Resource Placements in 2D Tori *

Bader Almohammad
Department of Mathematics and Computer Science
Kuwait University
P.O. Box 5969 Safat 13060, Kuwait
bdaiwi@mcc.sci.kuniv.edu.kw

Bella Bose
Department of Computer Science
Oregon State University
Corvallis, OR 97331, USA
bose@cs.orst.edu

Abstract

Results on how to place a limited number of resources in two dimensional torus-based parallel systems are described. The resources are placed so that every non-resource node is within a given distance d from some resource node. It is proved that the proposed methods are optimal in terms of reducing the maximum distance between the resource and the non-resource nodes. Simulation results show that the proposed methods are superior to the existing methods in terms of the average message latency.

1. Introduction

The family of torus graphs, such as mesh and k-ary n-cube, is becoming a popular topology for the interconnection networks of high performance parallel computers. Many practical systems, including Ametak 2010 [20], the MIT J-Machine [14] (3D mesh), the Mosaic [19], and the Cray T3D/T3E [18] (3D torus), have been built based on this network topology. In these multicomputer systems, there may be a limited number of resources, such as I/O nodes and software packages, that each processor needs to access. These limited number of resources are to be distributed in the system so that all processors can access them in a comparable manner. This problem is called "the resource placement problem".

In this paper, results on distance-d resource placements in 2D tori are given. In a distance-d placement the resources are placed in the system so that any non-resource node is at a distance of d or less from some resource node. Even though some work has been done on distance-d placement in hypercube [9, 12, 16], only a limited amount of work has been done for torus [3, 15]. In [3], the authors using Lee distance error correcting codes showed that a perfect distance-d placement is possible in a 2D torus with both dimensions of size a multiple of $(2d^2 + 2d + 1)$. The major contribution of this paper is to generalize the results given in [3]. New placements are given which are perfect or quasi-perfect (to be defined in Section 2) for a $k \times k$ torus for any k. Thus, these are the best possible placements in terms of reducing the maximum distance between the resource and the non-resource nodes.

The rest of the paper is organized as follows. Section 2 gives the definitions and the required mathematical background. The proposed scheme and some of the properties are described in Section 3. In Section 4, it is shown when and how the proposed scheme can be applied to a given 2D torus. Section 5 presents the quasi-perfect placements for $2^i \times 2^i$ tori. Section 6 shows the simulation results, in terms of inter-request time vs. average I/O communication latency, comparing the proposed scheme with a placement method used in practice. Finally, the conclusions are given in Section 7.

2. Definitions and Mathematical Background

The problem of resource placements in 2D tori has been investigated from two different point of views: error correcting codes and graph theory. Bae and Bose [3]

*This work is supported by the NSF grant MIP-9705738.

and Bose et al [7] have proposed solutions based on *Lee distance error correcting codes* [11, 5]. On the other hand, Livingston and Stout have investigated resource placements using the concept of perfect dominating sets used in graph theory [12, 13]. The Lee distance is a metric used in the field of error correcting codes. It has been shown in [7] that the Lee distance is a natural metric to use with toroidal networks. Many topological properties of a toroidal network can be derived from this useful metric [7].

Mixed Radix Notation: In a mixed radix notation, any integer I, $0 \leq I < K = k_{n-1}k_{n-2}\ldots k_0$, is represented as an n-dimensional vector $X = x_{n-1}x_{n-2}\ldots x_0$ over $K = k_{n-1}k_{n-2}\ldots k_0$, where $x_i \in \{0, 1, 2, \ldots, k_i - 1\}$. $I(X)$, the *Integer Value* of X, is defined as: $I(X) = x_0 + x_1k_0 + x_2k_0k_1 + \cdots + x_{n-1}k_0k_1\ldots k_{n-1} = (\sum_{i=1}^{n-1}(x_i\prod_{j=0}^{i-1}k_j)) + x_0$. For example, if 432 is a vector over 543; then $I(432) = 4(12) + 3(3) + 2 = 59$.

Lee Weight: Let X be a vector in the mixed radix notation over K. 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.

Lee Distance: Let X and Y be vectors in the mixed radix notation over K. 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.

Torus: Let T be an n-dimensional torus denoted as $T_{k_{n-1}kn-2\ldots k_0}$, with each dimension of size k_i, $i = 0, 1, \ldots, n-1$. T has $N = \prod_{i=0}^{n-1} k_i$ nodes. Each node of T is labeled with a mixed radix n-dimensional vector over $K = k_{n-1}k_{n-2}\ldots k_0$ called its address. 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 iff $D_L(X,Y)$ = 1. A k-ary n-cube graph, Q_k^n, is a torus in which all the dimensions have the same radix.

In the rest of this section an overview of the major related results achieved by the previous studies is presented. First, some terms and concepts are defined. Then, a concise summary of these results will follow.

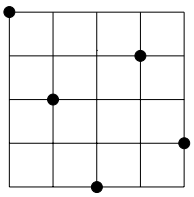


Figure 1. Perfect distance-1 placement in a 5 × 5 torus.

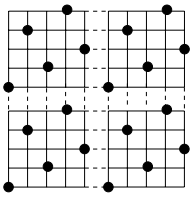


Figure 2. Perfect distance-1 placement for a 10 × 10 torus in which a 5 × 5 torus is used as a tiling block. Dashed lines indicate links among tiles.

Definition 2.1 *A placement is perfect distance-d if any non-resource node is within a distance of d or less from exactly one resource node (An example is given in Figure 1).*

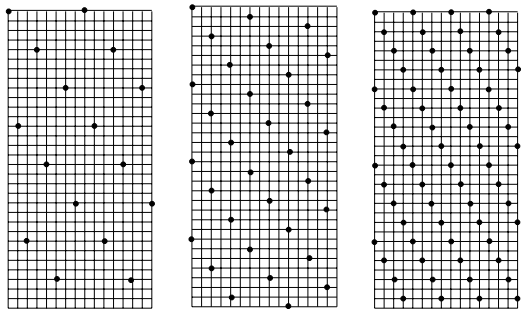
Definition 2.2 *Radius-d packing sphere of a resource node A is the set of all nodes within a distance of d or less from A.*

It has been proved in [3] that the number of nodes at a distance of d or less from a given node (called the volume of a radius-d sphere) is:

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

It is easy to verify that the maximum possible volume of a radius-d sphere in a 2D torus, $d \geq 1$, is $(2d^2 + 2d + 1)$.

Bae and Bose have shown that for a given $X \times Y$ 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 a $k \times k$ torus, $k = 2d^2 + 2d + 1$, is obtained by using *Lee distance error correcting codes*. The resources are placed



(a) (b) (c)

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

at $[i,\ 2d^2i]\ (mod\ k)$, $i = 0, 1, \ldots, (k-1)$. Next, the $k \times k$ torus is used to tile the larger $X \times Y$ torus [2, 3]. Figure 2 shows how a 5×5 torus is used to tile a 10×10 torus. They have also proved in [3] 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. Thus, perfect placements do not exist for a large number of 2D tori. This is the main motivation for investigating what is so-called *quasi-perfect placements*. The following section defines this term and presents the major results of this investigation.

3. Quasi-Perfect Placement Scheme for Q_k^2

As mentioned earlier, perfect placements do not exist for a large class of 2D tori. For those cases which do not have perfect placements *a quasi-perfect placement* can be achieved. A *quasi-perfect* placement is a placement in which the following two conditions are satisfied:

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 3.

Finding an allocation method for quasi-perfect placements would offer more flexibility in choosing the dimensions of an interconnection network as well as abilities 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 a power of 2 in each dimension). Figure 3-(b) shows a quasi-perfect distance-2 placement for a torus of size 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 3-(a) and (c), respectively.

3.1. The Proposed Method: QP_K Placement Scheme

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 the resources are placed in $(i,\ 2i\ mod\ 5),\ 0 \le i < 5$, as illustrated in Figure 1. A special linear placement scheme for Q_k^2 is defined as follows:

QP_k **Scheme** *is a linear resource placement method for Q_k^2 interconnection networks; in QP_k the generator matrix is given by $G = [d \quad (d+1)]$. Thus, the resource nodes are placed at $QP_k = \{(x,y) : x = id\ (mod\ k),\ y = i(d+1)\ (mod\ k), 0 \le i < k\}$ where $2d^2 + 2 \le k \le 2(d+1)^2 + 1,\ d \ge 0$.*

The following subsections prove that QP_K placements are:

(1) Quasi-perfect distance-$(d-1)$ when $2d^2 + 2 \le k \le 2d^2 + 2d$.
(2) Perfect distance-d placement when $k = 2d^2 + 2d + 1$.
(3) Quasi-perfect distance-d placement when $2d^2 + 2d + 2 \le k \le 2(d+1)^2 + 1$.

The rest of this subsection presents definitions and lemmas needed in the following subsections. Due to space limitations, the proofs for the lemmas and the theorems are not given here, but can be found in [1].

Definition 3.1 *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* $n_j \in QP_k$*. Further, let* $+_s$ *and* $-_s$ *denote the binary operators, 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 3.1 $(QP_k,\ +_s)$ *is a finite Abelian group. (i.e.* $(QP_k,\ +_s)$ *satisfies the closure, identity element, associativity, inverse, and commutativity properties).*

Lemma 3.2 *The cardinality of* QP_k *is* k.

Lemma 3.3 *Let* W_L *(x) be Lee weight of a given node* x. W_L *(*n_i*) =* W_L *(*n_{k-i}*),* $n_i \in QP_k$.

Lemma 3.4 *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* $W_L(n_i)$*:* $i \neq 0$ *and* $n_i \in QP_k\}$.

3.2. Minimum Distance Between any Two Resources

In this subsection, QP_k is proved to satisfy the first necessary condition of a quasi-perfect placement.

Theorem 3.1 *Let* $D(QP_k)$ *be the minimum Lee distance,* D_L*, between any two resource nodes in* QP_k*, then:*
(1) $D(QP_k) \geq 2d-1$*, when* $2d^2+2 \leq k \leq 2d^2+2d$.
(2) $D(QP_k) \geq 2d+1$*, when* $2d^2+2d+1 \leq k \leq 2(d+1)^2+1$.

Corollary 3.2 *Let* $R_{a,d}$ *be radius-*d *packing sphere of node* a*, then:*

- **(1)** $R_{a,(d-1)} \cap R_{b,(d-1)} = \phi$*, for any two nodes* a *and* $b \in QP_k$*, when* $2d^2+2 \leq k \leq 2d^2+2d$.
- **(2)** $R_{a,d} \cap R_{b,d} = \phi$*, for any two nodes* a *and* $b \in QP_k$*, when* $2d^2+2d+1 \leq k \leq 2(d+1)^2+1$.

Corollary 3.3 QP_k *is a perfect distance-*d *placement when* $k = 2d^2+2d+1$.

3.3. Maximum Distance Between a Non-Resource node and the Closest Resource Node

In the previous subsection, QP_K was proved to maintain the first necessary condition for a quasi-perfect placement. In this subsection, QP_k is shown to satisfy the second necessary condition for a quasi-perfect placement.

Lemma 3.5 *Let* T *be a* $k \times k$ *torus. If all the non-resource nodes at a distance of* $(d+i+1)$ *from any resource node are within a distance of* $(d+i)$ *or less from some other resource nodes, then each node in* T *is within a distance of* $(d+i)$ *or less from at least one resource node. i.e. no non-resource node is at a distance of* $(d+i+1)$ *or more from all resource nodes.*

Lemma 3.6 *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.*

Theorem 3.4 *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* $2d^2+2 \leq k \leq 2d^2+d$.

Theorem 3.5 *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* $2d^2+d+1 \leq k \leq 2d^2+2d$.

Corollary 3.6 QP_k *is a quasi-perfect distance-*$(d-1)$ *placement when* $2d^2+2 \leq k \leq 2d^2+2d$.

Theorem 3.7 *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* $2d^2+2d+2 \leq k \leq 2d^2+3d+1$.

Theorem 3.8 *Any node in a* $k \times k$ *torus is within a distance of* $(d+1)$ *from at least one resource node when* QP_k*, where* $2d^2+3d+2 \leq k \leq 2d^2+4d+2$.

Theorem 3.9 *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* $k = 2d^2+4d+3$.

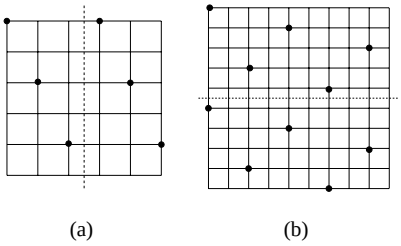
Corollary 3.10 QP_k *is a quasi-perfect distance-*d *placement when* $2d^2+2d+2 \leq k \leq 2d^2+4d+3$.

Placement	Tiling Block	Number of Resources
Quasi-Perfect Distance-0	2×2	450
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 1. The possible quasi-perfect placements for a 30×30 torus using QP_k placement scheme.

Placement	Tiling Block	Number of 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-2	12×12	72

Table 2. The possible quasi-perfect placements for a 24×36 torus using QP_k placement scheme.



(a) (b)

Figure 4. (a) vertical bisection in a 6×6 torus, (b) horizontal bisection in a 10×10 torus.

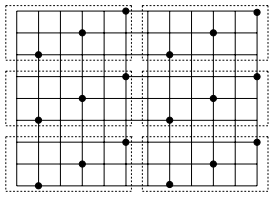


Figure 5. A 9×12 torus tiled by a vertical bisection of a 6×6 torus.

4. 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 $X \times Y$ torus. As shown in Section 2, the quasi-perfect placement in a $k \times k$ torus can be used in tiling a larger $X \times Y$ torus given that $(k \mid X)$ and $(k \mid Y)$. Some examples are shown in Tables 1 and 2, respectively, for a 30×30 and a 24×36 networks.

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 4.1 *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.*

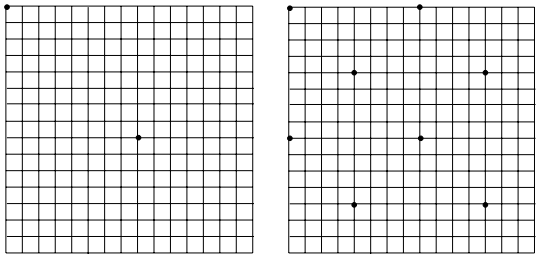
(1) If d is even then the horizontal bisection of T uses $\frac{k}{2}$ resources and maintains a quasi-perfect distance-l placement assuming there exist wraparound links among the corresponding nodes of row$_0$ and row$_{\frac{k}{2}}$.

(2) If d is odd then the vertical bisection of T uses $\frac{k}{2}$ resources and maintains a quasi-perfect distance-l placement assuming there exist wraparound links among the corresponding nodes of column$_0$ and column$_{\frac{k}{2}}$.

Examples of cases in which Theorem 4.1 holds are shown in Figure 4. In Figure 4-(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 4-(b) shows the placements 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. Figure 5 shows how a 6×3 torus can be used in tiling a 9×12 torus.

5. 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



(a) (b)

Figure 6. Quasi-perfect placements as applications of Theorem 5.1 in 16 × 16 tori: (a) quasi-perfect distance-7, (b) quasi-perfect distance-3.

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

Table 3. Scaled-down quasi-perfect placements for a 32 × 32 torus that result from applying PLACE-SCALED-DOWN algorithm.

represents the basis for scaling the number of resources down.

Theorem 5.1 *Let T be a $k \times k$ torus, and $k = 2^i$, i an integer ≥ 1. Then:*
$P = \{ (0,0),\ (\frac{k}{2}, \frac{k}{2}) \}$ is a quasi-perfect distance-$(\frac{k}{2} - 1)$ placement for T.

Figure 6 shows an application of this theorem. Figure 6-(a) shows the use of two resources in 16×16 torus to achieve a quasi-perfect distance-7 placement. On the other hand, Figure 6-(b) shows how to use eight resource 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. Then, use the 8×8 torus in tiling 16×16 torus.

Figure 7 formally describes a scaling down algorithm based on Theorem 5.1. For example, different scaled-down quasi-perfect placements can be used in a 32×32 torus. Table 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

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, $R = 2 \times 4^i, i \geq 0, R \leq k$.
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 7. PLACE-SCALED-DOWN Algorithm.

Placement	Number of Resources
Quasi-Perfect Distance-0	512
Quasi-Perfect Distance-0	256
Quasi-Perfect Distance-1	128
Quasi-Perfect Distance-2	64

Table 4. Possible scaled-up quasi-perfect placements for a 32 × 32 torus based on PLACE-SCALED-UP algorithm.

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 8 describes a possible method for scaling up the number of resources in a torus of this class. Table 4 lists the possible scaled-up quasi-perfect placements for a 32×32 torus based on PLACE-SCALED-UP algorithm.

Finally, it should be noticed that any placement that results from applying PLACE-SCALED-DOWN or PLACE-SCALED-UP algorithm can be used as a tiling block. For example, a 32×32 torus can be used to tile a 64×96 torus. Hence, any of the placements listed in Table 3 or Table 4 can be achieved in a 64×96 torus.

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, $R = k2^i \times, i \geq 0, R \geq k$.
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 8. PLACE-SCALED-UP algorithm.

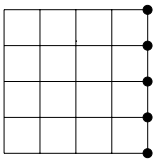
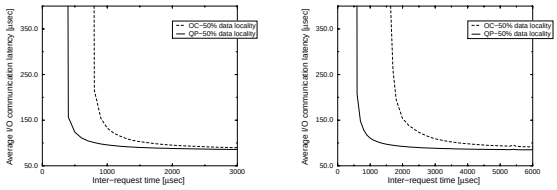


Figure 9. Outer-Column I/O placement for a 5 $\times$ 5 torus.

6. Simulation Results

A modified version of Proteus [8] simulator is employed to investigate the influence of I/O placement strategies on average network latency, i.e. average time a message needs to traverse from the source to the destination. QP_k scheme is compared to what we call Outer-Column (OC) strategy. In OC all the resources are placed in one column of a 2D torus or a 2D mesh. Intel uses a similar placement for the Paragon whose interconnection network is a 2D mesh [6]. Figure 9 shows OC in a 5×5 torus. The simulation considers both scientific and transaction environments. The main distinction between them is assumed to be the overlapping of I/O and computation communication in the latter but not in the former. 5×5, 13×13, 8×8, and 16×16 are the simulated networks. The first two represent perfect placements while the others illustrate quasi-perfect placements.



(a) 8×8 (b) 13×13

Figure 10. Scientific Environment: Inter-request time [μsec] vs. average I/O communication latency [μsec] when data locality is set to 50%.

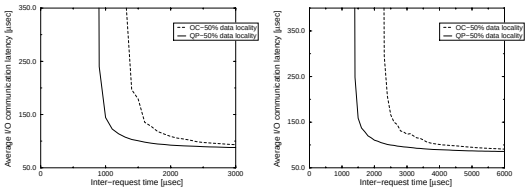
The simulation is designed to work as follows. The nodes generate requests at random instants. The average time between two sequential instants, Inter-request time (IR), specifies the network contention. Small IR means high volume of traffic and high contention while large IR indicates low volume of traffic and low contention. IR is 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 [4, 17, 10, 16].

All the simulation results indicate that QP_k is superior to OC in terms of reducing average network latency. Figures 10 and 11 show some of the simulation results.

Detailed overview and analysis for the results can be seen in [1].

7. Conclusions

QP_k is a placement scheme for the $k \times k$ tori. In this paper, QP_k has been defined and proven to maintain the following properties: (1) QP_k is a quasi-perfect distance-$(d-1)$ placement when $2d^2 + 2 \leq k \leq 2d^2 + 2d$, (2) QP_k is a perfect distance-d placement when $k =$



(a) 8×8 (b) 13×13

Figure 11. Transaction Environment: 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%.

$2d^2 + 2d + 1$, and (3) QP_k is a quasi-perfect distance-d placement when $2d^2 + 2d + 2 \leq k \leq 2d^2 + 4d + 3$. 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?"

References

[1] B. F. Almohammad. *On Resource Placements and Fault-Tolerant Broadcasting in Toroidal Networks*. PhD thesis, Department of Computer Science, Oregon State University, November 1997.

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

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

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

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

[6] 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.

[7] 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.

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

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

[10] 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.

[11] 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.

[12] M. Livingston and Q. Stout. "Distributing Resources in Hypercube Computers". In 3^{rd} *Conference on Hypercube Concurrent Computers and Applications*, volume 1, pages 222–231, 1988.

[13] M. Livingston and Q. Stout. "Perfect Dominating Sets". *Conressus Numerantium*, 79:187–203, 1990.

[14] M. Noakes, D. A. Walach, and W. J. Dally. "The J-Machine Multicomputer: An Architectural Evaluation". In $20^{t}h$ *International Symposium on Computer Architecture*, pages 224–235, 1993.

[15] P. Ramanathan and S. Chalasani. "Resource Placement in k-ary n-cubes". In *International Conference on Parallel Processing*, volume 2, pages 133–140, 1992.

[16] 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.

[17] T. Robertazzi. *Computer networks and systems: queuing theory and performance evaluation*. Springer-Verlag, Newyork, 1994.

[18] 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.

[19] C. Seitz et al. "Submicron Systems Architecture Project Semiannual Technical Report". Technical Report Caltec-CS-TR-88-18, California Inst. of Technology, November 1988.

[20] 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.