Mixed-radix Twisted Torus Interconnection Networks

José M. Cámara[1], Miquel Moretó[2], Enrique Vallejo[3], Ramón Beivide[3], José Miguel-Alonso[4], Carmen Martínez[3] and Javier Navaridas[4]

[1]University of Burgos
Dept. of Electromechanical Engineering
checam@ubu.es

[2]Technical University of Catalonia
Dept. of Computer Architecture
mmoreto@ac.upc.edu

[3]University of Cantabria
Computer Architecture Group
{enrique, mon,carmenmf}@atc.unican.es

[4]University of the Basque Country
Dept. of Computer Architecture and Technology
{j.miguel, javier-navaridas}@ehu.es

Abstract

Many parallel computers use Tori interconnection networks. Machines from Cray, HP and IBM, among others, exploit these topologies. In order to maintain full network symmetry, 2D and 3D Tori (k-ary 2-cubes and k-ary 3-cubes) must have the same number of nodes (k) per dimension resulting in square or cubic topologies. Nevertheless, for practical reasons, computer engineers have designed and built 2D and 3D Tori having a different number of nodes per dimension. These mixed-radix topologies are not edge-symmetric which translates into poor performance provoked by an unbalanced use of the network links.

In this paper, we propose and analyze twisted 2D and 3D Tori which remove the network bottlenecks present in mixed-radix standard Tori. These new topologies recover edge-symmetry and, consequently, balance the utilization of their links. We describe the distance-related parameters of these twisted networks and use simulation to asses their performance under synthetic loads. The obtained results show noticeable and consistent performance gains (up to a 88% increase in accepted load). In addition, we propose scalable and practicable packet routing and folding techniques for these interconnection subsystems. The complexity of the resulting architectural solutions is similar to the one exhibited by traditional routing and folding mechanisms employed in standard Tori. This fact together with the performance improvements obtained could justify the use of these twisted topologies in the future.

This work has been done with the support of the Spanish Ministerio de Educación y Ciencia, grants TIN2004-07440-C02-01, TIN2004-07440-C02-02, AP2004-6907 and AP-2005-3318. Mr. J. Navaridas is supported by a doctoral grant of the UPV/EHU.

1-4244-0910-1/07/$20.00 ©2007 IEEE.

1. Introduction

Multiple parallel computers using different direct interconnection networks have been designed in last decades. Mesh, Torus and Hypercube have been the most popular topologies. Nowadays, Hypercubes seem to decline in favor of lower degree networks such as two-dimensional and three-dimensional Tori and Meshes. Different machines, such as the Alpha 21364-based HP GS1280 [10] and the Cray X1E vector computer [9], have used 2D Tori. Others, such as the Cray T3D and T3E [19], which preceded the Cray XT3 [8], have used 3D Tori. The IBM BlueGene is a notable example of a massively parallel computer that joins $2^6 \times 2^5 \times 2^5$ nodes in a mixed-radix 3D Torus [1], .

Typically, a 2D Torus arranges its N nodes in a square Mesh with $\sqrt{N}$ nodes on each side and adding $2\sqrt{N}$ wraparound links. Above, more or less, a thousand nodes it has been shown that parallel computers should use 3D topologies, being a cubic 3D Torus of side $\sqrt[3]{N}$ the most desirable solutions [2].

Each dimension of a Torus network may have a different number of nodes, leading to rectangular and prismatic topologies for two and three dimensions respectively. These topologies are denoted in [11] as mixed-radix networks. Mixed-radix Tori are often built for practical reasons of packaging and modularity. For example, the HP GS1280 [10] employs a 2D rectangular network and the IBM BlueGene a 3D prismatic one [1]. However, mixed-radix Tori have two important drawbacks: first, they are no longer edge-symmetric and second, the distance-related network parameters (diameter and average distance) are quite far from the optimum values of square and cubic topologies. The edge asymmetry introduces load imbalance in these networks, and for many traffic patterns the load on the longer dimensions is larger than the load on the shorter ones

[11]. In addition, maximum and average packet delays are relatively long as they depend on the poor values of diameter and average distance exhibited by these networks.

In this paper, we propose and analyze alternative mixed-radix 2D and 3D Torus topologies that avoid the two above mentioned problems by adequately twisting the wraparound links of one or two network dimensions. As we will see, the performance exhibited by these twisted Tori is notably higher than the one obtained when using their standard mixed-radix counterparts.

In general, a twist can be applied to the wraparound links of any rectangular or prismatic Tori with aspect ratios $k:1$ (the longest dimension has k times the number of nodes of the shortest one) or $k:j:1$, for any $k \geq j \geq 1$. However, it is clear that as the aspect ratio increases, the links in the longest dimensions become a more severe bottleneck. In this work, we will focus just on networks with $2:1$ and $2:1:1$ aspect ratios. These ratios have been previously used by manufacturers [10, 7], allowing the number of nodes to be a power of two, which is sometimes a desired property. Even more, these ratios represent the simplest way to upgrade a square or cubic network, doubling their number of nodes by only rearranging some peripheral links. Anyway, the methodology we present here is applicable to any rectangular or prismatic network independently of the selected aspect ratio.

The main contributions of this research are:

- The proposal of twisted 2D and 3D Tori as alternative topologies to mitigate the performance flaws on existing rectangular and prismatic Tori.
- A detailed analysis of their topological properties.
- A novel and simple routing mechanism for the proposed topologies.
- A performance evaluation, both theoretical and by means of network simulations, showing performance increases up to 88%.
- A novel layout for the proposed networks that keeps link length under a bounded value, eliminating long peripheral wires and facilitating their implementation.

The rest of this paper is organized as follows. Section 2 considers related work. Section 3 is devoted to analyze 2D rectangular twisted Tori providing their distance-related properties and a minimal adaptive routing. In Section 4 we study 3D twisted Tori. Section 5 describes the simulation tools employed in our experiments and provides performance data for rectangular and prismatic networks. Section 6 deals with network folding and wireability and, finally, Section 7 concludes the paper summarizing the main findings of this research.

2. Related Work

The idea of twisting 2D square Tori in one of its two dimensions for obtaining architectural benefits is not new. A twist of 1 in the wraparound links of dimension X was employed in the Illiac IV in order to provide a Hamiltonian ring embedded on its topology which facilitated control operations. Doubly twisted Tori were introduced by Sequin at [18] looking for optimal mappings of binary trees onto a processor array. Other square meshes with twisted wraparound links were proposed in [5] to reduce maximum and average distance among their nodes.

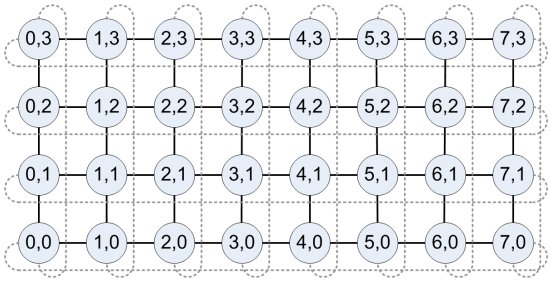
Notwithstanding, including a twist in a square Torus gives little improvement in packet throughput and delay. Just a 4% gain in average distance can be obtained [5]. In addition, the natural symmetry of the network dimensions is broken when introducing the twist. Other options such as including two twists, one per dimension, is an interesting solution for pursuing better performance. However, it has collateral drawbacks, such as a higher asymmetry and lack of optimal adaptive routing algorithms.

Although rectangular Tori have worse relative performance than square ones, engineers have built them to satisfy node count limits and other practical reasons. In addition, as stated before, they are very easy to build by joining two square Tori. Some parallel machines have used a rectangular Torus for their interconnection network [10]. Notable supercomputer architectures using prismatic 3D Torus are the IBM BlueGene [1] and the Cray XT3 [8].

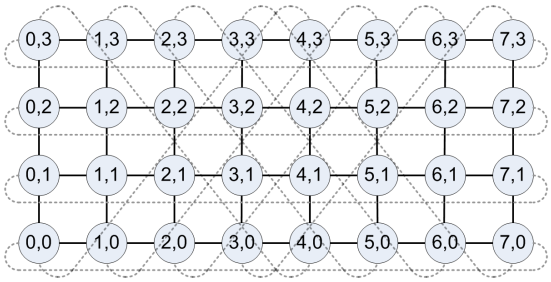
Not too much work has been done in order to improve the performance of rectangular Tori by twisting one of its two dimensions. The results introduced in [5] appear to be the first attempt in this direction. Similar topologies were used in [21] as a component of hierarchical networks denoted as Recursive Diagonal Tori. The work in [10] also empirically considers the use of a twist in 2D rectangular Tori. The study of 3D prismatic networks is an important extension of the 2D case. Up to our knowledge, no significative work has been published for optimizing network performance in prismatic 3D Tori by twisting one or two of the network dimensions.

3. Bi-dimensional Networks

In this Section we study Rectangular Twisted Tori (RTT) as an alternative to the standard $2:1$ Rectangular Tori (RT). Figures 1(b) and 1(a) present examples of both networks. Note that RTT is obtained from RT by adding a twist of a columns to the vertical wraparound links. RTTs as the one depicted in Figure 1(b) has been previously considered in [10, 21]. Nevertheless, the topological properties of these networks were not studied and no practicable routing algorithm neither implementability issues were considered for



(a) Rectangular Torus.



(b) Rectangular Twisted Torus.

Figure 1. RT and RTT of size 8×4 ($a = 4$).

them.

The first issue in the analysis of RTTs is why a twist of a columns in the vertical dimension of a $2a \times a$ rectangular network is the most attractive option. This question can be answered both empirically and theoretically.

From an empirical point of view, just a look to Figure 2 justifies that a twist of a columns in the vertical dimension is the most adequate choice. In this Figure, we report the average distance obtained with different twists for a 32×16 network ($a = 16$). It is clear that a twist equal to a leads to the minimum average distance and, as we will see later, it provides edge-symmetry, which positively affects network performance.

From a theoretical point of view, it can be proved that this choice minimizes the diameter and average distance

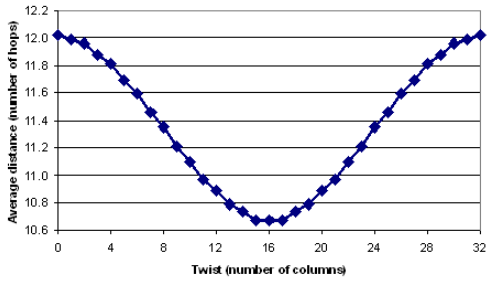


Figure 2. Average distance for different twist values in a 32×16 network.

among all the possible twists in Rectangular Tori (RT) of size $2a \times a$. However, the proof of these statements is quite laborious and does not contribute to a better understanding of the work presented here. For that reason, we have omitted such proofs. Nevertheless, the interested reader could deduce these assertions from the results presented in [14]. In the next Subsection, we present a more *digestive* analysis of the distance properties of RTTs based on the topological properties of their underlying rings.

3.1. Rectangular Twisted Tori

A standard RT of size $2a \times a$, as the one depicted in Figure 1(a), can be visualized as a composition of two sets of orthogonal rings of sizes $2a$ and a. Node labels correspond to pairs (x, y) indicating column and row. Nodes with the same x or the same y component belong to the same vertical or horizontal ring, respectively.

The diameter of a network, k, is the length of the longest minimum path. The average distance, $\bar{k}$, is the average length of all minimum paths from one node to any other different one. We also consider $\tilde{k}$, which represents the distance averaged among all of the nodes, including the source itself. Note that, in a network with N nodes, $\bar{k} = \tilde{k} \cdot \frac{N}{N-1}$. In order to study the distance properties of rectangular Tori, we first consider such properties in a ring.

Remark 1 *The distance parameters k and $\tilde{k}$ of a ring of n nodes are:*

$$k = \frac{n}{2} \quad ; \quad \tilde{k} = \frac{n}{4}.$$

Note that in all of the calculations we consider n to be even; having n odd would only slightly modify the resulting values.

Remark 2 *The diameter k and average distance $\bar{k}$ of a RT of $2a \times a$ nodes are:*

$$k = \frac{2a}{2} + \frac{a}{2} = \frac{3}{2} \cdot a \quad ; \quad \bar{k} = \left(\frac{2a}{4} + \frac{a}{4}\right) \cdot \frac{2a^2}{2a^2 - 1} \approx \frac{3}{4} \cdot a$$

It is interesting to note that in this $2:1$ Tori under uniform traffic the average number of hops on X will be twice as those on Y, so that if every X link is 100% busy, Y links will be at most 50% busy. Then, the maximum average link utilization will be 75%.

Next, we focus our attention on studying the distance properties of $2a \times a$ RTT networks as the one depicted in Figure 1(b). To obtain closed formulae for the diameter and average distance of the network, we describe its distance distribution $\Omega_a(d)$, that is, the number of nodes at distance d from node $(0, 0)$. As these networks are node symmetric all the nodes see the same distance distribution.

Proposition 3 *The distance distribution of a RTT with* $2a \times a$ *nodes is:*

$$\Omega_a(0) = 1; \quad \Omega_a(d) = 4d, \; if \; 0 < d < a; \quad \Omega_a(a) = 2a - 1.$$

This proposition is a direct consequence of the results introduced in [14], where the distance distribution of a general family of networks denoted as *Gaussian graphs* is presented. From this distance distribution we can easily obtain the following result:

Corollary 4 *The diameter* k *and the average distance* $\bar{k}$ *of a RTT with* $2a \times a$ *nodes are:*

$$k = a \quad ; \quad \bar{k} = \frac{\sum_{i=1}^{a} i \cdot \Omega_a(i)}{2a \cdot a - 1} = \frac{4a^2 - 1}{3(2a^2 - 1)} a \approx \frac{2}{3} \cdot a.$$

Thus, just adding a twist to $2a$ links in a RT produces a significant reduction of the diameter and average distance. More in detail, diameter is reduced in 33.3% and average distance in 11.1%. Furthermore, a very important property is regained in the RTT: the symmetry in dimensions X and Y. As we will see in Section 5, the use of this twist removes the network bottleneck in the longer dimension. This means that under uniform traffic, both X and Y links can be fully utilized (100%), leading to a 33.3% increase in link utilization with respect to RT.

3.2. Routing in Rectangular Twisted Tori

Now, we introduce a routing algorithm which computes the shortest path between any pair of nodes by applying simple operations on their coordinates. Although we restrict our routing to $2a \times a$ node networks, a generalization for other rectangular Tori can be straightforwardly obtained.

A routing record, $(\Delta X, \Delta Y)$, heading the packet will be generated by the source network interface. ΔX represents the number of links that the packet must traverse along the axis of the first coordinate and ΔY the number of links along the second coordinate's axis. Their signs indicate E/W and N/S directions. Routers will process the header information in the same way as in a Torus, decrementing the corresponding field header before sending the packet to the selected neighbor. A packet with $\Delta X = 0$ and $\Delta Y = 0$, will have reached its destination and will be delivered. There are multiple implementations for generating routing records in RTT but we present in Algorithm 1 the mechanism that has lower temporal complexity. We will employ in our experiments an optimal fully adaptive routing mechanism built on this basis.

The correctness of this routing mechanism is proved with a geometrical approach. This kind of regular graphs can be fully represented by plane tessellations [20]. The graph is characterized by a tile of area $2a^2$ that tessellates the plane. Each tile is defined by its origin node (left-lower corner

input : a: Parameter of the $2a \times a$ RTT.
(s_x, s_y): Source node.
(d_x, d_y): Destination node.
output: $(\Delta X, \Delta Y)$: routing_record
begin
DO IN PARALLEL:
begin

$$\begin{cases} \Delta x_0 := d_x - s_x; \\ \Delta y_0 := d_y - s_y; \end{cases} \quad \begin{cases} \Delta x_1 := d_x - s_x - a; \\ \Delta y_1 := d_y - s_y - a; \end{cases}$$

$$\begin{cases} \Delta x_2 := d_x - s_x + a; \\ \Delta y_2 := d_y - s_y - a; \end{cases} \quad \begin{cases} \Delta x_3 := d_x - s_x + 2a; \\ \Delta y_3 := d_y - s_y; \end{cases}$$

$$\begin{cases} \Delta x_4 := d_x - s_x - 2a; \\ \Delta y_4 := d_y - s_y; \end{cases} \quad \begin{cases} \Delta x_5 := d_x - s_x + a; \\ \Delta y_5 := d_y - s_y + a; \end{cases}$$

$$\begin{cases} \Delta x_6 := d_x - s_x - a; \\ \Delta y_6 := d_y - s_y + a; \end{cases}$$

end
$(\Delta X, \Delta Y) := (\Delta x_i, \Delta y_i)$ such that $|\Delta x_i| + |\Delta y_i|$ is minimum;
end

Algorithm 1: Routing Record Generator for RTT.

nodes highlighted in each rectangle of Figure 3 for a RTT of 8×4 nodes). Nodes in the same position of different tiles represent the same node of the network. As the diameter of the network is a, any node reaches any other destination with no more than a jumps.

The shadowed areas of Figure 3 show the possible destinations for two given source nodes, $(0, 2)$ and $(7, 3)$. Obviously, minimum paths from any node in the original tile will never go further than a jumps out of the tile, as shown in the figure with the black border that comprises the original tile and the six immediate neighbors. The next part of the proof is to show in terms of a that the eight segments of this border only comprise the seven tiles whose left-lower corners are the nodes $\{(0,0), (a,a), (-a,a), (-2a,0), (2a,0), (-a,-a), (a,-a)\}$. This implies that the minimal path within any pair of nodes can be found in that tile set. Details are omitted for their obviousness and for the sake of simplicity.

Our routing algorithm computes all the possible paths from a source node to the copy of the destination node in each one of the 7 possible tiles. After computing these paths' lengths in parallel, the algorithm returns the routing record having minimal length.

4. Three-dimensional Networks

The resource unbalance exhibited by mixed-radix 2D networks also happens in non-cubic 3D Tori, or Prismatic Tori (PT). In this section we focus our attention on building 3D networks derived from RTTs. We present two different approaches. The simplest one considers networks built with just one twist by piling up a RTTs. We have denoted such a network as Prismatic Twisted Torus (PTT). Next, we consider the network with twists on both Y and Z peripheral

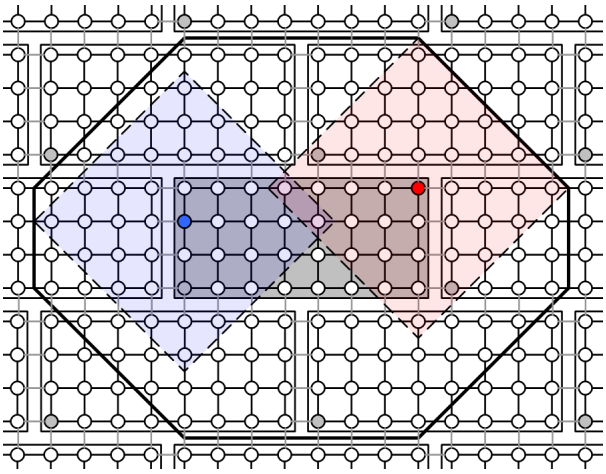


Figure 3. Possible Paths Departing from a 8×4 RTT.

links, leading to a Prismatic Doubly Twisted Torus (PDTT).

As we want to compare PT against PTT and PDTT let us consider first its distance-related parameters. The following result is direct as the PT is the product of three rings:

Proposition 5 *The diameter k and average distance $\bar{k}$ of a 3D PT of size $2a \times a \times a$ are:*

$$k = a + \frac{a}{2} + \frac{a}{2} = 2a \quad ; \quad \bar{k} = \left(\frac{2a}{4} + \frac{a}{4} + \frac{a}{4}\right)\frac{2a^3}{2a^3 - 1} \approx a$$

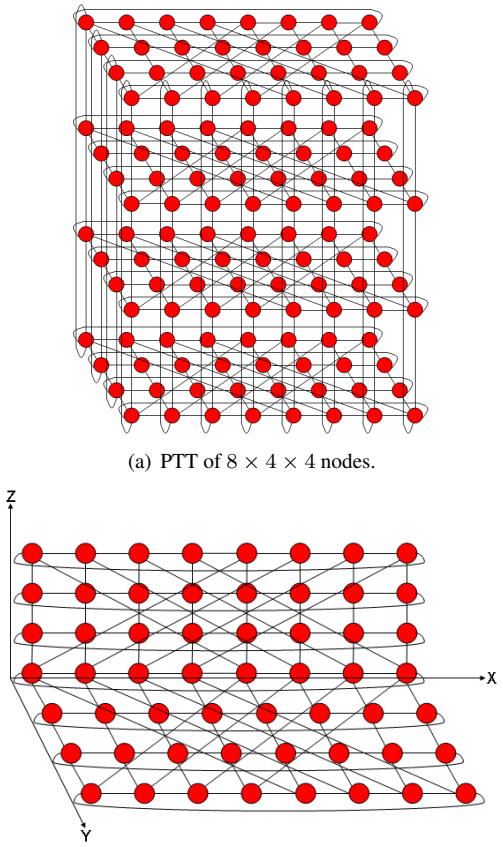
Also, note that, same as in RT, when the X links in PT are fully utilized, links in Y and Z can be at most 50% used under uniform traffic, leading to a global link utilization of 66.7% [6].

A PTT is composed of a RTTs of size $2a \times a$ on the XY planes, with links in Z forming rings. In Figure 4(a) we can see a PTT network of $8 \times 4 \times 4$ nodes, where most links in the Z dimension have been omitted for the sake of clarity. The expressions for the distance-related parameters of a PTT are:

Proposition 6 *The diameter k and average distance $\bar{k}$ of a PTT of size $2a \times a \times a$ are:*

$$k = \frac{3}{2} \cdot a \quad ; \quad \bar{k} \approx \frac{2a}{3} + \frac{a}{4} = \frac{11}{12} \cdot a$$

This result is direct if we see the PTT as a RTT in the XY plane and a ring of a nodes in the Z dimension. From a topological point of view, the PTT provides an improvement of approximately 25% in diameter and 8.33% in average distance, and provides symmetry in X, Y, allowing the full use of these links under uniform traffic and providing an increase of 25% in link utilization. A routing algorithm for PTTs is very simple as we can independently compute the



(a) PTT of $8 \times 4 \times 4$ nodes.

(b) PDTT of $8 \times 4 \times 4$ Nodes.

Figure 4. 3D Prismatic Twisted Tori and Doubly Twisted Tori.

routing record in the XY plane as proposed in Subsection 3.2, and in the ring of the Z dimension.

PDTT is built by applying a twist to both Y and Z peripheral links over the X dimension. Thus, this network consists in building RTTs in the XY and XZ planes. This makes the network fully symmetric on X, Y and Z, leading to an increase of up to 50% in global link utilization with respect to PT. Figure 4(b) shows horizontal and vertical cuts of a PDTT. Expressions for the distance-related parameters and the routing record generator for PDTTs can be derived analogously as in RTT. We omit details for simplicity.

Proposition 7 *The diameter k and average distance $\bar{k}$ of a PDTT sized $2a \times a \times a$ are:*

$$k = \frac{3}{2} \cdot a \quad ; \quad \bar{k} \approx \frac{7}{8} \cdot a$$

5. Performance Evaluation

We describe in this Section the behavior of these mixed-radix networks under the effect of different synthetic loads.

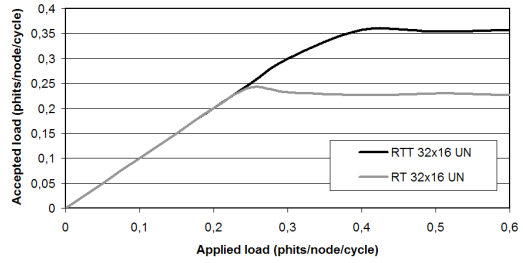
As a first approach, we have used uniform synthetic loads to better understand the advantages of the proposed topologies. Uniform traffic is useful for detecting network bottlenecks as it reflects the topological properties of the network under study. Afterwards, we have completed this study using different permutations and adverse traffic patterns. Synthetic loads capture the representative aspects of application-driven workloads and are easier to design and manipulate. Furthermore, we do not use traces or real traffic as it becomes unpracticable with large network sizes. In contrast with uniform traffic, other loads are sensitive to the order in which nodes are labeled as packet destinations are selected depending on their label. After evaluating different mappings, we found that the best results are obtained if the nodes are labeled following the shortest dimensions first (i.e., node label increases along a column in RT or RTT, or in YZX order in 3D networks). In order to make a fair comparison, we have used these mappings on our simulated experiments.

Network evaluations is performed using FSIN, a detailed network simulator developed in the UPV/EHU [17]. It can manage 1D, 2D and 3D routers. The router model is similar to the one implemented in the IBM BlueGene/L. Switching strategy is Virtual Cut-through [12]. Packet routing is fully adaptive and deadlock is avoided by means of Bubble flow control [16]. Packets are always 16 phits long (A phit is the amount of data bits that can be transmitted in parallel through a network link). The inter-injection interval at each node is random, chosen in such a way that injection can be modulated in terms of phits/cycle/node.

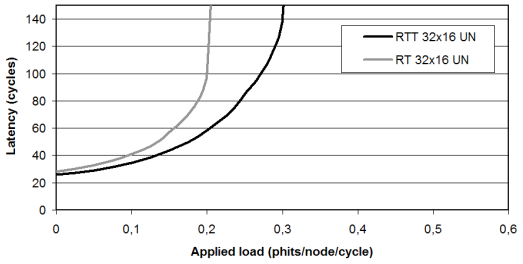
Performance data is shown in accepted load vs. provided load (throughput). Provided load is an input parameter, measured in phits/cycle/node. Accepted load is a measured value that indicates how many phits/cycle/node the network has been able to consume successfully. When the network is not saturated, both values are almost identical. However, when some of the routers (or all of them) saturate, actual throughput may be much smaller than applied load. In fact, under uniform traffic the theoretical limit of throughput is fixed by the network bisection bandwidth [11]. For a Torus with bidirectional links, this limit is $8/n$, where n is the number of nodes in the longest dimension. We also provide latency data in the form of plots of average packet delay vs. provided load. We measure the number of cycles between the instant a packet is injected (stored in the input buffer of the source router) until it is consumed at its destination node.

We have evaluated the performance of both RT and RTT with 128 (16×8), 512 (32×16) and 2048 (64×32) nodes. With 3D topologies, we have made a comparison of PT, PTT and PDTT of 1024 ($16 \times 8 \times 8$), 8192 ($32 \times 16 \times 16$) and 65536 ($64 \times 32 \times 32$) nodes.

The first set of experiments compare the results of 2D



(a) Throughput



(b) Latency

Figure 5. Throughput and delay for 32×16 RT and RTT under uniform traffic (UN).

networks, RT and RTT. We just show the case of the 32×16 network as all the remaining figures are equivalent. Figure 5(a) shows throughput for these topologies under uniform traffic. RTT arrives at saturation with a higher applied load and, therefore, the accepted load is significantly higher. As we introduced in Section 3, there are two factors behind this performance increase. On the one hand, the balance in the use of X and Y channels can contribute on a factor of 1.33. Figure 5 shows the link utilization for both RT and RTT on saturation. The bars on the left clearly show the limited use of Y channels on RT, while the bars on the right show their balanced use on RTT. On the other hand, the average distance is reduced in a factor of 1.11. This directly affects the base latency, which can be observed in Figure 5(b). Overall, there is a theoretical factor of $1.111 \times 1.333 = 1.481$ improvement in throughput, which is reflected in the Figure 5(a). This plot presents a throughput increase of 57% in saturation. Note that the result is slightly over the expected value because RT presents a throughput drop after reaching saturation. The speedup over the highest measured value is 46.4%.

However, this improvement under uniform traffic would be useless if the network performed poorly under other traffic patterns. Figures 8 show the network throughput for different random (hot-region [6]) and some permutation patterns (bit-complement, bit-reversal and perfect shuffle [11]). It can be observed that in all cases the RTT provides a significative improvement with respect to RT, ranging from

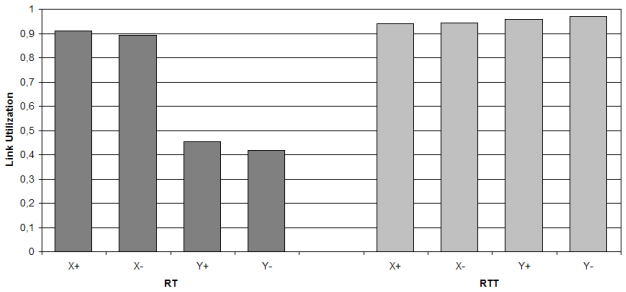


Figure 6. Link utilization for 32×16 RT and RTT in saturation.

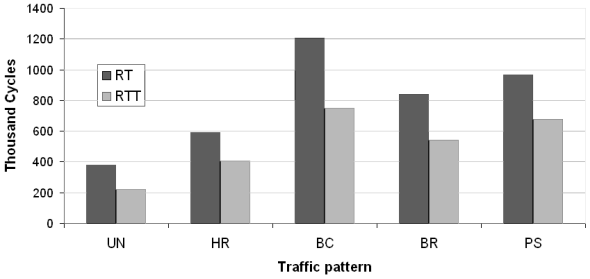


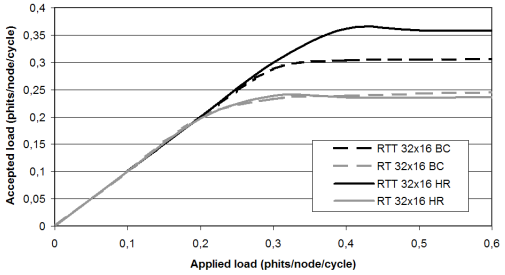
Figure 7. Time to complete 5 shots in an optimal RTT of 32×16 nodes

24.3% (Bit-complement) to 51.4% (Hot-Region).

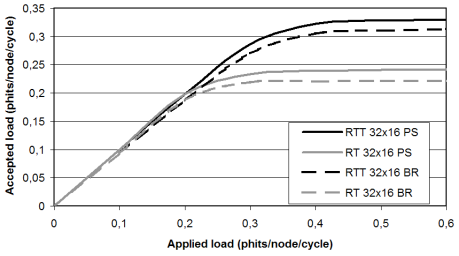
Next, we have estimated the reduction in execution time of programs running on these topologies using the synthetic traffic patterns previously introduced. This differs from latency data in that it analyzes the time to finish 5 "shots" of an application. Obviously, shorter times imply a higher network performance. The results are shown in Figure 5. We have obtained an average improvement of 29.8%, which demonstrates the great interest of this topology in front of the Rectangular Torus.

In the case of the 3D network, we have followed the same set of experiments with networks sized $64 \times 32 \times 32$. Note that these values correspond to the size of the Torus network of the largest configuration of the BlueGene [7]. Figure 9 compares PT, PTT and PDTT in terms of throughput and latency under uniform traffic. The figures for other sizes are equivalent.

The theoretical analysis of these topologies predicted an increase in throughput in a factor of $1.0833 \times 1.25 = 1.354$ in PTT and $1.143 \times 1.50 = 1.714$ in PDTT, both due to the reduction in average distance and the more balanced use of resources. Figure 9(a) shows that the maximum throughput is reached with higher applied load in PTT than in PT, and even higher in PDTT. The increases for the maximum injected load are respectively 33.3% and 59.8% compared with PT. However, note that these values are measured under heavy saturation, and the network performance decays after reaching maximum throughput, specially in networks

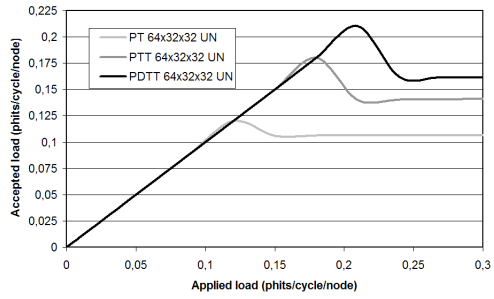


(a) Bit-Complement (BC) and Hot-Region (HR)

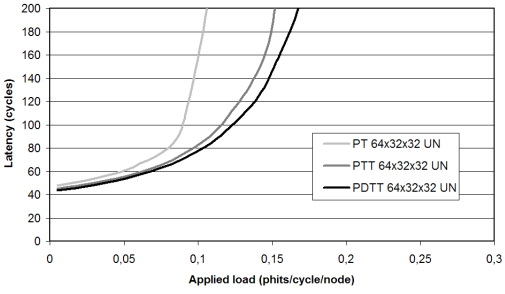


(b) Bit-Reversal (BR) and Perfect Shuffle (PS)

Figure 8. Throughput for different traffic patterns for 32×16 RT and RTT.



(a) Throughput



(b) Latency

Figure 9. Throughput and Delay, $64 \times 32 \times 32$ PT, PTT and PDTT, uniform traffic.

with large rings [15]. The increase of the values measured on the highest point on each curve are 49.9% and 74.9%. We must also consider that many times these large 3D ma-

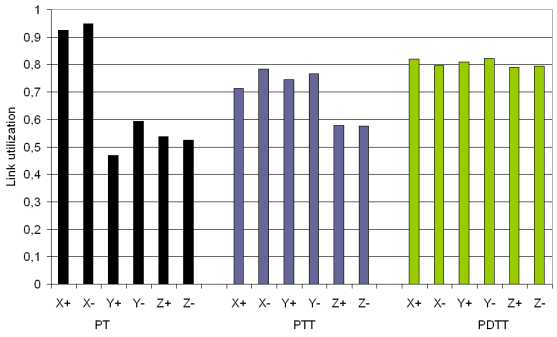


Figure 10. Link utilization for $64 \times 32 \times 32$ PT, PTT and PDTT in saturation.

chines are partitioned for different jobs. Though in these cases the improvement would not be so large, performance would still increase. For example, it is clear that considering the different $2a \times a$ planes as the different partitions leads to an increase equal to that of RTT over RT.

Figure 10 shows the channel use under uniform traffic for these topologies. As in the 2D case, PT presents a bottleneck in X, as packets have to travel twice as many hops as in the other dimensions. In PTT the utilization of Y is similar to X, but Z still remains under-used. Finally, PDTT presents a balanced utilization of all dimensions, which explains the performance increase.

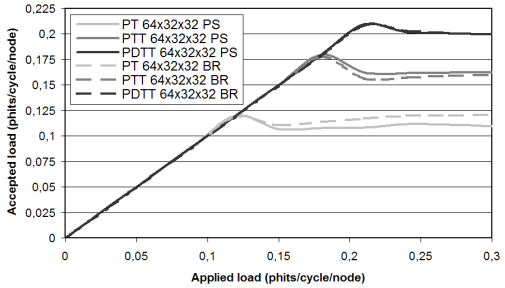
At last, Figure 11 shows throughput data for the same traffic permutations as in the 2D case. PTT always outperforms PT, with a speedup from 37.1% (in bit-reversal) to 53.2% (in perfect shuffle). PDTT improvements over PT are higher, from 59.7% (in bit-reversal) to 88.7% (in perfect shuffle).

6. Wiring RTT, PTT and PDTT Networks

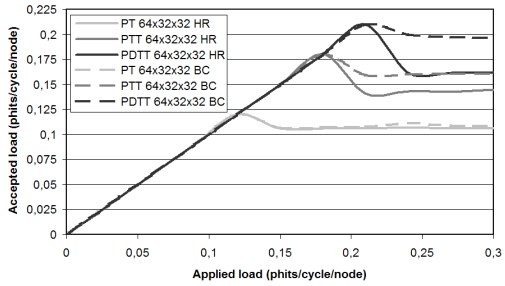
The notable performance gains exhibited by twisted rectangular and prismatic Tori compared with their standard Tori comes from a rearranging of the peripheral links. Hence, we have to consider the consequences derived from this rearrangement on the layout of the resulting network.

6.1. Bounded Link Length Layout for 2D RTT

In square Tori, the length of the wraparound links grows with the network size. While internal links are supposed to have unitary length, wraparound links grow as $\sqrt{N}$. As a consequence, real implementations could be negatively affected by this unbalance. The folded Torus is a good solution to equalize all the network links by increasing the individual wire length to 2 [11]. This idea is based on ap-



(a) Perfect Shuffle (PS) & Bit-Reversal (BR)



(b) Bit-Complement (BC) & Hot-Region (HR)

Figure 11. Throughput for different traffic patterns, $64 \times 32 \times 32$ networks.

plying certain shuffle transformations to rows and columns that interleaves the nodes. Two different shuffle transformations can be considered. Given a row (or column) of n nodes $(0, \ldots, n-1)$, the following transformations map every node location (x, y) onto a different one (x', y) on the same row (or column) (only row shuffle is shown):

- Shuffle A:

$$x' = 2x \quad \text{if } x < n/2.$$
$$x' = 2n - 2x - 1 \quad \text{if } x \geq n/2.$$

- Shuffle B:

$$x' = 2x + 1 \quad \text{if } x < n/2.$$
$$x' = 2n - 2x - 2 \quad \text{if } x \geq n/2.$$

Analogously to the folded Torus, we propose a new folding for RTT, denoted as Trellis Folded RTT, that generates a layout in which link lengths are equalized and bounded by $\sqrt{5}$. The whole process is detailed in Algorithm 2. Figure 12 shows a Trellis Folded layout of an 8×4 RTT. Note that, as in the Folded Torus, two planes are enough to lay all links without cutting each other.

This technique can be also applied in "block mode". As an example, consider a 32×16 RTT where each 4×4 submesh is a single block. This network can be represented as the basic 8×4 RTT in Figure 1(b), but each node becomes in a 4×4 sub-mesh and each link becomes in a group of 4 parallel links joining two such sub-meshes. After applying Algorithm 2 to the basic 8×4 network, the 32×16 block folded layout is obtained by simply substituting each node

Data: a
Step 1, Initial layout: Arrange the $2a^2$ nodes in a rows $(0, \ldots, a-1)$ and $2a$ columns $(0, \ldots, 2a-1)$.
Step 2, Row rotation: For each row i, rotate the row i positions to the right.
Step 3, Column rotation: For each column i, rotate the column $\left\lfloor \frac{i+1}{2} \right\rfloor$ positions down.
Step 4, Column shuffle: Apply an A shuffle to even rows and a B shuffle to odd ones.
Step 5, Row shuffle: Shuffle all rows according to shuffle A.

Algorithm 2: Trellis Folding Mapping Algorithm.

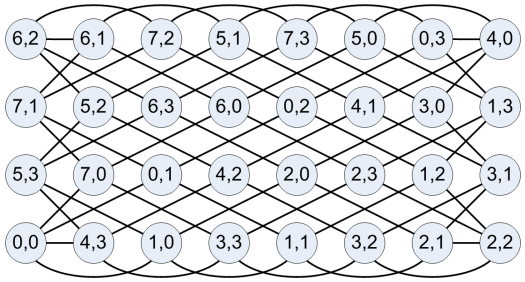


Figure 12. 8×4 **Trellis Folded RTT.**

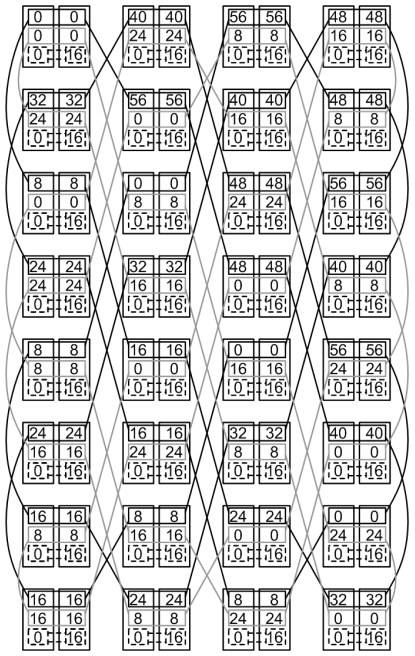
by the 4×4 sub-mesh, and each link by the four corresponding parallel links. In general, this method can be applied to any block size.

6.2. Layouts and Cabinet Distributions for 3D PTT and PDTT

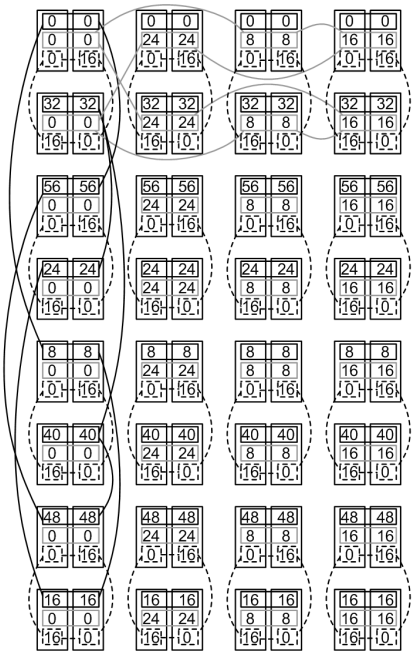
3D networks for large parallel systems are usually distributed among several cabinets. We take, as an example, the system layout of the larger configuration of the Bluegene/L ($64 \times 32 \times 32$ nodes) [7]. The system is organized as 8×8 cabinets of 1024 nodes each. Dimension Z evolves inside every pair of cabinets as every cabinet has an $8 \times 8 \times 16$ node configuration. Cabinets have been connected in pairs so that every pair shares X and Y dimension while Z grows inside it. These pairs of cabinets are laid on a rectangular array, with a standard folding applied on rows and columns to equalize cable lengths.

Now we deal with a folded layout for PTT. Considering that a close group of cabinets comprises the whole Z dimension for the nodes (x, y) within it, we can obviate such a dimension. Consequently, the folding problem is reduced to the previous bi-dimensional RTT case. Considering the network and cabinet sizes stated above and 8×8 blocks, each pair of cabinets would correspond with a node in a regular 8×4 RTT. Thus, the Trellis Folding can be directly applied. Figure 13(a) shows the resulting layout. Labels on cabinets mean x, y, z coordinates of each first node in the block. Note that only links between cabinets have to be modified, preserving the internal cabinet connectivity.

A PDTT is built by twisting both Y and Z dimensions.



(a) Trellis Folded PTT cabinet distribution.



(b) Folded PDTT cabinet distribution.

Figure 13. Cabinet Distribution, 3D Networks.

In this case, the dimension Z cannot be comprised inside a group of cabinets that share the same (x, y) nodes. Instead, dimension Z is spread between two groups of cabinets corresponding to different (x, y) nodes. Continuing with the previous example, Z should connect two pairs of cabinets. In this case, we cannot apply the previous Trellis Folding, as it places these pairs of cabinets in opposite locations. How-

ever, another folding technique based in [13] can be applied, which roughly corresponds to applying a shuffle once on Y and twice on X. The resulting layout leaves such pairs of cabinets together, as shown in Figure 13(b) (most links are omitted for the sake of clarity), but increases the maximum link length to 4. Note that, as before, only links between cabinets have to be reconnected.

7. Conclusions

There are several commercial parallel computers that use rectangular or prismatic Tori as topologies for their interconnection networks. We have evidenced severe communication bottlenecks in such networks that negatively affect their performance. Basically, performance degradation is provoked by the asymmetry induced in a network that has dimensions of different size.

In this paper, we have proposed and analyzed Twisted Torus networks that remove these bottlenecks by equalizing the length of the paths traversed by packets on each dimension. We have evaluated the performance of Twisted Tori both theoretically and by means of simulation. The node model employed in our experiments incorporates all the architectural features of current packet routers and resembles the one employed in the BlueGene/L. Simulation results obtained for different network sizes report throughput gains between 33% and 88% for uniform traffic. The imputable added costs of twisted networks come from both their packet routing mechanisms and their wireability. We propose in this research scalable and practicable solutions for these important architectural issues. As a conclusion, the proposed topologies appear as a clear option to improve the overall network performance by just rearranging the peripheral links of the network.

References

[1] N. R. Adiga et al. "An overview of the BlueGene/L Supercomputer". Supercomputing 2002 Technical Papers, November 2002.

[2] A. Agarwal. "Limits on interconnection network performance". IEEE Transactions on Parallel and Distributed Systems, 2(4):398–412, October 1991.

[3] G. Almasi et al. "Early experience with scientific applications on the BlueGene/L Supercomputer". Lecture Notes in Computer Science, Vol. 3648, pp. 560-570, 2005.

[4] B. W. Barrett, J. M. Squyres and A. Lumsdaine. "Implementation of Open MPI on Red Storm". Technical Report. Los Alamos National Laboratory, 2005.

[5] R. Beivide, E. Herrada, J. L. Balcazar and J. Labarta. "Optimized Mesh-Connected Networks for SIMD and MIMD Architectures". 14^{th} Annual International Symposium on Computer Architecture, pp. 163-169, 1987.

[6] M. Blumrich et al. "Design and Analysis of the BlueGene/L Torus Interconnection Network" IBM Research Report RC23025 (W0312-022) December 2003.

[7] P. Coteus *et al*."Packaging the Blue Gene/L supercomputer". IBM Journal of Research and Development, Vol. 49, Number 2/3, Page 213, 2005.

[8] Cray XT3 Datasheet. Available at http://www.cray.com/downloads/Cray _XT3_Datasheet.pdf.

[9] Cray X1E Datasheet. Available at http://www.cray.com/downloads/X1E_datasheet.pdf.

[10] Z. Cvetanovic."Performance Analysis of the Alpha 21364-based HP GS1280 Multiprocessor. Proceedings of the 30th Annual International Symposium on Computer Architecture. pp. 218-228, 2003.

[11] W. J. Dally and B. Towles."Principles and Practices of Interconnection Networks. Morgan Kaufmann, 2004.

[12] P. Kermani and L. Kleinrock, "Virtual cut-through: A new computer communication switching technique", Comput. Networks, Vol. 3, pp. 267-286, 1979.

[13] Francis C. M. Lau, Guihai Chen, "Optimal Layouts of Midimew Networks. IEEE Transactions on Parallel and Distributed Systems, Vol. 7, No 9, pp 954-961.

[14] C. Martínez, M. Moretó, R. Beivide and E. Gabidulin. "Perfect Dominating Sets in Gaussian Integers". Submitted for publication to Integers - Electronic Journal of Combinatorial Number Theory. Also available at http://www.atc.unican.es/ enrique/integers.pdf.

[15] J. Miguel-Alonso, J. A. Gregorio, V. Puente, F. Vallejo and R. Beivide. "Load Unbalance in k-ary n-cube Networks". Lecture Notes in Computer Science, Vol. 3149 (Proc. Euro-Par 2004), pp. 900-907, 2004.

[16] V. Puente, C. Izu, J.A. Gregorio, R. Beivide, and F. Vallejo, "The Adaptive Bubble router", Journal on Parallel and Distributed Computing, Vol. 61, no. 9, pp.1180-1208, 2001.

[17] F.J. Ridruejo, J. Miguel-Alonso. "INSEE: an Interconnection Network Simulation and Evaluation Environment". Lecture Notes in Computer Science, Vol. 3648 (Proc. Euro-Par 2005), pp 1014-1023, 2005.

[18] C. H. Sequin. "Doubly Twisted Torus Networks for VLSI processor arrays". 8^{th} Annual International Symposium on Computer Architecture, Minnesota, pp 471-480, 1981.

[19] S. L. Scott and G. M. Thorson. "The Cray T3E Network: Adaptive Routing in a High Performance 3D Torus". In Proceedings of HOT Interconnects IV, 1996.

[20] C. K. Wong and D. Coppersmith. "A Combinatorial Problem Related to Multimodule Memory Organizations". Journal of the ACM, Vol. 21, No. 3, pp. 392-402, 1974.

[21] Y. Yang et al. "Recursive Diagonal Torus: An Interconnection Network for Massively Parallel Computers". IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 7, 2001.