AN ABSTRACT OF THE THESIS OF

Bader Fahed AlBedaiwi AlMohammad for the degree of Doctor of Philosophy in Computer Science presented on November 13, 1997.

Title:

On Resource Placements and Fault-Tolerant Broadcasting in Toroidal Networks.

Redacted for Privacy

Abstract approved: ______________________________

Bella Bose

Parallel computers are classified into: Multiprocessors, and multicomputers. A multiprocessor system usually has a shared memory through which its processors can communicate. On the other hand, the processors of a multicomputer system communicate by message passing through an interconnection network. A widely used class of interconnection networks is the toroidal networks. Compared to a hypercube, a torus has a larger diameter, but better tradeoffs, such as higher channel bandwidth and lower node degree. Results on resource placements and fault-tolerant broadcasting in toroidal networks are presented.

Given a limited number of resources, it is desirable to distribute these resources over the interconnection network so that the distance between a non-resource and a closest resource is minimized. This problem is known as distance-d placement. In such a placement, each non-resource must be within a distance of d or less from at least one resource, where the number of resources used is the least possible. Solutions for distance-d placements in 2D and 3D tori are proposed. These solutions are compared with placements used so far in practice. Simulation experiments show that the proposed solutions are superior to the placements used in practice in terms of reducing average network latency.

The complexity of a multicomputer increases the chances of having processor failures. Therefore, designing fault-tolerant communication algorithms is quite necessary for a sufficient utilization of such a system. Broadcasting (single-node one-to-all) in a multicomputer is one of the important communication primitives. A non-redundant fault-tolerant broadcasting algorithm in a faulty toroidal network is designed. The algorithm can adapt up to $(2n-2)$ processor failures. Compared to the optimal algorithm in a fault-free n-dimensional toroidal network, the proposed algorithm requires at most 3 extra communication steps using cut through packet routing, and $(n+1)$ extra steps using store-and-forward routing.

On Resource Placements and Fault-Tolerant Broadcasting in Toroidal Networks

by

Bader Fahed AlBedaiwi AlMohammad

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Completed November 13, 1997
Commencement June 1998

Doctor of Philosophy thesis of Bader Fahed AlBedaiwi AlMohammad presented on November 13, 1997

APPROVED:

Redacted for Privacy

Major Professor, representing Computer Science

Redacted for Privacy

Head of Department of Computer Science

Redacted for Privacy

Dean of Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Redacted for Privacy

Bader Fahed AlBedaiwi AlMohammad, Author

ACKNOWLEDGMENT

All thanks are due to Allah the most generous the most merciful.

There are no humans to whom I owe as much as I do to my parents for they are my first teachers; for their love is unconditional.

I am deeply touched by the continuous support and encouragements of all the members of my family. I thank them all for everything.

I am grateful to my advisor Prof. Bella Bose who has shaped my research skills and taught me how to approach an open problem. Being a student of Prof. Bose is a great honor of mine.

I would like to thank the current and the former members of my committee: Prof. Paul Cull who has taught me a lot about the theory of computation, Prof. Timothy Budd, Prof. Çetin Koç, Prof. Gerald Smith, Prof. Vikram Saletore (moved to Intel since July 1997), and Prof. Mario Magaña (on sabbatical till January 1998). Their constructive comments, availability, and patience have smoothed my way toward finishing this thesis.

I am thankful to Kuwait University for sponsoring my graduate studies and to Oregon State University for providing me with excellent education and research environments.

Three people of Prof. Bose's research group deserve a distinct gratitude: Dr. Myung Mun Bae, Dr. Jong-Hyun Kim, and Dr. Suliman Albassam. I have learned many things from the discussions I have had with each of them.

Thanks to my friends and colleagues: Dr. Bob Broeg, Dr. Gowri Ramanathan, Dr. Pornsiri Muenchaisri, and Dr. Luca Tallini for their help and support.

Of course, I will not forget to thank the staff members of the computer science office for their help during my stay in OSU. A special salute to Bernadette Feyerherm and Sheryl Parker for their availability and prompt willingness to help.

Last, but not least I would like to thank all the staff of the writing center at OSU for their help and availability.

TABLE OF CONTENTS

Page

Chapter 1: Introduction **1**

1.1 Resource Placements 2

1.2 Fault Tolerant Communications 3

1.3 Organization 3

1.4 Terminologies 4

1.4.1 Mixed Radix Notation 4

1.4.2 Lee Weight 4

1.4.3 Lee Distance 4

1.4.4 Torus 5

1.4.5 Finite Groups 5

Chapter 2: Resource Placement for 2D Tori **6**

2.1 Previous Work 6

2.2 Quasi-Perfect Placement Scheme for $k \times k$ Tori 10

2.2.1 QP_K Placement Scheme - The Proposed Method 11

2.2.2 Minimum Distance Between any Two Resources 14

2.2.3 Maximum Distance Between a Non-Resource and the Closest Resource 20

2.3 Quasi-Perfect Placements for 2D Tori 41

2.4 Quasi-Perfect Placements for $2^i \times 2^i$ Tori 46

2.5 Conclusion 51

Chapter 3: I/O Placements and Network Latencies **52**

3.1 Average Distance Analysis 53

3.1.1 Average Traversal Distance of Local I/O Communication . . . 56

3.1.2 Average Traversal Distance of Non-Local I/O Communication 57

3.2 Workload and Simulation Environment 59

3.3 Scientific Environment Simulation Results 61

TABLE OF CONTENTS (Continued)

Page

3.4 Transaction Environment Simulation Results 64

3.5 Conclusion 71

Chapter 4: Resource Placement for 3D Tori 73

4.1 Previous Work 73

4.2 On Existence of Unknown Linear Perfect Distance-1 Placements . . . 76

4.2.1 On Existence of Irregular Linear Distance-1 Placements 77

4.2.2 On Existence of Regular Linear Distance-1 Placements 82

4.3 Irregular Perfect Distance-*d* Placements 83

4.4 Extending 2D tori Placements to 3D Tori 86

4.5 Sub-Mesh-Based Placements 87

4.6 Conclusion 92

Chapter 5: Broadcasting in Faulty Toroidal Networks 94

5.1 Fault-Free Broadcasting Algorithm 94

5.2 Fault-Tolerant Broadcasting Algorithm 98

5.2.1 Closest Fault-Free Q_k^{n-1} From the Source 101

5.2.2 Finding a Unique Fault-Free Adjacent Ring to a Faulty Ring . 106

5.2.3 Communication Complexity 108

5.2.4 The Algorithm in Toroidal Networks 109

5.3 Conclusion 110

Chapter 6: Future Research 111

6.1 Distance-*d* Resource Placements in Toroidal Networks 111

6.2 Simulation Studies 113

6.3 Fault-Tolerant Communication Algorithms 113

TABLE OF CONTENTS (Continued)

Page

6.4 Other Types of Dominations 113

Bibliography **115**

LIST OF FIGURES

Figure Page

2.1 Perfect placements in 2D tori 7

2.2 Perfect distance-1 placement for a 10×10 torus in which a 5×5 torus is used as a tiling block. Dashed lines indicate links among tiles. 8

2.3 Quasi-perfect resource placements for 32×16 torus: (a) quasi-perfect distance-3 (b) quasi-perfect distance-2 (c) quasi-perfect distance-1. . 10

2.4 Nodes at distance of d or $(d+1)$ from $(0,0)$ as described in the proofs of theorems 2.2.6-11. 22

2.5 Quasi-perfect placements result from horizontal and vertical bisecting. 45

2.6 A 9×12 torus tiled by a vertical bisection of a 6×6 torus. 45

2.7 Quasi-perfect placements as applications of Theorem 2.4.1 in a 16×16 torus. 48

2.8 PLACE-SCALED-DOWN algorithm. 49

2.9 PLACE-SCALED-UP algorithm. 50

3.1 Intel-Paragon 53

3.2 Outer-Column I/O placements for the architectures simulated in this study. (wraparound links are not shown). 54

3.3 Perfect/Quasi-Perfect I/O placements for the architectures simulated in this study: (a) perfect distance-1 (b) quasi-perfect distance-1 (c) perfect distance-2 (d) quasi-perfect distance-2 (wraparound links are not shown). 54

3.4 Architecture of PE-I/O node. 55

3.5 Utilized and non-utilized links, in terms of local I/O request flow, are represented by arrows and dashed lines, respectively. 58

3.6 Inter-request time [μsec] vs. average I/O communication latency [μsec] when data locality is set to 50%. 62

3.7 Inter-request time [μsec] vs. average I/O communication latency [μsec] when data locality is set to 20% and 80%. 63

LIST OF FIGURES (Continued)

Figure | Page

3.8 Inter-request time [μsec] vs. average I/O communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 50%. 65

3.9 Inter-request time [μsec] vs. average I/O communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 20% and 80%. 66

3.10 Inter-request time [μsec] vs. average I/O communication latency [μsec] when PE request block size is 32 byte. 67

3.11 Inter-request time [μsec] vs. average PE communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 50%. 69

3.12 Inter-request time [μsec] vs. average PE communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 20% and 50%. 70

3.13 Inter-request time [μsec] vs. average PE communication latency [μsec] when PE request block size is 32 byte. 71

4.1 Regular perfect distance-1 placement in a $7 \times 7 \times 7$ torus. 74

4.2 Irregular perfect distance-1 placement in a $2 \times 6 \times 3$ torus. 75

4.3 Irregular perfect distance-1 placement in a $2 \times 2 \times 2$ torus. 77

4.4 The impossibility of Case(4). 81

4.5 Irregular perfect distance-2 placement in a $2 \times 2 \times 12$ torus. 84

4.6 Irregular perfect distance-3 placement in a $2 \times 2 \times 20$ torus. 85

4.7 Irregular perfect distance-2 placement in a $2 \times 2 \times 24$ torus. 85

4.8 Irregular perfect distance-3 placement in a $2 \times 4 \times 16$ torus. 86

4.9 Irregular perfect distance-4 placement in a $2 \times 6 \times 20$ torus. 87

4.10 Sphere volume for perfect distance-d placement, sub-mesh-based distance-$3d$, distance-$(2d/3)$, and distance-d. 91

4.11 Average traversal distances for perfect distance-d placement, sub-mesh-based distance-$3d$, distance-$(2d/3)$, and distance-d. 92

LIST OF FIGURES (Continued)

Figure Page

5.1 A fault-free broadcasting algorithm for Q_k^n using cut through packet routing. 95

5.2 A fault-free broadcasting algorithm for Q_k^n using store-and-forward packet routing. 96

5.3 A broadcasting algorithm for a faulty Q_k^n. 99

5.4 Finding one of the closest sub-cubes to S. 102

5.5 Finding an adjacent fault-free ring along a given dimension. 105

5.6 Assigning each faulty ring along the X dimension an adjacent fault-free ring 106

LIST OF TABLES

Table Page

2.1 The possible quasi-perfect placements for a 30×30 torus using QP_k placement scheme. 43

2.2 The possible quasi-perfect placements for a 24×36 torus using QP_k placement scheme. 43

2.3 Scaled-down quasi-perfect placements for a 32×32 torus that result from applying PLACE-SCALED-DOWN algorithm. 49

2.4 Possible scaled-up quasi-perfect placements for a 32×32 torus based on PLACE-SCALED-UP algorithm. 50

3.1 Values of $d_l(OC)$ and $d_l(QP)$ for different k's rounded to two significant digits. * indicates a perfect placement. 57

4.1 Placements for $4 \times 8 \times 16$ based on 2D placements. 88

4.2 Placements for $9 \times 12 \times 16$ based on 2D placements. 88

4.3 Placements for $5 \times 32 \times 32$ based on 2D placements. 89

DEDICATION

To My Parents With Love.

ON RESOURCE PLACEMENTS AND FAULT-TOLERANT BROADCASTING IN TOROIDAL NETWORKS

Chapter 1

INTRODUCTION

Parallel computers are architecturally classified into: shared-memory multiprocessors, and distributed-memory multicomputers [44]. The processors of a system in the former class communicate through a shared memory, while in the latter, by message passing through an interconnection network. The focus of this thesis is made on toroidal interconnection networks. Several multicomputers have been built using this kind of networks, or networks that can be directly embedded into a torus. Examples are: Caltech Cosmic Cube (hypercube of 64 nodes) [68], Intel Paragon (2D mesh), Touchstone Delta (two dimensional mesh) [3], IBM Victor (16×16 mesh) [72], Ametak 2010 [71], Caltech Mosaic C multicomputer [70, 69], MIT J-Machine (three dimensional mesh) [3], Tera system (Q_k^3) [6], Cray T3D, and Cray T3E (three dimensional torus) [7, 54, 67].

A subclass of toroidal networks is k-ary n-cube (Q_k^n) graphs. In a torus, different dimensions may have different sizes, but in Q_k^n all dimensions have the same size k.

It has been shown that low-dimensional Q_k^n is a more wire efficient communication network than high-dimensional ones, under the assumption of constant node number and constant wire bisection [33]. For this reason, low-dimensional tori and Q_k^n interconnection networks have become significant from the practical point of view.

This thesis presents the results of investigating two problems: resource placements in 2D and 3D tori, and fault-tolerant broadcasting in toroidal networks.

The rest of this chapter is divided into four sections. Section 1.1 presents resource placement problems and a brief history of their development. Section 1.2 overviews the importance of fault-tolerant communications in practice. Section 1.3 describes the organization of the thesis. Finally, Section 1.4 introduces some terms and notations needed in later chapters.

1.1 Resource Placements

Providing each processor in a parallel computer with all the resources it needs is not cost effective. Methods for distributing a limited number of resources over an interconnection network have been investigated. Earlier, most of the resource placement research was concentrated on hypercube networks [25, 26, 50, 63]. Recently, more research has been conducted on the placement methods for tori and Q_k^n networks [8, 9, 10, 60, 61]. There are three subproblems associated with resource placements: distance-d placement, j-adjacency placement, and generalized placement. Distance-d placement involves in placing resources such that each non-resource node is within a distance of d or less from exactly one resource node. On the other hand, j-adjacency placement considers placing resources so that each non-resource node is adjacent to j resource nodes. Generalized placement combines both of distance-d and j-adjacency placements. It considers placing resources such that each non-resource node is within a distance $\leq d$ from j resource nodes [8, 9, 10].

In this thesis, the distance-d placements for 2D and 3D tori are investigated. Finding a distance-d placement for any given network is a "hard" problem. A single instance of this problem is the perfect distance-1 placement, which is equivalent to finding a perfect 1-error-correcting code or a perfect dominating set for a given graph. Both of these problems have been shown to be NP-complete [32, 37, 51].

1.2 Fault Tolerant Communications

The communication algorithms of a multicomputer are quite crucial to its performance. They are the basic tools by which the processors can cooperate. When some processors or links fail, it is not effective, and sometimes not affordable, to keep the entire system idle till the failures are resolved. For this reason, the design of efficient fault-tolerant communication algorithms is essential in the industry of parallel computers.

Broadcasting (single-node one-to-all) is one of the important communication patterns for multicomputer systems [45]. Broadcasting in faulty toroidal networks is investigated in Chapter 5 of this thesis.

1.3 Organization

The thesis is organized as follows.

Chapter 2 overviews the results of investigating distance-d resource placements in 2D tori.

Chapter 3 presents the results of a simulation study that compares the effects of different Input/Output resource placement strategies in 2D tori on average network latencies. In this study, the resource placement methods described in Chapter 2 are compared with strategies used so far in practice.

In Chapter 4, distance-d resource placements in 3D tori are investigated.

Chapter 5 describes a fault-tolerant broadcasting algorithm for toroidal networks in general.

Finally, future research topics are described in Chapter 6.

1.4 Terminologies

The *Lee distance* is a metric used in the field of error correcting codes. It has been shown in [19] that the Lee distance is a natural metric for toroidal networks. Many topological properties of a toroidal network can be derived from this useful metric [19]. In this section, some definitions and terms needed in later chapters are introduced.

1.4.1 Mixed Radix Notation

Let $X = x_{n-1}x_{n-2}\ldots x_0$ be an n-dimensional vector over $K = k_{n-1}k_{n-2}\ldots k_0$, where x_i has the radix k_i, then X is said to be in *mixed radix notation.* $I(X)$, the *Integer Value* of X, is defined as:

$$I(X) = x_0 + x_1k_0 + x_2k_0k_1 + \cdots + x_{n-1}k_0k_1\ldots k_{n-1} = \sum_{i=0}^{n-1}(x_i \prod_{j=0}^{i-1} k_j).$$

For example, if 432 is a vector over 543, then $I(432) = 4(12) + 3(3) + 2 = 59$.

1.4.2 Lee Weight

Let X be a vector in mixed radix notation over K, then the *Lee Weight*, W_L, of X is defined as:

$$W_L = \sum_{i=0}^{n-1} min(x_i, k_i - x_i).$$

For example, if 342 is a vector defined over 765, then $W_L(342) = 3 + 2 + 2 = 7$.

1.4.3 Lee Distance

Let X and Y be vectors in mixed radix notation over K, then the *Lee Distance*, D_L, is defined as:

$$D_L(X,Y) = \sum_{i=0}^{n-1} min\big(x_i - y_i(mod\ k_i), y_i - x_i(mod\ k_i)\big) = W_L(X - Y).$$

For example, let 131 and 554 be vectors over 765, then $D_L(131{,}554) = 7$.

1.4.4 Torus

Let T be an n-dimensional torus defined over $K = k_{n-1}k_{n-2}\dots k_0$, then T is denoted by $T_{k_{n-1},k_{n-2},\dots,k_0}$ over T_K and defined as follows. Let $N = \prod_{i=0}^{n-1} k_i$, T is a graph with N nodes numbered from 0 to $(N-1)$. Each node of T is labeled with a mixed radix n-dimensional vector over K and this vector is referred to as the address of the node. For any node $X \in T$, the node number of $X = I(X)$. For any two nodes X and Y $\in T$, there is an edge between X and Y if and only if $D_L(X,Y) = 1$.

A k-ary n-cube graph, Q_k^n, is a torus in which all the dimensions have the same radix.

1.4.5 Finite Groups

A *group*, as defined in [30], is a set S together with a binary operation $\oplus$ for which the following properties hold:

1. Closure: $\forall\ a,\ b \in S,\ a \oplus b \in S$.

2. Identity: There is an element $e \in S$ such that $\forall\ a \in S,\ e \oplus a = a \oplus e = a$.

3. Associativity: $\forall\ a, b, c \in S,\ (a \oplus b) \oplus c = a \oplus (b \oplus c)$.

4. Inverse: $\forall\ a \in S$, there exists a unique element $b \in S$ such that $a \oplus b = a \oplus b = e$.

If $\forall\ a, b \in S,\ a \oplus b = a \oplus b$ then $(S, \oplus)$ is an *Abelian group*. Further, If $|S| < \infty$, $(S, \oplus)$ is called a *finite group*.

Chapter 2

RESOURCE PLACEMENT FOR 2D TORI

Methods for perfect distance-d placements in 2D tori have been presented in [8, 9, 10, 50]. These methods can be applied to special cases of 2D tori, as shown in Section 2.1. In this chapter, a placement scheme for $k \times k$ tori is defined and proven to maintain special characteristics. Furthermore, a discussion on when and how to use the defined scheme in finding placements for 2D tori is included.

There are five sections in this chapter. Section 2.1 overviews the related literature. Section 2.2 presents a distance-d placement scheme for k-ary 2-cube (or $k \times k$ tori). Sections 2.3 and 2.4 present placements for 2D tori, based on the scheme of Section 2.2. Finally, Section 2.5 draws the conclusions of this chapter.

2.1 Previous Work

The problem of resource placements in 2D tori has been investigated from two different points of view : Error correcting codes and graph theory. Bose, Broeg, and Bae have proposed solutions based on *Lee error correcting codes* [40, 13, 19, 21, 8, 9, 10]. On the other hand, Livingston and Stout have investigated resource placements using the concept of *perfect dominating sets* of graph theory [50, 51].

In the rest of this section, an overview of the major related results achieved by the above previous studies will be presented. However, before this presentation some terms are needed to be defined.

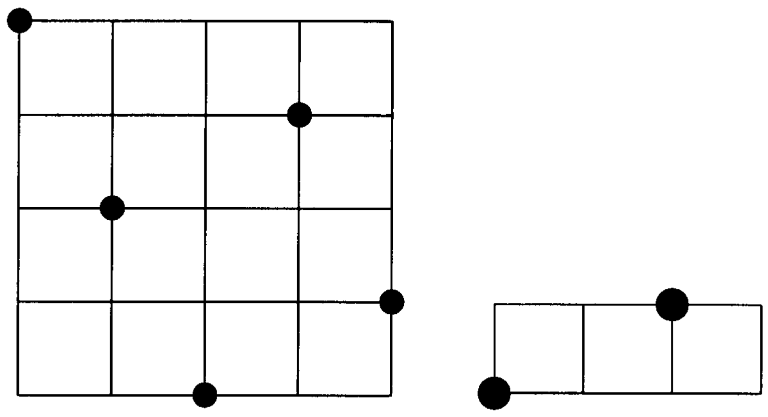

(a) 5×5 - Regular (b) 2×4 - Irregular

FIGURE 2.1: Perfect placements in 2D tori

Definition 2.1.1 *A placement is a perfect distance-d if any non-resource node is within a distance of d or less from exactly one resource node.*

Figure 2.1 shows two different perfect placements.

Definition 2.1.2 *A radius-d packing sphere of a resource node A is the set of all nodes within a distance of d or less from A.*

Definition 2.1.3 *A perfect distance-d placement is called regular if the radius-d sphere of each resource node has the maximum possible size; otherwise it is called irregular.*

Examples of regular and irregular perfect placements are shown in Figure 2.1-(a) and (b), respectively. As it can be noticed from these figures, the radius-1 packing sphere of a resource in the regular case does not collapse into itself, while it does in the irregular case. In general, a sphere can collapse into itself if and only if it exists in a space in which at least one of the dimensions is less than $(2d + 1)$.

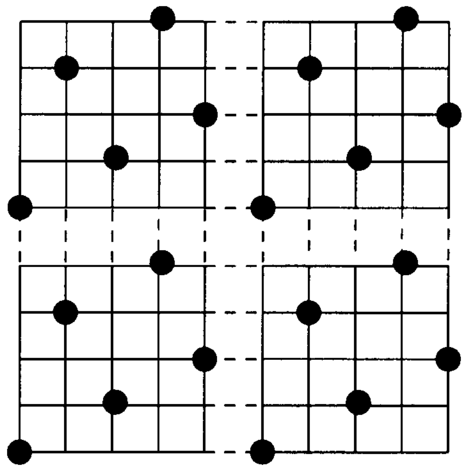


FIGURE 2.2: Perfect distance-1 placement for a 10 $\times$ 10 torus in which a 5 $\times$ 5 torus is used as a tiling block. Dashed lines indicate links among tiles.

The major related results of the above mentioned studies can be summarized in the following:

- Bose et al have proved that the surface area (i.e. the number of nodes at a distance exactly d from a given node) of a radius-d sphere in a Q_n^k is [19, 21]:

$$A_n^k(d) \;=\; \sum_{i=1}^{min(d,n)} \binom{d-1}{i-1}\binom{n}{i}2^i \tag{2.1}$$

  The volume (i.e. the number of nodes at a distance of d or less from a given node) of a radius-d sphere in a Q_n^k is [10, 21]:

$$V_n^k(d) \;=\; 1+\sum_{i=1}^{d} A_n^k(i) \;=\; 1+\sum_{i=1}^{min(d,n)} 2^i\binom{n}{i}\binom{d}{i} \tag{2.2}$$

  Based on that, it is easy to verify that the maximum possible area and volume of a radius-d sphere in a 2D torus, $d \geq 1$, are $(4d)$ and $(2d^2+2d+1)$, respectively.

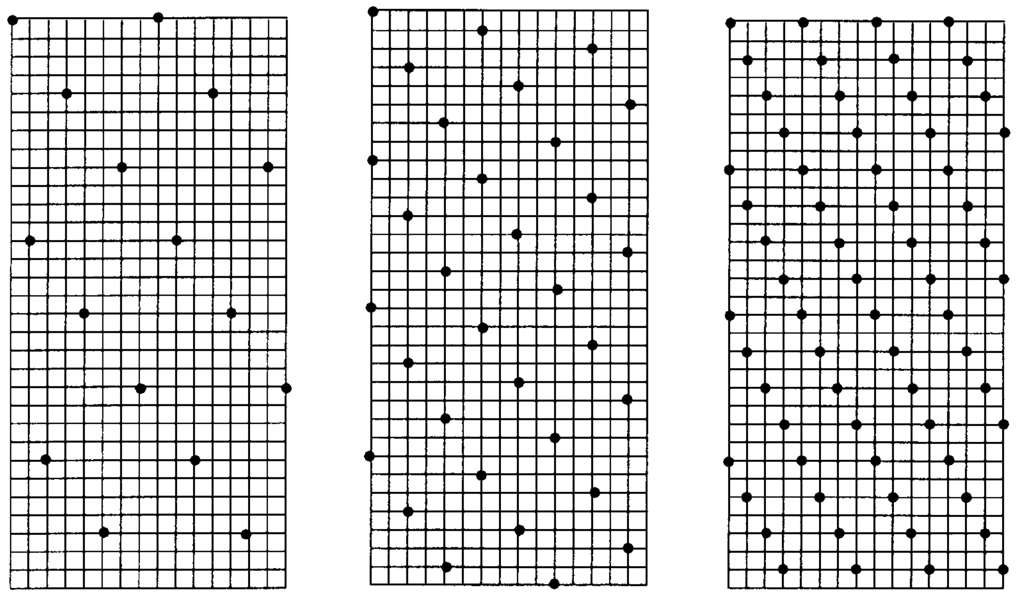
- Bae and Bose have shown that for an $X \times Y$ torus, a regular perfect distance-d placement exists, if both X and Y are divisible by $2d^2 + 2d + 1$. In this case, first, a perfect distance-d placement for $k \times k$ torus, $k = 2d^2+2d+1$, is obtained by using *Lee error correcting codes.* The resources are placed at $(i,\ 2d^2i)$, for $i = 0, 1, \ldots, k-1$. Next, $k \times k$ torus is used to tile the larger $X \times Y$ torus [8, 9, 10]. Figure 2.2 shows how a 5×5 torus is used to tile a 10×10 torus.

- Bae and Bose have proved that for any $X \times Y$ torus, there exists a regular perfect distance-1 placement, if and only if both X and Y are divisible by 5 [8, 9, 10].

- Livingston and Stout have claimed that a perfect distance-d placement exists for an $X \times Y$ torus, if and only if $\{X, Y\} \in$ $\big\{\{2, 4dp\}, \{4, (4d-2)p\}, \{6, (4d-4)p\}, \ldots, \{2d, (2d+2)p\} : p \geq 1\big\} \cup$ $\big\{\{(2d^2 + 2d + 1)p, (2d^2 + 2d + 1)q\} : p, q \geq 1\big\}$ [50].
  However, this claim is not quite right. According to this claim, a 4×4 torus should have a perfect distance-1 placement, but this can easily be proven to be wrong. Upon a discussion between M. L. Livingston* and the author, the above claim has been modified and proved by Livingston to be as follows:
  A perfect distance-d placement exists for an $X \times Y$ torus, if and only if $\{X, Y\}$ $\in \big\{\{2, 4dp\} : 0 \leq p \leq d\text{-}1\big\} \cup \big\{\{(2d^2 + 2d + 1)p, (2d^2 + 2d + 1)q\} : p, q \geq 1\big\}$.

Based on this, it is clear that perfect placements do not exist for the majority of 2D tori. This is the main motivation for investigating the so-called *quasi-perfect placements.* The following section defines this term and presents the major results of this investigation.

* Currently with the department of Computer Science at the University of Oregon



(a) 16 resources (b) 32 resources (c) 64 resources

FIGURE 2.3: Quasi-perfect resource placements for 32×16 torus: (a) quasi-perfect distance-3 (b) quasi-perfect distance-2 (c) quasi-perfect distance-1.

2.2 Quasi-Perfect Placement Scheme for $k \times k$ Tori

Perfect placements do not exist for a large class of 2D tori. Cases for which perfect placements are not possible, *quasi-perfect placements* might exist. A *quasi-perfect* placement satisfies the following two conditions:

1. Let a and b be any two resource nodes and S_a and S_b be the sets of nodes at a distance d or less from a and b, respectively, then $S_a \cap S_b = \phi$.

2. No non-resource node is at a distance of more than $d + 1$ from some resource nodes.

These two conditions imply that the maximum possible number of non-resource nodes are at a distance of d or less from the resource nodes, and the remaining nodes are at a distance of $d+1$ from some resource nodes. Examples of quasi-perfect placements are shown in Figure 2.3.

Finding an allocation method for quasi-perfect placements would offer more flexibility in choosing the dimensions of an interconnection network, as well as the ability to scale the number of used resources up or down. For instance, it would be possible to find a quasi-perfect placement for a torus of size $2^i \times 2^j$ (many practical systems have sizes 2^x in each dimension). Figure 2.3-(b) shows a quasi-perfect distance-2 placement for 32×16 using 32 resources. For the same network, it is possible to have a quasi-perfect distance-3 placement using 16 resources, or a quasi-perfect distance-1 placement using 64 resources, as illustrated in Figures 2.3-(a)&(c), respectively.

2.2.1 QP_K Placement Scheme - The Proposed Method

A linear resource placement for Q_k^2 can be described by a 1×2 generator matrix $[a_1 \quad a_2]$. For example, $G = [1 \quad 2]$ is a generator matrix for placing resources in a 5×5 torus where resources are placed in $(i,\ 2i \ mod\ 5),\ 0 \leq i < 5$, as illustrated in Figure 2.1-(a). A special linear placement scheme for Q_k^2 is defined as follows:

The QP_k Scheme *is a linear resource placement method for Q_k^2 interconnection networks, generated by $G = [d \quad (d+1)]$. QP_k is defined as follows:*

$$QP_k = \{(x,y) : x = id \ (mod\ k),\ y = i(d+1) \ (mod\ k), 0 \leq i < k\}$$

where $2d^2 + 2 \leq k \leq 2(d+1)^2 + 1,\ d \geq 0.$

The following subsections prove that the QP_K is:

- A quasi-perfect distance-$(d-1)$ placement when $2d^2 + 2 \leq k \leq 2d^2 + 2d$.

- A perfect distance-d placement when $k = 2d^2 + 2d + 1$.
- A quasi-perfect distance-d placement when $2d^2 + 2d + 2 \leq k \leq 2(d+1)^2 + 1$.

The rest of this subsection presents definitions and lemmas needed in the following subsections.

Definition 2.2.1 *Let $n_i = (id \ (mod\ k),\ i(d+1)\ (mod\ k))$ and $n_j = (jd\ (mod\ k),\ j(d+1)\ (mod\ k))$, then n_i and $n_j \in QP_k$. The operations $+_s$ and $-_s$ are defined as follows:*

$$n_i +_s n_j = (d(i+j)\ (mod\ k),\ (d+1)(i+j)\ (mod\ k)).$$

$$n_i -_s n_j = (d(i-j)\ (mod\ k),\ (d+1)(i-j)\ (mod\ k)).$$

Lemma 2.2.1 *$(QP_k,\ +_s)$ is a finite Abelian group.*

Proof:

1. Closure: For any two nodes n_i and $n_j \in QP_k$,

$$\begin{aligned} n_i +_s n_j &= (d(i+j)\ (mod\ k),\ (d+1)(i+j)\ (mod\ k)) \\ &= n_{(i+j)\ (mod\ k)} \in QP_k. \end{aligned}$$

2. Identity Element: $(0,0)$ is the identity element.

3. Associativity: For any three nodes n_i, n_j, and $n_l \in QP_k$,

$$\begin{aligned} (n_i +_s n_j) +_s n_l &= (d(i+j+l)\ (mod\ k),\ (d+1)(i+j+l)\ (mod\ k)) \\ &= n_i +_s (n_j +_s n_l) \end{aligned}$$

4. Inverse: The inverse of any node $n_i = n_{k-i}, 0 < i < k$:

$$\begin{aligned} n_i +_s n_{k-i} &= (d(i+k-i)\ (mod\ k),\ (d+1)(i+k-i)\ (mod\ k)) \\ &= (0,0). \end{aligned}$$

Suppose there is a node n_j, $0 < j < k$ such that $j \neq (k-i)$ and $n_i +_s n_j = (0,0)$ $\Rightarrow j = k-i$ which is a contradiction. Hence, the inverse of n_i is unique.

5. Commutativity: for any two nodes n_i and $n_j \in QP_k$,

$$\begin{aligned} n_i +_s n_j &= (d(i+j)\ (mod\ k),\ (d+1)(i+j)\ (mod\ k)) \\ &= (d(j+i)\ (mod\ k),\ (d+1)(j+i)\ (mod\ k)) \\ &= n_j +_s n_i. \end{aligned}$$

■

Lemma 2.2.2 *The cardinality of QP_k is k (i.e. $|QP_k| = k$).*

Proof: Assume there exists $n_i = n_j$, $i \neq j$, and $0 \leq i,j < k \Rightarrow n_i -_s n_j = (0,0)$ $\Rightarrow (i-j)\ (mod\ k) = 0$. This is a contradiction since $i \neq j$ and $0 \leq i,j < k$. ■

Lemma 2.2.3 *$W_L\ (n_i) = W_L\ (n_{k-i})$, $n_i \in QP_k$.*

Proof:

$$\begin{aligned} W_L(n_i) &= min\ (id\ (mod\ k),\ k - id\ (mod\ k)) + \\ &\quad min\ (id + i\ (mod\ k),\ k - id - i\ (mod\ k)) \\ &= min(id\ (mod\ k),\ -id\ (mod\ k)) + \\ &\quad min\ (id + i\ (mod\ k),\ -id - i\ (mod\ k)). \end{aligned}$$

Furthermore,

$$\begin{aligned} W_L(n_{k-i}) = \quad & min\ ((k-i)d\ (mod\ k),\ k-(k-i)d\ (mod\ k)) + \\ & min\ ((k-i)(id+i)\ (mod\ k),\ k-(k-i)(id+i)\ (mod\ k)) \\ = \quad & min\ (-id\ (mod\ k),\ id\ (mod\ k)) + \\ & min\ (-id-i\ (mod\ k),\ id+i\ (mod\ k)). \end{aligned}$$

Hence, $W_L(n_i) = W_L(n_{k-i})$. ■

Lemma 2.2.4 *Let $D(QP_k)$ be the minimum Lee distance, D_L, between any two nodes in QP_k. $D(QP_k) \geq$ minimum $\{$Lee weight $W_L(n_i)$: $n_i \in QP_k \ \wedge \ i \neq 0\}$.*

Proof: $D(QP_k) = D_L\ (n_i,n_j)$ for some n_i and $n_j \in QP_k$, $i \neq j$,

Hence, $D(QP_k) = W_L\ (n_i -_s n_j) = W_L\ (n_i +_s n_{k-j})$.

Since $(QP_k, +_s)$ is a group, $(n_i +_s n_{k-j}) \in QP_k$, and $(n_i +_s n_{k-j}) \neq (0,0)$. ■

2.2.2 Minimum Distance Between any Two Resources

In this subsection, QP_k is proven to satisfy the first necessary condition of a quasi-perfect placement. This will follow in two steps:

- Showing that the radius-$(d-1)$ packing spheres of any two resources in QP_k are disjoint when $2d^2+2 \leq k \leq 2d^2+2d$ (Theorem 2.2.1 and Corollary 2.2.2).

- Showing that the radius-d packing spheres of any two resources in QP_k are disjoint when $2d^2+2d+3 \leq k \leq 2d^2+4d+3$ (Theorem 2.2.3 and Corollary 2.2.4).

Theorem 2.2.1 *Let $D(QP_k)$ be the minimum Lee distance, D_L, between any two resource nodes in QP_k. $D(QP_k) \geq 2d-1$, when $2d^2+2 \leq k \leq 2d^2+2d$.*

Proof: The only nodes in a given torus with Lee weight $< (2d-1)$ are those at a distance of $(2d-2)$ or less from $(0,0)$. These nodes are nothing but the nodes of radius-$(2d-2)$ packing sphere of $(0,0)$. Let this sphere be denoted as $R_{n_0,(2d-2)}$. Showing that $n_i \notin R_{n_0,(2d-2)}$, for any $n_i \in QP_K$, $0 < i < k$, proves this theorem. The proof is by contradiction. Assume there exists $n_i \in R_{n_0,(2d-2)}$, $0 < i < k$. Then, n_i must be in one of the following classes:

- **Class 1:** $id \ (mod\ k) = 0 \ \wedge \ 0 < id + i \ (mod\ k) \leq 2d-2$.

  Assume there exists an $n_i \in$ **Class 1** $\Rightarrow 2d-2 \geq W_L\ (n_i) = i \ (mod\ k) = i$, since $i < k$ and $2d-2 < d^2+1 \leq \frac{k}{2}$. Furthermore, $id \ (mod\ k) = 0 \Rightarrow id = Ck,\ C > 0$ $\Rightarrow i \geq \frac{2d^2+2}{d} > 2d-1$, since $k \geq 2d^2+2$. This means that $W_L\ (n_i) > 2d-1$ (Contradiction) $\Rightarrow$ There exists no $n_i \in$ **Class 1**.

- **Class 2:** $id \ (mod\ k) = 0 \ \wedge \ k - 2d + 2 \leq id + i \ (mod\ k) < k$.

  Assume there exists an $n_i \in$ **Class 2** $\Rightarrow n_i$ has an inverse $\in QP_k$ since $(QP_k, +_s)$ is a group (Lemma 2.2.1). Suppose n_j is the inverse of n_i then $jd \ (mod\ k) = 0$ and $0 < jd + j \ (mod\ k) \leq 2d-2$. Hence, $n_j \in$ **Class 1**, but it was shown that no such $n_j \in$ **Class 1** (Contradiction).

- **Class 3:** $k - 2d + 2 \leq id \ (mod\ k) < k \ \wedge \ 0 \leq id + i \ (mod\ k) \leq 2d-2$.

  Assume there exists an $n_i \in$ **Class 3** $\Rightarrow W_L\ (n_i) = id + i - id \ (mod\ k) = i$ since $2d-2 < \frac{k}{2}$, $k-2d+2 > \frac{k}{2}$, and $i < k$. Furthermore, $id \ (mod\ k) > id+i \ (mod\ k)$ $\Rightarrow \lfloor \frac{id+i}{k} \rfloor \geq 1 \Rightarrow i \geq \frac{k}{d+1} > 2d-2$. Hence, $W_L\ (n_i) \geq 2d-1$ (Contradiction) $\Rightarrow$ There exists no $n_i \in$ **Class 3**.

- **Class 4:** $0 < id \ (mod\ k) \leq 2d-2 \ \wedge \ k - 2d + 2 \leq id + i \ (mod\ k) < k$.

  Assume there exists an $n_i \in$ **Class 4** $\Rightarrow$ its inverse $n_j \in$ **Class 3**. However, it was shown that no such $n_j \in$ **Class 4** (Contradiction) $\Rightarrow$ There is no $n_i \in$ **Class 4**.

- **Class 5:** $0 < id \ (mod\ k) \leq 2d-2 \ \wedge \ id+i \ (mod\ k) = 0$.

  Assume there exists an $n_i \in$ **Class 5** $\Rightarrow 2d-1 \geq W_L(n_i) = W_L(n_j) \in$ **Class 3**, where n_j is the inverse of n_i. However, it was shown that there is no such n_j $\in$ **Class 3** (Contradiction) $\Rightarrow$ There exists no $n_i \in$ **Class 5**.

- **Class 6:** $0 < id \ (mod\ k) \leq 2d-2 \ \wedge \ 0 < id+i \ (mod\ k) \leq 2d-2$.

  Assume there exists an $n_i \in$ **Class 6**. Since $id+i \ (mod\ k) \ \neq \ id \ (mod\ k) \Rightarrow$ one of the following cases holds:

  I. $id+i \ (mod\ k) \ > \ id \ (mod\ k) \Rightarrow$

  $id+i \ (mod\ k) \ - id \ (mod\ k) = i$, $1 \leq i \leq 2d-3$, $d \geq 3$.

  If $i = 1$ then $W_L\ (n_i) = 2d+1 > 2d-1$.

  If $2 \leq i \leq 2d-3$ then $2d \leq id \leq 2d^2-3d \Rightarrow id \ (mod\ k) > 2d-2$

  $\Rightarrow n_i \notin$ **Class 6** (Contradiction).

  II. $id \ (mod\ k) \ > \ id+i \ (mod\ k) \Rightarrow$

  $id \ (mod\ k) \ - id - i \ (mod\ k) = k-i$, $1 \leq k-i \leq 2d-3 \Rightarrow$

  $k-2d+3 \leq i \leq k-1$, $d \geq 3 \Rightarrow$

  $kd-2d^2+3d \leq id \leq kd-d \Rightarrow id \ (mod\ k) \geq 3d+2 \Rightarrow$

  $n_i \notin$ **Class 6** (Contradiction).

- **Class 7:** $k-2d+2 \leq id \ (mod\ k) < k \ \wedge \ k-2d+2 \leq id+i \ (mod\ k) < k$.

  Assume there exists an $n_i \in$ **Class 7** $\Rightarrow$ its inverse $n_j \in$ **Class 6**. However, it was shown that there is no such $n_j \in$ **Class 6** (Contradiction).

Hence, $W_L(n_i) \geq 2d-1$, $0 < i < k$. This proves that $D(QP_k) \geq 2d-1$. ■

Corollary 2.2.2 *Let $R_{a,d}$ be the radius-d packing sphere of node a. $R_{a,(d-1)} \cap R_{b,(d-1)} = \phi$, for any two nodes a and $b \in QP_k$, when $2d^2+2 \leq k \leq 2d^2+2d$.*

Proof: Assume there exist two resource nodes a and b such that:

$$R_{a,(d-1)} \cap R_{b,(d-1)} \neq \phi.$$

This means that there exists at least one node c that belongs to both $R_{a,(d-1)}$ and $R_{b,(d-1)}$. In such a case, the distance between a and $b \leq (2d-2)$ since ($a \dashrightarrow c \leq d-1$ and $c \dashrightarrow b \leq d-1$). However, this contradicts with Theorem 2.2.1. ∎

Theorem 2.2.3 *Let $D(QP_k)$ be the minimum Lee distance, D_L, between any two nodes in QP_k. $D(QP_k) \geq 2d+1$, when $2d^2+2d+1 \leq k \leq 2(d+1)^2+1$.*

Proof: The proof is quite similar to the proof of Theorem 2.2.1. The only nodes in a given torus with Lee weight $< (2d+1)$ are those at a distance of $2d$ or less from $(0,0)$. These nodes are nothing but the nodes of radius-$(2d)$ packing sphere of $(0,0)$. Let this sphere be denoted as $R_{n_0,(2d)}$. Showing that $n_i \notin R_{n_0,(2d)}$, for any $n_i \in QP_K$, $0 < i < k$, proves this theorem. The proof is by contradiction. Assume there exists $n_i \in R_{n_0,(2d)}$, $0 < i < k$. Then, n_i must be in one of the following classes:

- **Class 1:** $id \ (mod\ k) = 0 \ \wedge \ 0 < id + i \ (mod\ k) \leq 2d.$

  Assume there exists an n_i *in* **Class 1** $\Rightarrow 2d \geq W_L\ (n_i) = i\ (mod\ k) = i$, since $i < k$ and $2d < d^2 + d < \frac{k}{2}$. Furthermore, $id\ (mod\ k) = 0 \Rightarrow id = Ck,\ C > 0$ $\Rightarrow i \geq \frac{2d^2+2d+1}{d} > 2d+2$, since $k \geq 2d^2+2d+1$. This means that $W_L\ (n_i)$ $> 2d+1$ (Contradiction) $\Rightarrow$ There exists no $n_i \in$ **Class 1**.

- **Class 2:** $id\ (mod\ k) = 0 \ \wedge \ k - 2d \leq id + i\ (mod\ k) < k.$

  Assume there exists an $n_i \in$ **Class 2** $\Rightarrow n_i$ has an inverse $\in QP_k$ since $(QP_k, +_s)$ is a group (Lemma 2.2.1). Suppose n_j is the inverse of n_i then $jd\ (mod\ k) = 0$

and $0 < jd + j \ (mod\ k) \leq 2d$. Hence, $n_j \in$ **Class 1**, but it was shown that no such $n_j \in$ **Class 1** (Contradiction).

- **Class 3:** $k - 2d \leq id \ (mod\ k) < k \ \wedge \ 0 \leq id + i \ (mod\ k) \leq 2d$.

  Assume there exists an $n_i \in$ **Class 3** $\Rightarrow W_L\ (n_i) = id + i - id \ (mod\ k) = i$ since $2d - 2 < \frac{k}{2}$, $k - 2d > \frac{k}{2}$, and $i < k$. Furthermore, $id \ (mod\ k) > id + i \ (mod\ k)$ $\Rightarrow \lfloor \frac{id+i}{k} \rfloor \geq 1 \Rightarrow i \geq \frac{k}{d+1} > 2d$. Hence, $W_L\ (n_i) \geq 2d + 1$ (Contradiction) $\Rightarrow$ There exists no such $n_i \in$ **Class 3**.

- **Class 4:** $0 < id \ (mod\ k) \leq 2d \ \wedge \ k - 2d \leq id + i \ (mod\ k) < k$.

  Assume there exists an $n_i \in$ **Class 4** $\Rightarrow$ its inverse $n_j \in$ **Class 3**. However, it was shown that there is no such $n_j \in$ **Class 3** (Contradiction).

- **Class 5:** $0 < id \ (mod\ k) \leq 2d \ \wedge \ id + i \ (mod\ k) = 0$.

  Assume there exists an $n_i \in$ **Class 5** $\Rightarrow 2d - 1 \geq W_L(n_i) = W_L(n_j) \in$ **Class 3**, where n_j is inverse of n_i. However, it was shown that there is no such $n_j \in$ **Class 3** (Contradiction).

- **Class 6:** $0 < id \ (mod\ k) \leq 2d \ \wedge \ 0 < id + i \ (mod\ k) \leq 2d$.

  Assume there exists an $n_i \in$ **Class 6**. Since $id + i \ (mod\ k) \ \neq \ id \ (mod\ k) \Rightarrow$ one of the following cases holds:

  I. $id + i \ (mod\ k) \ > \ id \ (mod\ k) \Rightarrow$

     $id + i \ (mod\ k) \ - id \ (mod\ k) = i$, $1 \leq i \leq 2d - 1$, $d \geq 2$. If $1 \leq i \leq 2$ then $W_L\ (n_i) \geq 2d + 1$. If $3 \leq i \leq 2d - 1$ then $3d \leq id \leq 2d^2 - d \Rightarrow$ $id \ (mod\ k) > 3d \Rightarrow n_i \notin$ **Class 6** (Contradiction).

  II. $id \ (mod\ k) \ > \ id + i \ (mod\ k) \Rightarrow$

     $id \ (mod\ k) \ - id - i \ (mod\ k) = k - i$, $1 \leq k - i \leq 2d - 1$, $d \geq 2 \Rightarrow$

$k-2d+1 \leq i \leq k-1, \Rightarrow kd-2d^2+d \leq id \leq kd-d \Rightarrow id\ (mod\ k) \geq 3d+1$
$\Rightarrow n_i \notin$ **Class 6** (Contradiction).

- **Class 7:** $k-2d \leq id\ (mod\ k) < k \ \wedge\ k-2d \leq id+i\ (mod\ k) < k$. Assume there exists an $n_i \in$ **Class 7** $\Rightarrow$ its inverse $n_j \in$ **Class 6**. However, it was shown that there is no such $n_j \in$ **Class 6** (Contradiction).

Hence, $W_L(n_i) \geq 2d+1$, $0 < i < k$. This proves that $D(QP_k) \geq 2d+1$. ■

Corollary 2.2.4 *Let $R_{a,d}$ be the radius-d packing sphere of node a.*
$R_{a,d} \cap R_{b,d} = \phi$, for any two resource nodes a and $b \in QP_k$, when $2d^2+2d+1 \leq k \leq 2(d+1)^2+1$.

Proof: Assume there exist two resource nodes a and b such that:

$$R_{a,d} \cap R_{b,d} \neq \phi.$$

This means that there exists at least one node c that belongs to both $R_{a,d}$ and $R_{b,d}$. In such a case, the distance between a and $b \leq 2d$ since ($a \rightarrow c \leq d$ and $c \rightarrow b \leq d$). However, this contradicts with Theorem 3.1. ■

Corollary 2.2.5 *QP_k is a perfect distance-d placement when $k = 2d^2+2d+1$.*

Proof: In this case, the number of nodes in $R_{n_i,d} = 2d^2+2d+1 = k$ [21]. Further, $R_{n_i,d} \cap R_{n_j,d} = \phi$, where n_i and n_j are two different resource nodes (Corollary 2.2.4), Since $|QP_k| = k$ (Lemma 2.2.2), the total number of nodes covered by all the resources =

$$\textstyle\sum_{i=0}^{k-1} R_{n_i,d} = k^2, \text{ where } n_i \text{ is } i^{th} \text{ resource node.}$$

Thus, each non-resource node is within a distance of d or less from exactly one resource node. ■

2.2.3 Maximum Distance Between a Non-Resource and the Closest Resource

In the previous subsection, QP_K was proven to maintain the first necessary condition for a quasi-perfect placement. In this subsection, QP_k is proven to satisfy the second necessary condition. This will follow in several stages:

- Theorem 2.2.6 proves that any node in a $k \times k$ torus is within a distance of d or less from at least one resource node $\in QP_k$, for $2d^2 + 2 \le k \le 2d^2 + d$.
- Theorem 2.2.7 proves that any node in a $k \times k$ torus is within a distance of d or less from at least one resource node $\in QP_k$, for $2d^2 + d + 1 \le k \le 2d^2 + 2d$.
- Theorem 2.2.9 proves that any node in a $k \times k$ torus is within a distance of $(d+1)$ or less from at least one resource node $\in QP_k$, for $2d^2+2d+2 \le k \le 2d^2+3d+1$.
- Theorem 2.2.10 proves that any node in a $k \times k$ torus is within a distance of $(d+1)$ or less from at least one resource node $\in QP_k$, for $2d^2 + 3d + 2 \le k \le 2d^2 + 4d + 2$.
- Theorem 2.2.11 proves that any node in a $k \times k$ torus is within a distance of $(d+1)$ or less from at least one resource node $\in QP_k$, for $k = 2d^2 + 4d + 3$.

Lemma 2.2.5 *Let T be a $k \times k$ torus. Furthermore, Let S be a set defined as follows.*

$$S = \{n : n \in T \text{ and } D_L(n, R) = (d + i + 1), \text{ where } R \text{ is any resource in } T\ \}$$

If any node in S is at a distance of $(d+i)$ or less from some resource nodes, then each node in T is within a distance of $(d+i)$ or less from at least one resource node.

Proof: Let b be a non-resource node such that $D_L(a, b) = (d+i+j)$, $j \ge 1$, and a be one of the closest resource nodes to b. There exists a node, say c, at a distance of

$(j-1)$ from b and at a distance of $(d+i+1)$ from a. By the hypothesis, there exists a resource node r such that $D_L(r,c) = (d+i)$. Hence, $D_L(r,b) \leq (d+i+j-1) < (d+i+j)$. This contradicts the assumption of a being one of the closest resource nodes to b. Therefore, a node such as b does not exist in T. ■

Lemma 2.2.6 *Let T be a $k \times k$ torus. Let $r_i = (id \ (mod\ k),\ id + i(mod\ k))$ be the i^{th}-resource node in QP_k. Suppose every non-resource node, R, at a distance of $(d+j+k)$ from the resource node $r_0 = (0,0)$ is at a distance of $(d+j+k-1)$ from some resource node S. Then, every non-resource node in T is within a distance of $(d+j+k-1)$ or less from at least one resource node.*

Proof: Let b be a non-resource node such that $D_L(a,b) = (d+j+k)$, and a be one of the closest resource nodes to b. Let $-a$ be the additive inverse of a. Then, $(b-a \ (mod\ k))$ is at a distance of $(d+j+k)$ from $(0,0)$. By the lemma hypothesis, there must be a resource node, say r, at a distance of $(d+j+k-1)$ from $(b-a \ (mod\ k))$. Hence b must be at a distance of $(d+j+k-1)$ from $(r+a \ (mod\ k))$ which is a resource node since QP_k is a group. This contradicts the assumption that a is one of the closest resource nodes to b. Therefore, a node such as b does not exist in T. ■

Theorem 2.2.6 *Any node in a $k \times k$ torus is within a distance of d or less from at least one resource node when QP_k is used, where $2d^2+2 \leq k \leq 2d^2+d$.*

Proof: Suppose a node a is at a distance of $(d+1)$ from the resource node $(0,0)$. It will be shown that the node a is at a distance of d or less from some other resource nodes. Thus, based on Lemma 2.2.5 and Lemma 2.2.6, this theorem follows.

Any node at a distance of $(d+1)$ from $(0,0)$ must be in one of the following classes:

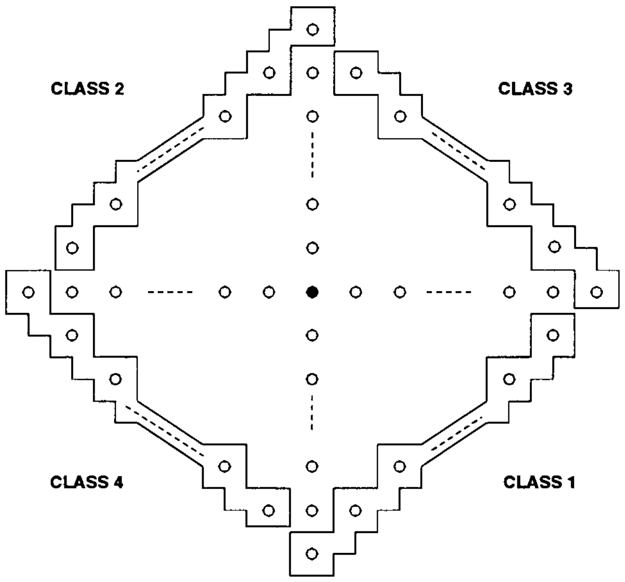


FIGURE 2.4: Nodes at distance of d or $(d+1)$ from $(0,0)$ as described in the proofs of theorems 2.2.6-11.

- **Class 1:** $\{\ (x,y) : x+y=(d+1),\ 1 \leq y \leq (d+1),\ \mathit{and}\ 0 \leq x \leq d\ \}$.

  Let $r_1 = (d,\ d+1)$. Obviously, $r_1 \in QP_k$, and

$$D_L(r_1,(x,y)) \leq\ 2d+1-x-y=d.$$

- **Class 2:** $\{\ (x,y) : -x-y=(d+1),\ (-d-1) \leq y \leq -1,\ \mathit{and}\ -d \leq x \leq 0\ \}$.

  Let $r_{-1} = (-d,\ -d-1)$. Obviously, $r_{(-1)}$ is a resource node, and

$$D_L(r_{(-1)},\ (x,y)) \leq\ 2d+1+x+y=d.$$

- **Class 3:** $\{\ (x,y) : x-y=(d+1),\ -d \leq y \leq 0,\ \mathit{and}\ 1 \leq x \leq (d+1)\ \}$.

  Define a matrix M of i rows and j columns to be:

$$M = \begin{bmatrix} (1,-d) & (2,1-d) & (3,2-d) & \dots & (d-1,-2) & (d,-1) & (d+1,0) \\ (1,-d) & (2,1-d) & (3,2-d) & \dots & (d-1,-2) & (d,-1) & (d+1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,-d) & (2,1-d) & (3,2-d) & \dots & (d-1,-2) & (d,-1) & (d+1,0) \end{bmatrix}$$

$, 1 \leq i \leq (d-1), 1 \leq j \leq (d+1).$

Note that $M[i,j]$ is a unique node in **Class 3** when $k = i + 2d^2 + 1$, $x = j$, and $y = j - d - 1$. More clearly, the i^{th} row represents all the nodes of **Class 3** in a $k \times k$ torus, $k = 2d^2 + i + 1$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (1,2d^2-d+2) & (2,2d^2-d+3) & \dots & (d,2d^2+1) & (d+1,2d^2+2) \\ (1,2d^2-d+3) & (2,2d^2-d+4) & \dots & (d,2d^2+2) & (d+1,2d^2+3) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,2d^2) & (2,2d^2+1) & \dots & (d,2d^2+d-1) & (d+1,2d^2+d) \end{bmatrix}$$

Let $r_{(-2d)}$ be resource node number $(k - 2d)$.
Thus, $r_{(-2d)} = (k - 2d^2, k - 2d^2 - 2d)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (2,2d^2-2d+4) = & r_{(-2d)} & \textit{when } k = 2d^2+2 \\ (3,2d^2-2d+6) = & r_{(-2d)} & \textit{when } k = 2d^2+3 \\ \vdots & \vdots & \vdots \\ (d,2d^2) = & r_{(-2d)} & \textit{when } k = 2d^2+d \end{bmatrix}$$

$, 1 \leq i \leq (d-1).$

Furthermore, let $r_{(1-2d)}$ be resource node number $(k - 2d + 1)$.
Thus, $r_{(1-2d)} = (k - 2d^2 + d, k - 2d^2 - d + 1)$.
Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (d+2, 2d^2-d+5) = & r_{(-2d+1)} & \textit{when } k = 2d^2+2 \\ (d+3, 2d^2-d+7) = & r_{(-2d+1)} & \textit{when } k = 2d^2+3 \\ \vdots & \vdots & \vdots \\ (2d, 1) = & r_{(-2d+1)} & \textit{when } k = 2d^2+d \end{bmatrix}$$

$, 1 \leq i \leq (d-1)$.

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i \geq j-1 \\ V_2[i], & i < j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \leq d$, $1 \leq i \leq (d-1)$ and $1 \leq j \leq (d+1)$. Hence, every node in **Class 3** is within a distance of d or less from at least one resource node.

- **Class 4:** $\{\ (x,y) : y - x = (d+1),\ 0 \leq y \leq d,\ \textit{and}\ (-d-1) \leq x \leq -1\ \}$.
Let M be a matrix of i rows and j columns:

$$M = \begin{bmatrix} (-1,d) & (-2,d-1) & (-3,d-2) & \ldots & (1-d,2) & (-d,1) & (-d-1,0) \\ (-1,d) & (-2,d-1) & (-3,d-2) & \ldots & (1-d,2) & (-d,1) & (-d-1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (-1,d) & (-2,d-1) & (-3,d-2) & \ldots & (1-d,2) & (-d,1) & (-d-1,0) \end{bmatrix}$$

$, 1 \leq i \leq (d-1), 1 \leq j \leq (d+1)$.

Note that $M[i,j]$ is a unique node in **Class 4** when $k = i + 2d^2 + 1$, $x = -j$, and $y = d - j + 1$. More clearly, the i^{th} row represents all the nodes of **Class 4** in a $k \times k$ torus, $k = 2d^2 + i + 1$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (2d^2+1,d) & (2d^2,d-1) & \dots & (2d^2-d+2,1) & (2d^2-d+1,0) \\ (2d^2+2,d) & (2d^2+1,d-1) & \dots & (2d^2-d+3,1) & (2d^2-d+2,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (2d^2+d-1,d) & (2d^2+d-2,d-1) & \dots & (2d^2,1) & (2d^2-1,0) \end{bmatrix}$$

Let $r_{(2d)}$ be resource node number $(2d)$.
Thus, $r_{(2d)} = (2d^2, 2d^2+2d)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (2d^2, 2d-2) = & r_{(2d)} & \textit{when } k = 2d^2+2 \\ (2d^2, 2d-3) = & r_{(2d)} & \textit{when } k = 2d^2+3 \\ \vdots & \vdots & \vdots \\ (2d^2, d) = & r_{(2d)} & \textit{when } k = 2d^2+d \end{bmatrix}$$

$, 1 \leq i \leq (d-1)$.

Furthermore, let $r_{(2d-1)}$ be resource node number $(2d-1)$.
Thus, $r_{(2d-1)} = (2d^2-d, 2d^2+d-1)$. Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (2d^2-d, d-3) = & r_{(2d-1)} & \textit{when } k = 2d^2+2 \\ (2d^2-d, d-4) = & r_{(2d-1)} & \textit{when } k = 2d^2+3 \\ \vdots & \vdots & \vdots \\ (2d^2-d, -1) = & r_{(2d-1)} & \textit{when } k = 2d^2+d \end{bmatrix}$$

$, 1 \leq i \leq (d-1)$.

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i \geq j-1 \\ V_2[i], & i < j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \leq d$, $1 \leq i \leq (d-1)$ and $1 \leq j \leq (d+1)$. Hence, every node in **Class 4** is within a distance of d or less from at least one resource node.

Thus, any node is within a distance of d or less from at least one resource node. ■

Theorem 2.2.7 *Any node in a $k \times k$ torus is within a distance of d or less from at least one resource node when QP_k is used, where $2d^2 + d + 1 \le k \le 2d^2 + 2d$.*

Proof: Suppose a node a is at a distance of $(d+1)$ from the resource node $(0,0)$. It will be shown that the node a is at a distance of d or less from some other resource nodes. Thus, based on Lemma 2.2.5 and Lemma 2.2.6, this theorem follows.

Any node at a distance of $(d+1)$ from $(0,0)$ must be in one of the following classes:

- **Class 1:** $\{ (x,y) : x + y = (d+1),\ 1 \le y \le (d+1),\ \textit{and}\ 0 \le x \le d \}$.
  Let $r_1 = (d,\ d+1)$. Obviously, $r_1 \in QP_k$, and

$$D_L(r_1, (x,y)) \le\ 2d + 1 - x - y = d.$$

- **Class 2:** $\{ (x,y) : -x - y = (d+1),\ (-d-1) \le y \le -1,\ \textit{and}\ -d \le x \le 0 \}$.
  Let $r_{-1} = (-d,\ -d-1)$. Obviously, $r_{(-1)}$ is a resource node, and

$$D_L(r_{(-1)},\ (x,y)) \le\ 2d + 1 + x + y = d.$$

- **Class 3:** $\{ (x,y) : x - y = (d+1),\ -d \le y \le 0,\ \textit{and}\ 1 \le x \le (d+1) \}$.
  Define a matrix M of i rows and j columns to be:

$$M = \begin{bmatrix} (1,-d) & (2,1-d) & (3,2-d) & \dots & (d-1,-2) & (d,-1) & (d+1,0) \\ (1,-d) & (2,1-d) & (3,2-d) & \dots & (d-1,-2) & (d,-1) & (d+1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,-d) & (2,1-d) & (3,2-d) & \dots & (d-1,-2) & (d,-1) & (d+1,0) \end{bmatrix}$$

, $1 \leq i \leq d$, $1 \leq j \leq (d+1)$.

Note that $M[i,j]$ is a unique node in **Class 3** when $k = i + 2d^2 + d$, $x = j$, and $y = j - d - 1$. More clearly, the i^{th} row represents all the nodes of **Class 3** in a $k \times k$ torus, $k = 2d^2 + i + d$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (1,2d^2+1) & (2,2d^2+2) & \dots & (d,2d^2+d) & (d+1,0) \\ (1,2d^2+2) & (2,2d^2+3) & \dots & (d,2d^2+d+1) & (d+1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,2d^2+d) & (2,2d^2+d+1) & \dots & (d,2d^2+2d-1) & (d+1,0) \end{bmatrix}$$

Let $r_{(-2d-1)}$ be resource node number $(k-2d-1)$.
Thus, $r_{(-2d-1)} = (k-2d^2-d, k-2d^2-3d-1)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (1,2d^2-d+1) = & r_{(-2d-1)} & \textit{when } k = 2d^2+d+1 \\ (2,2d^2-d+3) = & r_{(-2d-1)} & \textit{when } k = 2d^2+d+2 \\ \vdots & \vdots & \vdots \\ (d,2d^2+d-1) = & r_{(-2d-1)} & \textit{when } k = 2d^2+2d \end{bmatrix}$$

, $1 \leq i \leq d$.

Furthermore, let $r_{(-2d)}$ be resource node number $(k-2d)$.
Thus, $r_{(-2d)} = (k-2d^2, k-2d^2-2d)$.
Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (d+1, 2d^2+2) = & r_{(-2d)} & \textit{when } k = 2d^2+d+1 \\ (d+2, 2d^2+4) = & r_{(-2d)} & \textit{when } k = 2d^2+d+2 \\ \vdots & \vdots & \vdots \\ (2d, 0) = & r_{(-2d)} & \textit{when } k = 2d^2+2d \end{bmatrix}$$

$, 1 \leq i \leq d.$

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i > j-1 \\ V_2[i], & i \leq j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \leq d$, $1 \leq i \leq d$ and $1 \leq j \leq (d+1)$. Hence, every node in **Class 3** is within a distance of d or less from at least one resource node.

- **Class 4:** $\{ \ (x,y) : y - x = (d+1), \ 0 \leq y \leq d, \ \textit{and} \ (-d-1) \leq x \leq -1 \ \}$.
  Define a matrix M of i rows and j columns to be:

$$M = \begin{bmatrix} (-1,d) & (-2,d-1) & (-3,d-2) & \dots & (1-d,2) & (-d,1) & (-d-1,0) \\ (-1,d) & (-2,d-1) & (-3,d-2) & \dots & (1-d,2) & (-d,1) & (-d-1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (-1,d) & (-2,d-1) & (-3,d-2) & \dots & (1-d,2) & (-d,1) & (-d-1,0) \end{bmatrix}$$

$, 1 \leq i \leq d, \ 1 \leq j \leq (d+1).$

Note that $M[i,j]$ is a unique node in **Class 4** when $k = i + 2d^2 + d$, $x = -j$, and $y = d+1-j$. More clearly, the i^{th} row represents all the nodes of **Class 4** in a $k \times k$ torus, $k = 2d^2 + i + d$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (2d^2+d,d) & (2d^2+d-1,d-1) & \dots & (2d^2+1,1) & (2d+^2,0) \\ (2d^2+d+1,d) & (2d^2+d,d-1) & \dots & (2d^2,1) & (2d^2+1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (2d^2+2d-1,d) & (2d^2+2d-2,d-1) & \dots & (2d^2+d-2,1) & (2d^2+d-1,0) \end{bmatrix}$$

Let $r_{(2d+1)}$ be resource node number $(2d+1)$.
Thus, $r_{(2d+1)} = (2d^2+d, 2d^2+3d+1)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (2d^2+d,2d) = & r_{(2d+1)} & when\ k = 2d^2+d+1 \\ (2d^2+d,2d-1) = & r_{(2d+1)} & when\ k = 2d^2+d+2 \\ \vdots & \vdots & \vdots \\ (2d^2+d,d+1) = & r_{(2d+1)} & when\ k = 2d^2+2d \end{bmatrix}$$

$, 1 \leq i \leq d.$

Furthermore, let $r_{(2d)}$ be resource node number $(k - 2d)$.
Thus, $r_{(2d)} = (2d^2, 2d^2+2d)$.
Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (2d^2,d-1) = & r_{(2d)} & when\ k = 2d^2+d+1 \\ (2d^2,d-2) = & r_{(2d)} & when\ k = 2d^2+d+2 \\ \vdots & \vdots & \vdots \\ (2d^2,0) = & r_{(2d)} & when\ k = 2d^2+2d \end{bmatrix}$$

$, 1 \leq i \leq d.$

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i > j-1 \\ V_2[i], & i \leq j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \leq d$, $1 \leq i \leq d$ and $1 \leq j \leq (d+1)$. Hence, every node in **Class 4** is within a distance of d or less from at least one resource node.

Thus, any node is within a distance of d or less from at least one resource node. ■

Corollary 2.2.8 *QP_k is a quasi-perfect distance-$(d-1)$ placement when $2d^2+2 \leq k \leq 2d^2+2d$.*

Proof:

(1) In this range, the radius-$(d-1)$ packing spheres of any two nodes in QP_k are disjoint (Corollary 2.2.2).

(2) Also in this range, any node is at a distance of d or less from at least one resource node (Theorems 2.2.6 and 2.2.7).

Hence, the two conditions required for a quasi-perfect distance-$(d-1)$ placement are satisfied. ■

Theorem 2.2.9 *Any node in a $k \times k$ torus is within a distance of $(d+1)$ or less from at least one resource node when QP_k is used, where $2d^2+2d+2 \leq k \leq 2d^2+3d+1$.*

Proof: Suppose a node a is at a distance of $(d+2)$ from the resource node $(0,0)$. It will be shown that the node a is at a distance of $(d+1)$ or less from some other resource nodes. Thus, based on Lemma 2.2.5 and Lemma 2.2.6, this theorem follows.

Any node at a distance of $(d+2)$ from $(0,0)$ must be in one of the following classes:

- **Class 1:** $\{\ (x,y) : x+y=(d+2),\ 1 \leq y \leq (d+2),\ \textit{and}\ 0 \leq x \leq (d+1)\ \}$. Let $r_1=(d,\ d+1)$. Obviously, $r_1 \in QP_k$, and

$$D_L(r_1,(x,y)) \leq (d+1).$$

- **Class 2:** $\{\ (x,y) : -x-y=(d+2),\ (-d-2)\leq y\leq -1,\ \mathit{and}\ (-d-1)\leq x\leq 0\ \}$.

  Let $r_{-1} = (-d,\ -d-1)$. Obviously, $r_{(-1)}$ is a resource node, and

$$D_L(r_{(-1)},\ (x,y)) \leq (d+1).$$

- **Class 3:** $\{\ (x,y) : x-y=(d+2),\ (-d-1)\leq y\leq 0,\ \mathit{and}\ 1\leq x\leq (d+2)\ \}$. Define a matrix M of i rows and j columns to be:

$$M = \begin{bmatrix} (1,-d-1) & (2,-d) & (3,1-d) & \dots & (d,-2) & (d+1,-1) & (d+2,0) \\ (1,-d-1) & (2,-d) & (3,1-d) & \dots & (d,-2) & (d+1,-1) & (d+2,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,-d-1) & (2,-d) & (3,1-d) & \dots & (d,-2) & (d+1,-1) & (d+2,0) \end{bmatrix}$$

  $,\ 1\leq i\leq d,\ 1\leq j\leq (d+2)$.

  Note that $M[i,j]$ is a unique node in **Class 3** when $k = i+2d^2+2d+1$, $x=j$, and $y=j-d-2$. More clearly, the i^{th} row represents all the nodes of **Class 3** in a $k\times k$ torus, $k=i+2d^2+2d+1$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (1,2d^2+d+1) & (2,2d^2+d+2) & \dots & (d+1,2d^2+2d+1) & (d+2,0) \\ (1,2d^2+d+2) & (2,2d^2+d+3) & \dots & (d+1,2d^2+2d+2) & (d+2,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,2d^2+2d) & (2,2d^2+2d+1) & \dots & (d+1,2d^2+3d) & (d+2,0) \end{bmatrix}$$

  Let $r_{(-2d-2)}$ be resource node number $(k-2d-2)$.
  Thus, $r_{(-2d-2)} = (k-2d^2-2d, k-2d^2-4d-2)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (2, 2d^2+2) = & r_{(-2d-2)} & \textit{when } k = 2d^2+2d+2 \\ (3, 2d^2+4) = & r_{(-2d-2)} & \textit{when } k = 2d^2+2d+3 \\ \vdots & \vdots & \vdots \\ (d+1, 2d^2+2d) = & r_{(-2d-2)} & \textit{when } k = 2d^2+3d+1 \end{bmatrix}$$

$, 1 \le i \le d.$

Furthermore, let $r_{(-2d-1)}$ be resource node number $(k - 2d - 1)$.
Thus, $r_{(-2d-1)} = (k - 2d^2 - d, k - 2d^2 - 3d - 1)$.
Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (d+2, 2d^2+d+3) = & r_{(-2d-1)} & \textit{when } k = 2d^2+2d+2 \\ (d+3, 2d^2+d+5) = & r_{(-2d-1)} & \textit{when } k = 2d^2+2d+3 \\ \vdots & \vdots & \vdots \\ (2d, 0) = & r_{(-2d-1)} & \textit{when } k = 2d^2+3d+1 \end{bmatrix}$$

$, 1 \le i \le d.$

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i \ge j-1 \\ V_2[i], & i < j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \le (d+1)$, $1 \le i \le d$ and $1 \le j \le (d+2)$. Hence, every node in **Class 3** is within a distance of $(d+1)$ or less from at least one resource node.

- **Class 4:** $\{ (x,y) : y - x = (d+2),\ 0 \le y \le (d+1),\ \textit{and}\ (-d-2) \le x \le -1 \}$. Define a matrix M of i rows and j columns to be:

$$M = \begin{bmatrix} (-1,d+1) & (-2,d) & (-3,d-1) & \dots & (-d,2) & (-d-1,1) & (-d-2,0) \\ (-1,d+1) & (-2,d) & (-3,d-1) & \dots & (-d,2) & (-d-1,1) & (-d-2,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (-1,d+1) & (-2,d) & (-3,d-1) & \dots & (-d,2) & (-d-1,1) & (-d-2,0) \end{bmatrix}$$

$, 1 \leq i \leq d, 1 \leq j \leq (d+2).$

Note that $M[i,j]$ is a unique node in **Class 4** when $k = i + 2d^2 + 2d + 1$, $x = -j$, and $y = d + 2 - j$. More clearly, the i^{th} row represents all the nodes of **Class 4** in a $k \times k$ torus, $k = 2d^2 + 2d + 1 + i$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (2d^2+2d+1,d+1) & (2d^2+2d,d) & \dots & (2d^2+d+1,1) & (2d^2+d,0) \\ (2d^2+2d+2,d+1) & (2d^2+2d,d) & \dots & (2d^2+d+2,1) & (2d^2+d+1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (2d^2+3d,d+1) & (2d^2+3d-1,d) & \dots & (2d^2+2d,1) & (2d^2+2d-1,0) \end{bmatrix}$$

Let $r_{(2d+2)}$ be resource node number $(2d+2)$.
Thus, $r_{(2d+2)} = (2d^2+2d, 2d^2+4d+2)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (2d^2+2d,2d) = & r_{(2d+2)} & \textit{when } k = 2d^2+2d+2 \\ (2d^2+2d,2d-1) = & r_{(2d+2)} & \textit{when } k = 2d^2+2d+3 \\ \vdots & \vdots & \vdots \\ (2d^2+2d,d+1) = & r_{(2d+2)} & \textit{when } k = 2d^2+3d+1 \end{bmatrix}$$

$, 1 \leq i \leq d.$

Furthermore, let $r_{(2d+1)}$ be resource node number $(2d+1)$.
Thus, $r_{(2d+1)} = (2d^2+d, 2d^2+3d+1)$.
Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (2d^2+d, d-1) = & r_{(2d+1)} & \textit{when } k = 2d^2+2d+2 \\ (2d^2+d, d-2) = & r_{(2d+1)} & \textit{when } k = 2d^2+2d+3 \\ \vdots & \vdots & \vdots \\ (2d^2+d, 0) = & r_{(2d+1)} & \textit{when } k = 2d^2+3d+1 \end{bmatrix}$$

, $1 \le i \le d$.

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i \ge j-1 \\ V_2[i], & i < j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \le (d+1)$, $1 \le i \le d$ and $1 \le j \le (d+2)$. Hence, every node in **Class 4** is within a distance of $(d+1)$ or less from at least one resource node.

Thus, any node is within a distance of $(d+1)$ or less from at least one resource node. ■

Theorem 2.2.10 *Any node in a $k \times k$ torus is within a distance of $(d+1)$ from at least one resource node when QP_k is used, where $2d^2+3d+2 \le k \le 2d^2+4d+2$.*

Proof: Suppose a node a is at a distance of $(d+2)$ from the resource node $(0,0)$. It will be shown that the node a is at a distance of $(d+1)$ or less from some other resource nodes. Thus, based on Lemma 2.2.5 and Lemma 2.2.6, this theorem follows.

Any node at a distance of $(d+2)$ from $(0,0)$ must be in one of the following classes:

- **Class 1:** $\{\ (x,y) : x+y=(d+2),\ 1 \le y \le (d+2),\ and\ 0 \le x \le (d+1)\ \}$.
  Let $r_1 = (d,\ d+1)$. Obviously, $r_1 \in QP_k$, and

$$D_L(r_1,(x,y)) \le (d+1).$$

- **Class 2:** $\{\ (x,y) : -x-y=(d+2),\ (-d-2) \le y \le -1,\ and\ (-d-1) \le x \le 0\ \}$.
  Let $r_{-1} = (-d,\ -d-1)$. Obviously, $r_{(-1)}$ is a resource node, and

$$D_L(r_{(-1)},\ (x,y)) \le (d+1).$$

- **Class 3:** $\{\ (x,y) : x-y=(d+2),\ (-d-1) \le y \le 0,\ and\ 1 \le x \le (d+2)\ \}$.
  Define a matrix M of i rows and j columns to be:

$$M = \begin{bmatrix} (1,-d-1) & (2,-d) & (3,1-d) & \dots & (d,-2) & (d+1,-1) & (d+2,0) \\ (1,-d-1) & (2,-d) & (3,1-d) & \dots & (d,-2) & (d+1,-1) & (d+2,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,-d-1) & (2,-d) & (3,1-d) & \dots & (d,-2) & (d+1,-1) & (d+2,0) \end{bmatrix}$$

$,\ 1 \le i \le (d+1),\ 1 \le j \le (d+2).$

Note that $M[i,j]$ is a unique node in **Class 3** when $k = i + 2d^2 + 3d + 1$, $x = j$, and $y = j - d - 2$. More clearly, the i^{th} row represents all the nodes of **Class 3** in a $k \times k$ torus, $k = i + 2d^2 + 3d + 1$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (1,2d^2+2d+1) & (2,2d^2+2d+2) & \dots & (d+1,2d^2+3d+1) & (d+2,0) \\ (1,2d^2+2d+2) & (2,2d^2+2d+3) & \dots & (d+1,2d^2+3d+2) & (d+2,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (1,2d^2+3d+1) & (2,2d^2+3d+2) & \dots & (d+1,2d^2+4d+1) & (d+2,0) \end{bmatrix}$$

Let $r_{(-2d-3)}$ be resource node number $(k-2d-3)$.
Thus, $r_{(-2d-3)} = (k-2d^2-3d, k-2d^2-5d-3)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (2, 2d^2+d+1) = & r_{(-2d-3)} & \textit{when } k = 2d^2+3d+2 \\ (3, 2d^2+d+3) = & r_{(-2d-3)} & \textit{when } k = 2d^2+3d+3 \\ \vdots & \vdots & \vdots \\ (d+2, 2d^2+3d+1) = & r_{(-2d-3)} & \textit{when } k = 2d^2+4d+2 \end{bmatrix}$$

$, 1 \le i \le (d+1)$.

Furthermore, let $r_{(-2d-2)}$ be resource node number $(k-2d-2)$.
Thus, $r_{(-2d-2)} = (k-2d^2-2d, k-2d^2-4d-2)$.
Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (d+2, 2d^2+2d+2) = & r_{(-2d-2)} & \textit{when } k = 2d^2+3d+2 \\ (d+3, 2d^2+2d+4) = & r_{(-2d-2)} & \textit{when } k = 2d^2+3d+3 \\ \vdots & \vdots & \vdots \\ (2d+2, 0) = & r_{(-2d-2)} & \textit{when } k = 2d^2+4d+2 \end{bmatrix}$$

$, 1 \le i \le (d+1)$.

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i \ge j-1 \\ V_2[i], & i < j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \le (d+1)$, $1 \le i \le (d+1)$ and $1 \le j \le (d+2)$. Hence, every node in **Class 3** is within a distance of $(d+1)$ or less from at least one resource node.

- **Class 4:** $\{\ (x,y) : y-x=(d+2),\ 0 \le y \le (d+1),\ \textit{and}\ (-d-2) \le x \le -1\ \}$. Define a matrix M of i rows and j columns to be:

$$M = \begin{bmatrix} (-1,d+1) & (-2,d) & (-3,d-1) & \dots & (-d,2) & (-d-1,1) & (-d-2,0) \\ (-1,d+1) & (-2,d) & (-3,d-1) & \dots & (-d,2) & (-d-1,1) & (-d-2,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ (-1,d+1) & (-2,d) & (-3,d-1) & \dots & (-d,2) & (-d-1,1) & (-d-2,0) \end{bmatrix}$$

, $1 \le i \le (d+1)$, $1 \le j \le (d+2)$.

Note that $M[i,j]$ is a unique node in **Class 4** when $k = i + 2d^2 + 3d + 1$, $x = -j$, and $y = d + 2 - j$. More clearly, the i^{th} row represents all the nodes of **Class 4** in a $k \times k$ torus, $k = 2d^2 + 3d + 1 + i$. Hence, M can be rewritten by adding the corresponding k to the negative terms as follows:

$$M = \begin{bmatrix} (2d^2+3d+1,d+1) & (2d^2+3d,d) & \dots & (2d^2+2d+1,1) & (2d^2+2d,0) \\ (2d^2+3d+2,d+1) & (2d^2+3d,d) & \dots & (2d^2+2d+2,1) & (2d^2+2d+1,0) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ (2d^2+4d+1,d+1) & (2d^2+4d,d) & \dots & (2d^2+3d+1,1) & (2d^2+3d,0) \end{bmatrix}$$

Let $r_{(2d+3)}$ be resource node number $(2d+3)$.
Thus, $r_{(2d+3)} = (2d^2+3d, 2d^2+5d+3)$. Define V_1 to be a vector of i elements:

$$V_1 = \begin{bmatrix} (2d^2+3d,2d+1) = & r_{(2d+3)} & \textit{when}\ k = 2d^2+3d+2 \\ (2d^2+3d,2d) = & r_{(2d+3)} & \textit{when}\ k = 2d^2+3d+3 \\ \vdots & \vdots & \vdots \\ (2d^2+3d,d+1) = & r_{(2d+3)} & \textit{when}\ k = 2d^2+4d+2 \end{bmatrix}$$

, $1 \le i \le (d+1)$.

Furthermore, let $r_{(2d+2)}$ be resource node number $(2d+2)$.
Thus, $r_{(2d+2)} = (2d^2+2d, 2d^2+4d+2)$.

Define V_2 to be a vector of i elements:

$$V_2 = \begin{bmatrix} (2d^2+2d, d) = & r_{(2d+2)} & \textit{when } k = 2d^2+3d+2 \\ (2d^2+2d, d-1) = & r_{(2d+2)} & \textit{when } k = 2d^2+3d+3 \\ \vdots & \vdots & \vdots \\ (2d^2+2d, 0) = & r_{(2d+2)} & \textit{when } k = 2d^2+4d+2 \end{bmatrix}$$

$, 1 \leq i \leq (d+1)$.

Define a matrix M' of i rows and j columns such that:

$$M'[i,j] = \begin{cases} V_1[i], & i \geq j-1 \\ V_2[i], & i < j-1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \leq (d+1)$, $1 \leq i \leq (d+1)$ and $1 \leq j \leq (d+2)$. Hence, every node in **Class 4** is within a distance of $(d+1)$ or less from at least one resource node.

Thus, any node is within a distance of $(d+1)$ or less from at least one resource node. ■

Theorem 2.2.11 *Any node in a $k \times k$ torus is within a distance of $(d+1)$ or less from at least one resource node when QP_k is used, where $k = 2d^2+4d+3$.*

Proof: Suppose a node a is at a distance of $(d+2)$ from the resource node $(0,0)$. It will be shown that the node a is at a distance of $(d+1)$ or less from some other resource nodes. Thus, based on Lemma 2.2.5 and Lemma 2.2.6, this theorem follows.

Any node at a distance of $(d+2)$ from $(0,0)$ must be in one of the following classes:

- **Class 1:** $\{\ (x, y) : x + y = (d+2),\ 1 \le y \le (d+2),\ \mathit{and}\ 0 \le x \le (d+1)\ \}$.

  Let $r_1 = (d,\ d+1)$. Obviously, $r_1 \in QP_k$, and

$$D_L(r_1, (x, y)) \le (d+1).$$

- **Class 2:** $\{\ (x, y)\ :\ -x-y = (d+2),\ (-d-2) \le y \le -1,\ \mathit{and}\ (-d-1) \le x \le 0\ \}$.

  Let $r_{-1} = (-d,\ -d-1)$. Obviously, $r_{(-1)}$ is a resource node, and

$$D_L(r_{(-1)},\ (x, y)) \le (d+1).$$

- **Class 3:** $\{\ (x, y) : x - y = (d+2),\ (-d-1) \le y \le 0,\ \mathit{and}\ 1 \le x \le (d+2)\ \}$.

  Define a matrix M of one row and j columns to be:

$$M = \begin{bmatrix} (1, -d-1) & (2, -d) & (3, 1-d) & \dots & (d, -2) & (d+1, -1) & (d+2, 0) \end{bmatrix}$$

  $, 1 \le j \le (d+2)$.

  Note that $M[1, j]$ is a unique node in **Class 3** when $k = 2d^2 + 4d + 3$, $x = j$, and $y = j - d - 2$. Hence, M can be rewritten by adding k to the negative terms as follows:

$$M = \begin{bmatrix} (1, 2d^2+3d+2) & (2, 2d^2+3d+3) & \dots & (d+1, 2d^2+4d+2) & (d+2, 0) \end{bmatrix}$$

  Let $r_{(-2d-3)}$ be resource node number $(k - 2d - 3)$.
  Thus, $r_{(-2d-3)} = (k - 2d^2 - 3d, k - 2d^2 - 5d - 3) = (d+3, 2d^2+3d+3)$.
  Furthermore, let $r_{(-1)}$ be resource node number $(k-1)$.
  Thus, $r_{(-1)} = (k-d, k-d-1) = (2d^2+3d+3, 2d^2+3d+2)$.

Define a matrix M' of one row and j columns such that:

$$M'[i,j] = \begin{cases} r_{(-2d-3)}, & j \neq 1 \\ r_{(-1)}, & j = 1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \leq (d+1)$, $i = 1$ and $1 \leq j \leq (d+2)$. Hence, every node in **Class 3** is within a distance of $(d+1)$ or less from at least one resource node.

- **Class 4:** $\{\ (x,y) : y - x = (d+2),\ 0 \leq y \leq (d+1),\ \mathit{and}\ (-d-2) \leq x \leq -1\ \}$. Define a matrix M of one row and j columns to be:

$$M = \left[\ (-1, d+1) \quad (-2, d) \quad (-3, d-1) \quad \ldots \quad (-d, 2) \quad (-d-1, 1) \quad (-d-2, 0)\ \right]$$

$, 1 \leq j \leq (d+2)$.

Note that $M[i,j]$ is a unique node in **Class 4** when $k = 2d^2 + 4d + 3$, $x = -j$, and $y = d + 2 - j$. Hence, M can be rewritten by adding k to the negative terms as follows:

$$M = \left[\ (-1, d+1) \quad (-2, d) \quad (-3, d-1) \quad \ldots \quad (-d, 2) \quad (-d-1, 1) \quad (-d-2, 0)\ \right]$$

$, 1 \leq j \leq (d+2)$.

Note that $M[1,j]$ is a unique node in **Class 4** when $k = 2d^2 + 4d + 3$, $x = j$, and $y = j - d - 2$. Hence, M can be rewritten by adding k to the negative terms as follows:

$$M = \left[\ (2d^2 + 4d + 2, d+1) \quad (2d^2 + 4d + 1, d) \quad \ldots \quad (2d^2 + 3d + 2, 1) \quad (2d^2 + 3d + 1, 0)\ \right]$$

Let $r_{(2d+3)}$ be resource node number $(2d+3)$.

Thus, $r_{(2d+3)} = (2d^2 + 3d, 2d^2 + 5d + 3) = (2d^2 + 3d, d)$.

Furthermore, let r_1 be the first resource node. Thus, $r_1 = (d, d + 1)$.

Define a matrix M' of one row and j columns such that:

$$M'[1,j] = \begin{cases} r_{(2d+3)}, & j \neq 1 \\ r_1, & j = 1 \end{cases}$$

$W_L(M[i,j] - M'[i,j]) \leq (d+1)$, $i = 1$ and $1 \leq j \leq (d+2)$.

Thus, any node is within a distance of $(d + 1)$ or less from at least one resource node. ■

Corollary 2.2.12 *QP_k is a quasi-perfect distance-d placement when* $2d^2 + 2d + 2 \leq k \leq 2d^2 + 4d + 3$.

Proof:

(1) In this range, the radius-d packing spheres of any two resources are disjoint (Corollary 2.2.4).

(2) Also in this range, any node is at a distance of $(d+1)$ or less from at least one resource node (Theorems 2.2.9, 2.2.10, and 2.2.11).

Hence, the two conditions required for a quasi-perfect distance-d placement are satisfied. ■

2.3 Quasi-Perfect Placements for 2D Tori

The QP_k placement scheme is designed to define a placement for a $k \times k$ torus using k resources, where k is an integer ≥ 2. In this section, it is shown when and how QP_k can be used in finding placements for other $X \times Y$ tori.

Lemma 2.3.1 *let $T_{X\times Y}$ be an $X \times Y$ torus. Let $k|X$ and $k|Y$, where k is an integer ≥ 2. Furthermore, let $T_{k\times k}$ be a $k \times k$ torus in which QP_k is used to achieve a perfect or quasi-perfect distance-l placement. Then $T_{k\times k}$ can be used to tile $T_{X\times Y}$. In this case, $T_{X\times Y}$ will maintain a perfect or quasi-perfect distance-l placement exactly as $T_{k\times k}$. The number of resources needed for $T_{X\times Y}$ is $\frac{XY}{k}$.*

Proof: The tiling blocks are identical. Thus, linking a block to an identical one is equivalent to wrapping around the links to the block itself. Hence, the placement characteristics of the basic tiling block must hold in the tiled $T_{X\times Y}$.

With respect to the number of resources needed, there are k resources in each $T_{k\times k}$. Furthermore, $\frac{XY}{k^2}$ tiling blocks are needed to tile $T_{X\times Y}$. Hence, the total number of the needed resources will be:

$$\frac{XY}{k^2}k = \frac{XY}{k}.$$

■

For example, there are several placements possibilities based on the QP_k scheme for a 30×30 or a 24×36 torus. Tables 2.1 and 2.2 show these possible placements.

The vertical or the horizontal bisection of some placements are quasi-perfect exactly as the placement of the whole space before bisecting. The following theorem formally explains the idea.

Theorem 2.3.1 *Let T be a $k \times k$ torus in which QP_k is used, $2|k$. Assume the resultant placement is quasi-perfect distance-l. Let $[d, (d+1)]$ be the generator matrix used for the placement.*

(1) If d is even then the horizontal bisection of T uses $\frac{k}{2}$ resources and maintains a quasi-perfect distance-l placement, assuming the bisections are $k \times \frac{k}{2}$ tori.

Placement	Tiling Block	Number of Needed Resources
Quasi-Perfect Distance-0	2×2	450
Quasi-Perfect Distance-0	4×4	225
Perfect Distance-1	5×5	180
Quasi-Perfect Distance-1	6×6	150
Quasi-Perfect Distance-1	10×10	90
Quasi-Perfect Distance-2	15×15	60

TABLE 2.1: The possible quasi-perfect placements for a 30×30 torus using QP_k placement scheme.

Placement	Tiling Block	Number of Needed Resources
Quasi-Perfect Distance-0	2×2	432
Quasi-Perfect Distance-0	4×4	216
Quasi-Perfect Distance-1	6×6	144
Quasi-Perfect Distance-1	12×12	72

TABLE 2.2: The possible quasi-perfect placements for a 24×36 torus using QP_k placement scheme.

(2) If d is odd then the vertical bisection of T uses $\frac{k}{2}$ resources and maintains a quasi-perfect distance-1 placement, assuming the bisections are $\frac{k}{2} \times k$ tori.

Proof: Let $i = \frac{k}{2}$. i will be a positive integer less than k, since $2|k$.

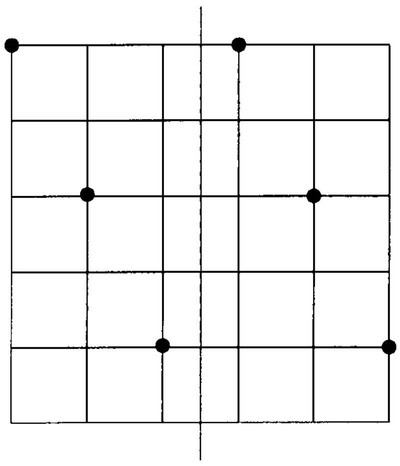
- If d is even then $\frac{kd}{2}$ $(mod\ k) = 0$. Hence $i(d, d+1) = (0, 0+i) = (0, \frac{k}{2})$ is a resource node. Thus, the horizontal bisections of T are identical in terms of resource node locations.

- If d is odd then $\frac{kd}{2}$ $(mod\ k) = \frac{k}{2}$. Hence, $i(d, d+1) = (\frac{k}{2}, \frac{k}{2}+\frac{k}{2}\ (mod\ k)) = (\frac{k}{2}, 0)$ is a resource node. Thus, the vertical bisections of T are identical in terms of resource node locations.

Since in both cases T will be partitioned into two identical parts, the links between these parts are equivalent to the wraparound links around each part itself. Furthermore, those parts being identical implies that the number of resources in each of them is $\frac{k}{2}$. ■

Examples of cases in which Theorem 2.3.1 holds are shown in Figure 2.5. In Figure 2.5-(a), QP_k is applied to achieve a quasi-perfect distance-1 placement. The generator matrix used in this case is $[1\ \ 2]$. Since $(d = 1)$ is odd and $2|(k = 6)$, the vertical bisection is quasi-perfect distance-1. Figure 2.5-(b) shows the placement results of using QP_k in a 10×10 torus . This placement is a quasi-perfect distance-1 and uses the generator matrix $[2\ \ 3]$. Since $(d = 2)$ is even and $2|(k = 10)$, the horizontal bisection is quasi-perfect distance-1.

From another perspective, these bisections can be used as tiling blocks. For example, Figure 2.6 shows how a 6×3 torus can be used in tiling a 9×12 torus. Note that the general conditions for using a tiling block in tiling a given torus are slightly different than what was stated in Lemma 2.3.1. Since the dimensions of a


(a) 6×6

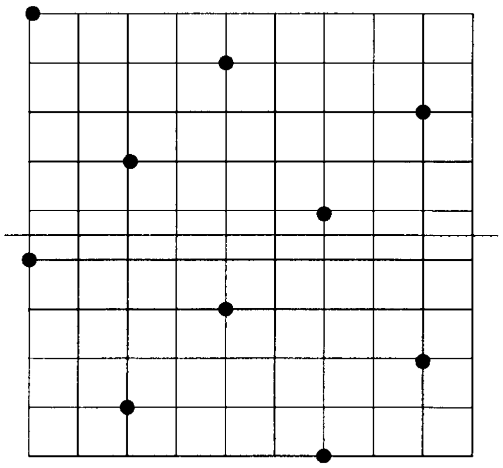

(b) 10×10

FIGURE 2.5: Quasi-perfect placements result from horizontal and vertical bisecting.

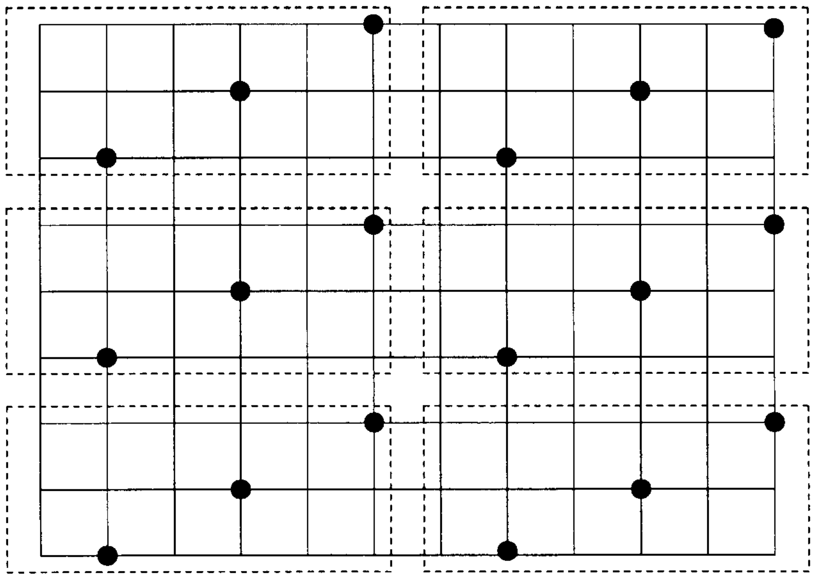


FIGURE 2.6: A 9 × 12 torus tiled by a vertical bisection of a 6 × 6 torus.

tiling block might not be equal, the conditions needed can be formally described as follows:

Given a tiling block, B, of dimensions k_1 and k_2, and a 2D torus, T, of dimensions X and Y, then B can be used to tile T if:

- $(k_1 \mid X)$ and $(k_2 \mid Y)$.

OR

- $(k_1 \mid Y)$ and $(k_2 \mid X)$.

2.4 Quasi-Perfect Placements for $2^i \times 2^i$ Tori

The 2D tori with dimension sizes as power of two are of special interest in industry. In this section, a special attention has been made on resource placements in a $k \times k$ torus, when $k = 2^i$. The main goal to be achieved is the ability of scaling the number of resources down or up for a given 2D torus in this class. The following theorem represents the basis for scaling the number of resources down.

Theorem 2.4.1 *Given T, a $k \times k$ torus, $k = 2^i$, i is an integer ≥ 1.*
$P = \{ (0,0), (\frac{k}{2}, \frac{k}{2}) \}$ is a quasi-perfect distance-$(\frac{k}{2} - 1)$ placement for T.

Proof: Let R_0 and R_1 be $(0,0)$ and $(\frac{k}{2}, \frac{k}{2})$, respectively.

(1) $D_L(R_0, R_1) = k \geq (k-1)$. Hence, the radius-$(\frac{k}{2} - 1)$ packing spheres of R_0 and R_1 are disjoint (First necessary condition for P to be a quasi-perfect distance-$(\frac{k}{2} - 1)$ placement).

(2) It is clear that P is a finite group and so is T. Hence, R_0 and R_1 can be transformed to each other as it was shown in Lemma 2.2.6. Thus, showing that every node at a distance of $(\frac{k}{2} + 1)$ from R_0 is at a distance of $(\frac{k}{2})$ or less

from R_1 proves that every node in T is within a distance of $(\frac{k}{2})$ or less from at least one of R_0 or R_1.

Any node at a distance of $(\frac{k}{2}+1)$ from R_0 can be written in the following form:

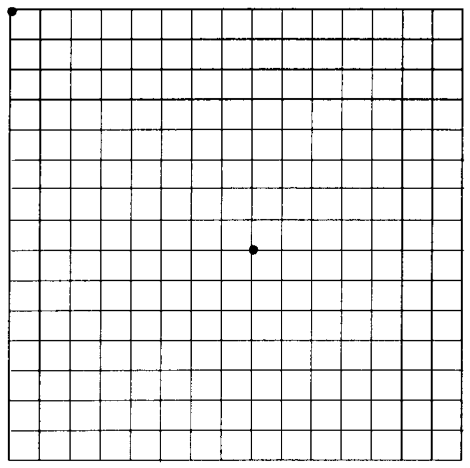
$$(x, y) \text{ such that } (\ min(x, k-x)\ +\ min(y, k-y)\ = \tfrac{k}{2}+1\ ),\ x, y \neq 0.$$

Note that assuming x to be zero makes $|y| = (\frac{k}{2}+1)$. This implies that (x, y) is at a distance of $(\frac{k}{2}-1)$ from $(0,0)$ (Contradiction since (x, y) must be at a distance of exactly $(\frac{k}{2}+1)$ from $(0,0)$). The same contradiction follows if y assumed to be zero. Therefore, for any (x, y) $D_L(R_1, (x, y)) = k - (\frac{k}{2}+1) < \frac{k}{2}$. Hence, any node in T is within a distance of $(\frac{k}{2})$ from at least one of R_0 or R_1 (Second necessary condition for P to be a quasi-perfect distance-$(\frac{k}{2}-1)$ placement).

■

Figure 2.7 shows an application of this theorem. Figure 2.7-(a) shows the use of two resources in a 16×16 torus to achieve a quasi-perfect distance-7 placement. On the other hand, Figure 2.7-(b) shows how to use eight resources and achieve a quasi-perfect distance-3 placement. The idea is to use two resources in an 8×8 torus to obtain a quasi-perfect distance-3 placement, and then, use the 8×8 torus in tiling 16×16 torus.

Unfortunately, finding a quasi-perfect placement using four resources in a $k \times k$ torus, when $k = 2^i$ and $i > 3$, seems to not be possible. However, proving this claim is still under investigation. Hence, at least for the current time, the number of resources to be used in the scaling down process for a torus in this class must be in the form $R = 2 \times 4^j, 0 \leq 2j < i-1$ (i.e. $R < 2^i$). The algorithm in Figure 2.8 formally describes the scaling down process.


(a) Quasi-Perfect Distance-7

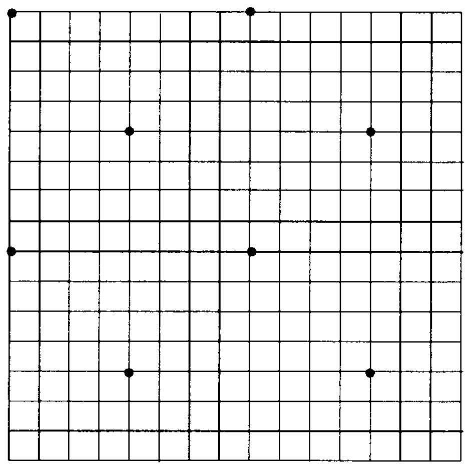

(b) Quasi-Perfect Distance-3

FIGURE 2.7: Quasi-perfect placements as applications of Theorem 2.4.1 in a 16×16 torus.

For example, a 32×32 torus can use different scaled-down quasi-perfect placements. Table 2.3 lists these placements and the number of resources needed in each case.

The idea of scaling up the number of resources is based on using a tiling block in which QP_k is used. For example, it is possible to use an 8×8 torus in which QP_k is used to tile a 16×16 torus. In this case, the latter will maintain a quasi-perfect distance-1 placement using 32 resources. The algorithm in Figure 2.9 describes a possible method for scaling up the number of resources in a torus of this class. Table 2.4 lists the possible scaled-up quasi-perfect placements for a 32×32 torus based on the PLACE-SCALED-UP algorithm. Finally, it should be noticed that any placement that results from applying the PLACE-SCALED-DOWN or the PLACE-SCALED-UP algorithm can be used as a tiling block. For example, a 32×32 torus

```
PLACE-SCALED-DOWN ( $A$, $B$, $R$, $k$ )

$A$: upper left node of the given torus.
$B$: lower right node of the given torus.
$R$: number of resources, $R = 2 \times 4^j, 0 \leq 2j < i - 1$ (i.e. $R < k$).
$k$: dimension, $k = 2^i, i > 0$.

(1) If $R = 2$
        place resource$_0$ in $(0, 0)$.
        place resource$_1$ in $(\frac{k}{2}, \frac{k}{2})$.
(2) Else
        CALL PLACE-SCALED-DOWN ( $(0,0)$ , $(\frac{k}{2}-1, \frac{k}{2}-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
        CALL PLACE-SCALED-DOWN ( $(\frac{k}{2},0)$ , $(k-1, \frac{k}{2}-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
        CALL PLACE-SCALED-DOWN ( $(0,\frac{k}{2})$ , $(\frac{k}{2}-1, k-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
        CALL PLACE-SCALED-DOWN ( $(\frac{k}{2},\frac{k}{2})$ , $(k-1, k-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
```

FIGURE 2.8: PLACE-SCALED-DOWN algorithm.

Placement	Number of Needed Resources
Quasi-Perfect Distance-15	2
Quasi-Perfect Distance-7	8

TABLE 2.3: Scaled-down quasi-perfect placements for a 32×32 torus that result from applying PLACE-SCALED-DOWN algorithm.

```
PLACE-SCALED-UP ( $A$, $B$, $R$, $k$ )

$A$: upper left node of the given torus.
$B$: lower right node of the given torus.
$R$: number of resources, $R = 2^j k, 0 \leq j \leq i$ (i.e $k \leq R \leq k^2$).
$k$: dimension, $k = 2^i, i > 0$.

(1) If $R = k$
        Use $QP_k$.
(2) Else
        CALL PLACE-SCALED-UP ( $(0,0)$ , $(\frac{k}{2}-1, \frac{k}{2}-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
        CALL PLACE-SCALED-UP ( $(\frac{k}{2},0)$ , $(k-1, \frac{k}{2}-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
        CALL PLACE-SCALED-UP ( $(0,\frac{k}{2})$ , $(\frac{k}{2}-1, k-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
        CALL PLACE-SCALED-UP ( $(\frac{k}{2},\frac{k}{2})$ , $(k-1, k-1)$, $\frac{R}{4}$, $\frac{k}{2}$ ).
```

FIGURE 2.9: PLACE-SCALED-UP algorithm.

Placement	Tiling Block	Number of Resources
Quasi-Perfect Distance-0	2×2	512
Quasi-Perfect Distance-0	4×4	256
Quasi-Perfect Distance-1	8×8	128
Quasi-Perfect Distance-2	16×16	64

TABLE 2.4: Possible scaled-up quasi-perfect placements for a 32×32 torus based on PLACE-SCALED-UP algorithm.

can be used to tile a 64×96 torus. Hence, any of the placements listed in Table 2.3 or Table 2.4 can be achieved in a 64×96 torus.

2.5 Conclusion

QP_k is a placement scheme for $k \times k$ tori. In this chapter, QP_k has been defined and proven to maintain the following properties:

- QP_k is a quasi-perfect distance-$(d-1)$ placement, when $2d^2+2 \leq k \leq 2d^2+2d$ (Corollary 2.2.8).
- QP_k is a perfect distance-d placement, when $k = 2d^2+2d+1$ (Corollary 2.2.5).
- QP_k is a quasi-perfect distance-d placement, when $2d^2+2d+2 \leq k \leq 2d^2+4d+3$ (Corollary 2.2.12).

Furthermore, it has been shown when and how to use the QP_k in finding placements for certain classes of 2D tori. The general distance-d placement problem of 2D tori is still under investigation. This problem can be stated as: "Given a 2D torus and R resources, how to place the R resources and minimize the maximum distance between a non-resource node and its closest resource node?"

Chapter 3

I/O PLACEMENTS AND NETWORK LATENCIES

The low cost of the high performance microprocessors makes building massively parallel supercomputers very cost effective. In this trend, several systems were built including Intel-Paragon, Cray, and IBM SP. However, the throughput of an Input/Output (I/O) device is too poor when it is compared to the Processing Element's (PE). This problem prevents utilizing the full power of a parallel computer. For this reason, many studies have tried to balance the throughputs of I/O devices and PE's [38, 59, 63, 62, 73]. In this chapter, it is shown that the I/O placement strategy used in a multicomputer has a major effect on the average network latency. The I/O placement strategy is a method by which I/O devices are distributed over the interconnection network. In practice, a simple I/O placement is usually used. For example, in the Intel-Paragon, whose interconnection network is a two-dimensional mesh, service nodes are placed at the boundary as shown in Figure 3.1. This placement is called Outer-Column (OC). A major problem with OC is the potentiality of *hot spot* growth. A hot spot is a part of the network in which traffic is concentrated. When OC placement is used, it is quite possible that the column in which service nodes are placed would become a hot spot. As described in [33, 59], hot spot traffic degrades performance of the entire network. Any message that traverses through the hot spot has a high probability of getting blocked while other parts of the network are not being utilized.

A uniform distribution of the network traffic guarantees a higher utilization of its links and, hence, a better performance [15, 16, 17, 18]. One factor that effects traffic

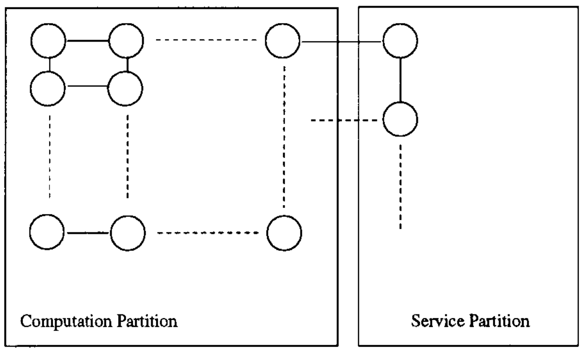


FIGURE 3.1: Intel-Paragon

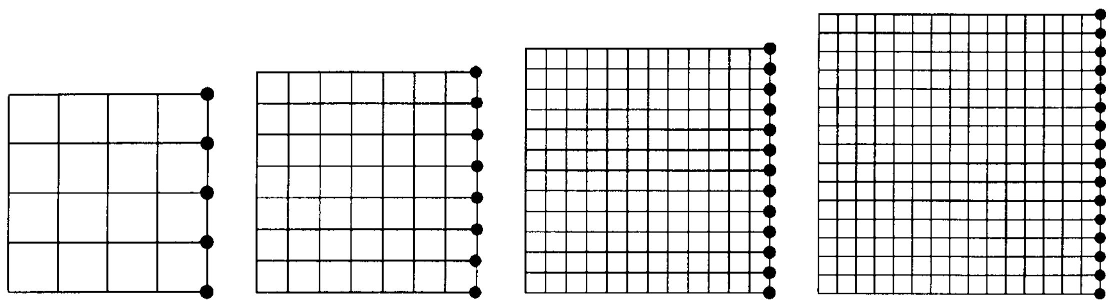
distribution is the I/O placement strategy. A placement strategy that uniformly distributes service-request traffic, (or simply I/O traffic), in 2D toroidal networks is Perfect/Quasi-perfect, (QP_k or simply QP). In this chapter, the performances of OC and QP are compared in terms of average network latency [4, 5].

The rest of this chapter is divided into five sections. Section 3.1 presents the derivations of average distance an I/O request traverses in OC and in QP. Section 3.2 describes the workload characteristics and the simulation environment. Simulation results are shown in Sections 3.3 and 3.4. Finally, conclusions are highlighted in Section 3.5.

3.1 Average Distance Analysis

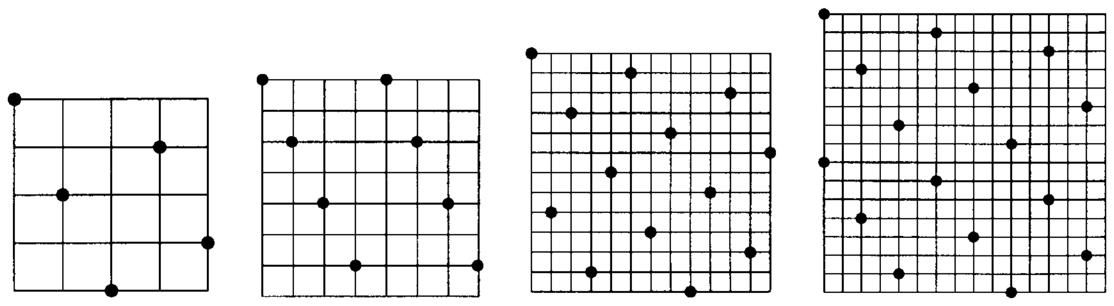
The network is assumed to use *wormhole* routing instead of *store-and-forward.* Under this assumption, the time, t, needed to send a message from A to B can be computed as follows:

$$t_{A,B} = t_c(d + \frac{l}{w}) \tag{3.1}$$



(a) 5×5 torus (b) 8×8 torus (c) 13×13 torus (d) 16×16 torus

FIGURE 3.2: Outer-Column I/O placements for the architectures simulated in this study. (wraparound links are not shown).



(a) 5×5 torus (b) 8×8 torus (c) 13×13 torus (d) 16×16 torus

FIGURE 3.3: Perfect/Quasi-Perfect I/O placements for the architectures simulated in this study: (a) perfect distance-1 (b) quasi-perfect distance-1 (c) perfect distance-2 (d) quasi-perfect distance-2 (wraparound links are not shown).

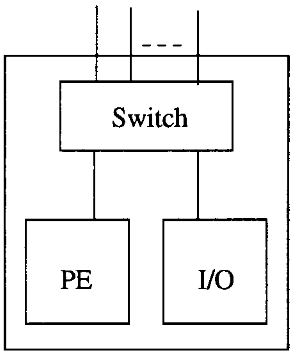


FIGURE 3.4: Architecture of PE-I/O node.

where t_c is the channel cycle time, d is the number of hops between A and B, l is the message length, and w is the channel width. Since OC and QP are compared in a set of equivalent networks, t_c, l, and w are the same in both cases, but d may vary. Therefore, the average traversal distances of an I/O request in OC and in QP are derived. The derivation is based on the following assumptions:

- I/O requests are divided into two types: Local and non-local I/O requests. Local I/O requests are forwarded to the nearest, or local, I/O node. Non-local I/O requests are uniformly distributed among all I/O nodes excluding the local one.

- Rates of local and non-local I/O requests are p and $(1 - p)$, respectively.

- The amount of local I/O requests is greater than or equal to the amount of non-local I/O requests forwarded to a single non-local I/O node. Formally, $\frac{1-p}{k-1} \leq p \leq 1$, where k is the number of I/O nodes.

- The network is a $k \times k$ torus, and the number of I/O nodes is k.

- Any I/O node embodies a PE as shown in Figure 3.4. This architecture has been described in [62].

Based on the above assumptions, the average traversal distance, d_{Avg}, of an I/O request is given by:

$$d_{Avg} = p(d_l) + (1-p)(d_n) \tag{3.2}$$

where d_l and d_n are the average traversal distances for local and non-local I/O requests, respectively. d_{Avg} is obtained by solving three subproblems: deriving d_l of OC, deriving d_l of QP, and deriving d_n which will be shown to be independent of the placement strategy. The rest of this section is divided into two subsections. Subsection 3.1.1 gives the derivations of d_l of OC and QP. Subsection 3.1.2 gives a derivation of d_n which is independent of the used placement strategy.

3.1.1 Average Traversal Distance of Local I/O Communication

The d_l of OC, $d_l(OC)$, is equal to the average distance of a node a in a ring of size k to all the nodes including a in this ring. Thus, $d_l(OC)$ of a $k \times k$ torus can be derived as follows:

Case1: k is odd:

$$d_l(OC) = \frac{2\sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} i}{k} = \frac{(\lfloor \frac{k}{2} \rfloor)(\lfloor \frac{k}{2} \rfloor + 1)}{k} = \frac{(\frac{k-1}{2})(\frac{k-1}{2}+1)}{k} = \frac{k^2-1}{4k} = \frac{k}{4} - \frac{1}{4k} \tag{3.3}$$

Case2: k is even:

$$d_l(OC) = \frac{(2\sum_{i=1}^{\frac{k}{2}} i) - \frac{k}{2}}{k} = \frac{(\frac{k}{2})(\frac{k}{2}+1) - \frac{k}{2}}{k} = \frac{k^2}{4k} = \frac{k}{4} \tag{3.4}$$

Deriving d_l of QP, $d_l(QP)$, is based on the following fact: The number of the unique nodes at a distance of exactly t hops from any node in a $k \times k$ torus is $4t$, given that

k	5*	8	13*	16	221*	256
$d_l(OC)$	1.24	2.00	3.23	8.00	55.25	64.00
$d_l(QP)$	0.8	1.25	1.54	1.81	6.97	7.52

TABLE 3.1: Values of $d_l(OC)$ and $d_l(QP)$ for different k's rounded to two significant digits. * indicates a perfect placement.

$t \leq \lfloor \frac{k-1}{2} \rfloor$. For a perfect-$t$ placement in a $k \times k$ torus, the number of nodes in the local sphere of a single I/O node is $(\sum_{i=1}^{t} 4i) + 1 = 2t^2 + 2t + 1$. Thus, $d_l(QP)$ of a $k \times k$ torus is given by:

Case1: perfect distance-t placement:

$$d_l(QP) = \frac{4\sum_{i=1}^{t} i^2}{2t^2 + 2t + 1} = \frac{4t(2t^2 + 3t + 1)}{6(2t^2 + 2t + 1)} = \frac{2t(k+t)}{3k} \tag{3.5}$$

Case 2: quasi-perfect distance-t placement:

$$d_l(QP) = \frac{(4\sum_{i=1}^{t} i^2) + (t+1)(k - (2t^2 + 2t + 1))}{k} = \frac{(t+1)(3k - 2t^2 - 4t - 3)}{3k} \tag{3.6}$$

Table 3.1 shows $d_l(OC)$ and $d_l(QP)$ for different k's. It is clear that QP is superior to OC in terms of reducing d_l. In addition, QP is superior in terms of link utilization. In OC, local I/O requests are sent/received only through horizontal links as shown in Figure 3.5-(a). However, in QP local I/O, requests are sent/received through both horizontal and vertical links, as illustrated in Figure 3.5-(b). This makes the interconnection network less content when QP is applied.

3.1.2 *Average Traversal Distance of Non-Local I/O Communication*

Making non-local I/O requests to be uniformly distributed among all I/O nodes including the local one, would simplify the derivation of d_n. Fortunately, this can

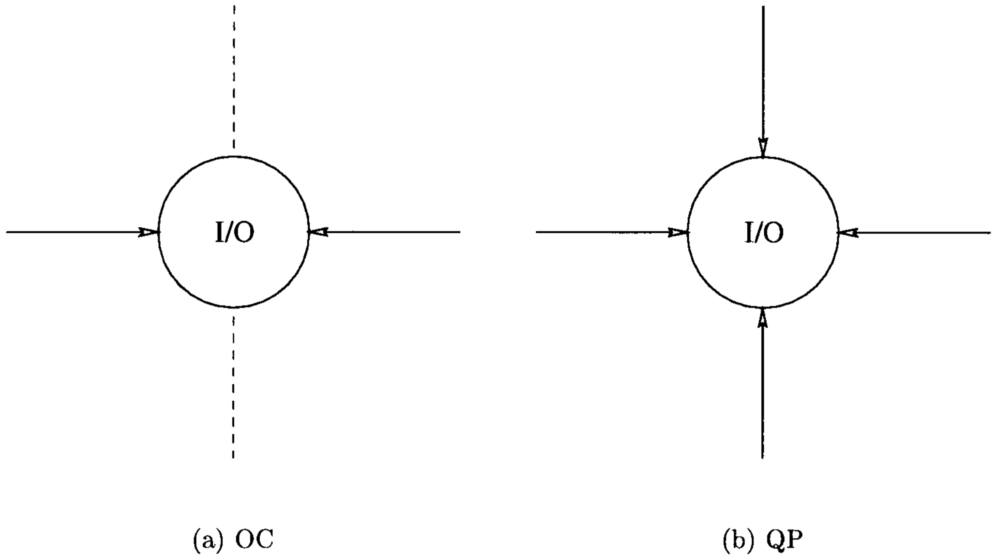


FIGURE 3.5: Utilized and non-utilized links, in terms of local I/O request flow, are represented by arrows and dashed lines, respectively.

be achieved easily by taking a portion of the local I/O requests and adding it to the non-local requests. Define p' as follows:

$$p' = p - \frac{1-p}{k-1}$$

According to the assumptions stated earlier, the following holds:

$$0 \leq p' \leq 1 - \tfrac{1-p}{k-1}$$
$$(1-p') = 1 - p + \tfrac{1-p}{k-1}$$

Hence, (3.2) can be rewritten as:

$$d_{Avg} = p'd_l + (1-p')(d_n) \tag{3.7}$$

Now, non-local I/O requests, of rate $(1 - p')$, are uniformly distributed among all I/O nodes including local one. Let $d_n(x)$ be:

$$d_n(x) = \frac{\sum_{i=1}^{k} \sum_{j=1}^{k} \ distance\ (x^{th}\ I/O\ node,\ node(i,j))}{k^2}$$

In this case, d_n is given by:

$$d_n = \frac{\sum_{i=1}^{k} d_n(x)}{k} \tag{3.8}$$

Because of symmetry of $d_n(i)$, (3.8) becomes:

$$d_n = d_n(x)\ for\ any\ i = 1, 2, \ldots, k \tag{3.9}$$

This proves that d_n does not depend on location of I/O nodes nor on their number.

Case 1: k is odd:

$$d_n = \frac{4k(\sum_{i=1}^{\lfloor \frac{k}{2} \rfloor} i)}{k^2} = \frac{k^2 - 1}{2k} = \frac{k}{2} - \frac{1}{2k} \tag{3.10}$$

Case 2: k is even:

$$d_n = \frac{k[(4\sum_{i=1}^{\frac{k}{2}} i) - k]}{k^2} = \frac{k^2}{2k} = \frac{k}{2} \tag{3.11}$$

This means that the performances of OC and QP will be the same if the rate of non-local I/O requests is high under the assumption of contention free network. However, OC suffers from the potentiality of hot-spot growth. The simulation results, to be shown in Sections 3.3 and 3.4, indicate that the non-local I/O requests in OC accelerate hot-spot growth. Hence, even in low data locality environments, QP is superior to OC.

3.2 Workload and Simulation Environment

OC and QP placements for simulated architectures are shown in Figures 3.2, and 3.3. A modified version of Proteus [20] is used to simulate 5×5, 8×8, 13×13, and

16×16 tori. 5×5 and 13×13 represent perfect cases, while 8×8 and 16×16 illustrate quasi-perfect placements. The simulation considers both transaction and scientific computing environments. It is assumed that the main distinction between these environments is in the overlapping of PE-PE communication and I/O requests. Also, it is assumed that computations in scientific environments go in disjoint cycles: computation cycle in which PE-PE communication is performed, and I/O cycle in which I/O requests are issued. Under this assumption, I/O requests and PE-PE communication do not overlap. Contrarily, in transaction environments, it is assumed that PE-PE communication and I/O requests do overlap.

The simulation is designed to work as follows: the nodes generate requests at random instants. The average time between two sequential instants, Inter-Request time (IR), is a metric for specifying network contention. Small IR's mean high volume of traffic and high contention, while large IR's indicate low volume of traffic and low contention. IR is assumed to be exponentially distributed. A specified percentage of generated requests is considered to be I/O requests. The rest of the requests are set to be PE-PE communication requests (abbreviated as PE requests in what follows). The destination of a PE request is selected from other PE nodes with a uniform probability. A percentage of I/O requests that matches data locality factor is forwarded to the closest I/O node. The rest of I/O requests are uniformly distributed among the other I/O nodes. Each I/O node is assumed to satisfy all the requests it receives [11, 65, 66, 64].

In scientific environment, I/O request percentage is set to 100% since it is assumed that there is no overlap between PE and I/O requests. In transaction environment, the percentage of I/O requests out of the total requests varies from 5% to 20% [62]. In this simulation, the I/O request percentage is set to 10%. In both environments, I/O request block and flit sizes are set to 4 kilobyte and 1 byte, respectively. PE request block size, however, may vary from 32 byte to 1 kilobyte [38, 62]. Therefore, simulating the transaction environment has been carried out for the cases in which PE request sizes are 32 byte and 1 kilobyte.

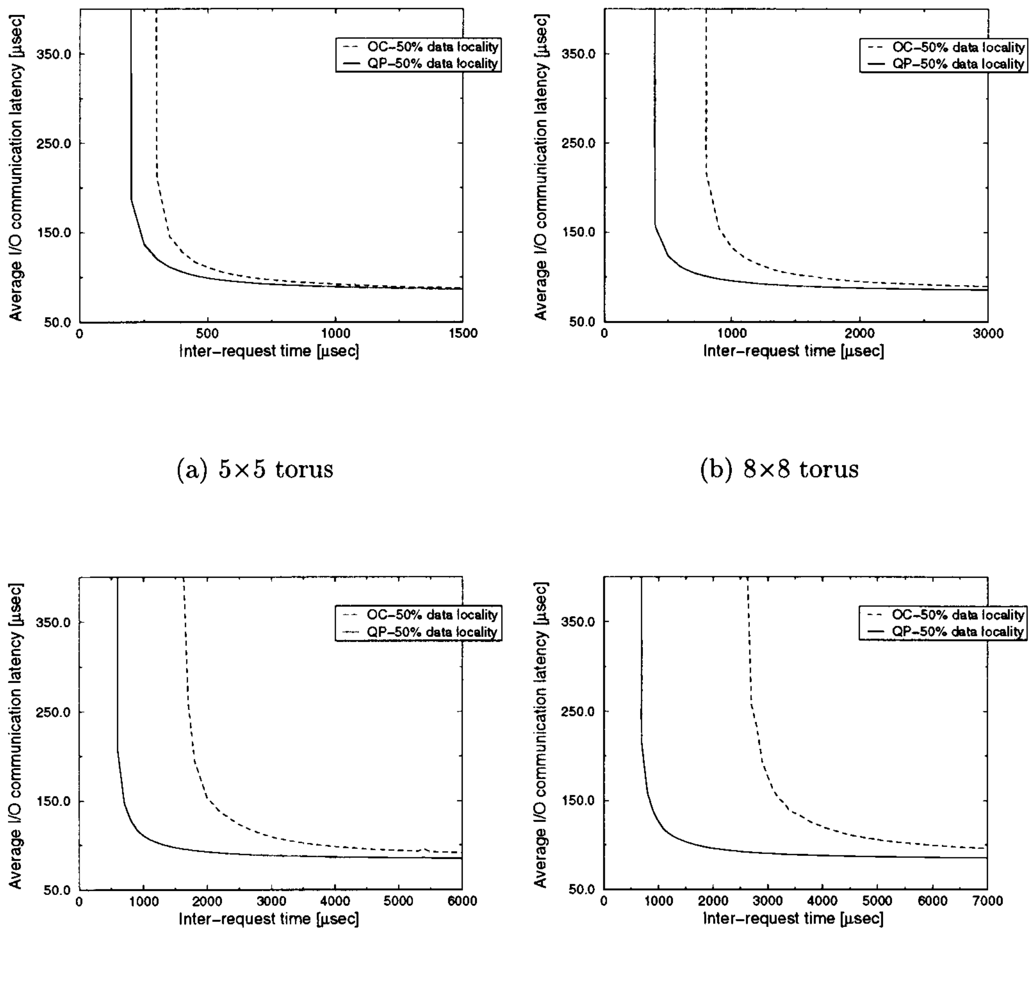
3.3 Scientific Environment Simulation Results

Figure 3.6 shows some of the simulation results. The data locality indicates how many of those I/O requests generated by any node will be forwarded to the local I/O node. For example, 40% data locality means that 40% of the requests generated in any node are forwarded to the nearest I/O node, while the remaining requests are uniformly distributed among the other I/O nodes. From Figure 3.6, the following observations can be made:

- QP, in general, has a less average I/O communication latency than OC.
- The gap between average network latencies of OC and QP increases as IR decreases.
- As dimension of interconnection network becomes larger, gaps between average network latencies of OC and QP increase. As shown in Section 3.1, the increase of interconnection network size causes average traversal distance of local I/O communication to increase faster in OC case than in QP case.

Figure 3.7 shows simulation results when data locality is set to 20% and 80%. From these results, we observe the following:

- Average network latencies of OC and QP are always less when data locality is high, than when it is low. Non-local I/O requests have to traverse longer distances and, hence, their chances to block or to be blocked are higher.



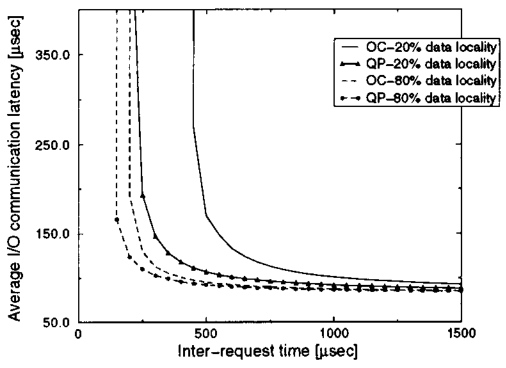
(a) 5×5 torus

(b) 8×8 torus

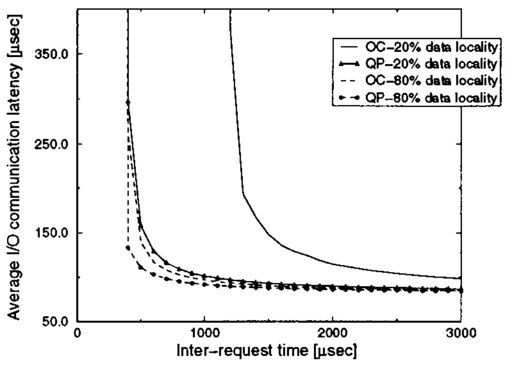
(c) 13×13 torus

(d) 16×16 torus

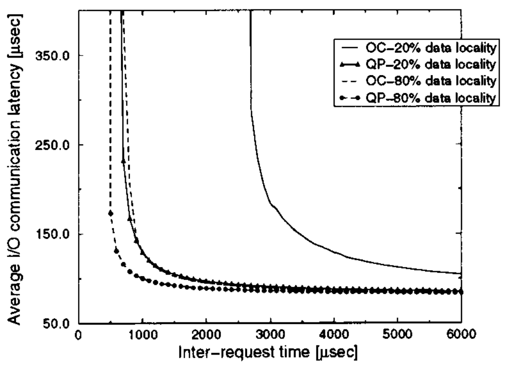
FIGURE 3.6: Inter-request time [μsec] vs. average I/O communication latency [μsec] when data locality is set to 50%.



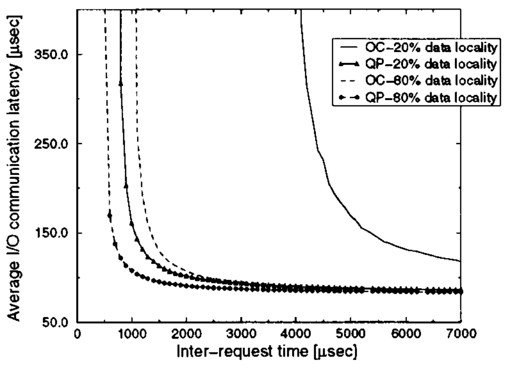
(a) 5×5 torus



(b) 8×8 torus



(c) 13×13 torus



(d) 16×16 torus

FIGURE 3.7: Inter-request time [μsec] vs. average I/O communication latency [μsec] when data locality is set to 20% and 80%.

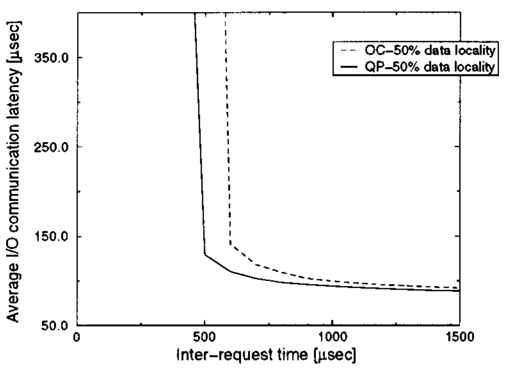
- Data locality sensitivity of QP is less than OC. This means that changing data locality in QP does not effect average latency as much as it does in OC.

- The gaps between OC-20% and QP-20% are larger than the gaps between OC-80% and QP-80%. This indicates that low data locality accelerates the hot-spot growth in OC case. As pointed above, the non-local I/O requests have a higher potentiality toward blocking or getting blocked. QP scatters non-local I/O requests uniformly since I/O nodes are uniformly distributed. This reduces the chance of a non-local I/O request to block or to be blocked. OC, on the other hand, concentrates I/O requests in one column. This increases the chance of a non-local I/O request to block or to be blocked.

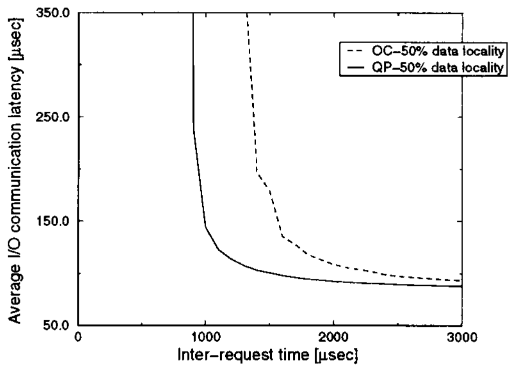
- Gap between OC-80% and QP-80% becomes greater in larger interconnection networks. This is also true for gap between OC-20% and QP-20%. This is because the average traversal distance of local I/O requests grows faster in the case of OC than that of QP.

3.4 Transaction Environment Simulation Results

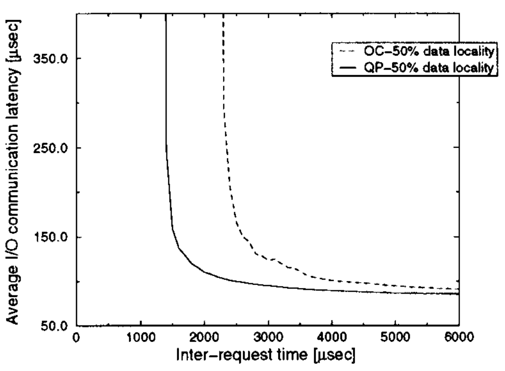
The simulation results of this environment are evaluated with respect to two metrics: average I/O communication latency, and average PE communication latency. It is desired to investigate the influence of I/O placement strategy on each of I/O and PE requests. Using average network latency, i.e. network entire traffic latency average, in evaluating the simulation results would have not satisfied the intended goal. PE communication is set to be 90% of the total communication in this environment. Hence, the average network latency is mainly influenced by the average PE communication latency. In this case, the influence of I/O placement method on I/O communication will not be noticed.



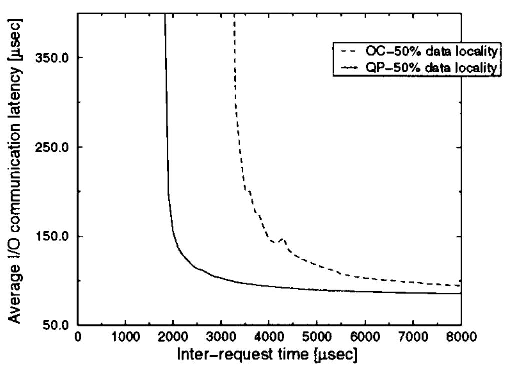
(a) 5×5 torus



(b) 8×8 torus

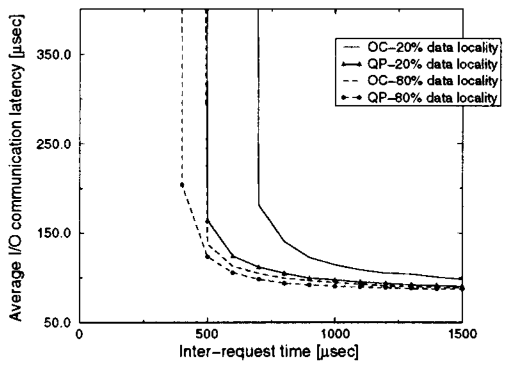


(c) 13×13 torus

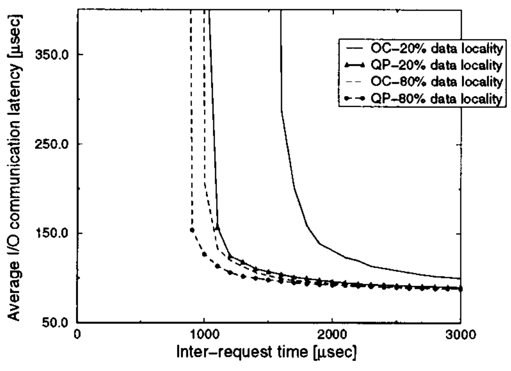


(d) 16×16 torus

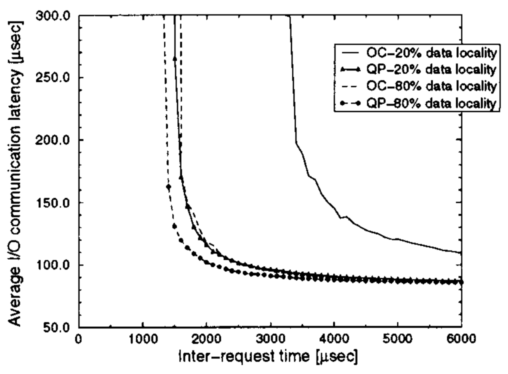
FIGURE 3.8: Inter-request time [μsec] vs. average I/O communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 50%.



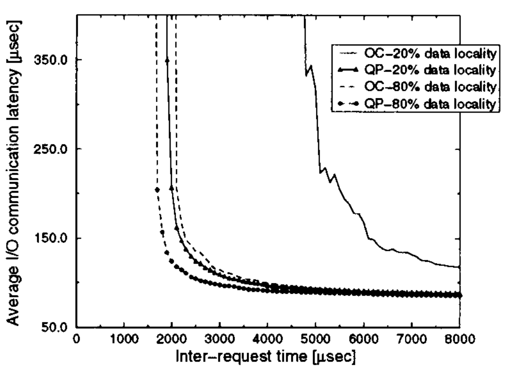
(a) 5×5 torus



(b) 8×8 torus

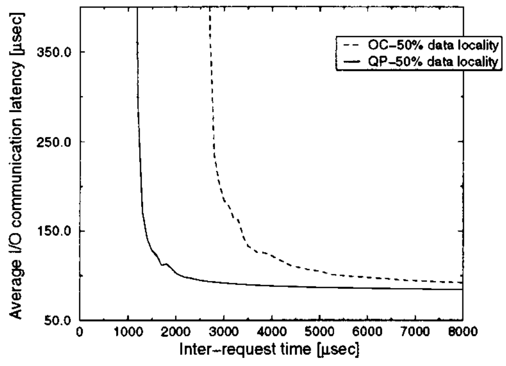


(c) 13×13 torus

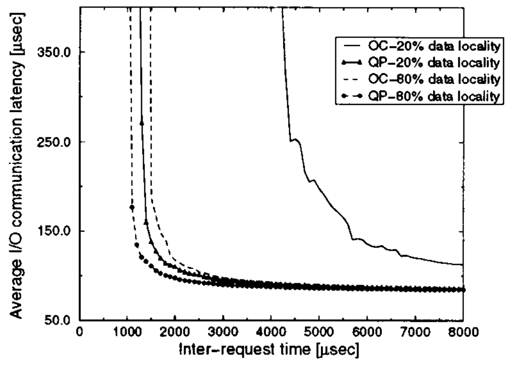


(d) 16×16 torus

FIGURE 3.9: Inter-request time [μsec] vs. average I/O communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 20% and 80%.



(a) 16 × 16 torus



(b) 16 × 16 torus

FIGURE 3.10: Inter-request time [μsec] vs. average I/O communication latency [μsec] when PE request block size is 32 byte.

The simulation has been carried out for the cases in which PE request sizes are set to 1 kilobyte and 32 byte. T-1k and T-32 will refer to transaction environments when PE request sizes are set to 1 kilobyte and 32 byte, respectively. Figures 3.8, and 3.9 show the simulation results with respect to average I/O communication latency in T-1k. From these figures, the following observations were made:

- All the observations were made in Section 3.3 hold for I/O communication in T-1k and T-32.

- In T-1k, even though PE request size is smaller than I/O requests which represent only 10% of the total communication, simulated networks saturate faster than the cases of scientific environments. In order to gain more understanding of this observation, T-32 was decided to be simulated. The results show that the simulated networks saturate almost in the same time in T-32 and scientific environment experiments. This indicates that the "over-uniform" distribution of the traffic may result in a higher network contention. Indeed, this observation needs further study and investigation. The results with respect to the average I/O latency of simulating a 16×16 torus in T-32 are shown in Figure 3.10.

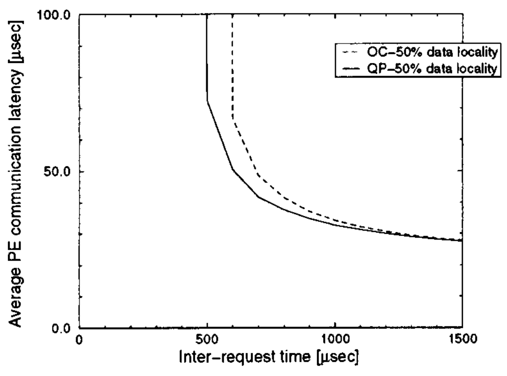
- The gaps between average I/O latencies of QP and OC in scientific environments are larger than those in T-1k and T-32. This is expected, since in this simulation, all the traffic of scientific environment is due to I/O requests. Therefore, the placement strategy has more influence on reducing the network latency.

Figures 3.11, and 3.12 show the results of simulating T-1k with respect to average PE network latency. Part of the results of simulating T-32 is shown in Figure 3.13. From these results, the followings are observed:

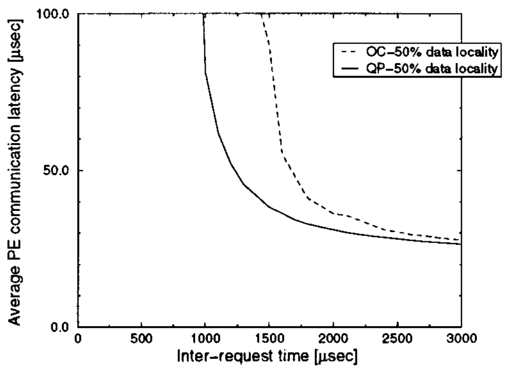
- QP has a less average PE-latency than OC in both of T-1k and T-32 experiments. This shows that I/O placement strategy influences the traffic of the entire network.

- All the observations of Section 3.4 hold here also, but with respect to average PE-latency.

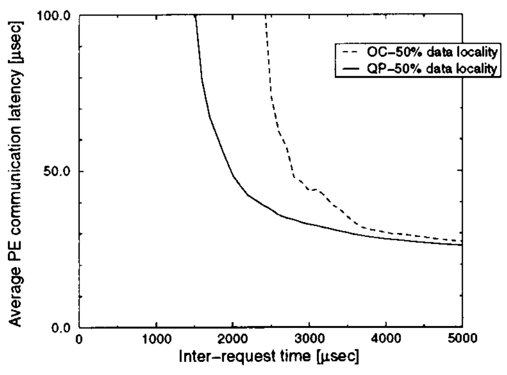
- The simulated networks saturate in T-1k faster than in T-32. This is expected since the larger size of PE-request in T-1k experiment causes more network contention.



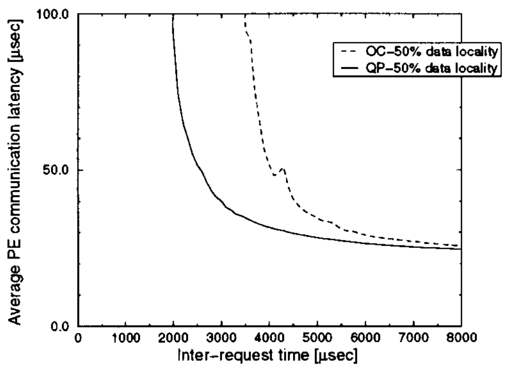
(a) 5×5 torus



(b) 8×8 torus

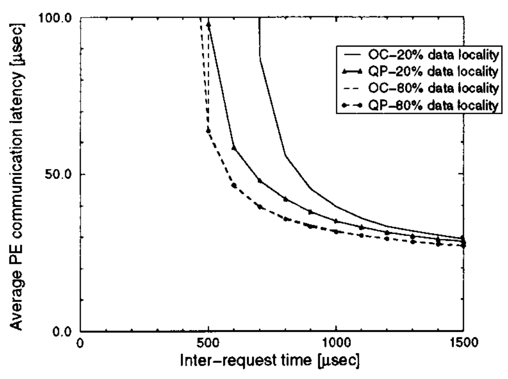


(c) 13×13 torus

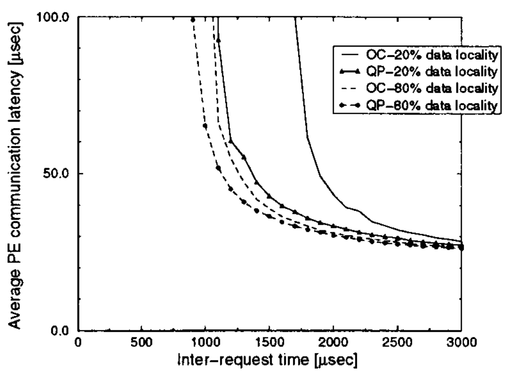


(d) 16×16 torus

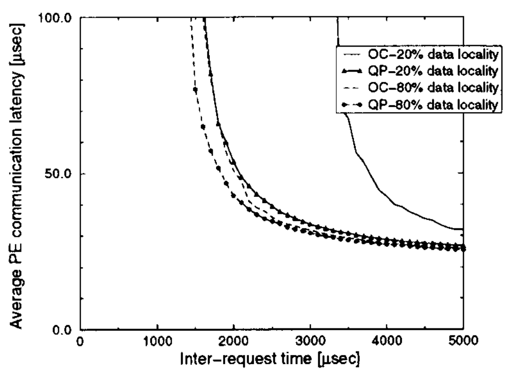
FIGURE 3.11: Inter-request time [μsec] vs. average PE communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 50%.



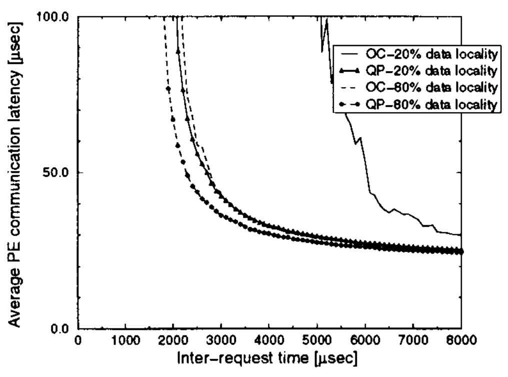
(a) 5×5 torus



(b) 8×8 torus

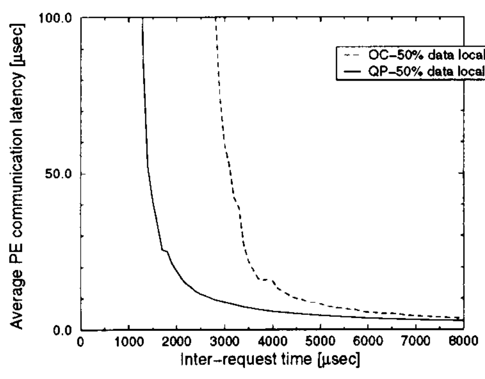


(c) 13×13 torus

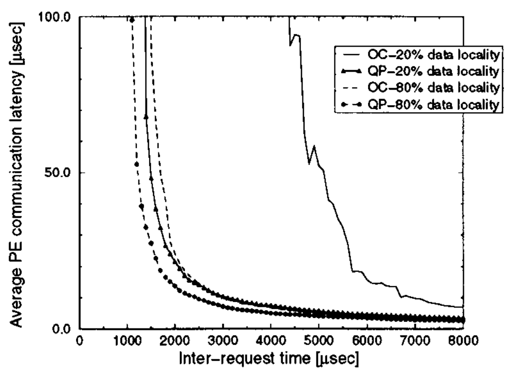


(d) 16×16 torus

FIGURE 3.12: Inter-request time [μsec] vs. average PE communication latency [μsec] when PE request block size is 1 kilobyte and data locality is set to 20% and 50%.



(a) 16 × 16 torus



(b) 16 × 16 torus

FIGURE 3.13: Inter-request time [μsec] vs. average PE communication latency [μsec] when PE request block size is 32 byte.

3.5 Conclusion

In this chapter, it is shown that the I/O placement strategy in a massively parallel computer influences interconnection network contention and average network latency even if worm hole routing is used.

Outer-Column (OC) and Perfect/Quasi-perfect (QP) I/O placement strategies are compared. The latter is proved to be superior to the former in terms of reducing average traversal distance of I/O requests. QP uniformly distributes I/O devices in an interconnection network, and hence, I/O requests are uniformly distributed. This increases the network links utilization and eliminates the potentiality of hot-spot growth from which OC suffers.

Performance evaluation shows that a $k \times k$ torus , in general, has a smaller or equal average message latency when QP is used rather than OC. This holds even if

I/O data locality varies. The gaps between average message latency of OC and QP increase with the increase of network size or network contention.

Chapter 4

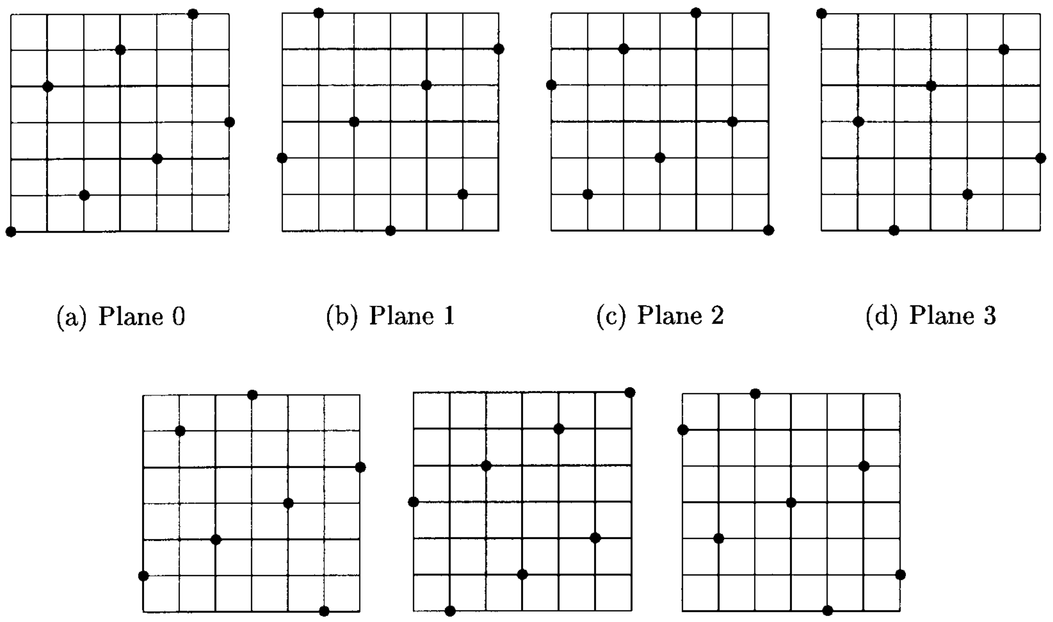
RESOURCE PLACEMENT FOR 3D TORI

The problem of distance-d resource placement is more challenging in 3D tori than in 2D ones. The only known regular perfect distance-d placements in 3D toroidal spaces are the perfect distance-1 placements for $(7i \times 7j \times 7k)$ tori, where i, j, and k are positive integers. In this chapter, three issues related to distance-d resource placements in 3D tori are discussed. First, the existence of linear perfect distance-1 placements for 3D tori is investigated. Second, irregular perfect placements for 3D tori are defined. Third, alternative resource placement approaches for 3D tori are introduced.

There are six sections in this chapter. Section 4.1 presents the previous related results. Section 4.2 shows the results of investigating the existence of linear perfect distance-1 placements. Section 4.3 defines irregular perfect distance-d placements. In section 4.4, the placements for 2D tori, proposed in Chapter 2, are extended to 3D tori. Sub-mesh-based resource placement approach for 3D tori is defined in Section 4.5. Finally, conclusions of this chapter are highlighted in Section 4.6.

4.1 Previous Work

The distance-d placements in 3D tori can be classified, as in 2D tori, into regular and irregular placements. As defined in Chapter 2, the volume of a distance-d packing sphere of any resource node (i.e. number of nodes at a distance of d or less from the given resource) in a regular distance-d placement must be of a full size. However, the volume of a distance-d packing sphere in an irregular placement is less than the full


(a) Plane 0 (b) Plane 1 (c) Plane 2 (d) Plane 3

(e) Plane 4 (f) Plane 5 (g) Plane 6

FIGURE 4.1: Regular perfect distance-1 placement in a $7 \times 7 \times 7$ torus.

size. The full volume size of a distance-d sphere can be derived from equation 2.2 and proved to be:

$$(\sum_{i=1}^{d} 4i^2 + 2) + 1 = \frac{(2d)(2d+1)(d+1)}{3} + 2d + 1.$$

Furthermore, it can be shown, using equation 2.1, that the full surface area of a radius-d sphere (i.e. number of nodes at a distance of exactly d from a given node) in a 3D torus is:

$$4d^2 + 2.$$

Examples of a regular and an irregular perfect placements are shown in Figures 4.1 and 4.2, respectively. As it can be noticed, the packing sphere of a resource node in a

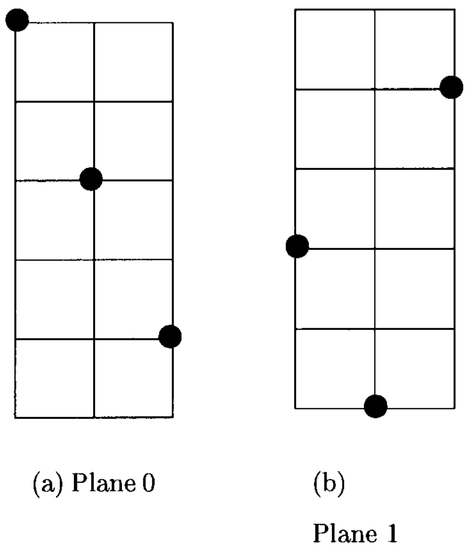


FIGURE 4.2: Irregular perfect distance-1 placement in a $2 \times 6 \times 3$ torus.

regular placement does not collapse into itself, while it does in an irregular case. All the dimensions in a regular distance-d placement must be greater than $(2d)$. In an irregular distance-d placement, however, at least one of the dimensions is less than $(2d+1)$.

The related previous results, according to my knowledge, have been achieved by: Lee, Golomb, Welch, Livingston, Stout, Bae, Bose, and Broeg. The related results can be summarized in the following:

- Lee's error correcting code can be used to achieve a regular perfect distance-1 placement in an $X \times Y \times Z$ torus if all of X, Y, and Z are divisible by 7 [13, 40, 8, 9, 10, 51]. The generator matrix for this code is:

$$G = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 2 \end{bmatrix}$$

- Golomb and Welch proved that a full size packing sphere, whose radius $= 2$, cannot tile a closed 3D toroidal space. They also showed that there exists an integer d such that a full size packing sphere of radius r, $r \geq d$, cannot tile a closed 3D toroidal space [40].

- Golomb and Welch conjectured that in a 3D toroidal space there exists no perfect t-error correcting codes, $t \geq 2$. However, I have discovered irregular perfect distance-d placements for 3D tori. These placements can be considered as error correcting codes (to be shown in Section 4.3). Still, this conjecture may hold for the regular perfect codes.

- Livingston and Stout have shown that there exists an irregular prefect distance-1 placement for an $X \times Y \times Z$ torus if [51]:

$$\{X, Y, Z\} \in \big\{\{2, 3i, 6j\}, i \text{ and } j \geq 1\big\}.$$

- Bae and Bose have defined an open related problem that can be rephrased as follows : "There exists a perfect distance-1 placement for a 3D torus if all of its dimensions are divisible by 7. However, is it necessary that all the dimensions be divisible by 7 for the existence of a perfect distance-1 placement?" [8, 9, 10]. Note that they have meant regular perfect distance-1 placements.

4.2 On Existence of Unknown Linear Perfect Distance-1 Placements

The known perfect distance-1 placements are:

(1) The regular perfect distance-1 placements for $(7i \times 7j \times 7k)$ tori, i, j, and k are positive integers (the basic tiling block is shown in Figure 4.1).

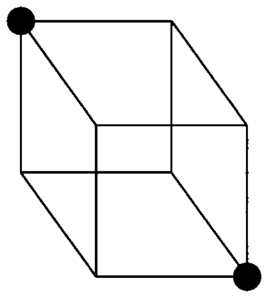


FIGURE 4.3: Irregular perfect distance-1 placement in a $2 \times 2 \times 2$ torus.

(2) The irregular perfect distance-1 placements for $(2 \times 3i \times 6j)$ tori, i and j are positive integers (the basic tiling block is shown in Figure 4.2).

(3) The irregular perfect distance-1 placement for a $(2 \times 2 \times 2)$ torus (shown in Figure 4.3).

It is clear that these placements are linear. Hence, they maintain several properties among which [52, 58, 75]:

(1) Adding any two resource nodes results in a resource node,

(2) Every resource node has an additive inverse which is a resource node as well.

These two properties will be used to prove the non-existence of unknown linear distance-1 perfect placements. This claim is proved for the regular and the irregular placements in Subsections 4.2.1 and 4.2.2, respectively.

4.2.1 On Existence of Irregular Linear Distance-1 Placements

A 3D torus that has an irregular perfect distance-1 placement must have at least one dimension equal to two. This gives three possibilities: All dimensions are equal to two, only two dimensions are equal to two, or only one dimension is equal to two.

The following lemma proves that if two dimensions are equal to two, then the third must be equal to two in order to have a perfect distance-1 placement.

Lemma 4.2.1 *Let T be a 3D torus that has a perfect distance-*1 *placement and two of its dimensions are equal to two. Then, the third dimension is equal to two.*

Proof: T can be viewed as a set of 2×2 planes. Each of these planes must have exactly one resource node for the following reason. If there were two or more resources in the same plane, some nodes in this plane would have been shared by more than a resource. On the other hand, if a plane had no resources, then there would have been at least two resources in the next upper or the next lower plane which is just shown to be impossible.

Assume that the third dimension in T is X, $X > 2$. Since each of the 2×2 planes must have exactly one resource, there will be at least one node in the first plane covered by at least two resources. Hence, X cannot be greater than two. ■

Theorem 4.2.1 *Let T be a 3D torus that has a linear perfect distance-*1 *placement, and only one of its dimensions, say Z, is equal to two. Then, the other two dimensions must be in $\{\{3i, 6j\} : i \text{ and } j \geq 1\}$.*

Proof: Let X and Y be the other two dimensions of T. T can be thought of as two planes of $X \times Y$ tori, say P_0 and P_1. In these two planes, there are four possibilities with respect to resources distribution:

(1) Every row in P_0 and P_1 has at least one resource. Furthermore, every column in P_0 and P_1 has at least one resource.

(2) There exists a row without any resources, but every column has at least one resource.

(3) There exists a column without any resources, but every row has at least one resource.

(4) There exist a row and a column without any resources.

In what follows, it is proved that:

- When Case(1) holds, then X and Y must be divisible by 6.

- When Case(2) or (3) holds, then $\{X, Y\}$ must be in:

$$\{\{3i,\ 6j\ \} : \ i \ and \ j \ are \ integers \ > 0\}.$$

- Case(4) is impossible.

Case(1)

Since the placement is linear, every row must have the same number of resources. This is true because of the following reason. Assume r_{ip}, i.e. the ith row in plane p, has the maximum number of resources, say R, among all the rows in both planes. Add the additive inverse of one resource in r_{ip} to all the resources in this row. This results in at least R resources in r_{00}. Since r_{00} cannot have more than R resources, r_{00} has exactly R resources. Adding the resources of r_{00} to a resource in $r_{jp'}$ results in at least R resources in $r_{jp'}$. Again, since $r_{jp'}$ cannot have more than R resources, $r_{jp'}$ has exactly R resources. Since every row has at least one resource then every row has R resources. The same can be proved with respect to the columns. Note that this does not imply that a row and a column must have the same number of resources.

Since every row has the same number of resources, each resource has three correspondent resources in the three adjacent rows (the upper and the lower rows in the same plane, and the same row in the other plane). Hence, for each resource a in every row exactly six nodes are needed. Three are covered by a, and three are covered by the correspondent resources of a in the adjacent rows. Therefore, the number of nodes in each row must be divisible by 6. The same can be said about the columns. Hence, X and Y must be divisible by 6.

Case(2) and Case(3)

Because of symmetry of a 2D torus, Case(1) and (2) are identical. In the following, Case(2) is assumed to hold. Let r_{ip}, i.e. the ith row in plane p, be the row without any resources. Then, the three rows adjacent to r_{ip} must have $\frac{Y}{3}$ resources each. This is true since each row cannot have more than $\lfloor\frac{Y}{3}\rfloor$ and the three adjacent rows to r_{ip} must have a total of Y resources. If $(3 \nmid Y)$, then $(3\lfloor\frac{Y}{3}\rfloor < Y)$. Thus, some nodes in r_{ip} would not be covered by any resource. Hence, Y must be divisible by 3.

Since each column is assumed to have at least one resource, the argument of Case(1) holds here also. This means the number of nodes in each column must be divisible by 6. Hence, $\{ X, Y \}$ must be in:

$$\{ \{ 3i,\ 6j \} : \ i \ and \ j \ are \ integers \ > 0\}.$$

Case(4)

First, this case does not hold if $(X = 3)$ or $(Y = 3)$. Let r be a row without resources. If $(X = 3)$ and $(Y \neq 3)$, r must be adjacent to three rows each of which has $\frac{Y}{3}$ resources. Since $(X = 3)$, two of adjacent rows to r are adjacent. Hence, it is not possible to place $\frac{Y}{3}$ resources in each of them. Therefore, some nodes in r would not be covered. The same holds, but with respect to Y, when it is assumed that $(Y = 3)$ and $(X \neq 3)$.

Second, assume that both X and Y are greater than 3. Without loss of generality, let the empty row be r_{i0}, i.e. the ith row in plane 0.

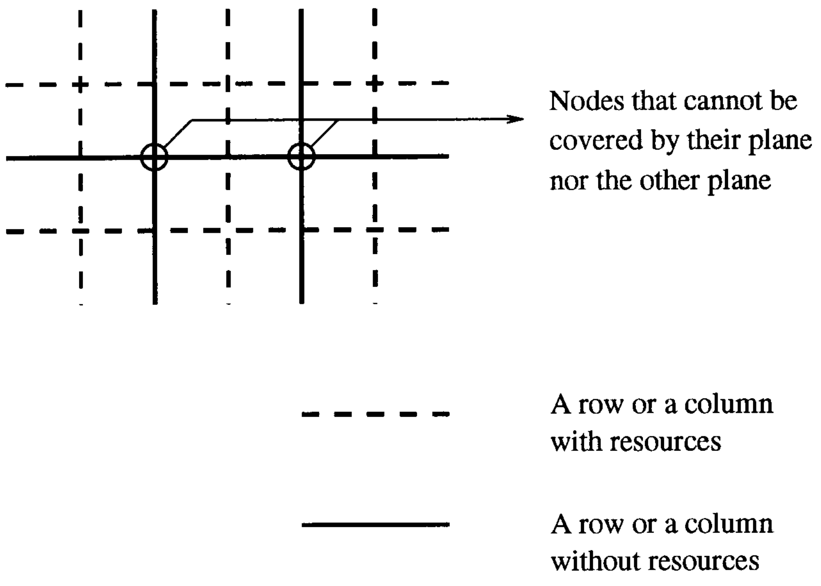

Nodes that cannot be covered by their plane nor the other plane
A row or a column with resources
A row or a column without resources

FIGURE 4.4: The impossibility of Case(4).

- If the empty column is c_{j1}, then $c_{(j+1)0}$ and $c_{(j-1)0}$ have no resources since c_{j0} must have $\frac{X}{3}$ resources.

- If the empty column is c_{j0}, then $c_{(j+2)0}$ has no resources since $c_{(j+1)0}$ must have $\frac{X}{3}$ resources.

Therefore, there is always an empty row, say r, and two empty columns, say c_i and $c_{(i+2)}$, in the same plane, say p. The nodes (p, r, c_i) and (p, r, c_{i+2}) cannot be covered by resources in plane p. This is because all the nodes at a distance of one in the same plane p from these nodes lay on r, c_i, or c_{i+2}. Hence, no resource can be placed in any of them. Furthermore, $(p+1, r, c_i)$ and $(p+1, r, c_{i+2})$ cannot both be resources. If they were, $(p+1, r, c_{i+1})$ would have been covered by both of them. Figure 4.4 illustrates this argument. ■

*4.2.2 On Existence of Regular Linear Distance-*1 *Placements*

A 3D torus that has a regular perfect distance-1 placement must have all of its dimensions to be greater than two. An open question raised by Bae and Bose was [8, 9]: "If a perfect distance-1 placement exists for a 3D torus, is it possible to have one or two of its dimensions to not be divisible by 7 ?" In this subsection, it is proved that a regular linear perfect distance-1 placement exists for a 3D torus, if and only if all of its dimensions are divisible by 7.

Theorem 4.2.2 *Let T be an $X \times Y \times Z$ torus that has a regular linear perfect distance-*1 *placement. Then, all of X, Y, and Z are divisible by 7.*

Proof: It is helpful to point out that in a linear placement, any two rings along the same dimension have the same number of resources, or one of them has no resources. This is true for the following reason. Let r_i be the ring with the maximum number of resources along the D^{th} dimension, and R be the number of resources in r_i. Adding the additive inverse of a resource in r_i to all the resources in that ring results in at least R resources in r_0, where r_0 is a ring along the D^{th} dimension in which the identity node $(0, 0, \ldots, 0)$ exists. Since R is the maximum number of resources in any ring along the D^{th} dimension, r_0 must have exactly R resources. Let r_j be a ring along the D^{th} dimension that has at least one resource. Adding the R resources of r_0 to a resource in r_j results in at least R resources in r_j. Again, since R is the maximum number of resources in any ring along the D^{th} dimension, r_j must have exactly R resources.

The main idea in this proof is to show that all the rings along the same dimension must have the same number of resources. Let r_{ip} be the i^{th} row in plane p of a 3D torus. Assume that r_{ip} has no resources. Then, r_{ip} must be covered by rows adjacent to it, say s_1, s_2, s_3, and s_4. Let N be the number of nodes in r_{ip}. Since any row cannot have more than $\frac{N}{3}$ resources, an empty row cannot be covered by one or two of its adjacent rows. Hence, r_{ip} must be covered by three or four of s_1, s_2, s_3, and s_4.

Assume that r_{ip} is covered by three rows adjacent to it. Without loss of generality, assume these rows are s_1, s_2, and s_3. This implies that each of s_1, s_2, and s_3 has $\frac{N}{3}$ resources. Furthermore, it implies that three of the rows adjacent to s_4 cannot have any resource. This is because any row adjacent to s_1, s_2, or s_3 cannot have a resource. Hence, some nodes in s_4 cannot be covered.

Assume that r_{ip} is covered by all of its four adjacent rows. Then each of s_1, s_2, s_3, and s_4 must have $\frac{N}{4}$ resources since the placement is linear. This means that each of s_1, s_2, s_3, and s_4 has $\frac{N}{4}$ nodes that must be covered by an adjacent row. This implies that either r_{10} or r_{01} has a resource, where r_{mn} is the m^{th} row in the n^{th} plane. In any of these cases, r_{ip} will have a resource by adding r_{10} to $r_{(i-1)p}$ or r_{01} to $r_{i(p-1)}$. Thus, there exists no row such as r_{ip}.

Since every ring must have at least a resource, all the rings along the same dimension must have the same number of resources. This means every resource in a ring has four correspondent resources in the rings adjacent to it. Thus, for every resource in a ring there must be six nodes plus the resource node; three to be covered by the resource and four by its correspondents. Hence, the number of nodes in a ring must be divisible by 7. Therefore, any row or column along any dimension must be divisible by 7. ■

4.3 Irregular Perfect Distance-d Placements

Golomb and Welch have proved that a full packing sphere of radius 2 cannot tile a closed 3D toroidal space. Furthermore, they conjectured that a perfect code does not exist in a 3D toroidal space if $t > 1$ [40]. However, irregular codes (or placements)

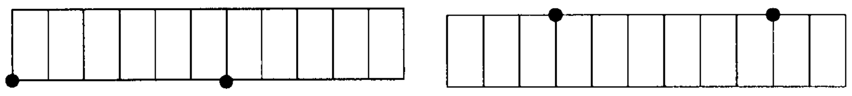

(a) Plane 0 (b) Plane 1

FIGURE 4.5: Irregular perfect distance-2 placement in a $2 \times 2 \times 12$ torus.

exist for any t (or d) in a 3D toroidal space as it will be shown in the following theorem. Still, the proof of Golomb and Welch holds for the regular cases and their conjecture might hold for them as well.

Theorem 4.3.1 *Let T be a $2 \times 2 \times (8d-4)$ torus, $d \geq 2$. Then, the placement P is an irregular perfect distance-d placement,*

$$P = \{\ (0,0,0),\ (0,4d-2,0),\ (1,2d-1,1),\ (1,6d-3,1)\ \}.$$

Proof: The distance between any two nodes in P is $\geq (2d+1)$, $d \geq 2$. Hence, any node at a distance of d or less from a resource node R is at a distance of $(d+1)$ or more from any other resource. In addition, the size of a radius-d sphere in such a space is $(8d-4)$. Assume there exists a node at a distance of $(d+1)$ from all the four resources. This implies that the total number of nodes in T is $> 4(8d-4)$ which is not true. Hence, every node in T is at a distance of d or less from exactly one resource node. ■

Figures 4.5 and 4.6 show an irregular perfect distance-2 and distance-3 placements, respectively. An irregular perfect distance-d placement for a $2 \times 2 \times (8d-4)j$ torus, $j > 0$, can be achieved by using the $2 \times 2 \times (8d-4)$ torus as a tiling block. Figure 4.7 illustrates how to achieve a perfect placement for a $2 \times 2 \times 24$ torus by using a $2 \times 2 \times 12$ torus as a tiling block.

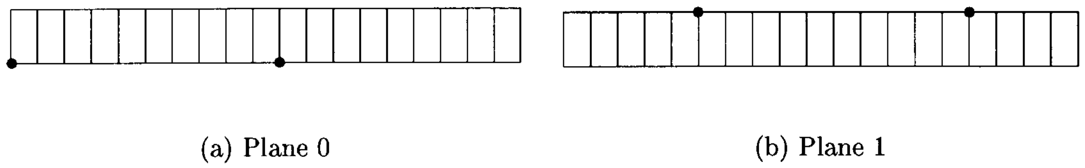


FIGURE 4.6: Irregular perfect distance-3 placement in a $2 \times 2 \times 20$ torus.

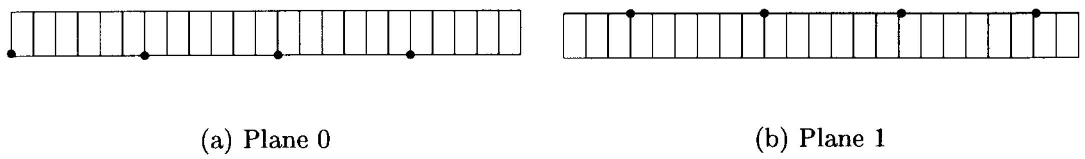


FIGURE 4.7: Irregular perfect distance-2 placement in a $2 \times 2 \times 24$ torus.

More irregular perfect placements can be found for a larger set of tori. Theorem 4.3.2 generalizes Theorem 4.3.1 as follows.

Theorem 4.3.2 *Let T be a $2 \times 2i \times (8d-4i)$ torus, $i \geq 1$, $d > i$, and both i and d are positive integers. Then, the placement P is an irregular perfect distance-d placement,*

$$P = \{(0,0,0),\ (0,4d-2i,0),\ (1,2d-i,i),\ (1,6d-3i,i)\ \}.$$

Proof: Since $d > i \geq 1$, the Lee distance between any two nodes in P is $\geq (2d+1)$. Hence, any node at a distance of d or less from a resource node R is at a distance of (d+1) or more from the other resources.

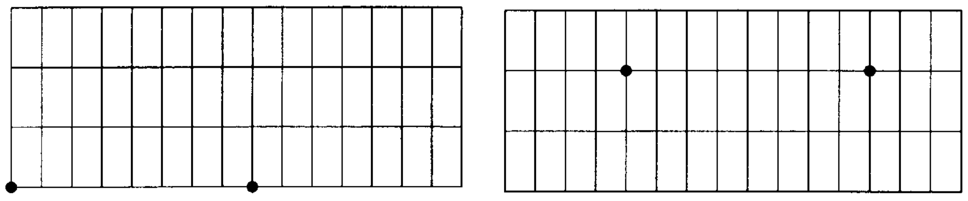

(a) Plane 0 (b) Plane 1

FIGURE 4.8: Irregular perfect distance-3 placement in a $2 \times 4 \times 16$ torus.

The volume of a radius-d sphere in such a space is :

$$= (2d+1) + 2\left[\sum_{j=1}^{i-1}(2d-2j+1)\right] + (2d-2i+1) +$$
$$(2(d-1)+1) + 2\left[\sum_{j=1}^{i-1}(2(d-1)-2j+1)\right] + (2(d-1)-2i+1)$$
$$= 2i(2d-i+1) + 2i(2d-i-1)$$
$$= 2i(4d-2i) = 8id - 4i^2.$$

Assume there exists a node at a distance of $(d+1)$ or more from all the resources. This implies that the number of nodes in T is strictly more than $4(8id-4i^2)$, which is not true. Therefore, every node is within a distance of d or less from exactly one resource. ■

Figures 4.8 and 4.9 show examples for cases in which Theorem 4.3.2 holds.

4.4 Extending 2D tori Placements to 3D Tori

The perfect and quasi-perfect placements for 2D tori were introduced in Chapter 2. These placements can be extended to 3D tori by taking into account that a 3D torus is comprised of several 2D tori. For example, a $3 \times 4 \times 5$ torus is:

(1) 3 planes each of which is a 4×5 torus,

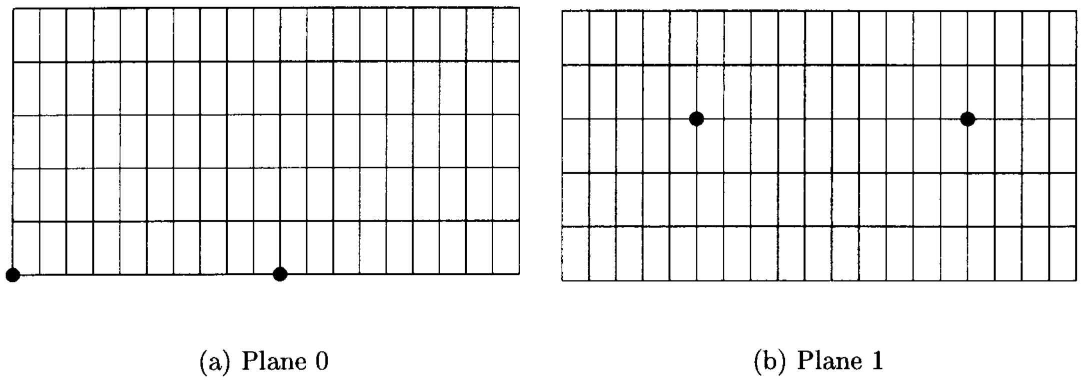


FIGURE 4.9: Irregular perfect distance-4 placement in a $2 \times 6 \times 20$ torus.

(2) 4 planes each of which is a 3×5 torus, or

(3) 5 planes each of which is a 3×4 torus.

Tables 4.1, 4.2, and 4.3 show the possible placements for $4 \times 8 \times 16$, $9 \times 12 \times 16$, and $5 \times 32 \times 32$ tori using placements introduced in Chapter 2. Note that a distance-d placement in these tables means that the placement is a quasi-perfect distance-$(d-1)$ in a tiled 2D plane. Hence, any node in the 3D space would be within a distance of d or less from at least one resource node.

This placement scheme, however, has a major draw back. The links among the planes will not be utilized by local communications. This may cause over-contention in some links while others are lightly utilized. The Sub-Mesh-Based placements, presented in the following section, are designed to overcome this problem.

4.5 Sub-Mesh-Based Placements

A 3D torus can be divided into 3D sub-meshes. For example, a $9 \times 12 \times 15$ torus can be divided into 27 meshes of size $3 \times 4 \times 5$ each of them can be defined as a

Placement	Tiling Block	Number of Resources
Distance-1	2×2 in 8×16 planes	258
Distance-1	4×4 in 8×16 planes	128
Distance-2	8×8 in 8×16 planes	64
Distance-3	8×16 in 8×16 planes	32

TABLE 4.1: Placements for $4 \times 8 \times 16$ based on 2D placements.

Placement	Tiling Block	Number of Resources
Distance-1	3×3 in 9×12 planes	576
Distance-1	2×2 in 12×16 planes	864
Distance-1	4×4 in 12×16 planes	432
Distance-2	3×6 in 9×12 planes	288
Distance-2	4×8 in 12×16 planes	216

TABLE 4.2: Placements for $9 \times 12 \times 16$ based on 2D placements.

Placement	Tiling Block	Number of Resources
Distance-1	2×2 in 32×32 planes	3560
Distance-1	4×4 in 32×32 planes	1280
Distance-2	8×8 in 32×32 planes	640
Distance-3	16×16 in 32×32 planes	320
Distance-4	32×32 in 32×32 planes	160
Distance-8	32×32 scaled down to 8 resources in 32×32 planes	40
Distance-15	32×32 scaled down to 2 resources in 32×32 planes	10

TABLE 4.3: Placements for $5 \times 32 \times 32$ based on 2D placements.

resource sphere. Even if the dimensions of a given torus cannot be divided by the dimensions of the desired sub-mesh, a placement can be achieved. The following is an example illustrating possible solutions for this problem. Let T be an $8 \times 10 \times 11$ torus. One solution to divide T into $3 \times 3 \times 3$ meshes is to use a $2 \times 1 \times 2$ mesh in tiling the region that would be left of $3 \times 3 \times 3$ tiling. Another solution is to add the remaining space to the last tiling blocks. In this case, the last region will be tiled by a $5 \times 4 \times 5$ mesh. A third solution is to distribute the remaining nodes on to the maximum number of the other tiling blocks. In such a case, the tiling blocks to be used are $4 \times 3 \times 4$, and $4 \times 4 \times 4$.

Even though this approach does not take advantage of torus wraparound links, it is quite flexible in finding different placements for a given 3D torus. Furthermore, it can be applied to a 3D mesh as well.

A comparison between sub-mesh-based placements and perfect placements, assuming they exist, is conducted in the rest of this section. The comparison is based on two factors. The first factor is sphere size. The larger size means less number of resources needed to cover a given space. The second factor is the average local traversal distance between a resource and the nodes in its sphere.

Assume there exists a regular perfect distance-d placement for a 3D torus. The sphere size in this case would be:

$$\left(\frac{2d(2d+1)(d+1)}{3}\right)+2d+1$$

Since the surface area or a regular radius-d sphere is $4d^2+2$, the average traversal distance between the resource node and the other nodes within a sphere is computed as follows:

$$\frac{\left(4\sum_{i=1}^{d} i.i^2 \;+\; 2\sum_{i=1}^{d} i\right)}{\textit{sphere size}}$$

$$=\frac{(\;d(d+1)(d^2+d)\;+\;d(d+1)\;)}{\textit{sphere size}}$$

$$=\frac{(\;d(d+1)(d^2+d+1))}{(\;2d(2d+1)(d+1)/3\;)+2d+1}$$

On the other hand, for the sub-mesh-based placements, the volume of a $(2x+1)\times(2y+1)\times(2z+1)$ mesh is $(2x+1)(2y+1)(2z+1)$. Assume the resource node is located at (x,y,z). Let P be the sum of the plane distances in which the resource exists.

$$P=(2y+1)2\sum_{i=1}^{x} i \;+\; (2x+1)2\sum_{i=1}^{y} i$$
$$=(2y+1)(x^2+x)+(2x+1)(y^2+y)$$

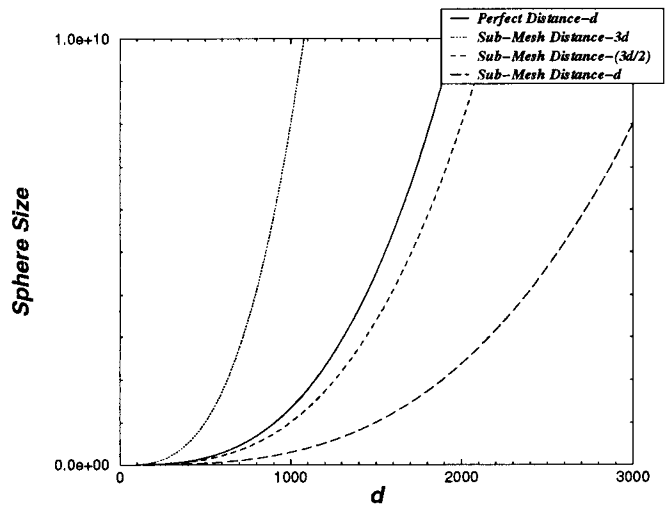


FIGURE 4.10: Sphere volume for perfect distance-d placement, sub-mesh-based distance-$3d$, distance-$(2d/3)$, and distance-d.

Based on that, the average traversal distance can be computed as follows:

$$\frac{((2z+1)P+(2x+1)(2y+1)2\sum_{i=1}^{z} i)}{(\ (2x+1)(2y+1)(2z+1)\ )}$$

$$=\frac{(2y+1)(x^2+x)+(2x+1)(y^2+y)+(2x+1)(2y+1)(z+z^2)}{(2x+1)(2y+1)(2z+1)}$$

Let $x = y = z = d$, the volume of the mesh will be $(2d+1)^3$. Furthermore, the average traversal distance would be:

$$\frac{(2d+1)(d^2+d)(2d+3)}{(2d+1)^3}=\frac{(d^2+d)(2d+3)}{(2d+1)^2}$$

The comparison has been conducted among perfect distance-d, sub-mesh-based distance-$3d$ ($x = y = z = d$), sub-mesh-based distance-$(3d/2)$ ($x = y = z = \frac{d}{2}$), and sub-mesh-based distance-d ($x = y = z = \frac{d}{3}$). Figures 4.10 and 4.11 show the

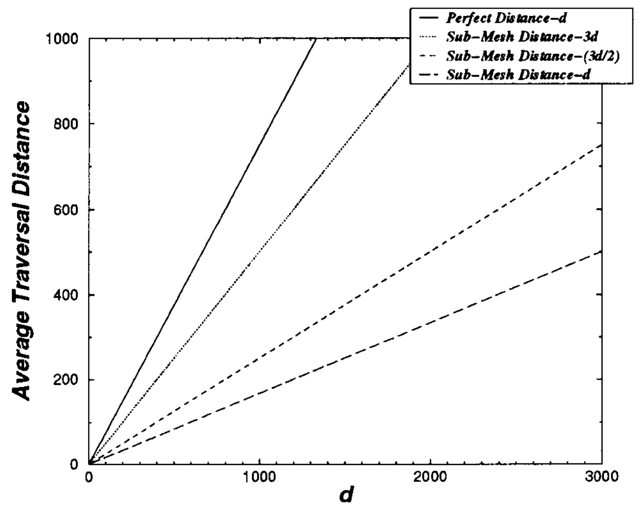


FIGURE 4.11: Average traversal distances for perfect distance-d placement, sub-mesh-based distance-$3d$, distance-$(2d/3)$, and distance-d.

comparison results. The perfect distance-d placement has the second largest sphere size after sub-mesh-based distance-$3d$ and still the latter has less average traversal distance than the former. This makes this placement a considerable alternative for perfect placements in 3D tori.

4.6 Conclusion

Several topics related to resource placements in 3D tori were investigated in this chapter. It is shown that there exists no unknown linear perfect distance-1 placements. Irregular perfect distance-d placements were ignored in most of the previous related studies; irregular perfect distance-d placements are introduced in this chapter. Alternative placement approaches for 3D tori are described. The placements for 2D tori introduced in Chapter 2 are extended to 3D tori. Furthermore, the sub-mesh-based placement approach is defined. This approach is quite similar to the perfect placements in terms of sphere size, local average traversal distance, and uniform distribution of resources over an interconnection network.

Chapter 5

BROADCASTING IN FAULTY TOROIDAL NETWORKS

Low-dimensional toroidal networks are more wire-efficient communication networks than high-dimensional ones under the assumptions of constant wire bisection and constant number of nodes [33]. For this reason, low-dimensional toroidal interconnection networks have become more widely used. Examples of systems that use toroidal interconnection networks are: Tera systems [6], Cray T3D [54], and Cray T3E [67].

Broadcasting (single-node one-to-all) in a multicomputer is one of the important communication primitives [45]. In the past, many fault-tolerant broadcasting algorithms have been published for hypercube interconnection networks [1, 49, 55, 56, 57, 76]. Not much work has been done in this area for toroidal networks. In [22], a fault-tolerant broadcasting algorithm for k-ary n-cube (Q_k^n) that can adapt up to $(n-1)$ failures is given. This chapter presents a fault-tolerant broadcasting algorithm for toroidal networks that can adapt up to $(2n-2)$ failures. The algorithm is non-redundant, requires a global fault information, and is close to optimal in terms of communication time.

5.1 Fault-Free Broadcasting Algorithm

MECA is one of the first routing and broadcasting algorithms for Q_k^n's [36]. It uses *wormhole packet routing* technique [34, 53], and to avoid deadlocks, it uses virtual channels [35]. Later, Bose et al introduced the Basic Broadcasting Algorithm which is similar to the e-cube algorithm used in hypercubes [19]. The algorithm needs

```
Broadcast Using Cut-Through ($S$, $M$)

$S$ : Source node,
$M$ : Broadcasted message,
An uncovered-node is a node that has not received $M$ yet, and
A covered-node is a node that has received $M$.

(1) Mark $S$ as a covered-node.

(2) For i = 1 to $n$

    (2.1) For j = 1 to $\lceil lg\ k \rceil$

        (2.1.1) Each covered-node sends $M$ to an uncovered-node at a dis-
                tance of $\lceil \frac{k}{2^j} \rceil$ along $i^{th}$ dimension.
        (2.1.2) The nodes which received $M$ in Step (2.1.1) are marked as
                covered-nodes.
```

FIGURE 5.1: A fault-free broadcasting algorithm for Q_k^n using cut through packet routing.

Broadcast Using Store and Forward (S, M)

S : Source node,

M : Broadcasted message,

A covered-node is a node that has already received M,

An uncovered-node is a node that has not received M yet, and

```
(1) Mark $S$ as an old-covered-node.
(2) For i = 1 to $n$
    (2.1) Mark all old-covered-nodes to be new-covered-nodes.
    (2.2) For j = 1 to $\lceil \frac{k}{2} \rceil$
        (2.2.1) Each new-covered-node sends $M$ to an adjacent uncovered-
                node along $i^{th}$ dimension.
        (2.2.2) The nodes which received $M$ in Step (2.2.1) are marked as
                new-covered-nodes.
        (2.2.3) Each new-covered-node adjacent to two covered-nodes (old
                or new) along $i^{th}$ dimension is marked as an old-covered-
                node.
```

FIGURE 5.2: A fault-free broadcasting algorithm for Q_k^n using store-and-forward packet routing.

n phases to broadcast in a Q_k^n. In the 0^{th}-phase, the source node performs a ring broadcasting along the 0^{th}-dimension. In the i^{th}-phase, each node that has received the broadcasted message from a previous phase performs a ring broadcasting along the i^{th}-dimension. In the case of $store-and-forward$ using a single port I/O model, this algorithm has been proved to be optimal for k even; for k odd, the algorithm takes one extra step than the known lower bound [19].

Figures 5.1 and 5.2 describe fault-free broadcasting algorithms for a Q_k^n when *cut through* [46] or *store-and-forward* [48] routing strategy is used, respectively. The basic idea of both algorithms is the same as that of the e-cube algorithm described above. However, broadcasting in a ring differs depending on the packet routing strategy used. When cut though strategy is used, the message path length becomes less significant. In such a case, the communication complexity of sending a message of size M words along a path of length P becomes:

$$t_s + Mt_w + Pt_h,$$

where t_s is startup time, t_w is per word time, and t_h is per hop time. On the other hand, the same message sent over the same path using store-and-forward routing has a communication complexity of:

$$P(t_s + Mt_w + t_h).$$

Since t_h is small compared to $(t_s + Mt_w)$, broadcasting in a ring R of k nodes using cut through routing can be performed in $\lceil \log k \rceil$ phases as follows. In the 0^{th}-phase, the source node S sends the message M to a node at a distance of $\lceil \frac{k}{2} \rceil$. In the i^{th}-phase, all nodes which have received M in a previous phase send M to distinct nodes at a distance of $\lceil \frac{k}{2^i} \rceil$. The communication complexity of this algorithm is:

$$\lceil \log k \rceil (t_s + Mt_w) + \sum_{i=1}^{\log k} \lceil \frac{k}{2^i} \rceil = \lceil \log k \rceil (t_s + Mt_w) + t_h(k-1).$$

In a fault-free Q_k^n, broadcasting using cut through routing can be achieved by performing n ring broadcastings. Figure 5.1 describes such an algorithm which has a communication complexity of:

$$n\lceil \log k \rceil (t_s + Mt_w) + nt_h(k-1).$$

On the other hand, broadcasting in a ring using store-and-forward routing strategy has a communication complexity of:

$$\lceil \frac{k}{2} \rceil (t_s + t_w m + t_h).$$

Hence, the broadcasting algorithm for Q_k^n described in Figure 5.2 has a communication complexity of:

$$n\lceil \frac{k}{2} \rceil (t_s + t_w m + t_h).$$

5.2 Fault-Tolerant Broadcasting Algorithm

In [22], the authors have proposed a non-redundant fault-tolerant broadcasting algorithm that can adapt up to $(n-1)$ node failures and requires global fault information [22]. In this chapter, this algorithm is extended to adapt up to $(2n-2)$ node failures. The algorithm is given in Figure 5.3. Its basic idea is as follows. Let the number of faulty nodes be $t \leq (2n-2)$ and the address of these nodes be:

$$\begin{aligned}
f_1 &= (a_{n-1,1}\ a_{n-2,1}\ a_{n-3,1}\ \cdots a_{0,1}) \\
f_2 &= (a_{n-1,2}\ a_{n-2,2}\ a_{n-3,2}\ \cdots a_{0,2}) \\
\vdots & \qquad\qquad \vdots \\
f_t &= (a_{n-1,t}\ a_{n-2,t}\ a_{n-3,t}\ \cdots a_{0,t}).
\end{aligned}$$

If $k > (2n-2)$, then at least one of the digits, say b_i, in $Z_k = \{0, 1, \ldots, (k-1)\}$ will not appear in the i^{th} digits, $i = 0, 1, \ldots, (n-1)$, of the faulty node addresses. Then, $(* * \ldots b_i \cdots * *)$ is a fault-free $(k-1)$-ary n-cube.

Broadcasting in a Faulty Q_k^n (S, M)

Assumptions:

1) Total number of faults $< 2n - 1$.

2) $k > 2n - 2$ and $k > 3$.

Definitions:

S : is source node,

M : is message to be broadcasted,

Q : is Q_k^n to broadcast M in,

C_i's : are sub-cubes of Q along a certain dimension X each of which is a Q_k^{n-1},

C : is one of the C_i's that is a fault-free, and

R_N : is a ring a long the X dimension to which the node N belongs and $N \in C$.

(1) **S sends M to a node in C, say S'**

(2) **S' broadcasts M in C using one of the algorithms given in Figures 5.1 or 5.2.**

(3) **Each node $A \in C$ starts a ring broadcasting along the X dimension iff R_A is fault-free.**

(4) **Each faulty ring along the X dimension gets M from a fault-free adjacent ring (one communication step).**

FIGURE 5.3: A broadcasting algorithm for a faulty Q_k^n.

Example: In Q_5^3, let the faulty nodes be:

$$\begin{aligned} f_1 &= (3\ 2\ 1) \\ f_2 &= (1\ 3\ 2) \\ f_3 &= (0\ 4\ 3) \\ f_4 &= (2\ 0\ 4). \end{aligned}$$

Then, the sub-cube $(4 * *)$ is fault-free because the digit "4" does not appear in the second digit addresses of the faulty nodes. Similarly, the sub-cubes $(*1*)$ and $(* * 0)$ are also fault-free.

Let $S = (s_{n-1}s_{n-2}s_{n-3} \ldots s_0)$ be the source node. For the time being, assume S is in a fault-free Q_k^{n-1}, say $(b_{n-1} * * \cdots *)$. This means that the faulty nodes are along the $(n-1)^{th}$ dimensional rings. The number of faulty rings will be $\leq t$, where t is the number of faulty nodes. This is because there may be more than one faulty node in the same ring. In the first phase, S broadcasts the message in the fault-free Q_k^{n-1} using the basic broadcasting algorithm. In the second phase, each node A in the fault-free sub-cube Q_k^{n-1} will broadcast the message along the $(n-1)^{th}$ dimensional ring, provided this ring is fault-free. In the last phase, the fault-free nodes in the faulty rings receive the message from nodes of a unique adjacent fault-free ring. It will be shown shortly that there exists a unique fault-free ring adjacent to a given faulty ring.

If the source node S is not in any of the fault-free k-ary $(n-1)$-cubes, then S can send the message to one of the closest nodes in a fault-free k-ary $(n-1)$-cube. After this, the above steps can be repeated.

The rest of this section is divided into four subsections. In Subsection 5.1, an algorithm to find a fault-free Q_k^{n-1} (and so a node in this Q_k^{n-1}) closest to the source node S is given. Subsection 5.2 describes an algorithm to find a unique fault-free ring adjacent to each of the faulty rings. The communication complexity of the algorithm is derived in Subsection 5.3. Subsection 5.4 discusses conditions under which the algorithm can be used in a general toroidal network.

5.2.1 Closest Fault-Free Q_k^{n-1} From the Source

The *Lee* distance between two rings along the same dimension is defined in the following.

Definition 5.2.1 *Let R_A and R_B be two rings along the same dimension X in a toroidal space. R_A and R_B can be addressed as $(a_{n-1}a_{n-2}\dots a_{x+1} * a_{x-1}\dots a_0)$ and $(b_{n-1}b_{n-2}\dots b_{x+1} * b_{x-1}\dots b_0)$, respectively where * is a don't care character. The Lee Distance between R_A and R_B, $D_L(R_A,\ R_B)$, is defined as:*

$$D_L(R_A, R_B) = \sum_{i=0 \text{ and } i \neq x}^{n-2} min\ (|a_i - b_i|,\ |k_i - a_i + b_i|),$$

where k_i is the size of the i^{th} dimension.

For example, if $R_A = (*223)$, $R_B = (*364)$, and $K = (4578)$, $D_L(R_A, R_B) = 1+3+1 = 5$.

Suppose $(* \cdots * \ b \ * \ *)$ is one of the closest fault-free Q_k^{n-1} from S. Then the node in this sub-cube closest to S is $S' = (s_{n-1}, s_{n-2}, \dots, s_{i+1}, b_i, s_{i-1}, \dots, s_0)$ and $D_L(S, S') = D_L(s_i, b_i)$. If the shortest path from S to S' is fault-free, then this path can be used in Step(1) of the algorithm given in Figure 5.4 to send the message. However, if some node in this path is faulty, then S can send the message to a node T in an adjacent fault-free ring, and then T can send the message along its ring to a node in the fault-free Q_k^{n-1} (Note that the faulty nodes are along the i^{th} dimensional rings, and there are $(2n-2)$ adjacent rings to a faulty ring, so at least one of the adjacent rings is fault-free). The following lemma is useful to find the shortest distance fault-free sub-cube, Q_k^{n-1}, from S.

Finding Closest Fault-Free Sub-Cube to S

S : is source node in a Q_k^n,

Output is $(i,\ j)$ indicating that the j^{th} sub-cube along the i^{th} dimension is one of the closest sub-cubes to S.

```
(1) Let faulty nodes be denoted by $f_i$, $0 \le i \le (2n-3)$.

(2) Define a 2 dimensional matrix $MAT[n,k]$, and initialize it to be 0's.

(3) For $i = 0$ to $(n-1)$
    (3.1) For $j = 0$ to $(k-1)$
        (3.1.1) $MAT[i, i^{th} digit\ of\ f_j] = 1$.

(4) Found = False, $i = 0$.

(5) While not Found
    (5.1) If $MAT[i, i^{th} digit\ of\ S] = 0$
        (5.1.1) Found = True, Return($i$, $i^{th} digit\ of\ S$).
    (5.2) Else $i = i + 1$.

(6) Found = False, $i = -1$.

(7) While not Found
    (7.1) $i = i + 1$, $j = 0$.
    (7.2) While (not Found) and $(j < n)$
        (7.2.1) If $MAT[j, j^{th} digit\ of\ S\ + i] = 0$
            (7.2.1.1) Found = True, Return($j$, $j^{th} digit\ of\ S + j$).
        (7.2.2) Else If $MAT[j, j^{th} digit\ of\ S\ - i] = 0$;
            (7.2.2.1) Found = True, Return($j$, $j^{th} digit\ of\ S - j$).
        (7.2.2) Else $j = j + 1$;
```

FIGURE 5.4: Finding one of the closest sub-cubes to S.

Lemma 5.2.1 *Suppose $S = (s_{n-1}, s_{n-2}, \ldots, s_0)$ is the source node and the faulty nodes are $f_i = (a_{n-1,i}, a_{n-2,i}, \ldots, a_{0,i})$, $i = 1, 2, \ldots, t$, where $t \leq (2n-2)$. Then, one of the sub-cubes $(* \cdots * s_p + j * \cdots *)$, where $j = 0, \pm 1, \pm 2, \ldots, \pm(n-1)$, is fault-free.*

Proof: The t faulty nodes can have a maximum of t distinct digits in the p^{th} address digits. However, the set $\{s_p + j : j = 0, \pm 1, \pm 2, \ldots, \pm(n-1)\}$ contains $(2n-1)$ digits and so at least one of the digits, say $(s_p + j')$, is distinct from the p^{th} digits of the faulty node address digits. Thus, $(* \cdots * s_p + j' * \cdots *)$ is a fault-free Q_k^{n-1}. ■

To find the closest fault-free Q_k^{n-1} from S, we proceed in the following order. First, check whether any of the sub-cubes $(* \cdots * s_p * \cdots *)$, $p = 0, 1, \ldots, (n-1)$, is fault-free. If so, the source node is in a fault-free sub-cube and stop this process. Otherwise, check any one of the sub-cubes $(* \cdots * s_p + j * \cdots *)$, where $j = \pm 1$, and $p = 0, 1, 2, \ldots, (n-1)$ is fault-free. Continue this process for $j = \pm 2$, $j = \pm 3$, $\ldots$, $j = \pm(n-1)$. When a fault-free sub-cube is found, the search process is stopped. Lemma 1 guarantees finding a fault-free sub-cube, say $(* \cdots * s_{p'} + j' * \cdots *)$, where $0 \leq p' \leq (n-1)$ and $j = 0, \pm 1, \pm 2, \ldots, \pm(n-1)$. This fault-free sub-cube must be one of the closest to S and the distance from S to a node in $(* \cdots * s_{p'} + j' * \cdots *)$ closest to S is j'. Suppose there is a fault-free sub-cube $(* \cdots * s_{p'} + j'' * \cdots *)$ which is closer to S than $(* \cdots * s_{p'} + j' * \cdots *)$. In that case $|j''| < |j'|$. Then in the above search process, the sub-cube $(* \cdots * s_{p'} + j'' * \cdots *)$ would have been declared a fault-free before searching for values $j = j'$ (i.e. the search process for $j = +j'$ and $j = -j'$ would not have occurred). Thus, this search process finds one of the fault-free k-ary $(n-1)$-cubes closest to S.

The distance between the source node S and S' which is a node in a fault-free sub-cube can be at most $(n-1)$ if S sends the message along the same ring. Alternatively, this distance can be at most n if S first sends the message to its adjacent fault-free ring and then that node sends the message along its ring to the node in the fault-free Q_k^{n-1}.

The algorithm given in Figure 5.4 formally describes how to find the closest fault-free sub-cube from S. This algorithm has a local computation complexity of $O(n^2)$. First, the algorithm constructs a 2D bit-map matrix $M[(n-1),(k-1)]$ in which all the faulty sub-cubes are marked (i.e. if $M[i,j] = 1$ then the j^{th} sub-cube along the i^{th} dimension is faulty; otherwise it is fault-free). This is given in Step(3) of the algorithm. Then, it tries to find a fault-free sub-cube along some dimension to which S belongs (Step 5). If there are none, then it tries to find a fault-free sub-cube along some dimension at a minimum distance, $D_M = 1, 2, \ldots, (n-1)$, from S (Step 7). From the arguments given in the previous paragraphs, it is easy to verify that $D_M \leq (n-1)$.

If there are two or more fault-free sub-cubes at a distance of D_M from S, the algorithm can be enhanced by choosing the one to which S can send the message M directly. Furthermore, it can be modified to try to find a fault-free sub-cube that is connected to S through a fault-free path (this fault-free sub-cube may not be closest to S). In this case only one routing operation needs to be done through a longer path rather than two routing operations. In cut through routing, the message start up time is more significant than the path length, and therefore, this approach might be favored. The given algorithm, even with the enhancements, has a local computation complexity of $O(n^2)$. In fact, it is possible to find a fault-free sub-cube, not necessarily the closest, in $O(n)$ local steps. Instead of searching all the dimensions, only one dimension is searched. In this case also, the fault-free sub-cube will be at a distance of $(n-1)$ or less from S.

```
Finding An Adjacent Fault-Free Ring $(R, X)$
$X$ : A dimension,
$R$ : A ring along $X$ to which a fault-free adjacent ring is needed to be found.
  (1) Let $FR$ be the set of faulty rings along a given dimension $X$.
  (2) Let the rings adjacent to $R$ be denoted by $R_i$, $0 \leq i \leq (2n-2)$.
  (3) Found = False, i = 0.
  (4) While (not Found)
      (4.1) If $R_i \notin FR$
          (4.1.1) Found = True, Return(i).
      (4.2) Else $i = i + 1$.
```

FIGURE 5.5: Finding an adjacent fault-free ring along a given dimension.

Assigning Each Faulty Ring to A Unique Adjacent Fault-Free Ring

```
(1) Let $FR$ be the set of faulty rings along the $X$ dimension.
(2) For each faulty ring $f_i$ do
    (2.1) Find the set of adjacent rings to $f_i$, let it be $A_i$.
    (2.2) $A_i = A_i$ - $FR$.
    (2.3) Choose any element in $A_i$, let it be $R_a$. Assign $R_a$ to $f_i$.
    (2.4) $FR = FR \cup \{R_a\}$.
```

FIGURE 5.6: Assigning each faulty ring along the X dimension an adjacent fault-free ring

5.2.2 Finding a Unique Fault-Free Adjacent Ring to a Faulty Ring

If the source node S is part of the fault-free $(n-1)$ dimensional sub-cube, C, then Step(1) of the algorithm, given in Figure 5.4, is not needed. However, if $S \notin C$, then S is in a ring along the X dimension, say R_A. If R_A is fault-free, Step(1) can be done in one communication step. But if R_A is faulty, then R_A must have an adjacent fault-free ring along the dimension X, say R_B. Hence, S sends M to its correspondent node in R_B and from there to a node in C. The existence of a fault-free ring R_B adjacent to R_A is proved in the following lemma.

Lemma 5.2.2 *Suppose the source S is in a faulty ring R along the X dimension. If the total number of faults in the given Q_k^n is $< (2n-2)$, and $k \geq 3$, then R_A has an adjacent fault-free ring R_B along the X dimension.*

Proof: The number of faults is $\leq (2n-2)$. Since R_A is faulty, at most $(2n-3)$ other faulty rings can exist. But R_A has $(2n-2)$ distinct adjacent rings because $k \geq 3$. Thus, at least one of them is fault-free. ■

In order to perform Step(4) of the algorithm we need to show that each faulty ring, R_N where $N \in C$, along the X dimension can be uniquely assigned a fault-free adjacent ring along the same dimension. By uniquely, it is meant that every faulty ring will be assigned to a distinct fault-free ring so that all the faulty rings are covered in a single communication step.

Lemma 5.2.3 *Let the faulty rings along the X dimension be denoted as f_i, $0 \leq i < 2n-2$. Furthermore, let A_i be the set of rings along the X dimension that are adjacent to f_i. If $f_j \in A_i$ and $k > 3$, then $A_i \cap A_j = \phi$.*

Proof: If $f_j \in A_i$ then $D_L(f_i, f_j) = 1$. Assume that $f_j \in A_i$ and $A_i \cap A_j \neq \phi$. This implies that there exists a ring R along the X dimension such that $R \in A_i$ and $R \in A_j$. Then, $D_L(R, f_i) = 1$ and $D_L(R, f_j) = 1$. This means either $[D_L(f_i, f_j) = 1$ and $k = 3]$ or $[D_L(f_i, f_j) = 2$ and $k > 3]$. The first case gives a contradiction because it is assumed that $k > 3$. Since $D_L(f_i, f_j) = 1$, the second case cannot hold. ■

Theorem 5.2.1 *In Q_k^n, $k > 3$, if the number of faulty rings is $\leq (2n-2)$ in some dimension, then for each faulty ring there exists a unique fault-free ring adjacent to it.*

Proof: The proof is by construction, meaning that it shows how to assign a unique fault-free ring to a given faulty ring. Let the faulty rings be $f_1, f_2, \ldots, f_t$ where $t \leq (2n-2)$. Let A_i be the adjacent rings of f_i, $i = 1, 2, \ldots, t$. Note that $|A_i| = (2n-2)$, for $i = 1, 2, \ldots, t$. For each faulty ring f_i, we assign a unique fault-free ring as follows. For $i = 1$, assign a fault-free ring R_1 in A_1 to f_1; this is possible because f_1 has $(2n-2)$ adjacent rings (i.e. $|A_1| = 2n-2)$ and at most $(2n-3)$

of them can be faulty. Then remove R_1 from other A_i's, $i = 2, 3, \ldots, t$, provided R_1 is in A_i. At the j^{th} step, assign a non-assigned fault-free ring in A_j to f_j. This is always possible because of the reasons given below. Let R_p be the fault-free ring assigned to the faulty ring f_p, $p = 1, 2, \ldots, (j-1)$. If $D_L(f_p, f_j) = 1$, then $f_p \in A_j$ and $R_p \notin A_p$ since $D_L(f_p, f_j) = 2$. On the other hand, if $D_L(f_p, f_j) = 2$ then $f_p \notin A_j$ and R_p might be in A_j. In all other cases, both f_p and R_p are not in A_j. Thus, while trying to assign a fault-free ring to f_j, there should be at least $(2n-2) - (j-1)$ rings adjacent to f_j available; of these the number of fault-free rings should be $(2n-2) - (j-1) - (t-j) \geq (2n-2) - (j-1) - (2n-2-j) \geq 1$. Thus, this fault-free ring can be assigned to f_j. ■

5.2.3 *Communication Complexity*

Step(2) and Step(3) are exactly equivalent to broadcasting in a Q_k^{n-1} and a Q_k^1, respectively. Hence, the communication complexity of the proposed algorithm is the complexity of broadcasting in a Q_k^n plus the complexities of Step(2) and Step(3). The complexities of Step(2) and Step(3) depend on the packet routing strategy used. In the following the complexities are derived when cut through or store-and-forward is used.

Cut Through: In the worst case, two communication steps are needed in Step(1). First, the source node S sends the message M to an adjacent node S' in a fault-free adjacent ring. Next, S' sends M to any node in the fault-free sub-cube which is at most $(n-1)$ hops away. Thus, Step(1) needs at most:

$$2(t_s + t_w M) + n t_h,$$

where t_s is the startup time, t_w is time per word, M is the size of broadcasted message in words, and t_h is time per hop. Furthermore, Step(4) needs exactly one communication step in which each faulty ring gets M from an adjacent fault-free ring:

$$t_s + t_w M + t_h,$$

Hence, the total communication complexity is:

$$n\lceil lg\ k\rceil(t_s + t_w M) + nt_h(k-1) + 3(t_s + t_w M) + t_h(n+1).$$

Store-and-Forward: In the worst case, here also, Step(1) needs to send M to an adjacent node S' in a fault-free ring. Next, M is sent from S' to a node in the fault-free sub-cube which is at a distance of $(n-1)$ or less. The time required for this is:

$$n(t_s + t_w M + t_h) - (n-2)t_s,$$

In addition, Step(4) needs exactly one communication step in which each faulty ring gets the message from a fault-free adjacent ring:

$$t_s + t_w M + t_h,$$

Hence, the resultant communication complexity in this case is:

$$(n\lceil\frac{k}{2}\rceil + n + 1)(t_s + t_w M + t_h) - (n-2)t_s.$$

5.2.4 The Algorithm in Toroidal Networks

The above algorithm also works for any general toroidal network provided that at least one of the dimensions, say X, is greater than $(2n-2)$ and that all the other dimensions are greater than 3. The dimension X being greater than $(2n-2)$ assures the existence of a fault-free sub-cube along this dimension. Furthermore, the other dimensions being greater than 3 guarantees that each faulty ring along the X

dimension can be assigned a unique fault-free adjacent ring using a greedy-like strategy. These two conditions imply that Lemma 1 and Theorem 1 also hold in those cases and, hence, the algorithm works correctly. Furthermore, the communication complexities are exactly the same as those of Q_k^n case.

5.3 Conclusion

This chapter presents a fault-tolerant broadcasting algorithm for toroidal networks of n dimensions that can adapt up to $(2n - 2)$ failures. Since the connectivity of toroidal networks is $(2n)$, the algorithm is very close to optimal with respect to the maximum possible number of failures a broadcasting algorithm can adapt.

The algorithm in the worst case needs three communication steps more than broadcasting in a fault-free toroidal network when cut through routing is used. When the system uses store-and-forward method, the proposed algorithm requires at most $(n + 1)$ more steps than the optimal fault-free algorithm.

Chapter 6

FUTURE RESEARCH

There are several open research problems related to the topics of this thesis; this chapter introduces some of them. The introduced problems are divided into four categories: Those related to resource placements, to simulation studies, to communication algorithms, and to the domination concept. The problems of each category are discussed in a separate section of this chapter..

6.1 Distance-d Resource Placements in Toroidal Networks

Chapters 2, and 4 of this thesis provided solutions for distance-d problems in 2D and 3D tori. Still, there are many related open problems in need of further investigation. The following lists some of these problems.

1. In Chapter 2, a scheme for the quasi-perfect placements in 2D tori was given. In this scheme, placing a resource is a single step procedure. However, finding the closest resource, or resources, from a non-resource node cannot be done efficiently. The only method, the author is aware of, is searching all resources till the closest ones are found. Designing a more efficient algorithm would be helpful especially for applications like the spare processor placements [24, 47, 74].

2. Quasi-perfect distance-1 placements for a Q_k^3, where $8 \leq k \leq 19$, have been found by the author. However, is it possible to construct a quasi-perfect placement scheme for 3D tori?

3. Brualdi and Pless have designed *greedy codes* based on the *Hamming distance* [23]. These codes have been proved to be linear. Is it possible to construct greedy codes based on the *Lee distance* so it can be used for distance-d placements?

4. Irregular perfect distance-d placements have been ignored in most of the previous studies. More investigation on the existence of these placements should be conducted. Furthermore, the existence of irregular quasi-perfect distance-d placements might be considered.

5. Is it possible to have a non-linear perfect distance-d placement for a 2D torus, a 3D torus, or any toroidal network? If there are non-linear perfect placements, then characterizing their existence would be a very interesting research topic. Otherwise, proofing the non-existence of unknown perfect placements would be simpler.

6. Characterizing the existence of linear perfect/quasi-perfect distance-d placements for toroidal networks is another interesting research problem. However, this problem might be too difficult to handle all at once. Breaking it into the special cases of 2D tori, 3D tori, and higher dimensional tori might be a good idea.

7. The general distance-d placement for toroidal networks is one of the most challenging related open problems. It is defined by the following question. Given R resources, and a toroidal network T; what is the best way by which the R resources are distributed over T so the maximum distance among the non-resource nodes and the resources is minimized? This problem can also be broken into the special cases of 2D tori, 3D tori, and higher dimensional tori.

6.2 Simulation Studies

In Chapter 2, it has been pointed out that the traffic over-distribution resulted in a faster network saturation. A simulation study to this phenomena, in different work loads and different networks, could be an interesting research problem.

Another simulation project is to investigate the relationship between the average message latency and the used placement strategy in 3D tori. The placement strategies to be compared are the ones proposed in Chapter 4 and the placements used in practice. The ones proposed in Chapter 4 are: Sub-Mesh-Based, and 2D perfect/quasi-perfect extended to 3D.

The Sub-Mesh-Based strategy can be applied in 2D tori as well. Comparing the Sub-Mesh-Based and perfect/quasi-perfect placements in 2D tori is another interesting simulation study.

6.3 Fault-Tolerant Communication Algorithms

Chapter 5 has introduced a fault-tolerant one-to-all broadcasting algorithm designed for toroidal networks. There are primitive communication patters other than single broadcasting. These primitives are: All-to-all, one-to-all personalized, and all-to-all personalized. Designing efficient fault-tolerant algorithms for these communication patterns is a challenging research topic.

6.4 Other Types of Dominations

In this thesis, the distance-d placement problem is investigated. This problem is an instance of "dominating sets" in graph theory. The concept of dominating sets has existed since the mid-1800's [43]. The very first problem related to this concept were the placements of chess pieces to attack all the squares of a chess board. There are many types of dominations. Examples are: typical domination [12], perfect domination [27], independent domination [28, 41], total domination [2, 14], connected domination [39], paired domination [42], clique domination [31], and cycle domination [29]. A comprehensive list of domination types can be found in [43]. An interesting research topic would be investigating the potential applications of different domination types to the popular network topologies.

BIBLIOGRAPHY

[1] A. Al-Dhelaan and B. Bose. "An Efficient Fault-Tolerant Broadcasting Algorithm for the Hypercube". In 4th *Conference on Hypercube Concurrent Computers and Applications*, pages 123–128, 1989.

[2] R. Allan, R. Laskar, and S. Hedetniemi. "A Note on Total Domination". *Discrete Mathematics*, 49:7–13, 1984.

[3] G. Almasi and A. Gottlieb. *Highly parallel computing.* The Benjamin/Cummings publishing company Inc., Redwood City, CA, 2nd edition, 1994.

[4] B. Almohammad, J. H. Kim, and B. Bose. "I/O Placement strategies and their Data Locality Sensitivities in $k \times k$ Torodidal Networks". In *HPC Asia '97 Conference and Exhibition - Seoul Korea*, pages 343–348, 1997.

[5] B. Almohammad, J. H. Kim, and B. Bose. "I/O Placements and Network Latencies in $k \times k$ Toroidal Networks". *Mathematical Modelling and Scientific Computing*, September 1997.

[6] R. Alverson, D. Callahan, D. Cummings, B. Koblenz, A. Porterfield, B. Smith. "The Tera Computer System". Technical report, Tera Computer Company, 1991.

[7] Anonymous. *Cray T3D system architecture overview manual.* Cray Research, Inc., 1993.

[8] M. Bae. *Resource Placement, Data Rearrangement, and Hamiltonian Cycles in Tours Networks.* PhD thesis, Department of Computer Science, Oregon State University, 1996.

[9] M. Bae and B. Bose. "Resource Placement in torus-based networks". In *IEEE International Parallel Processing Symposium*, pages 327–331, April 1996.

[10] M. Bae and B. Bose. "Resource placement in torus-based networks". *IEEE Transaction on Computers*, 46(10):1083–1092, October 1997.

[11] J. Banks and J. Carson. *Discrete-event system simulation.* Printice-hall, Englewood Cliffs, NJ, 1984.

[12] C. Berge. *Theory of Graphs and its Applications.* Methuen, London, 1962.

[13] E. Berlekamp. *Algebraic coding theory.* McGraw-Hill Inc., Newyork, 1968.

[14] A. Bertossi. "Total Domination in Interval Graphs". *Information Processing Letters*, 23:131–134, 1986.

[15] S. H. Bokhari. *"Communication Overhead on the Intel Paragon, IBM SP2 and Meiko".* CS-2, ICASE Interim Report 28, NASA Contractor Report 198211, Inst. for Computer Applications in Science & Engineering, Langley, Va, 1995.

[16] S.H. Bokhari. "Multiphase Complete Exchange: A Theoritcal Analysis". *IEEE Transactions on Computers*, 45(2):220–229, February 1996.

[17] S.H. Bokhari. "Multiphase Complete Exchange on Paragon, SP2, and CS-2". *IEEE Parallel and Distributed Technology*, 4(3):45–59, Fall 1996.

[18] S.H. Bokhari and D. M. Nicol. "Balancing Contention and Synchronization on the Intel Paragon". *IEEE Concurrency, Parallel, Distributed & Mobile Computing*, 5(2):74–83, April-June 1997.

[19] B. Bose, R. Broeg, Y. Kwon, and Y. Ashir. "Lee Distance and Topological Properties of k-ary n-cubes". *IEEE Transactions on Computers*, 44(8):1021–1030, August 1995.

[20] E. Brewer, C. Dellarocas, and W. Weihl. "PROTEUS: A High-Performance Parallel Architecture Simulator". Technical Report MIT/LCS/TR-516, MIT, 1991.

[21] R. Broeg. *Topics in Toroidal Interconnection Networks.* PhD thesis, Department of Computer Science, Oregon State University, 1995.

[22] R. Broeg and B. Bose. "Fault-Tolerant Broadcasting in A K-ary N-cube". In *Annual IEEE Workshop on Fault-Tolerant Parallel and Distributed Systems*, 1996. Chapter 15 of Fault-Tolerant Parallel and Distributed Systems, to be published by Kluwer Academic Publishers, Boston.

[23] R. Brualdi and V. Pless. "Greedy Codes". In *IEEE International Symposium on Information Theory*, page 366, 1993.

[24] J. Bruck, R. Cypher, and C. Ho. "Fault-Tolerant Meshes and Hypercubes with Minimal Number of Spares". *IEEE Transactions on Computers*, 42(9):1089–1103, September 1993.

[25] H. Chen and N. Tzeng. "Fault-Tolerant Resource Placement in Hypercube Computers". In *International Conference on Parallel Processing*, volume 1, pages 515–524, 1991.

[26] H. Chen and N. Tzeng. "Efficient Resource Placement in Hypercubes Using Multiple-Adjacency Codes". *IEEE Transaction on Computers*, 43(1):23–33, January 1994.

[27] E. Cockayne, B. Hartnell, S. Hedetniemi, and R. Laskar. "Perfect Domination in Graphs". *Journal of Combinatorics, Information, and System Sciences*, 18:136–148, 1993.

[28] E. Cockayne, S. Hedetniemi, and D. Miller. "Properties of Hereditary Hypergraphs and Middle Graphs". *Canadian Mathematical Bulletin*, 21(4):461–468, 1978.

[29] C. Colbourn and L. Stewart. "Dominating Cycles in Series-Parallel Graphs". *Ars Combinatoria*, 19:107–112, 1985.

[30] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to algorithms.* MIT Press, Cambridge, MA, 1989.

[31] M. Cozzens and L. Kelleher. "Dominating Cliques in Graphs". *Discrete Mathematics*, 86:101–116, 1990.

[32] P. Cull and I. Nelson. "Error-Correcting Codes on the Towers of Hanoi Graphs". Technical Report 96-20-4, Department of Computer Science, Oregon State University, October 1996.

[33] W. Dally. "Performance Analysis of k-ary n-cube Interconnection Networks". *IEEE Transaction on Computers*, 39(6):775–785, June 1990.

[34] W. Dally and C. Seitz. "The Torus Routing Chip". *Journal of Distributed Computing*, 1(3):187–196, 1986.

[35] W. Dally and C. Seitz. "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks". *IEEE Transaction on Computers*, 36(5):547–553, May 1987.

[36] J. Draper and J. Ghosh. "Multipath E-cube Algorithms (MECA) for Adaptive Wormhole Routing and Broadcasting in k-ary n-cube". In 6th *International Parallel Processing Symposium*, pages 407–410, March 1992.

[37] M. Garey and D. Johnson. *Computer and intractability: a guide to the theory of NP-completeness.* W. H. Freeman, Newyork, 1979.

[38] J. Ghosh, K. Goveas, and J. Draper. "Performance Evaluation of Parallel I/O Subsystems for Hypercube Multicomputers". *Journal of Parallel and Distributed Computing*, 17(1&2):90–106, January/February 1993.

[39] J. Ghoshal, R. Laskar, and D. Pillone. "Connected Domination and C-Irredundance". *Congressus Numerantium*, 107:161–171, 1995.

[40] S. Golomb and L. Welch. "Algebric Coding and the Lee Metric". In H. Mann, editor, *Error Correcting Codes: Proceedings of a Symposium Conducted by Mathematics Research Center*, pages 175–194. John Wiley & Sons, Inc., May 1968.

[41] J. Haviland. "Independent Domination in Regular Graphs". *Discrete Mathematics*, 143:275–280, 1995.

[42] T. Haynes and P. Slater. "Paired-Domination and the Paired-Domatic Number". *Congressus Numerantium*, 109:65–72, 1995.

[43] S. Hedetniemi. *"A Census of Models of Domination"*. Clemson University, 1996.

[44] K. Hwang. *Advanced computer architecture: parallelism, scalability, programmability.* McGraw-Hill Inc., 1993.

[45] S. Johnson and C. Ho. "Optimum Broadcasting and Personalized Communication in Hypercubes". *IEEE Transactions on Computers*, 38(9):1249–1268, September 1989.

[46] P. Kermani and L. Kleinrock. "Virtual Cut-Through: A New Communication Switching Technique". *Computer Networks*, 3(4):267–286, 1979.

[47] S. Kung, S. Jean, and C. Chang. "Fault-Tolerant Array Processors Using Single-Track Switches". *IEEE Transactions on Computers*, 38(4):501–514, April 1989.

[48] S. Lam. "Store-and-Forward Buffer Requirements in a Packet Switching Network". *IEEE Transactions on Communications*, COM-24(4):394–403, April 1976.

[49] T. Chiang Lee and J. Hayes. "A Fault-Tolerant Communication Scheme for Hypercube Computers". *IEEE Transactions on Computers*, 41(10):1242–1256, October 1992.

[50] M. Livingston and Q. Stout. "Distributing Resources in Hypercube Computers". In 3rd *Conference on Hypercube Concurrent Computers and Applications*, volume 1, pages 222–231, 1988.

[51] M. Livingston and Q. Stout. "Perfect Dominating Sets". *Conressus Numerantium*, 79:187–203, 1990.

[52] F. MacWilliams and N. Sloane. *The theory of error correcting codes.* North-Holland, Newyork, 1977.

[53] L. Ni and McKinley. "A Survey of Wormhole Routing Techniques in Direct Connect Networks". *IEEE Computer*, 26(2):62–76, February 1993.

[54] W. Oed. "Massively Parallel Processor System CRAY T3D". Technical report, Cray Research GmbH, November 1993.

[55] S. Park and B. Bose. "Broadcasting in Hypercubes with Link/Node Failures". In 4th *Symposium on the Frontiers of Massively Parallel Computation*, pages 286–290, 1992.

[56] S. Park and B. Bose. "All-to-All Broadcasting in Faulty Hypercubes". *IEEE Transactions on Computers*, 46(7):749–755, July 1997.

[57] S. Park, B. Bose, and R. Broeg. "Algorithms for Broadcasting in Faulty Hypercubes". In 6th *International Conference on Parallel and Distributed Computing Systems*, October 1993.

[58] W. Peterson and E. Waldon. *Error-correcting codes.* MIT Press, Boston, MA, 2nd edition, 1972.

[59] G. Pfister and V. Norton. ""Hot Spot" Contention and Combining in Multistage Interconnection Networks". *IEEE Transactions on Computers*, c-34(10):943–948, October 1985.

[60] P. Ramanathan and S. Chalasani. "Resource Placement in k-ary n-cubes". In *International Conference on Parallel Processing*, volume 2, pages 133–140, 1992.

[61] P. Ramanathan and S. Chalasani. "Resource Placement with Multiple Adjacency Constraints in k-ary n-cubes". *IEEE Transaction on Parallel and Distributed Systems*, 6(5):511–519, May 1995.

[62] A. Reddy and P. Banerjee. “Design, Analysis, and Simulation of I/O Architectures for Hypercube Multiprocessors”. *IEEE Transactions on Parallel and Distributed Systems*, 1(2):140–151, April 1990.

[63] A. Reddy, P. Banerjee, and S. Abraham. “I/O Embedding in Hypercubes”. In *International Conference on Parallel Processing*, pages 331–338, 1988.

[64] J. Rexford, W. Feng, J. Dolter, and K. Shin. “PP-MESS-SIM: A Flexible and Extensible Simulator for Evaluating Multicomputer Networks”. *IEEE Transactions on Parallel and Distributed Systems*, 8(1):25–40, January 1997.

[65] T. Robertazzi. *Computer networks and systems: queuing theory and performance evaluation.* Springer-Verlag, Newyork, 1994.

[66] C. Sauer and M. Chandy. *Computer systems performance modelling.* Printice-hall, Englewood Cliffs, NJ, 1981.

[67] S. Scott and G. Thorson. “The Cray T3E Network: Adaptive Routing in High Performance 3D Torus”. In *HOT Interconnects IV, Stanford University*, August 15-16 1991.

[68] C. Seitz. “The Cosmic Cube”. *Communication of the ACM*, 28(1):22–33, January 1985.

[69] C. Seitz et al. “The Design of the Caltech Mosaic C Multicomputer”. Technical report, California Inst. of Technology.

[70] C. Seitz et al. “Submicron Systems Architecture Project Semiannual Technical Report”. Technical Report Caltec-CS-TR-88-18, California Inst. of Technology, November 1988.

[71] C. Seitz et al. “The Architecture and Programming of the Ametak Series 2010”. In *Third Conf. Hypercube Concurrent Computers and Applications*, pages 33–37, January 1988.

[72] D. Shea et al. “The IBM Victor V256 Patitionable Multiprocessor”. *IBM Journal of Research and Development*, 35(5/6):573–590, September/November 1991.

[73] R. Thakur, A. Choudhary, R. Bordawekar, S. More, and S. Kuditipudi. “Passion: Optimized I/O for Parallel Applications”. *Computer*, 29(6):70–78, June 1996.

[74] T. Varvarigou, V. Roychowdhury, and T. Kailath. "Reconfiguring Processor Arrays Using Multiple-Track Models: The 3-Track-1-Spare-Approach". *IEEE Transactions on Computers*, 42(11):1281–1293, November 1993.

[75] S. Wicker. *Error control systems for digital communication and storage.* Prentice-Hall Inc., Upper saddle river, New Jersey, 1995.

[76] J. Wu and E. Fernandez. "Reliable Broadcasting in Faulty Hypercube Computers". In 11^{th} *Symposium on Reliable Distributed Systems*, October 1992.