Abstract

Interconnection Network Design Based on Packaging Considerations

by

Mandayam Thondanur Raghunath

Doctor of Philosophy in Computer Science

University of California at Berkeley

Professor Abhiram G. Ranade, Chair

An important problem in building large scale parallel computers is the design of the interconnection network. The network affects the performance, cost, scalability, and availability of the parallel computer. In this dissertation, we examine in detail the problem of interconnection network design for a large scale parallel machine. Our objective is to design networks that achieve a high performance for a low cost.

In general, the cost of a network is a complex function of a wide variety of parameters and is difficult to compute. We first develop a packaging model to characterize network costs. Our model is based on the fact that most large scale machines have to be packaged in a hierarchical fashion. We argue for a hierarchical design strategy, where the design of the network proceeds in levels corresponding to the levels of the packaging hierarchy. We evaluate several network designs using analysis and detailed simulations of random traffic. We identify families of networks (product of complete graphs, high-degree de Bruijn networks) that we believe to be useful for multi-level packaging technologies. Our results also indicate that making the networks denser at the lower levels of the packaging hierarchy has a significant positive impact on the overall performance, even when the higher levels use a sparser interconnect.

We also characterize network designs in terms of scalability, dividing network families into three broad categories of scalability based on the flexibility with which the networks can be scaled. The three categories illustrate the trade-offs between hardware costs and scaling flexibility. We provide quantitative evaluations of these trade-offs.

Many of the interesting networks, namely those that provide high performance for a low cost, also have the disadvantage of being vulnerable to faults because they have a single path between any pair of processors. We devise a scheme to tolerate faults in such networks. The scheme is simple to implement and allows the network to tolerate a large number of faults with a slight degradation in performance.

Professor Abhiram G. Ranade
Dissertation Committee Chair

Acknowledgements

It is a pleasure to acknowledge the years of guidance, support, and encouragement that I received from my thesis advisor, Abhiram Ranade. I am extremely grateful to Abhiram for providing my research with a sense of direction and helping me develop a better understanding of the research issues involved. I am also grateful to him for offering me sufficient challenge and carefully evaluating my work.

I would also like to express my gratitude to David Culler and George Shantikumar for reading through this lengthy dissertation. Their detailed comments helped improve the content and presentation of this thesis.

I am also grateful to all the instructors of the various courses I have attended at U. C. Berkeley and at I. I. T. Madras, for enabling me to acquire sufficient knowledge to carry out independent research. U. C. Berkeley, gave me the opportunity to take a wide range of classes, not just in Computer Science, but also in Sanskrit, Music, etc.

I would also like to thank Howard Ho, and many others at the IBM Almaden Research Center, for comments and suggestions relating to my research.

Many of my fellow graduate students at Berkeley have helped me academically, and also in making my stay at Berkeley enjoyable and fun. I would especially like to thank Bob Boothe, Jeff Rothman, and Brian O'Krafka for many technical discussions relating to my thesis. Besides helping me develop and evaluate research ideas, and commenting on drafts of papers, Bob Boothe and Jeff Rothman helped relieve the occasional monotony of work and moments of depression or cynicism with interesting discussions on a wide variety of topics such as vegetarianism, US foreign policy, sports, etc. I also wish to thank the numerous friends I developed while at Berkeley; my senior students who helped me settle down and get to know Berkeley, my roommates and others who braved my culinary attempts, and many others who joined me in various activities; games, movies, concerts, etc.

I would also like to thank Kathryn Crabtree and the other members of the administrative staff of the CS division for making the rather difficult bureaucracy of the university administration appear friendly and cordial.

Finally, I also wish to thank my parents for bringing me up with love and affection, for giving me a strong sense of determination and self-esteem, and for providing me with the best opportunity in education.

This research was supported in part by the Air Force Office of Scientific Research (AFOSR/JSEP) under contract F49620-90-C-0029, National Science Foundation Grant Number CCR-9005448, and a fellowship from IBM Corp. Computing resources were provided by NSF Infrastructure Grant number CDA-8722788. Their support is gratefully acknowledged.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Rapid technological advances and increasing demands for computer power have ushered in large scale parallelism. Several machines consisting of large numbers of processors have been built, either commercially or as research vehicles [GGK+83, Hil85, PBG+85, BBN86, H+86, Int86, LT88, A+90, R+90, Nic90, ND90, L+91, TMC91, Ken92, FIR93, DKN93, KS93]. Large scale parallelism has the potential for providing quantum jumps in computing power; complex computational problems that were once thought to be intractable, now seem feasible. However, there are a number of open research issues that need to be solved before the power of large scale parallelism can be conveniently applied to solve general problems.

One central issue is the design of the interconnection network that handles the communication between processors. Since parallel programs typically exchange large amounts of data between processors during the course of their execution, substantial design effort is usually directed towards the design of the interconnection network. The network takes up a significant fraction of the total cost; is often the hardest part of the system to engineer and for many applications determines the final performance.

The design of the interconnection network is also important from the point of view of making parallel machines easy to program. Parallel machines are generally regarded as being difficult to program and this difficulty stems from a variety of factors, one of which is the time spent in communication. When communication is slow, the programmer has to expend a great deal of effort in carefully partitioning the data among processors so as to reduce the amount of data movement. By designing a fast communication network that reduces the time spent in communication, we can simplify this aspect of parallel programming.

Fast interconnection networks also permit the programmer to use higher level programming models which provide programming simplicity at the cost of increased communication.

Since the network is a key aspect of a parallel machine, the problem of designing efficient interconnection networks has received much attention from researchers. However there is no apparent consensus: each of the different parallel machines currently on the market has a completely different interconnection mechanism. For instance, while the CM - 5 uses a fat-tree topology, the Intel Paragon uses a two dimensional mesh and the KSR - 1 uses a ring-of-rings interconnect. Further, the machines recently announced by both Thinking Machines and Intel have an entirely different network organization compared to their previous models. In both cases, they have abandoned the hypercube network topology in favor of the above mentioned topologies.

The lack of consensus clearly indicates that designing efficient networks is still an open problem. Some of the reasons for this lack of consensus are as follows:

1. *Modeling network cost:* In general, modeling the cost of a network is a difficult problem. Existing models of network cost (discussed in section 1.5) are often inadequate to model the complex costs associated with multi-level packaging hierarchies that most large scale network designs are forced to contend with. In the absence of a good way to model network costs, it is difficult to evaluate the cost-performance trade-offs of different network designs. While one design may promise greater performance than another (on the basis of simulation studies, for example), it is difficult to determine their relative costs until they are actually built. As a result, it is difficult to arrive at a consensus on whether one network is better than another.

2. *Traffic patterns:* We do not yet have a clear understanding of the characteristics of the communication traffic seen in parallel machines. Some applications (e.g. matrix multiplication) have communication traffic patterns that are extremely regular, while others (e.g. graph algorithms) have patterns that display no such regularity and may also vary with time. There is some disagreement on whether network designs should be optimized for regular traffic patterns. While some may argue that regularity in communication traffic is indeed the common case, and should be speeded up, others argue in favor of presenting the programmer with a uniform view of the network where any processor can send data to any other processor, with equal ease.

3. *Scalability:* Parallel machine manufacturers usually want to be able to supply machines of different sizes, because customers have different computing requirements. It is common for manufacturers to target a wide range of systems, starting from low-cost entry-level systems, all the way to a high-end system consisting of thousands of processors. It is also desirable to provide customers with the ability to incrementally upgrade their machines. Therefore, networks must be designed for *scalability*. However, there is no consensus on a formal definition of scalability, and there is no definitive method of characterizing network designs in terms of scalability. While some manufacturers may abandon a design because they consider it to be unscalable, others may regard it as being scalable.

4. *Fault-Tolerance:* Large scale networks typically consist of thousands of components and it is useful to build networks that are resilient to faults in at least a small fraction of their components. The extent of fault-tolerance required, depends on the reliability of the individual components, the time required to service faulty components, and the desired reliability of the entire machine. In many cases, neither of these three aspects is easy to quantify. There are also several different approaches to achieving fault-tolerance. One approach to providing fault-tolerance is to augment the network with extra components that are brought into use when faults occur. In a scheme of this type, the stand-by components are used to replace the faulty-ones, effectively retaining the structure and performance of a fault-free network. Another approach is to re-route messages around faulty components instead of replacing them. While the first approach usually implies an increase in cost without degradation in performance, the second approach might be cheaper but suffers performance degradation when faults occur. It is not clear as to which of these two approaches is better because the trade-offs between fault-tolerance, cost, and performance are not well understood.

In this dissertation we attempt to clarify some of the above issues and seek to design efficient networks for large scale parallel machines. By "large scale" we mean a machine size of around 1000 uniprocessors, with each processor as powerful as a state-of-the-art work-station.

The set of questions that come up while designing networks range from the very high-level to the very mundane, but many of these questions can make or break a network design. One of the important high-level questions we are faced with is: "What topology should we use?". For example, we may have a choice between a two dimensional mesh and a butterfly. We may consider the butterfly to be better because the worst case distance between the processors is shorter in the butterfly. However, the mesh has multiple paths between any pair of processors and may have better fault-tolerance properties. It is not clear which of the two topologies should be chosen. It is also conceivable that there may exist other topologies that have the advantages of both the mesh and the butterfly, and therefore, are better than both of them.

Even after we resolve the high level questions, we are still left with a number of lower level details. For example, let us consider the case where we have decided to build a network with processors on one side connected to memory units on the other using a log-depth multi-stage switching network with some specific communication bandwidth. Even now, there are a number of unresolved questions. Should we use a simple butterfly as shown in figure 1.1(a) or a dilated butterfly as shown in figure 1.1(b)? Since both networks are required to have the same communication bandwidth, the widths of channels in the dilated butterfly should be half the widths of the channels of the simple butterfly. Both networks have nominally the same bandwidth and very similar costs since both networks have the same number of wires going between any pair of nodes. How do we discriminate between the two networks? Another possibility is shown in figure 1.1(c). In this network we have a butterfly with half the number of rows, and we have connected two processors to each input of the butterfly using a tree of height 1. In order for this network to have the same bandwidth as the two previous networks, the channel widths in this network have to be double that of the simple butterfly. How does this network compare with the ones shown in figure 1.1(a) and (b)? How do these three compare with the network shown in figure 1.1(d), a butterfly of higher radix, and fewer stages of switching? We may also consider yet another possibility shown in figure 1.1(e). Here we use fewer memory units than processors. To achieve the same bandwidth as the other networks, we have to increase the channel widths to the memory. How does this network compare with the others?

The answers to many of these questions are more complex than they appear superficially because networks must be packaged in a hierarchical fashion, and we may be faced with a similar set of questions at each level of the packaging hierarchy. We are also faced with the question of whether it is advantageous to use different network architectures at different levels of the packaging hierarchy. In other words, are hybrid network architectures better than uniform ones? For example, is there any benefit to using different channel

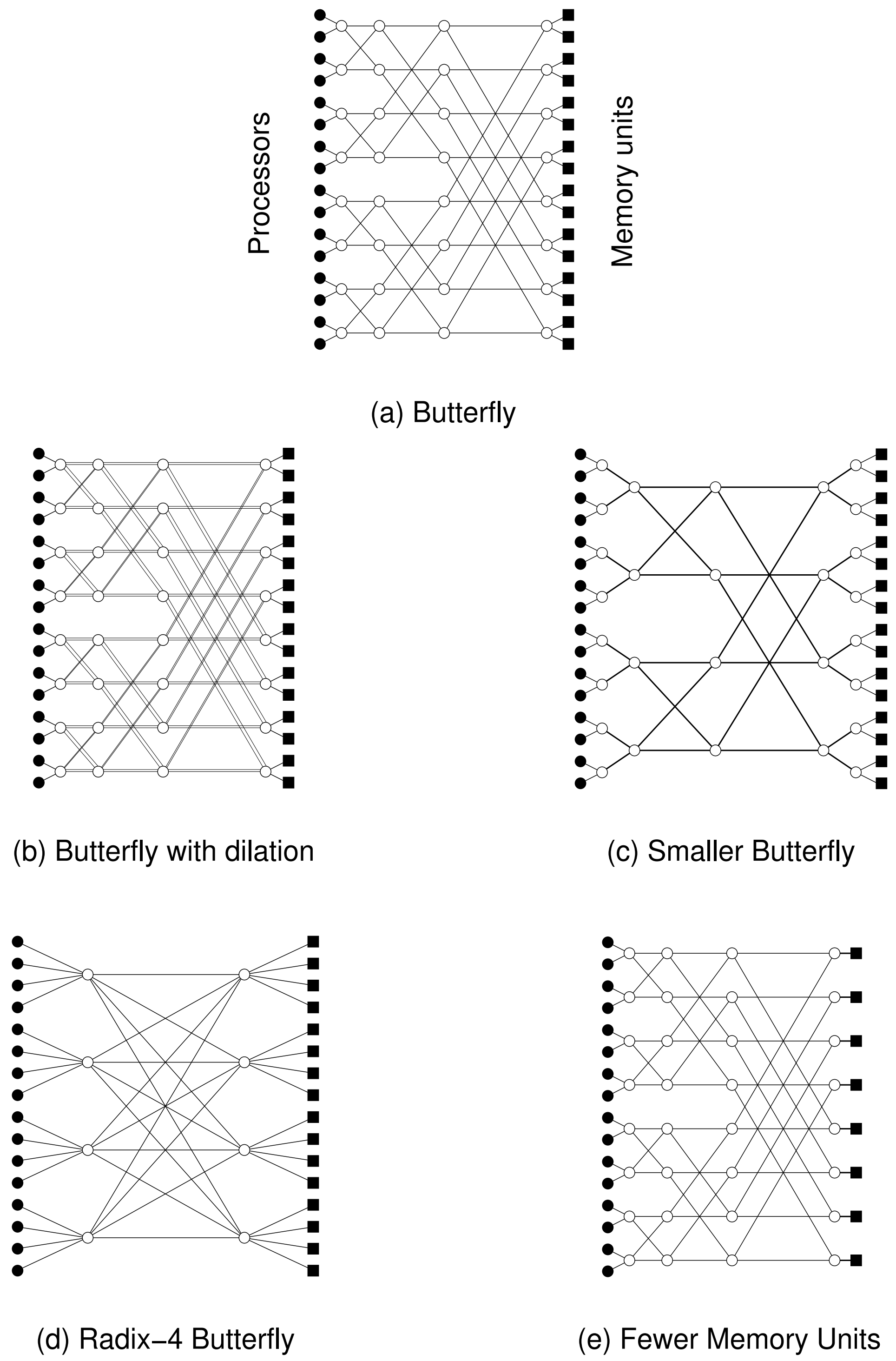

Figure 1.1: Butterfly Choices

widths at different levels of the hierarchy?

The objective of this dissertation is to develop a systematic method for dealing with questions of the type raised above. We believe that many of these decisions are currently made in a somewhat ad-hoc manner. Instead, we would like to formalize the decision process. We would like develop a systematic way of exploring the search space of design choices to enable us to pick out the promising ones while discarding the uninteresting choices. In order to motivate the exploration of the search space, we shall pursue the goal of designing the *best* network. We define the *best* network to be one that achieves the maximum performance for a fixed cost, or equivalently one that achieves a fixed performance at minimum cost. It is also important for the *best* network to possess good scalability and fault-tolerance properties.

Although trying to design the *best* network is a good way to drive the design process, we shall see that it is not always possible to obtain a design that can be proved to be the best. In some cases, we shall be able to prove that a particular design is optimal, but in many other cases we shall only be able to demonstrate that one set of designs is better than all other designs, thereby implying that the best network belongs to this set. Determining which element of the set is the best, will require knowing more details of the packaging technology than we are willing to consider as part of this thesis.

In this chapter we consider the above issues in greater detail (Sections 1.1, 1.2, 1.3 and 1.4), survey some of the related work (Section 1.5), list the main contributions of this dissertation(Section 1.6), and finally conclude with an overview of the remaining chapters.

Part of the work described in this thesis has already been presented in [RR90] and [RR93].

1.1 Modeling Network Cost

Modeling the cost of a large scale network is a difficult problem. The cost of such a network depends on the amounts of various packaging resources used to build the network. Since large scale networks are typically packaged in a hierarchical fashion, we have to take into account a large number of different packaging resources at each level of the packaging hierarchy in order to accurately model network costs. For example, under current packaging technology, a typical hierarchy would consist of chips, boards, and racks. The network is first partitioned so that it fits into some number of racks with wires going between them.

The contents of a rack are further partitioned into smaller units of the packaging hierarchy. To compute the cost of the network, we have to take into account many resources, such as, the number of racks, the number of wires that go between racks, number of wires connecting to a board, the pin-counts of the chips, etc.

Existing cost models deal primarily with the lowest levels of the hierarchy, and a better model is required to characterize multi-level packaging hierarchies. We develop a model of packaging technology that uses a number of parameters to explicitly characterize the constraints and costs associated with multi-level packaging hierarchies. The parameters of our model can be used to specify limits on pin-counts, bounds on layout areas, cabling restrictions, etc.

Although the model is general enough to characterize a wide range of current and future packaging technologies, we choose not to base our network designs on a specific packaging technology, because packaging technology continues to evolve with time; with new packaging methods being discovered, costs of existing technologies being driven down by economies of scale, etc. Instead we define three generic models, that capture fundamental technology trends that are likely to be valid even in the future, and study various network designs based on these generic models.

The purpose of defining generic models, is to obtain results that are likely to apply to a wide range of packaging technologies instead of being specific to just one technology. In our generic models, we leave some of the parameters unspecified; by appropriately fixing these parameters, the generic model can be used to characterize a variety of real technologies. Under each generic model, we broadly classify networks into *interesting* and *uninteresting* ones. A network is considered *interesting* if it is better than all other networks for *some* instantiation of the unspecified parameters. A network is *uninteresting* if there are other networks that are better, for *all* instantiations of the unspecified parameters. We provide detailed evaluations of the *interesting* networks.

1.2 Traffic Patterns

As mentioned earlier, applications can be broadly classified into two categories: those that have regular communication patterns and those that do not. When the communication patterns of an application are regular, it may be possible to transform this regularity into *network locality*, where the messages predominantly travel between processors that are proximal in the network topology. Assuming that all the communication traffic in the network has network locality can usually simplify the task of designing efficient networks. However, there are a number of reasons why assuming network locality may not always be justified.

First, a large number of applications have communication patterns that are either irregular or keep changing with time, and for such applications achieving network locality is either impossible or overwhelmingly difficult. Second, for many applications, algorithmic optimizations that reduce the running time, generally destroy any regularity in communication patterns that was present before the optimizations were performed. Third, when parallel machines are used to solve problems that are much larger than the number of processors (as is usually the case), it is difficult to achieve network locality. Fourth, exploiting network locality usually requires either the programmer or the compiler to know the precise manner in which the processors are interconnected, and this may not always be desired. Fifth, many high-level programming constructs attempt to hide the details of the interconnection network and usually rely on the ability to communicate data from any processor to any other processor. Finally, the presence of network faults generally implies that the physical interconnect of a machine changes with time, making it difficult to write programs that exploit network locality.

For all of these reasons, we shall assume that the communication traffic pattern is not characterized by network locality. It is common practice to model such traffic using randomly distributed communication traffic. In this dissertation, we make a similar assumption in our evaluations of network performance. We evaluate networks using simulation, and for the purposes of simulation, we assume a shared-memory model, but we believe that our results are equally applicable to other models of inter-processor communication where the messages are evenly distributed over the entire network.

1.3 Scalability

Scalability is a property that is often used to characterize interconnection networks, but is not very well understood [Hil90, NA91]. There is considerable disagreement about which networks are considered scalable and which ones are not. One of the reasons that prompted some manufacturers to move from the hypercube topology to a two or three dimensional mesh for interconnecting processors, was the opinion that hypercubes were not scalable because the degree of each node was not constant. Constant node degree is one of the desirable properties from the point of view of scalability, because we can easily build networks of various sizes using the same nodes. On the other hand, it is not clear that meshes are scalable either, because their bisection bandwidth does not grow linearly in the number of processors. The bisection bandwidth of a machine is a rough estimator of its performance and we usually want network performance to scale linearly in the size of the machine. In the absence of a formal definition of scalability, it is difficult to decide whether a particular network design is scalable or unscalable.

In this dissertation we provide a formal definition of scalability, in terms of how the performance and the cost of a network vary as we increase the number of processors. Briefly, we consider a network to be scalable if its performance meets a certain lower bound, and its cost is lower than a certain upper bound, over a specified range of machine sizes.

We also classify networks into three broad scalability categories depending on the fraction of the machine that remains unaltered as the size of the machine is changed. The larger this fraction, the greater the flexibility in scaling the machine. We analyze some of the network designs studied in this dissertation in terms of scalability, and demonstrate how to extend these designs as per each of the three broad scalability categories. We shall see that greater scaling flexibility is usually accompanied by lower performance or greater cost. We provide a detailed analysis and quantification of the trade-offs between scalability, cost and performance in Chapter 7.

1.4 Fault-Tolerance

Designing networks with the ability to tolerate faulty components is becoming increasingly important as the networks increase in size and complexity. There is an extensive amount of work in the area of fault-tolerant network design. A large fraction of this work is concerned with augmenting the network by adding extra components that are basically idle until failures occur. We believe that this approach is wasteful of resources, primarily because failures are usually rare (but not non-existent). Hardware components generally tend to be fairly reliable after the initial developmental errors are fixed and the common case is for networks to be either fault-free or have a small number of faults.

We believe that a better approach to fault-tolerance is to devise mechanisms to use the existing network to route messages around faulty components. This approach results in a reduction in performance when faults occur, since messages share fewer pathways in the network. However, if the number of faults is small, and the fault-tolerance scheme is efficient, the performance impact can be maintained at a minimal level. We present one such fault-tolerance scheme in this dissertation.

Most of the networks considered in this thesis have a single path from any input to any output. When a path is affected by a fault, we send messages using two passes over the network. In the first pass, the message travels from the source processor to a relay processor and in the second pass the relay processor sends the message to its appropriate destination. We select the relay processors in a manner that minimizes the performance impact of the faults.

1.5 Related Work

A major portion of this dissertation is concerned with modeling the cost of networks and designing efficient networks under our models of network cost. Accordingly, we classify the existing related research into two broad areas: cost modeling and network design. Related work on fault-tolerance is presented in Chapter 8, along with our proposed fault-tolerance scheme.

1.5.1 Cost Models

The two cost models that have been formally defined and theoretically explored in detail are the VLSI grid model and the Pin-requirements model. In the VLSI grid model, the layout area of a network is used as the metric of the network cost. This model was developed by Thompson [Tho80] primarily for circuit layouts in VLSI. In this model, layouts are assumed to conform to a regular grid consisting of unit squares. Each unit square contains either a node (defined as a transistor or a small cluster of transistors), or a single wire, or a wire crossing. Since the model deals with VLSI layouts, a natural cost metric under this model is the number of unit squares (= area) required by a layout of the network. As opposed to the grid model, the Pin-requirements model focuses on I/O limitations rather than on area. In the Pin-requirements model [Cyp90], the network is assumed to be partitioned into some number of chips. In this model, the maximum pin-count per chip is used as the cost metric.

We define three generic models (Chapter 2) that explicitly characterize packaging hierarchies to different levels of detail using cost functions that are more comprehensive. The first of our generic models is similar to the Pin-requirements model, but is applied at the top level of the packaging hierarchy. The other two generic models are more detailed, and account for other packaging resources. Since our models are more representative of the costs in the presence of packaging hierarchies, we believe that results have a greater applicability than the results under the two previous models.

1.5.2 Designing Networks

In this thesis, we design networks that achieve high performance for a low cost as specified by our cost model. There have been similar design studies under the two cost models mentioned above. In many of the existing design studies, performance is measured under the assumption of randomly distributed communication traffic. A similar assumption is made in this thesis.

Lower bounds

One way of tackling the problem of designing efficient networks is to prove theoretical lower bounds on the cost or the ratio of cost to performance. Then, if we are able to demonstrate that a network is close to the lower bound, we can claim that this network is one of the *better* networks.

Some of the initial work under the VLSI model and the Pin-requirements model consisted of proving such lower bounds. For example, under the VLSI grid model, Thompson [Tho80] showed that the area of a network is lower bounded by the square of its bisection width[1]. Other researchers have proved tighter lower bounds on the area of various network topologies [Lei82, BL84, Lei84, Ull84]. Since the cost of a network and its area are synonymous in this model, these lower bounds are essentially lower bounds on network cost. In addition, Thompson also proved lower bounds of the form AT^2, where A is the area of the chip, and T is the time taken to solve a specific problem (e.g. sorting). These bounds represent a trade-off between the cost of the chip and its performance (represented by $1/T$). Similarly, in the Pin-requirements model, Cypher [Cyp90] proved lower bounds on Q, the

[1]The bisection width of a network is the minimum number of wires that need to be cut in order to divide the network into two equal parts.

maximum number of pins per chip (= cost) for a number of commonly used network topologies. He also proved lower bounds on the product $Q * C * T$, where C is the number of chips and T is the time taken to solve a specific problem.

Universal Networks

Another approach to designing efficient interconnection networks is to demonstrate the existence of *universal* networks. A universal network is one that can emulate any other network of the same cost with minimal difference in performance. Based on the VLSI grid model, Leiserson demonstrated that a class of topologies called fat-trees [Lei85] are universal, in that they can simulate (off-line) any other network with the same area with only a polylogarithmic slowdown. Greenberg [Gre90] later extended Leiserson's work and proved the universality of a similar class of networks called fat-pyramids. The fat-pyramid network has the additional property that any network occupying the same area, can be mapped onto the fat-pyramid in a manner which ensures that the path lengths in the fat-pyramid are not much greater than the path lengths of the original network. This property makes the fat-pyramid applicable under many different assumptions with respect to wire delays. While universal networks are interesting from a theoretical point of view, we approach the network design problem from an engineering viewpoint where even constant factors of difference in performance are important, and polylogarithmic slowdowns are unacceptable. We are also interested in designing networks under cost models that are more detailed than the VLSI grid model.

Network comparisons

A third approach to designing efficient networks is to explicitly define several interconnections and compare them in terms of cost and performance. In this thesis we primarily take this approach, but we shall also prove that some of our networks satisfy theoretical bounds on performance and cost.

A large number of different interconnection topologies have been proposed for parallel machines [Fen81, AG89], but only a few classes of topologies have received substantial attention with respect to comparative evaluations of performance. One such class of topologies is the family of k-ary n-cubes. Rings, multi-dimensional meshes and hypercubes belong to this class of networks. William Dally [Dal86] analyzed the cost-performance trade-offs of this class of networks under Thompson's VLSI grid model, where the cost of a network is its layout area. Different network topologies can be made to occupy the same area by suitably varying the channel widths so that all the networks have the same bisection width. Dally's analysis shows that when bisection width is held constant, all networks have the same bandwidth for delivering messages with random destinations. However, he argues that low dimensional networks are better since they have lower latencies [Dal87].

Abraham and Padmanabhan [AP90] analyzed the k-ary n-cube family of networks under the Pin-requirements model, where the cost metric is the pin-count per node. They compared various networks of this family under the assumption that the pin-count per node was held constant. In contrast to Dally's results, their results indicate that high dimensional networks are better because they can provide higher bandwidths for random communication.

Agarwal [Aga91] conducted a similar analysis of the same class of networks under three different constraints: constant bisection width, constant channel width and constant pin-count. The first two constraints correspond to the VLSI model and the Pin-requirements model respectively, but Agarwal's analysis is based on a more comprehensive model of node and wire delays. His results indicate that the relative ranking of networks is highly sensitive to which cost model is used.

Another class of networks that has been considered by many researchers are multi-stage switching networks such as butterflies, omega networks, delta networks, etc. We can show that most of these networks are isomorphic to one another and are therefore equivalent [WF80]. One interesting variant of the butterfly network that was proposed recently is the multibutterfly network [Upf89, LM89, LLM90]. Konstantinidou and Upfal [KU91] compared the performance of the butterfly network to the multibutterfly network. Their results indicate that the performance of the multibutterfly network is comparable to the butterfly. However, the multibutterfly has far superior fault-tolerance properties in comparison to the butterfly network.

Dandamudi [Dan88] compares networks under a cost model that is different from the two cost models mentioned above. He uses the number of channels in the network as the cost metric. He compares the performance of several *hierarchical* networks, where different topologies are used at different levels of the packaging hierarchy. His work is similar to ours to the extent that the packaging hierarchy is explicitly considered in the design process. However, his cost model is different from ours; under his model, each channel costs the same, irrespective of which level in the hierarchy it belongs to. We believe that in most current packaging hierarchies the cost per channel is usually lower at the lower levels of the packaging hierarchy. Our cost models reflect this variation in costs.

1.6 Contributions

The main contributions of this thesis are:

1. *A model for computing network cost:* We develop a detailed model to characterize packaging costs and provide a method to compute the cost of a network in the presence of a complex hierarchy with a large number of cost factors.

2. *A systematic method for exploring the design space:* We develop a systematic method for exploring the search space of network designs. We argue for a top-down hierarchical network design strategy and use this design strategy to identify several network organizations that offer high performance for relatively low cost. Some of these network organizations (e.g. products of complete graphs, high degree de Bruijn networks) have not received much attention in the literature, even though they are better than many of the popular networks in terms of cost and performance.

3. *General principles for network design:* From extensive simulation studies of networks we arrive at the following general principles for network design: 1) Making the networks denser at the lower levels of the packaging hierarchy has a significant positive impact on global communication performance, 2) It is better to organize a fixed amount of communication bandwidth as a smaller number of high bandwidth channels, 3) For shared memory based communication primitives it is better to make the number of memory modules smaller than the number of processors and 4) Providing the processors with the ability to tolerate latencies (by using *multithreading*) is very useful in improving performance.

4. *A definition and categorization of scalability:* We provide a formal definition of scalability and give a method to compare two network designs in terms of scalability. We categorize scalability based on the level of flexibility in scaling networks. As examples of these categories, we describe extensions to some of the networks discussed in this dissertation. By comparing networks from these categories in terms of performance,

we demonstrate the trade-offs between the level of scaling flexibility and network performance.

5. *A simple fault-tolerance mechanism:* We present a simple and efficient scheme to tolerate faults in single path networks and evaluate this scheme in the presence of a large number of faults.

1.7 Overview of the dissertation

In Chapter 2 we develop a model of packaging technology. The model characterizes multi-level packaging hierarchies and provides a method to compute the cost of a network. In Chapter 3 we describe the strategy that we adopt in our network design process. We first define three generic models of packaging that deal with the costs of packaging at progressively increasing levels of detail. Chapter 3 also describes the methodology we use to evaluate network performance. In Chapters 4, 5, and 6 we consider networks designed under each of the three generic models. For each model, we compare several networks in terms of cost and performance. We shall see that the interconnect at the top level of the hierarchy has the greatest impact on network performance. For each model, we identify top-level networks that offer high performance at low cost, and then consider the lower levels of the hierarchy. The lower level interconnects also affect network performance, but not to the same extent as the top level network. The lower level interconnects affect the utilization of the top level network. We explore various alternatives for the lower level networks that include varying channel widths, channel dilation, etc., and draw quantitative comparisons between them. In Chapter 7, we provide a formal definition of scalability, and classify networks into three scalability categories depending on the flexibility that the networks provide, in going from one size to the other. In Chapter 8 we design a fault-tolerance scheme for the networks presented in this thesis. Finally in Chapter 9 we summarize the main results of this thesis and conclude.

Chapter 2

Packaging Model

Neither the VLSI model nor the Pin-requirements model, described in the previous chapter, is adequate for large machines, since both models deal only with the lowest levels of the packaging hierarchy, where as large parallel machines typically employ several levels of packaging [Nic92, McM92]. There are inherent technological constraints that limit VLSI die sizes, board areas, number of boards per card-cage, number of card-cages per cabinet, etc. As a result, if a system does not fit on a chip, it will first have to be partitioned into multiple chips; if all the chips do not fit on a board, we need multiple boards, and so on. Multi-level hierarchies become inevitable for packaging large parallel systems consisting of thousands of processors.

In the presence of a multi-level packaging hierarchy, it becomes harder to compare the cost of different designs. Cost is a function of a large number of parameters, such as the number of wires at each level of the hierarchy, the size of the printed circuit boards, the number of boards, number of connectors, etc. In order to get accurate cost estimates, we need a better model of packaging technology that explicitly deals with the costs associated with the various parameters. In addition to the costs, the model should also characterize technology-specific constraints such as limits on pin-counts and areas of chips, boards, etc.

In this chapter we define a model of packaging technology that characterizes multi-level packaging hierarchies (Section 2.1). Our model has a number of parameters to explicitly deal with the various constraints and costs associated with each level of the packaging hierarchy. By appropriately specifying these parameters we can model a wide range of packaging technologies (Section 2.2). Nevertheless, our model is an approximation and cannot model all the details of packaging technologies. We discuss the approximations made by the model and its limitations in Section 2.3.

2.1 Model

We model packaging technology as consisting of l levels (numbered from 0 to $l-1$). We call the ith level of the hierarchy as a level-i module. Each level-i module contains some number of level-$(i-1)$ modules and these, in turn contain level-$(i-2)$ modules and so on. Level-0 modules are processing elements, memory units and routing nodes. Figure 2.1 illustrates the hierarchy of modules. *Pins* are used to communicate across module boundaries. Pins are also classified into levels; a level-i pin is a connection that crosses a level-i module boundary. A connection between any two pins is called a *wire*. Wires do not cross module boundaries. A wire within a level-$(i+1)$ module, but outside a level-i module, is called a level-i wire. A level-i wire is used to make a connection between two level-i pins, or between a level-i pin and level-$(i+1)$ pin (figure 2.2). If we wish to make a connection between two level-i modules in different level-$(i+1)$ modules, but the same level-$(i+2)$ module, the connection has to cross level-$(i+1)$ module boundaries. This connection would be considered as 3 wires; two level-i wires and one level-$(i+1)$ wire. A *bundle* is a group of wires, all of which connect the same pair of modules. A bundle of level-i wires is called a level-i bundle.

Our packaging model is defined using two sets of parameters: constraint parameters and cost parameters. Packaging constraints are specified using a set of vectors, $(s_{max}, p_{max}, b_{max}, w_{max}, t)$ of length l, which are defined as follows:

- Module sizes (s_{max}): $s_{max}[i]$ is the maximum number of level-i modules that can be contained in a level-$(i+1)$ module. $s_{max}[l-1]$ is the maximum number of modules at the top of the hierarchy.

- Pin counts (p_{max}): $p_{max}[i]$ is the maximum level-i pin-count per level-i module.

- Bundle counts (b_{max}): $b_{max}[i]$ is the maximum number of level-i bundles per level-i module.

- Bisection widths (w_{max}): $w_{max}[i]$ is the maximum bisection width of the network connecting the level-i modules within a level-$(i+1)$ module. The bisection width of

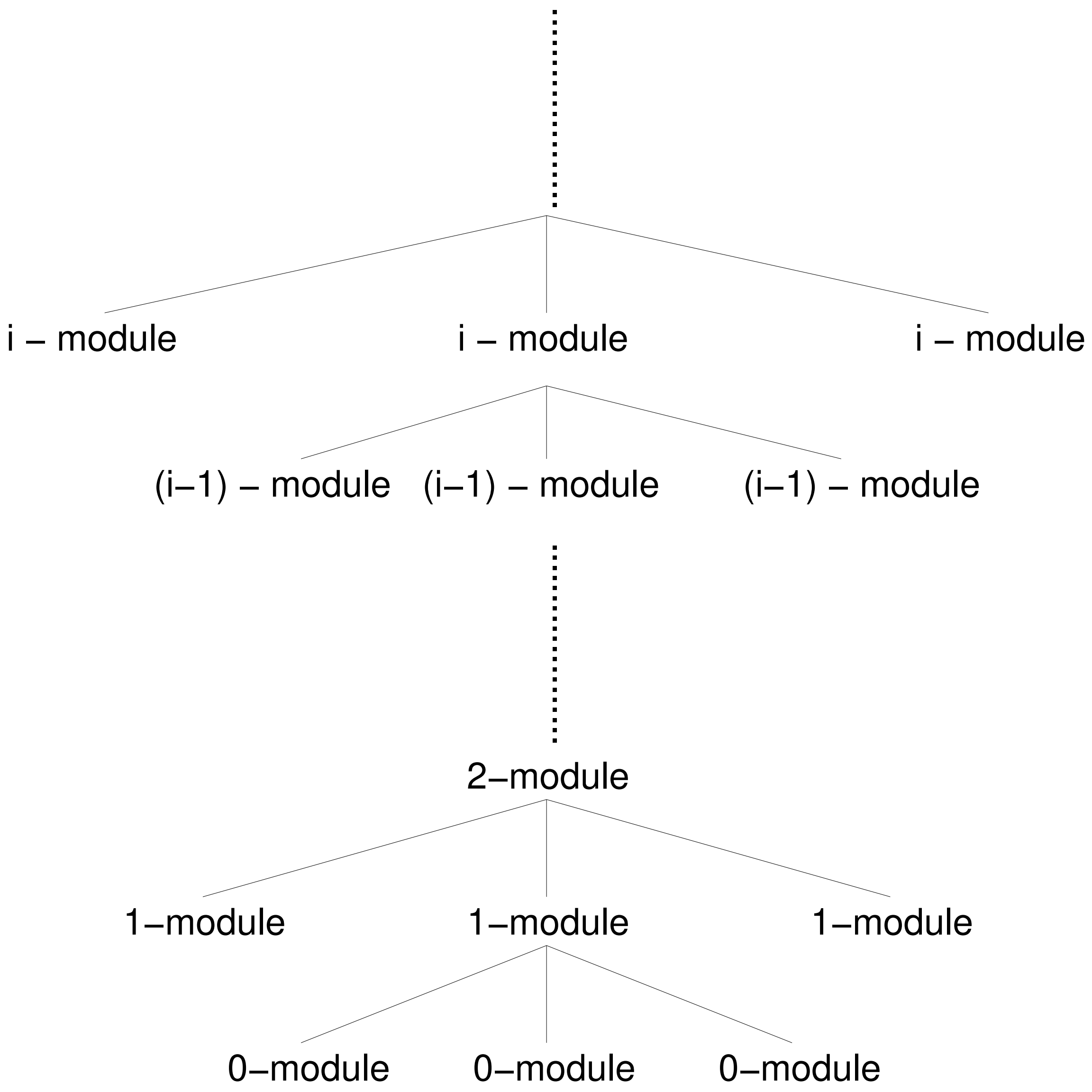

Figure 2.1: Packaging Hierarchy

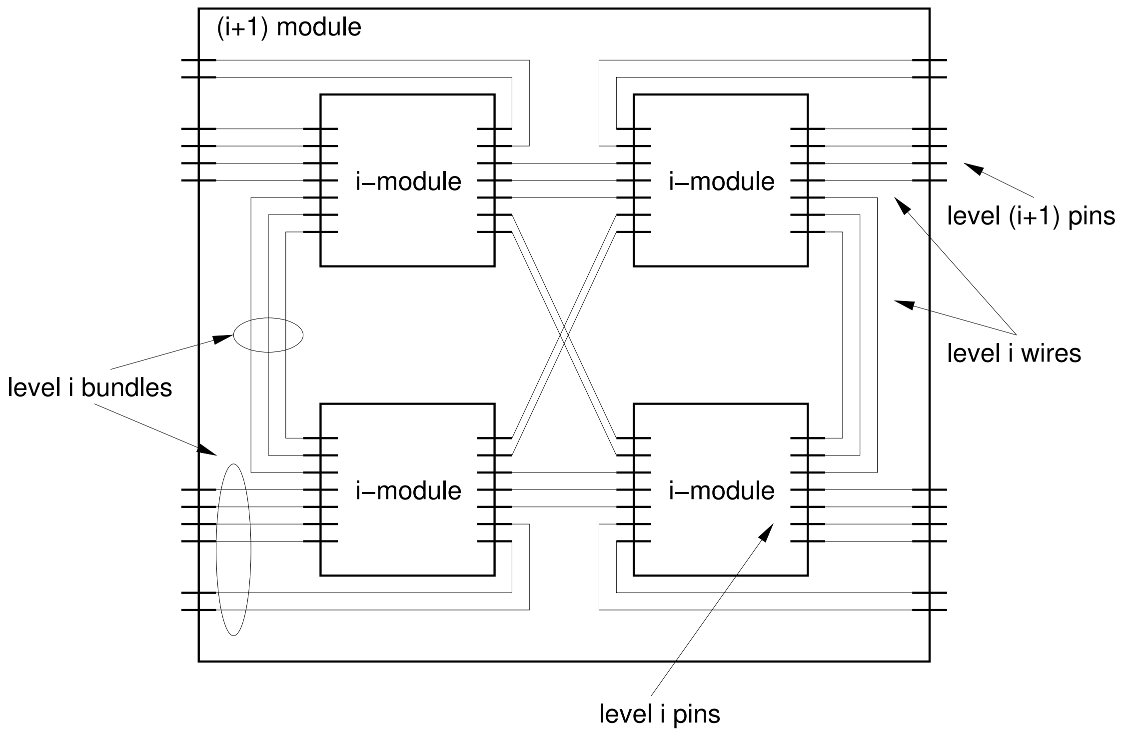

Figure 2.2: Classification of pins, wires and bundles

a network is the minimum number of wires that are cut when the network is divided into two equal parts.

- Clock rates (t): $t[i]$ is the rate at which bits are transmitted over level-i wires.

A valid network design under our model is one that satisfies all the constraint parameters. Such a design is described using the set of vectors $(s_{tot}, p_{tot}, b_{tot}, w_{tot})$. $s_{tot}[l-1]$, where $s_{tot}[l-1] \leq s_{max}[l-1]$, is the number of level-$(l-1)$ modules (top level modules) used by the design. Each level-$(l-1)$ module could contain different numbers of level-$(l-2)$ modules, provided each level-$(l-1)$ module contains no more than $s_{max}[l-2]$ modules. $s_{tot}[l-2]$ is the total number of level-$(l-2)$ modules used by the design ($s_{tot}[l-2] \leq s_{tot}[l-1] * s_{max}[l-2]$). In general $s_{tot}[i]$ is the total number of level-i modules used by the network[1]. $p_{tot}[i]$, $b_{tot}[i]$ and $w_{tot}[i]$ are similarly defined to be the total level-i pin-count, the total number of level-i bundles and the total level-i bisection respectively. We treat the clock rates as being fixed by the packaging technology, so they are not part of the description of the network.

The cost parameters of the model are the vectors (c_s, c_p, c_b, c_w) and are the relative weights of the corresponding quantities s_{tot}, p_{tot}, b_{tot} and w_{tot} respectively. The cost of a network is computed using the function:

$$Cost = \sum_{i=0}^{l-1}(c_s[i] * s_{tot}[i] + c_p[i] * p_{tot}[i] + c_b[i] * b_{tot}[i] + c_w[i] * w_{tot}[i])$$

2.2 Choosing the model parameters

Both the VLSI grid model and the Pin-requirements model are special cases of our model. The VLSI grid model is based on a single level hierarchy and uses layout area as the cost function. To convert our model to the VLSI grid model, we choose $l = 1$, $c_w[0] = 1$, and set all other cost parameters to zero. Since the bisection width of a network is proportional to the square root of its area, if one network is more expensive than another under the grid model, it will also more expensive under our model. However, the cost ratios will not be the same since our cost function is linear in the bisection width, while the cost function of the grid model is quadratic in the bisection width. To convert our model to the

[1]Note that a network design need not use all the l levels of the packaging hierarchy. Small networks may use fewer than l levels. The values of $s_{tot}[i]$ for unused levels is taken to be 0.

Pin-requirements model, we choose $l = 1$, $c_p[0] = 1$, and set all other cost parameters to zero.

Our packaging model is general enough to characterize a wide range of packaging technologies. By appropriately chosing the parameters, we can model most packaging technologies to reasonable accuracy. For example, consider a conventional packaging technology consisting of printed circuit boards in a card-cage, with the boards connected over a back plane. Each processor or routing node is a separate VLSI chip, and several VLSI chips are placed on a board and connected to each other using copper wires etched on the board. A natural representation of this technology in the model will consist of a two level hierarchy ($l = 2$) where the boards are considered to be level-1 modules and the chips are level-0 modules. The values of $s_{max}[0]$ and $s_{max}[1]$ will depend on the number of chips per board and the number of slots in the back plane respectively. The values of $p_{max}[0]$ and $p_{max}[1]$ correspond to pins per chip and pins per board respectively. In this technology there is no real limitation on the number of bundles, so $b_{max}[i] = \infty$. However, since the number of pins is a natural limit on the number of bundles, it is equivalent to say $b_{max}[i] = p_{max}[i]$. In this technology, we usually have a tight restriction on the number of wires in the backplane, and this is modeled using an appropriate value for $w_{max}[1]$. The value of $w_{max}[0]$ would depend on the area of the printed circuit board and number of layers of wiring. Choosing $t_{max}[0]$ and $t_{max}[1]$ is straight-forward. However, choosing the cost parameters is more difficult since our cost function is linear, while realistic cost functions will usually be non-linear. For example, the incremental cost of a wire on a printed circuit board will be small unless it requires increasing the number of layers, at which point the cost will increase by a step function. The cost parameters have to be chosen in a manner that provides a good approximation to the actual cost function.

If we have multiple card-cages within a rack, we may consider connecting boards in different card-cages using cables, while the boards within a card-cage are connected using the back-plane. To model this technology, we would require an additional level of the hierarchy, where the card-cages are considered to be level-2 modules. Since the connections between different card-cages (level-2 modules) are actually made between boards (level-1 modules), there are no physical level-2 pins. However, each board has two types of pins: those that connect to the back plane and those that connect to cables. We define the level-2 pin-count as the total number of pins connecting to cables, over all the boards in a card-cage.

We can also model technologies that use direct die mounting [Mey89, Ath90], multichip modules [Bak90] or wafer-scale integration, by using more levels in the abstract hierarchy. In such technologies, the number of pins per chip can be significantly higher than technologies which place single dies on a chip carrier. This can be modeled using lower per-pin costs and less stringent pin constraints. There may also be stricter limitations on bisection width, since there may be constraints on layout area. The number of bundles is not an important cost metric at this level.

When the machine is too large to fit within a rack, and we need to partition it among multiple racks, the bundling criteria becomes important. Racks are typically interconnected using cables and each cable corresponds to a bundle in our model. Each bundle contributes to the cost of the network in the form of connectors at either end. The number of bundles is also important from the point of view of the complexity involved in wiring the network.

Free-space optical interconnection technologies are being proposed for future parallel machines [TG92, Nef92]. In such technologies, laser beams are used to make connections, instead of pins and wires. Some of these technology proposals also include diffractive or reflective holograms to steer the laser beams. Since laser beams can cross each other in free space without interference, it appears that restrictions on bisection width and bundling may be irrelevant under such technologies. However, the laser sources and receivers may be expensive. In order to fit our model to such a technology, we could model each light beam as a wire and use high pin costs to model the costs of the laser sources and receivers.

In our model of the packaging hierarchy, the processors, routing nodes, and memory units form the leaf nodes (or level-0) modules. It is straightforward to extend our hierarchy further, so that the leaf nodes are smaller units, perhaps individual gates. However, since the primary focus of this dissertation is the interconnection network between processors, this extension is not required.

2.3 Limitations

Our model places no restrictions on the interconnection between modules at a level of the hierarchy, except for pin counts, bundling and bisection width constraints. Pins on any two modules can be connected by a wire, provided the constraints are satisfied. This flexibility may not be permitted in some packaging technologies. For instance the packaging technology proposed for the NuMesh project at M.I.T. uses modules which are connected by abutting each other in a *lego block* fashion [War92]. Each pin of a module can connect only to a physically adjacent module. The modules are assembled in a three dimensional lattice structure and the geometrical positioning of the pins on a module determines the kinds of structures that can be built using these modules. It is not possible to use the parameters of our model to restrict network designs to such lattice structures. However, we can overcome this problem to a certain extent by using additional levels of hierarchy in the model. We can recursively partition the lattice structure into sub-units and treat each sub-unit as a separate level of the hierarchy.

Another limitation of our model is that, at a given level of the hierarchy, all wires are assumed to cost the same. In reality, the wires may have different costs depending on how long they are. For example, wires that go between adjacent boards in a back-plane may be cheaper than wires that traverse the entire length of the back-plane, because longer wires need more powerful drivers. We can use a similar technique of using additional levels of hierarchy to overcome the problem of modeling the difference in costs. We can get a better cost approximation by sub-dividing the card-cage into two or more levels and use different cost parameters for the different levels.

Our model treats the clock rate (rate of bit transmission over wires) as being independent of network topology, but, clock rates may depend on wire length. In general, longer wires have larger capacitances and it takes longer to charge and discharge them, resulting in higher signal propagation delays and lower clock rates. However, in order to overcome the problem of propagation delays on long wires, many current systems treat wires as transmission lines and send bits along them in a pipelined fashion. When pipelined data transmission is used, the assumption made by our model is justified since the clock rates become independent of wire length.

There are many reasons to believe that pipelined data transmission will be the norm in future large scale systems. Technological advances have decreased processor cycle times by a much larger factor than the physical size of parallel computers, resulting in a substantial mismatch between processor speeds and wire delays. This makes it essential to use pipelined data transmission to achieve good performance. Some of the currently available commercial parallel machines and research machines, such as the CM5 [TMC91, L^+92], the MIT Alewife [A^+91], and the Monsoon [Joe90] use pipelined data transmission. The IEEE Standard, Scalable Coherent Interface (SCI) [IEE], that is currently under development, also includes pipelined data transmission. If machines use optical fiber interconnects instead of wires, pipelining is easier and usually the default method of data transmission.

Our model permits different clock rates at different levels of the hierarchy, but for the remaining part of the thesis we assume that the clock rates are the same over all the levels of the hierarchy. The clock rates at the lower levels of the hierarchy are usually higher. We model this using wider channels at the lower levels while keeping the clock rate constant over the entire hierarchy. Both higher clock rates and wider channels essentially have the same effect; namely increasing channel bandwidth. Since we find that network evaluation is simplified under the assumption of constant clock rate, and does not make a difference as far as our results are concerned, henceforth we will assume constant clock rates.

Chapter 3

Methodology

The main goal we pursue in this dissertation is to design the *best* possible network for a large scale parallel machine. We define the *best* network to be the one that achieves the maximum performance for a fixed cost or equivalently one that achieves a fixed performance for a minimum cost. In the preceding chapter, we defined a packaging model using which we can compute network cost. It is possible to use the model to characterize many packaging technologies and design the best network for that technology. However, packaging technologies change with time; a technology that is popular today may be superseded by a different one that becomes feasible in the future. When the technology changes, we would have to repeat the entire design process.

However, instead of basing our designs on a specific technology, we would like to obtain qualitative results with respect network design for packaging hierarchies in general. Therefore, instead of choosing the model parameters specific to a packaging technology, we wish to choose them in a manner that reflects the generic characteristics of packaging hierarchies. We would like choose parameters in a manner that represents general technological trends and omits details particular to a specific technology. The objective of choosing parameters in this manner, is to make our results more general and applicable over a wide range of packaging technologies, both current and future.

In order to choose the parameters in this fashion, we model the fundamental properties that are true of all packaging hierarchies. One such property exhibited by all current packaging hierarchies is *packaging locality*, i.e., there is a progressive increase in costs and decrease in density as we go up the levels of the hierarchy. It is clear that as we proceed up the hierarchy, wire length and pitch (spacing between wires) increase. For