2x + 1 \text{ from Result 2.3(a)} \\ &= 2\gamma_{N/2}(\gamma_{N/2}(x)) + 1 \text{ ... from Result 2.1} \\ &= 2\gamma_{N/2}(y) + 1 \\ &= \text{RHS.}\end{aligned}$$

A.2 Procedure to obtain the Equivalence Class Leader, given an LS

Let MIN(i,j) be a function that returns the minimum-valued leaf node label in the subtree rooted at the i^{th} descendant of node j.

Then the following procedure will permute zero or more non-leaf nodes to obtain an LS that is *equivalent* to the given LS.

```
For x=1 to n
{
   For y=0 to $m^{n-x}$ - 1
   {
      Permute the descendants of <-x,y> so that:
      $c_1<c_2$ => MIN($c_1$,<-x,y>)<MIN($c_2$,<-x,y>)
      where $0\leq c_1,c_2<m$.
   }
}
```

Appendix B.

Roots of the Characteristic Polynomial of the pp2' Recurrence Relations

B.1 Co-ordinate Transformation

In Chapter 3 (Theorem 3.6), it was shown that

$$p_{p2}'(d,n) = \mu[p_{p2}'(d-1,n-1) + p_{p2}'(d-2,n-2) + \ldots\ n\text{-}d+1 \text{ terms}] \tag{B.1}$$

where

$$\mu = m-1 \tag{B.2}$$

Substituting

$$p_{p2}'(d,n) = a_{2n-d+1,n-d+1} \tag{B.3}$$

gives

$$a_{n+k,k} = \mu[a_{n+k-1,k} + a_{n+k-2,k} + \ldots + a_{n,k}] \tag{B.4}$$

where

$$k = n\text{-}d+1 \tag{B.5}$$

is the order of the recurrence.

In Table B-1 $a_{n+k,k}$ are mapped onto integral coordinates in the octant bounded by d=n, d=0 of the d,n plane. The above transformation allows a pictorial representation of the recurrence in (B.4). The sequence

$a_{1,1}, a_{2,1}, a_{3,1}, \ldots$

appears on the semi-infinite line originating at d=0, n=0, inclined at 45^o to the n-axis and extending in a south-easterly direction to infinity. (This has been referred to as the *principal diagonal* in Section 3.2.5). Sequences parameterized by k, k>1,

$a_{2k-1,k}, a_{2k,k}, a_{2k+1,k}, \ldots$

are mapped to lines parallel to the principal diagonal beginning at d=0, n=k-1. (These semi-infinite lines are referred to as *paradiagonals* in Section 3.2.5).

d	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7
0	$a_{1,1}$	$a_{3,2}$	$a_{5,3}$	$a_{7,4}$	$a_{9,5}$	$a_{11,6}$	$a_{13,7}$	$a_{15,8}$
1	-	$a_{2,1}$	$a_{4,2}$	$a_{6,3}$	$a_{8,4}$	$a_{10,5}$	$a_{12,6}$	$a_{14,7}$
2	-	-	$a_{3,1}$	$a_{5,2}$	$a_{7,3}$	$a_{9,4}$	$a_{11,5}$	$a_{13,6}$
3	-	-	-	$a_{4,1}$	$a_{6,2}$	$a_{8,3}$	$a_{10,4}$	$a_{12,5}$
4	-	-	-	-	$a_{5,1}$	$a_{7,2}$	$a_{9,3}$	$a_{11,4}$
5	-	-	-	-	-	$a_{6,1}$	$a_{8,2}$	$a_{10,3}$
6	-	-	-	-	-	-	$a_{7,1}$	$a_{9,2}$
7	-	-	-	-	-	-	-	$a_{8,1}$

Table B-1: Mapping $a_{i,j}$'s onto the d,n plane

Solution of (B.4) for μ=1 with predefined initial conditions has been attempted[MILE67, FLOR67, GODS83]. This work attempts to generalize some of those results. The characteristic polynomial, $E_1(z)$ of (B.4) is

$$E_1(z) = z^k - \mu(z^{k-1}+z^{k-2}+ \ldots +1) \qquad \text{(B.6)}$$

If $E_1(z)=0$ has distinct roots, $r_{1,k,m}$, $r_{2,k,m}$, . . ., $r_{k,k,m}$, then the general solution for (B.4) is

$$a_{n+k,k}=c_{1,k,m}r^n_{1,k,m}+c_{2,k,m}r^n_{2,k,m}+\ldots+c_{k,k,m}r^n_{k,k,m} \qquad (B.7)$$

where $c_{1,k,m}$, $c_{2,k,m}$, . . ., $c_{k,k,m}$ are constants.

In the next section, we proceed to gain some insight into the location of these roots.

B.2 Roots of the Characteristic Equation

From (B.6),

$$E_1(z) = z^k - \mu(z^k-1)/(z-1), \quad z \neq 1 \tag{B.8}$$

Multiplying by z-1 gives

$$E_2(z) = (z-1)E_1 = z^{k+1} - (\mu+1)z^k + \mu \tag{B.9}$$

From the Fundamental Theorem of Algebra, $E_1(z)$ and $E_2(z)$ have k and k+1 roots respectively which are investigated below.

Statement B.1 *$E_1(z)$ has at least one root between μ and $\mu+1$.*

From (B.6),

$$\begin{aligned} E_1(\mu) &= \mu^k - \mu(\mu^{k-1}+\mu^{k-2} \ldots +1) \\ &= -(\mu^{k-1}+\mu^{k-2} \ldots +\mu) < 0. \end{aligned} \tag{B.10}$$

Also from (B.6),

$$\begin{aligned} E_1(\mu+1) &= (\mu+1)^k - \mu[(\mu+1)^{k-1} + (\mu+1)^{k-2} \ldots +1] \\ &= 1 \end{aligned} \tag{B.11}$$

Since $E_1(\mu)<0$, $E_1(\mu+1) = 1$ and $E_1(z)$ is continuous, there must be a real root between μ and $\mu+1$.

The goal of the remainder of this subsection is to show that the remaining k-1 roots of $E_1(z)=0$ are distinct and lie inside the unit circle.

Statement B.2 *$E_2(z)$ has no roots on the unit circle except for $z=1$ (with multiplicity 1).*

Let z_1 be a root of $E_2(z)=0$ on the unit circle.

$$\text{So } z_1 = \text{Cos}\theta + i\text{Sin}\theta \tag{B.12}$$

Substituting for z_1 in $E_2(z_1)=0$ and noting that

$$(\text{Cos}\theta + i\text{Sin}\theta)^n = \text{Cos}(n\theta) + i\text{Sin}(n\theta) \tag{B.13}$$

gives

$$E_2(\theta) = \text{Cos}(k+1)\theta + i\text{Sin}(k+1)\theta - (\mu+1)[\text{Cos}(k\theta) + i\text{Sin}(k\theta)] + \mu = 0 \tag{B.14}$$

Subtracting μ from both sides, taking magnitudes and rearranging terms obtains

$$\begin{aligned}&\text{Cos}^2(k+1)\theta + \text{Sin}^2(k+1)\theta + (\mu+1)^2[\text{Cos}^2(k\theta) + \text{Sin}^2(k\theta)]\\ &-2(\mu+1)\{\text{Cos}[(k+1)\theta]\text{Cos}(k\theta) + \text{Sin}[(k+1)\theta]\text{Sin}(k\theta)\} = \mu^2\end{aligned} \tag{B.15}$$

or $1 + (\mu+1)^2 - 2(\mu+1)\text{Cos}\theta = \mu^2 \Rightarrow \text{Cos}\theta = 1$

From (B.12),

$$z_1 = 1. \tag{B.16}$$

(That there are no multiple roots at z=1 is a consequence of Statement B.3).

Statement B.3 E_2 has distinct roots in and on the unit circle.

Differentiating $E_2(z)$ (Eqn. (B.9)) gives

$$E_2'(z) = (k+1)z^k - k(\mu+1)z^{k-1} \tag{B.17}$$

The roots of $E_2'(z)$ are

$$z = 0 \quad and \quad z = \frac{k(\mu+1)}{(k+1)}$$

Of these the first is not a root of $E_2(z)$ and the latter is outside the unit circle for $k \geq 2$, $\mu \geq 1$. So $E_2(z)$ has no repeated root on or in the unit circle.

Statement B.4 E_2 has k distinct roots in or on the unit circle.

Define

$$f(z) = z^{k+1} \tag{B.18}$$

and

$$g(z) = -(\mu+1)z^k + \mu \tag{B.19}$$

so that

$$E_2(z) = f(z) + g(z). \tag{B.20}$$

We consider the behavior of $|g(z)|/|f(z)|$ *just outside* the unit circle i.e. at $|z| = (1+\varepsilon)^{1/k}$, $\varepsilon>0$ and small

$$|g(z)| = |-(\mu+1)(1+\varepsilon)e^{i\theta} + \mu| \tag{B.21}$$

Since the magnitude of the sum of two vectors is greater or equal to the difference of their magnitudes,

$$\begin{aligned} |g(z)| &\geq (\mu+1)(1+\varepsilon) - \mu \\ &= 1 + (\mu+1)\varepsilon \end{aligned} \tag{B.22}$$

$$\text{So } |g(z)|/|f(z)| \geq [1+(\mu+1)\varepsilon]/(1+\varepsilon)^{(k+1)/k} \tag{B.23}$$

Denoting the RHS of the above inequality as $F(\varepsilon)$,

$$F'(\varepsilon) = \frac{(\mu+1)(1+\varepsilon)^{(k+1)/k} - [1+(\mu+1)\varepsilon][(k+1)/k](1+\varepsilon)^{1/k}}{(1+\varepsilon)^{(2k+2)/k}} \tag{B.24}$$

Now $F(0) = 1$ and $F'(0) = (\mu+1) - [(k+1)/k] > 0$. So $|g(z)|/|f(z)| \geq 1$ on the unit circle and is > 1 just outside it.

From Rouche's Theorem, $g(z)$ and $g(z)+f(z) = E_2(z)$ have the same number of roots on and in the unit circle.

Since $g(z) = -(\mu+1)z^k + \mu$ has all k roots inside the unit circle, $E_2(z)$ has k roots on or in the unit circle.

From statements 2 and 3, $E_2(z)$ has k-1 distinct roots in the unit circle. Finally, since $E_1(z)$ and $E_2(z)$ have the same roots except for z=1, we conclude using statement 1 that $E_1(z)=0$ has k-1 roots inside the unit circle and 1 real root between μ and $\mu+1$.

$$\boxed{\begin{array}{l}\text{Let the k roots be denoted } r_{1,k,m}, r_{2,k,m}, \ldots r_{k,k,m} \\ \text{where} \\ \mu < r_{k,k,m} < \mu+1 \\ \text{and} \\ |r_{i,k,m}| < 1, \; i = 1,2,\ldots,(k-1).\end{array}} \qquad (B.25)$$

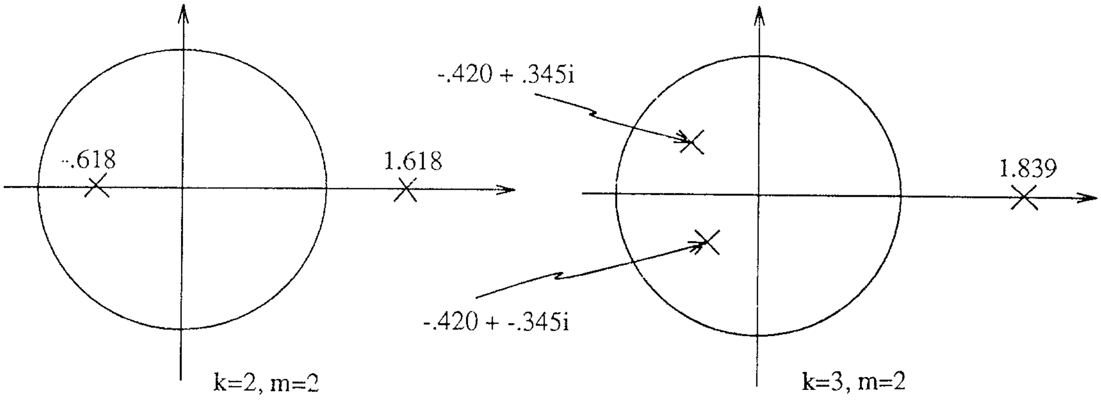


Figure B-1: Roots of the Characteristic Polynomial: k=2,3; m=2

Finally, from (B.3) and (B.7),

$$\boxed{p_{p2}'(d,n)=c_{1,k,m}r^{n}_{1,k,m}+c_{2,k,m}r^{n}_{2,k,m}+\ldots+c_{k,k,m}r^{n}_{k,k,m} \quad 1\leq d\leq n, n\geq 2.} \qquad (B.26)$$

where $c_{1,k,m}$, $c_{2,k,m}$, . . ., $c_{k,k,m}$ are constants and $r_{1,k,m}$, . . . $r_{k,k,m}$ are described in (B.25).

Fig. B-1 shows the roots for k=2,3 and m=2.

B.2.1 The real root, $r_{k,k,m}$

The root $r_{2,2,m}$ (k=2 in $E_1(z)=0$) is the solution of the quadratic equation

$$z^2 - \mu z - \mu = 0$$

which lies outside the unit circle

$$or \quad r_{2,2,k} = \frac{\mu + \sqrt{\mu^2+4\mu}}{2} \qquad (B.27)$$

In the binary case, this is the well-known golden ratio, $(1+\sqrt{5})/2$.

In the general case of k>2, the real root outside the unit circle may be obtained quite easily as follows:

Let

$$r_{k,k,m} = 1+\mu-\delta \qquad (B.28)$$

where

$$\delta<1 \qquad (B.29)$$

Upon substitution in $E_2(z) = 0$

$$(1+\mu-\delta)^{k+1} - (\mu+1)(\mu+1-\delta)^k + \mu = 0 \qquad (B.30)$$

So

$$\delta(\mu+1-\delta)^k = \mu$$

or

$$\delta = \frac{\mu}{(1+\mu-\delta)^k} \qquad (B.31)$$

At this point we note that as $k \rightarrow \infty$, $\delta \rightarrow 0$. This serves as a lower bound for δ and may be used as an initial value for δ which on substitution into the RHS of (B.31) will obtain $r_{k,k,m}$ by iteration. Values of $r_{k,k,2}$ are tabulated as a function of k (Table B-2). Note that $r_{k,k,2}$ increases monotonically with k and has $\mu+1$ as asymptote.

k	$\mathbf{r}_{k,k,2}$
2	1.618
3	1.839
4	1.927
5	1.966
6	1.984
7	1.991
8	1.996
9	1.998
10	1.999

Table B-2: $r_{k,k,2}$ as a function of k

Appendix C.

Auxilary Results for Traffic

C.1 Monotonicity of $p_{m2}(d,n)$ with respect to d, m>2

We need to prove that $p_{m2}(d,n) > P_{m2}(d-1,n)$, $2 \leq d \leq n$, in KYKLOS-II<m,2,n>, $m \geq 3$, $n \geq 1$. We first consider the case of n odd. For this purpose, we split the problem into three parts viz.

$$p_{m2}(d,n) > p_{m2}(d-1,n) \qquad 2 \leq d \leq \lfloor n/2 \rfloor \tag{C.1}$$

$$p_{m2}(d,n) > p_{m2}(d-1,n) \qquad n \geq d > \lceil n/2 \rceil \tag{C.2}$$

$$p_{m2}((n-1)/2,n) > p_{m2}((n+1)/2,n) > p_{m2}((n+3)/2,n) \tag{C.3}$$

<u>Proving Result 3-3</u>

Since $m \geq 3$,

$2(m-1)m^{d-1} > 2(m-1)m^{d-2}$

Using Theorem 3.2,

$p_{m2}(d,n) > p_{m2}(d-1,n) \qquad 2 \leq d \leq \lfloor n/2 \rfloor$

<u>Proving Result (C.2)</u>

For $m > 3$, $d \leq n$

$2 > [(m+1)/m]^2 m^{d-n}$

So $2(m-1)^2 m^{d-2} > (m-1)^2(m+1)^2 m^{2d-n-2}/m^2$

So $2(m-1)m^{d-1}-(1-1/m^2)m^{2d-n} > 2(m-1)m^{d-2}-(1-1/m^2)m^{2d-n-2}$

Using Theorem 3.2, we get

$p_{m2}(d,n) > p_{m2}(d-1,n) \quad n \geq d > \lceil n/2 \rceil$

Proving Result (C.3)

We first show that $p_{m2}((n+3)/2,n) > p_{m2}((n+1)/2,n)$, $n \geq 3$.

For $m \geq 3$,

$m^2-3m+1 > 0$

So $2(m-1)m > m^2+m-1$

For $n \geq 3$ (and odd)

$2(m-1)m^{(n-1)/2} > m^2+m-1$

So $2(m-1)m^{(n-1)/2}(m-1) > (m-1)(m^2+m-1)$

So $2(m-1)(m^{(n+1)/2}-m^{(n-1)/2}) > (1-1/m^2)m^3-(m-1)$

So $2(m-1)m^{(n+3)/2-1} - (1-1/m^2)m^{2(n+3)/2-n} > 2(m-1)m^{(n+1)/2-1}-\lfloor(1-1/m^2)m^{2(n+1)/2-n}\rfloor$

Using Theorem 3.2,

$p_{m2}((n+3)/2,n) > p_{m2}((n+1)/2,n)$

For $m \geq 3$, $n \geq 3$

$2(m-1)m^{(n-3)/2}(m-1) > (m-1)$

So $2(m-1)m^{(n-1)/2} - 2(m-1)m^{(n-3)/2} > (m-1)$

$2(m-1)m^{(n+1)/2-1}-\lfloor(1-1/m^2)m^{2(n+1)/2-n}\rfloor > 2(m-1)m^{(n-1)/2-1}$

Using Theorem 3.2,

$p_{m2}((n+1)/2,n) > p_{m2}((n-1)/2,n)$

In a similar way, it can be shown that for even n, $m \geq 3$, $p_{m2}(d,n)$ increases monotonically with respect to d so that

$$\boxed{\begin{array}{l}\text{In a KYKLOS-II}<m,2,n>,\ m\geq 3,\ n>1,\\ p_{m2}(d,n) > p_{m2}(d-1,n),\ 2\leq d\leq n\end{array}} \qquad \text{(C.4)}$$

C.2 Bitonicity of $p_{m2}(d,n)$ with respect to d, m=2

The proof for m=2 is similar except in the special case when d=n. For this case $p_{m2}(d,n) < p_{m2}(d\text{-}1,n)$ so that

$$\boxed{\begin{array}{l}\text{In a KYKLOS-II<2,2,n>, } n>2,\\ p_{m2}(d,n) > p_{m2}(d\text{-}1,n),\ 2\leq d\leq n\text{-}1\\ \text{and}\\ p_{m2}(n,n) < p_{m2}(n\text{-}1,n)\end{array}} \qquad \text{(C.5)}$$

C.3 Monotonicity of $T_{h2}(i,n)$ with respect to i, 1<=i<=(n+1)/2

For $1\leq i\leq\lceil n/2\rceil$, $m\geq 2$,

$m^{\lceil n/2\rceil} > m^{i}(m+1)/m^{2}$

So $m^{n}(m\text{-}1) > m^{\lfloor n/2\rfloor+i-2}(m^{2}\text{-}1)$

So $2m^{i-1}(m^{n}\text{-}m^{\lfloor n/2\rfloor+i-1}) > 2m^{i-2}(m^{n}\text{-}m^{\lfloor n/2\rfloor+i-2})$

Combining this with Theorem 4.4 we get,

$$\boxed{\begin{array}{l}\text{In a KYKLOS-II<m,2,n>,}\\ T_{h2}(i,n) > T_{h2}(i\text{-}1,n),\ 1\leq i\leq\lceil n/2\rceil\end{array}} \qquad \text{(C.6)}$$

Appendix D.

Reliability of a Multiple-Tree Network

Let N_s be the number of switches in the original simplex tree.

(For the binary tree, $N_s = N - 1$).

Let λ be the constant failure rate of a single switch node.

Let R_s be the reliability of a single switch node = $e^{-\lambda t}$.

(Reliability of a switch node represents reliability of its logic and connections.)

We need to compute R_{net} defined as the probability that at least one of r complete trees is operational. Using the classical approach, this reliability may be expressed as

$$R_{net} = 1 - (1-R_o)^r \qquad \text{(D.1)}$$

where R_o = Reliability of the original simplex network.

In terms of switch reliability

$$R_{net} = 1 - (1-R_s^{N_s})^r \qquad \text{(D.2)}$$

Defining the failure probability (or unreliability) of the network as

$F_{net} = 1 - R_{net}$

and failure probability of a switch as

$F_s = 1 - R_s$

enables us to rewrite (D.2) in terms of failure probabilities as

$$F_{net} = [1-(1-F_s)^{N_s}]^r \qquad \text{(D.3)}$$

Failure probability (FP) of the network is shown plotted against failure probability of the switch (Fig. D-1) for the simplex, duplex and triple-tree cases with 128 processors. Observe that for small values of F_s ($F_s << 1/N_s$), F_{net} is far more sensitive to F_s in the simplex case as compared to the duplex and triplex trees. The improvement in reliability of the multiple tree structure over the simplex tree may be obtained by examining the above equation for FP of the network.

$$\begin{aligned} F_{net} &= [1 - (1-F_s)^N]^r \\ &= [1 - (1 - N_sF_s + . . .)^N]^r \end{aligned} \qquad (D.4)$$

For small values of F_s (F_s<<$1/N_s$), this may be approximated to

$$F_{net} = (N_sF_s)^r \qquad (D.5)$$

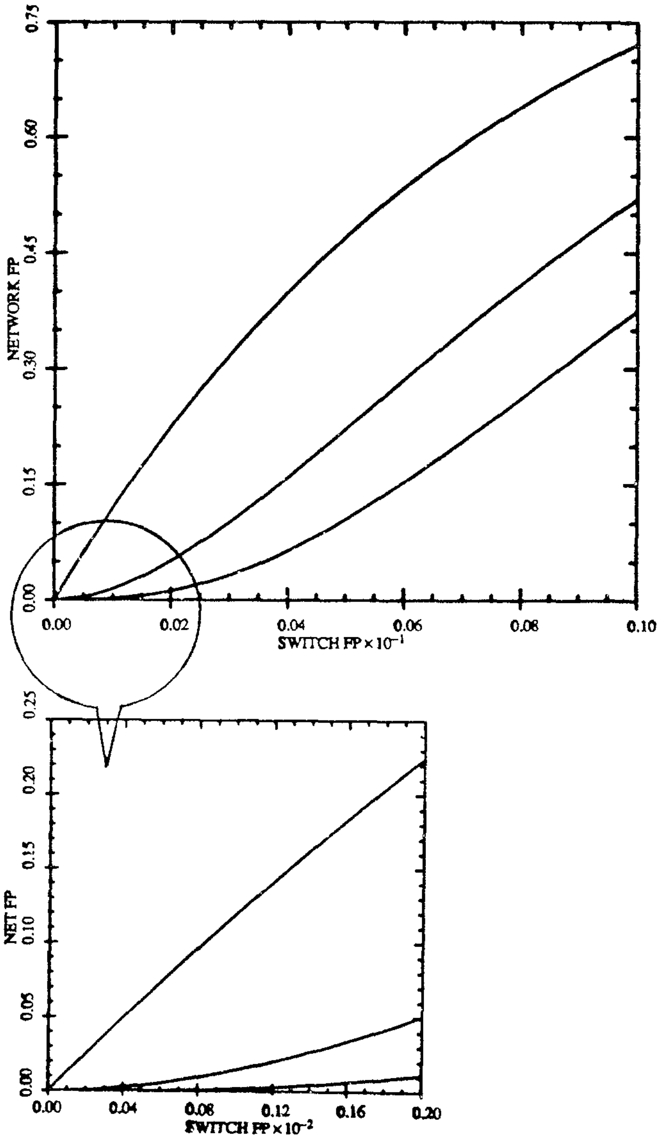


Figure D-1: Failure Probability vs. N

For example, at $F_s = .002$, F_s is .225 in the simplex tree but .050 and .011 in the duplex and triplex trees respectively. This implies that the simplex tree is more than 4 times as likely to fail as compared to the double tree and over 20 times as likely to fail as compared to the triple tree when the reliability of the switch is 0.998.

The equation for network reliability may be rewritten to show the explicit dependence of R_{net} on time as

$$R_{net} = 1 - (1 - e^{-\lambda N_s t})^r \qquad (D.6)$$

Reliability of the network (simplex, duplex and triplex cases) is plotted as a function of time for N = 128 (Fig. D-2). It is assumed that the switch failure rate is 0.1 per million hours - a fairly representative value in view of the fact that a switch node in KYKLOS has low fanout and corresponding low complexity. Note that the plots for r>1 have a low gradient during the first few thousands of hours of operation. This is in marked contrast to the simplex case where the slope is much steeper. Differentiation of the reliability expression with respect to time confirms this observation. Thus

$$\begin{aligned} dR_{net}/dt &= d[1 - (1-e^{-\lambda N t})^r]/dt && && (D.7) \\ &= -\lambda\, N\, e^{-\lambda N t}, && r=1 \\ &= -r\lambda\, N\, e^{-\lambda N t}\,(1-e^{-\lambda N t})^{r-1}, && r>1 \end{aligned}$$

This feature of the plots for multiple trees is responsible for the great improvement in mission time (MT) in KYKLOS.

Let k be the mission time reliability of the r-replica network. Substituting for R_{net} in (D.6) and solving for MT yields

$$MT = \{-\ln[1-(1-k)^{1/r}]\}/(\lambda N) \qquad (D.8)$$

For k~1, the above expression may be approximated (using the series expansion for ln(1-x), x~0) to

$$MT = (1-k)^{1/r}/(\lambda N) \qquad (D.9)$$

The mission time improvement (MTI), defined as the ratio of the MT in the r-replica case to the MT in the simplex case, is easily estimated from

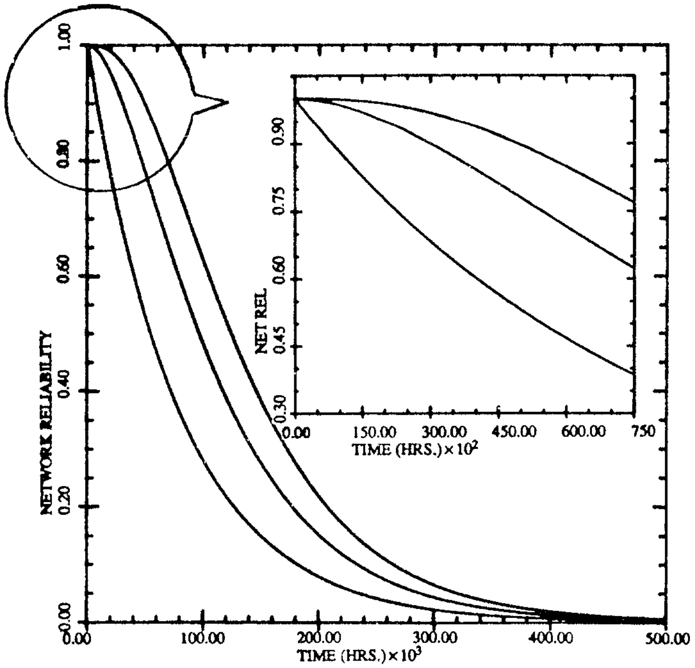


Figure D-2: Reliability vs. t

$$MTI = (1\text{-}k)^{1/r-1} \tag{D.10}$$

Fig. D-2 also shows a detailed view of R(t) during the first few years of the operation of the network. Consider an MT reliability of 0.9. While the MT for the simplex tree is about 8000 hours of operation or less than a year, that for the double tree shows close to a fourfold improvement (over 3 years) while that for the triple tree shows a sixfold improvement (nearly 6 years of continuous operation).

For some ultra-reliable applications where a high value of MTTF is at a premium, it is meaningful to compare the MTTF's of the multiple tree structure with that of the simplex tree.

Using (D.6), the MTTF for the general r-replica case may be expressed as

$$
\begin{aligned}
MTTF &= \int_0^{\infty} [1-(1-e^{-\lambda N_s t})^r]dt \\
&= \int_0^{1} (1/\lambda N_s) \ \ [(1-x^r)/(1-x)]dx \qquad where \ \ x \ = \ 1-e^{-\lambda N_s t} \\
&= (1/\lambda N_s)\int_0^{1} [1+x+x^2....+x^{r-1}]dx \\
&= (1/\lambda N_s)[1+1/2+1/3.....+1/r] \qquad (D.11)
\end{aligned}
$$

This implies that the use of a double tree increases the MTTF (graphically the area under the Reliability curve of Fig. D-2) by 50% over that of a single tree while adding a second replica has a cumulative effect of increasing the MTTF by about 83%.

Appendix E.

Bibliography

[ADAM87] G.B.Adams, D.P.Agrawal and H.J.Siegel, "A Survey and Comparison of Fault-tolerant Multistage Interconnection Networks", 20-6, *Computer*, May 1982, pp. 14-27.

[AGRA86] D.P.Agrawal, V.K.Janakiram and G.C.Pathak, "Evaluating the Performance of Multicomputer Configurations", 19-5, *Computer*, May 1986, pp. 23-37.

[AKER65] S.B.Akers, Jr. "On the construction of (d,k) graphs," *IEEE Transactions on Electronic Computers,* EC-14, June 1965, pp. 488.

[ARDE78] B.W.Arden and H.Lee, "A Multi-Tree Structured Network", *Proceedings COMPCON 78,* Sept. 1978, pp. 201-210.

[BATC76] K.E.Batcher, "The Flip Network in STARAN," *Proceedings of the 1976 International Conference on Parallel Processing,* Aug. 1976, pp. 65-71.

[BENE62] V.E. Benes, "On Rearrangeable Three-Stage Connecting Networks," *Bell System Technical Journal*, 41, Sept. 1962, pp. 1481-1492.

[BENE65] V.E.Benes, *Mathematical Theory of Connecting Networks and Telephone Traffic,* Academic Press, 1965, 319 pp.

[BROW79] S.A.Browning, "Computations on a tree of processors," *Proceedings VLSI Conference,* California Institute of Technology, Pasadena, Jan 22-24, 1979.

[BROW85] J.C.Browne, A.G.Dale, C.Leung and R.M.Jenevein, "A Parallel Multi-stage I/O Architecture with a Self-managing Disk Cache for Database Management Applications," *Proceedings of the Fourth International Workshop on Database Machines*, March 1985, pp. 330-345.

[CHER84] V.Cherkassky and M.Malek, "Reliability and Fail-softness Analysis of Multistage Interconnection Networks", *IEEE Transactions on Reliability*, R-34, #5, Dec. 1985, pp. 524-528.

[CLOS53] C.Clos, "A Study of Nonblocking Switching Networks", *Bell System Technical Journal*, 32, March 1953, pp. 406-424.

[DeME86] J.DeMelo and R.M.Jenevein, "SK-Banyans: A Unified Class of Banyan Networks", *Proceedings of the 1986 International Conference on Parallel Processing*, Aug. 1986, pp. 100-107.

[DESH86] Deshpande, S.R. and R.M.Jenevein, "Scaleability of a Binary Tree on a Hypercube", *Proceedings of the 1986 International Conference on Parallel Processing*, Aug. 1986, pp. 661-668.

[DESP78] A.M.Despain and D.A.Patterson, "X-Tree: A Tree Structured Multi-processor Computer Architecture", *Proceedings of the Fifth International Symposium on Computer Architecture*, April 1978, pp. 144-151.

[DeWI87] DeWitt, D.J, M.Smith, H.Boral, "A Single-User Performance Utilization of the Teradata Database Machine" *MCC Technical Report # DB-081-87*, March 5, 1987.

[DOTY84] K.W.Doty, "New Designs for Dense Processor Interconnection Networks," *IEEE Transactions on Computers*, C-33, #5, May 1984, pp. 447-450.

[EHRE84] M.J.Ehrensberger, "The DBC/102 Data Base Computer's System-Architecture, Components, and Performance" *Paper presented at the Minnowbrook Workshop on Database Machines*, 1984.

[ELSP64] B.Elspas, "Topological constraints on interconnection limited logic," *Switching Circuit Theory and Logical Design,* vol. s.164, Oct. 1964, pp. 133-147.

[ERDO66] P.Erdos, A.Renyi and V.T.Sos, "On a Problem of Graph Theory," *Stud. Sci. Math. Hungarica,* Vol. 1, 1966, pp. 215-235.

[FENG74] T.Feng, "Data Manipulating Functions in Parallel Processors and their Implementations," *IEEE Transactions on Computers,* Vol C-23, #3, March 1974, pp. 309-318.

[FENG81] T.Feng, "A Survey of Interconnection Networks," *Computer,* Vol. 14, Dec. 1981, pp. 12-27.

[FLOR67] Flores, I., "Direct Calculation of k-generalized Fibonacci Numbers", *Fibonacci Quarterly*, Vol 5, 1967, pp. 259-266.

[GARD83] Gardarin, G., et al., "SABRE: A Relational Database System for a Multimicroprocessor Machine," *Advanced Database Machine Architecture*, D. Hsiao (ed), Prentice-Hall, 1983.

[GODS83] C.D.Godsil and R.Razen, "A Property of Fibonacci and Tribonacci Numbers," *The Fibonacci Quarterly,* Vol. 21, #1, Feb. 1983, pp. 13-17.

[GOKE73] L.R.Goke and G.J.Lipovski,"Banyan Networks for Partitioning Multiprocessor Systems", *Proceedings of the 1st Annual Symposium on Computer Architecture,* Dec. 1973, pp. 21-28.

[GOOD81] J.R.Goodman and C.H.Sequin, "Hypertree: A Multiprocessor Interconnection Topology," *IEEE Transactions on Computers*, C-30, Dec. 1981, pp. 923-933.

[GREY84] B.O.A.Grey, A.Avizienis and D.A.Rennels,"A Fault-Tolerant Architecture for Network Storage Systems," *Proceedings of the 14th International Conference on Fault-Tolerant Computing*, May 1984, pp. 232-239.

[HAYE76] J.P.Hayes, "A Graph Model for Fault-Tolerant Computing Systems," *IEEE Transactions on Computers,* C-25, Sept. 1976, pp. 875-884.

[IMAI84] M.Imai, et al., "The Architecture and Efficiency of DON: A Combinatorial Problem Oriented Multicomputer System," *Proceedings of the 4th International Conference on Distributed Computing Systems,* pp. 174-182, May 1974.

[IMAS81] M.Imase and M.Itoh, "Design to minimize diameter on building-block network," *IEEE Transactions on Computers,* vol. C-30, June 1981, pp. 439-442.

[JENE86] R.Jenevein and B.Menezes, "KYKLOS: Low Tide High Flow" *Proceedings of the Sixth International Conference on Distributed Computing*, pp. 8-15, May 1986.

[LAWR75] D.H.Lawrie, "Access and Alignment of Data in an Array Processor," *IEEE Transactions on Computers,* C-24, Dec. 1975, pp. 1145-1155.

[LEIS85] C.E. Leiserson, "FAT-TREES: Universal Networks for Hardware-Efficient Supercomputing", *Proceedings of the 1985 International Conference on Parallel Processing*, Aug. 1985, pp. 393-402.

[LIPO82] G.J.Lipovski, A.Goyal and M.Malek, "Lookahead Networks," *Proceedings of AFIPS,* v.51, June 1982, pp. 153-166.

[LIPO87] G.J.Lipovski and M.Malek, *Parallel Computers: Theory and Comparisons,* John Wiley and Sons, Inc., 1987.

[MEMM82] G.Memmi and Y.Raillard, "Some New Results about the (d,k) Graph Problem," *IEEE Transactions on Computers,* C-31, Aug. 1982, pp. 784-791.

[MENE85a] B.L.Menezes and R.M.Jenevein, "Mathematical Foundations of KYKLOS", Tech. Report TRAC-32, Dept. ECE., Univ. of Texas at Austin.

[MENE85b] B.L.Menezes and R.M.Jenevein, "KYKLOS: A Linear growth Fault-tolerant Interconnection Network," *Proceedings of the International Conference on Parallel Processing,* Aug. 1985, pp. 498-502.

[MENE85c] B.L.Menezes, "Properties and Potential of the KYKLOS Interconnection Network," *Term Report in Interconnection Networks Course*, Department of Electrical and Computer Engineering, the University of Texas at Austin, Dec. 1985.

[MENE86a] B.L.Menezes, R.Jenevein and M.Malek, "Reliability Analysis of the KYKLOS Interconnection Network," *Proceedings of the Sixth International Conference on Distributed Computing*, May 1986, pp. 46-51.

[MENE86b] B.L.Menezes, "Managing Combinatorial Explosions in a Family of Interconnection Networks," *International Computer Symposium*, December 1986.

[MENE87a] B.L.Menezes, K.Thadani, A.Dale and R.Jenevein, "Design of a HyperKYKLOS-based Multiprocessor Architecture for High-Performance Join Operations", *Dept. of Computer Sciences, Technical Report TR-87-18*, the University of Texas at Austin, Austin, Tx., May 1987.

[MENE87b] B.L.Menezes, "An Interconnection Network Supporting Relational Join Operations," *Proceedings of the Seventh International Conference on Distributed Computing*, September 1987, pp. 128-135.

[MILES67] Miles, E.P., "Generalized Fibonacci Numbers and Associated Matrices", *American Mathematical Monthly*, Vol 67, 1960, pp. 745-752.

[PRAD82] D.K.Pradhan, "On a Class: Fault-Tolerant Multiprocessor Network Architectures," *Proceedings of the 3rd International Conference on Distributed Computing Systems*, 1982, pp. 302-311.

[PREP81] F.P.Preparata and J.Vuillemin, "The Cube-Connected Cycle: A Versatile Network for Parallel Computation," *Communications of ACM*, 24, May 1981, pp. 300-309.

[RAGH84] C.S.Raghavendra, A.Avizienis and M.Ercegov, "Fault-Tolerance in Binary Tree Architectures", *IEEE Transactions on Computers,* vol. C-33, June 1984, pp. 569-572.

[REED87] D.A.Reed and D.C.Grunwald, "The Performance of Multicomputer Networks," *Computer,* 20-6, June 1987, pp. 63-73.

[SHAW79] D.E.Shaw, "A Hierarchical Associative Architecture for the Parallel Evaluation of Relational Algebraic Database Primitives" *Dept. of Computer Science, Technical Report STAN-CS-79-778*, Stanford University, Stanford, Ca., 1979.

[SHIB84] S.Shibayama, "A Relational Database Machine with large Semiconductor Disk and Hardware Relational Algebra Processor," *New Generation Computing,* No. 2, 1984, pp. 131-155.

[SIEG81] H.J.Siegel and R.J.McMillen, "The Multistage Cube: A Versatile Interconnection Network," *Computer,* Vol. 14, Dec. 1981, pp. 65-76.

[SONG80] S.W.Song, "On a High-Performance VLSI Solution to Database Problems", *Ph.D. Thesis,* Department of Computer Science, Carnegie Mellon University, Pittsburgh, Pa., 1980.

[STON71] H.S.Stone, "Parallel Processing with the Perfect Shuffle," *IEEE Transactions on Computers,* C-20, Feb. 1971, pp. 153-161.

[STOR70] R.M.Storwick, "Improved Construction Techniques for (d,k) graphs," *IEEE Transactions on Computers,* C-19, Dec. 1970, pp. 1214-1216.

[TANA84] Tanaka, Y., "MPDC: Massive Parallel Architecture for Very Large Databases," *Proceedings of the International Conference on Fifth Generation Computer Systems*, 1984, pp. 113-137.

[TOUE79] S.Toueg and K.Steiglitz, "The design of small-diameter networks by local search," *IEEE Transactions on Computers,* C-28, July 1979, pp. 537-542.

[ULLM84] J.D.Ullman, *Computational Aspects of VLSI,* Computer Science Press, Rockville, Md., 1984.

[WILK70] R.S.Wilkov, "Construction of maximally reliable communication networks with minimum transmission delay," *Proceedings IEEE International Conference on Communication,* vol 6, June 1970, pp. 4210-4215.

[WU80a] C.Wu and T.Feng, "On a Class of Multistage Interconnection Networks," *IEEE Transactions on Computers,* C-28, Aug. 1980, pp. 694-702.

[WU80b] C.Wu and T.Feng, "The Reverse-Exchange Interconnection Network", *IEEE Transactions on Computers,* C-29, Sept. 1980, pp. 801-811.

[WU84] C.Wu and T.Feng, *Tutorial: Interconnection Networks for Parallel and Distributed Processing,* IEEE Computer Society Press, 1984, 647 pp.

VITA

Bernard L. Menezes was born in Bombay, India, on February 19, 1958, the son of Marie Esther Menezes and Frank Julio Menezes. After completing his work in St. Xavier's High School, Bombay, India, in 1974, he entered Jai Hind College, Bombay, where he completed his F.Y.Sc. He entered the Indian Institute of Technolgy, Bombay, in 1975 and graduated with a Bachelor of Technology in Electrical Engineering in 1980. He entered the University of Notre Dame, Indiana, in August, 1980 and was awarded the degree of Master of Science in Electrical Engineering in December, 1983. In January 1983, he entered the graduate school of the University of Texas at Austin.

Permanent Address: 301 Beach View
77 Chimbai Rd.
Bandra, Bombay 400050
India.

This dissertation was typed by Bernard L. Menezes