University of Castilla–La Mancha

A publication of

Department of Computing Systems

N-dimensional Twin Torus Networks*

by

F.J. Andújar, J.A. Villar, F.J. Alfaro, J.L. Sánchez, J. Duato

Technical Report #**DIAB-13-10-1** October 2013

(*) This work has been jointly supported by the MINECO and European Commission (FEDER funds) under the projects TIN2012-38341-C01 and TIN2012-38341-C04. Francisco J. Andujar is also funded by the Spanish Ministry of Science and Innovation MICINN under FPU grant AP2010-4680.

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS
ESCUELA SUPERIOR DE INGENIERÍA INFORMÁTICA
UNIVERSIDAD DE CASTILLA–LA MANCHA
CAMPUS UNIVERSITARIO s/n
02071, ALBACETE, SPAIN
Phone +34.967.599200, Fax +34.967.599224

N-dimensional Twin Torus Networks

Francisco J. Andújar, Juan A. Villar, Francisco J. Alfaro, José L. Sánchez
Computing Systems Department
Faculty of Computer Science Engineering
University of Castilla-La Mancha
02071 – Albacete, Spain
{fandujar, juanan, falfaro, jsanchez}@dsi.uclm.es

José Duato
Department of Systems Data Processing and Computers
Polytechnic University of Valencia
46022 – Valencia, Spain
jduato@gap.upv.es

Contents

Abstract

Torus topology is one of the most common topologies used in the largest supercomputers. It is chosen for its attractive properties related to cost, implementation or scalability. In the market, there are low–profile communication expansion cards that have a reduced number of ports that is not enough to build tori of a certain number of dimensions. For instance, by means of one four-port card per node, a 2D torus topology could be built, but not a 3D torus topology. However, two of these cards can be used per node to build a 3D torus topology [4, 5] (3DT torus). Although there are several ways of assigning the dimension and direction of the ports of the two cards for a 3DT torus in the previous reference, it has been proved which is the optimal port configuration. This paper extends and generalizes that previous work in order to obtain the optimal port configuration when n dimensions are considered. Thus, the nDT torus topology is defined and a detailed formal analysis leads to the optimal port configuration. Moreover, some comments about routing algorithm in nDT torus are included.

1 Introduction

The high number of nodes in large supercomputers imposes severe requirements on the interconnect system design, and therefore high–performance interconnection networks are mandatory in large supercomputers and clusters dominating the supercomputing market. In such systems, the network topology plays a major role in determining the overall system performance. There are many factors that may affect the choice of an appropriate network topology, but for this kind of systems, fat–tree [13] and torus [11] are usually the preferred topologies for indirect and direct networks, respectively.

In a fat–tree network every node has equal access bandwidth to every node. Therefore, this topology is very appropriate for running large scale applications which generate a lot of communication among all the nodes. However, the more a system grows, the more the fat–tree topology shows limits of cost, consumption, reliability and, very important, scalability. In contrast, the 3D torus topology provides a reduced hardware cost and an excellent scalability.

Torus topology belongs to the n–cube k–ary family that consists of n dimensions with k nodes in each dimension, with a total of k^n nodes. In particular, a 3D torus is a 3–cube k–ary topology. This topology has low radix and diameter, allowing an easy implementation and reducing the latency of the communications. In general, the cabling of the 3D torus network is simpler, allowing shorter cables, and if expansions are required, this topology can be added to with little re-cabling. Moreover, it is also important that the scalability cost is linear. Finally, torus supports several routing algorithms that increase path diversity so that the fault tolerance and load balance become feasible. Additionally, the topology maps very well several well–known traffic patterns generated by current scientific-purpose applications. In particular, torus topology is the best for applications with high locality level, such as applications that use 3D mathematical models.

The 3D torus topology is one of the most common topologies used in the largest supercomputers in the Top500 list [10]. For example, the Gemini system interconnect [3] employed in Titan [14] (actually, the second supercomputer at the Top500 list), the supercomputers of Cray's XT5 [8] family or the supercomputers of the Blue Gene/L [1] family. Some of these systems were also at the top in previous Top500 list.

To obtain a 3D torus, six ports (or links) per node are needed, two for each dimension. These six ports can be offered by a single switch or several low–profile expansion cards. Usually these low–profile expansion cards are incorporated in each node of a cluster. As today it is also usual that each node of the cluster is 1U (1.75 inches) tall, manufactures provide low–profile expansion cards with few communication ports. For instance, it is possible to build 2D torus with four–port cards (one per node), but not a 3D torus. However, two of these cards can be used per node to build a 3D torus topology.

As shown in Figure 1, one port of each card would be used to interconnect the two cards, and the remaining ports for inter–node communication. We called this topology 3D Twin torus or just 3DT torus. The 3DT torus topology offers a great advantage respect to the 2D torus topology: using the same 4–port card, the 3DT torus offers a lower distances between nodes. So, the network latency is reduced and the throughput is increased, only changing the topology and without extra economic investment.

There are significant challenges in 3DT torus design. Two of them stand out specially: port configuration and deadlock problem. The manner in which the ports of the two cards are assigned to the dimensions and their directions has a great influence on the communication performance. In order to reduce the latency, it is necessary to avoid as much as possible the paths that pass through the node using the two cards. If successful, the cost of the communication would be noticeably reduced. The six ports of the node are split in two groups, and every group is assigned to one of the cards, as shown in Figure 1. The ports of the 3DT torus topology have an assigned dimension (i.e., d_0, d_1, d_2) and direction (i.e., positive or negative), which have to be established in the network deployment. There are several ways of assigning the dimension and direction to the ports (port configurations), and each one has a different performance level. In [4], we presented a detailed study of the behavior of all the port configurations, and determined by means a formal analysis which is the best of them.

On the other hand, it is probable that *deadlock* appears in the 3DT torus network. This problem occurs because the link interconnecting the two cards can be used for any message in the network, independently of the dimension that the message is crossing. This causes new cycles in the 3DT torus network that do not appear in a 3D torus built directly with 6-port cards.

Recently, some networks used in large supercomputers build n-cube topologies with more than three dimension. For instance, supercomputers of IBM's Blue Gene/Q family [7] uses a five dimensional torus network, while the proprietary network Tofu [2] uses a six dimensional torus. Also, these networks are implemented in some of the supercomputers at the Top500 list, like Fujitsu's K-Computer [15] (Tofu) or Sequoia [12], Juqueen and Vulcan (IBM's Blue Gene/Q).

For this reason, we extend the work realized in [4, 5] in this paper, in order to define the nDT torus topology and to obtain the optimal port configuration, when n dimensions are considered instead of only three dimensions. We have focused our attention on the first challenge described above and have presented in this work a detailed formal analysis to obtain the optimal port configuration. Also, we have included a brief analysis about the deadlock in the nDT topology and how to avoid it.

The rest of the paper is organized as follows: In Section 2 the nDT torus topology is defined. Section 3 provides the formal analysis that leads to the optimal port configuration in Section 3.3. Finally, in Section 4, we presented a brief description of the routing algorithm implemented in the nDT torus.

2 The nDT torus topology

In this section we define the nDT torus topology. Previously, we introduce the notation to be used in the rest of the document, which is commonly used to treat with general torus topologies.

2.1 Notation

The following notation is used bellow:

- n: number of dimensions of the torus, where $n \geq 2$.
- k: number of nodes in every dimension of the nD torus. Note that the same number of nodes in each dimension is assumed and $k \geq 3$.
- $\langle o_0 o_1 \dots o_i \dots o_{n-2} o_{n-1} \rangle$: nD torus node identifier, $0 \leq i < n$ and $0 \leq o_i < k$.
- d_i: a dimension of the nD torus, $0 \leq i < n$.
- d_i^+, d_i^-: ports of the dimension d_i, corresponding to the two directions.
- $\mathcal{P}$: set of ports of a node, $\mathcal{P} = \{d_0^-, d_0^+, \dots, d_i^-, d_i^+, \dots, , d_{n-1}^-, d_{n-1}^+\}$.
- P: port of a node, $P \in \mathcal{P}$.
- $d_{[i,j]}$: a subset of ports of $\mathcal{P}$, including the ports from the dimension d_i to the dimension d_j, $0 \leq i, j < n$ and $i \leq j$. Note that if $j < i$, the subset would be an empty set.
- $PE0$, $PE1$: processing elements of a node.
- $\langle o_0 \dots o_{n-1} | pe \rangle$: processing element identifier in nDT torus, $0 \leq i < n$, $0 \leq o_i < k$, and pe is zero or one. Note that the left side of identifier $(o_0 \dots o_{n-1})$ is the torus node identifier, while the right side (pe) is the PE identifier.

- $N_s^P(\langle o_0 \dots o_{n-1} \rangle)$: set of nodes that send messages to node $\langle o_0 \dots o_{n-1} \rangle$ and reach it through the port P.

- $N_d^P(\langle o_0 \dots o_{n-1} \rangle)$: set of nodes to which the node $\langle o_0 \dots o_{n-1} \rangle$ sends messages from its port P.

- $D_s^P(\langle o_0 \dots o_{n-1} \rangle)$: cardinal of the set $N_s^P(\langle o_0 \dots o_{n-1} \rangle)$.

- $D_d^P(\langle o_0 \dots o_{n-1} \rangle)$: cardinal of the set $N_d^P(\langle o_0 \dots o_{n-1} \rangle)$.

- $R_{P \to P'}(\langle o_0 \dots o_{n-1} \rangle)$: number of paths that pass through the node $\langle o_0 \dots o_{n-1} \rangle$ from its input port P to its output port P'. If both P and P' belong to the same dimension, sometimes we refer to the addition of $R_{P \to P'}(\langle o_0 \dots o_{n-1} \rangle)$ and $R_{P' \to P}(\langle o_0 \dots o_{n-1} \rangle)$ by an expression, using only the identifier of the dimension, hiding the sign of the direction, and substituting the double arrow symbol for a single arrow ($R_{d_i \leftrightarrow d_i}(\langle o_0 \dots o_{n-1} \rangle)$).

  Also, P or P' can be a subset of ports. For example, the number of paths that pass through the node $\langle o_0 \dots o_{n-1} \rangle$ entering by ports of the dimensions d_0, d_1 and d_2 and leaving to output port d_{n-1}^+ is referred as $R_{d_{[0,2]} \to d_{n-1}^+}(\langle o_0 \dots o_{n-1} \rangle)$.

- $PATH(i)$: number of paths that pass through the node $\langle o_0 \dots o_{n-1} \rangle$ between any two ports P and P', where $P \in d_j$, $P' \in d_k$, $k = j + i$ and $0 \leq i, j, k < n$.

- $[a, b]^n$: with $b = a + m$; $a, b \in \mathbb{Z}$ and $n, m \in \mathbb{N}$, defines a set of integers whose members are $\{(a) \ mod \ n, (a+1) \ mod \ n, \dots, (a+m-1) \ mod \ n, (a+m) \ mod \ n\}$, where the operation module, $a \ mod \ k$, $a \in \mathbb{Z}$ and $k > 0$, is the remainder of integer division: $a \ mod \ k = (a + k) \ mod \ k$. It is easy to prove that the cardinal of the set that is defined by the interval $[a, b]^n$ is $b - a + 1$.

2.2 nDT torus definition

In this section we formally define the nD Twin torus topology.

Definition 2.1 *A nD Twin torus, or just nDT torus, is a n–cube k–ary (nD torus) topology, with $k \in \mathbb{N}^*$, $k \geq 2$ and $n \geq 3$, where each node is a virtual node*[1] *consisting basically of the following main components:*

- *Hardware for communications: it consists of two $(n+1)$–port cards*[2]*, offering a total of $2n+2$ ports. Two of these ports (one of each card) are used to interconnect both cards to each other, and the $2n$ remaining ports are used to connect the node to the rest of dimensions, building a torus topology with n dimensions.*

[1] In this point, we use this term for better explaining how a node is formed in this topology. However, in most of the paper we will use the term *node* to refer it.

[2] When only one $(n+1)$–port card is used per node, a n_0D torus ($n_0 = \frac{n+1}{2}$) is obtained

- *Computing hardware: each internal* $(n+1)$*–port card is connected to a processing element, and so there are two processing elements in each virtual node. Therefore, there are a total of* $2k^n$ *processing elements interconnected by the network.*

Example 2.1 *For building a* $3DT$ *torus* $(n = 3)$*, each virtual node consists of two* 4*–port cards and two processing elements. The* $3DT$ *torus is obtained using the available six ports after both cards are interconnected by means one port of each one (Figure 1).*

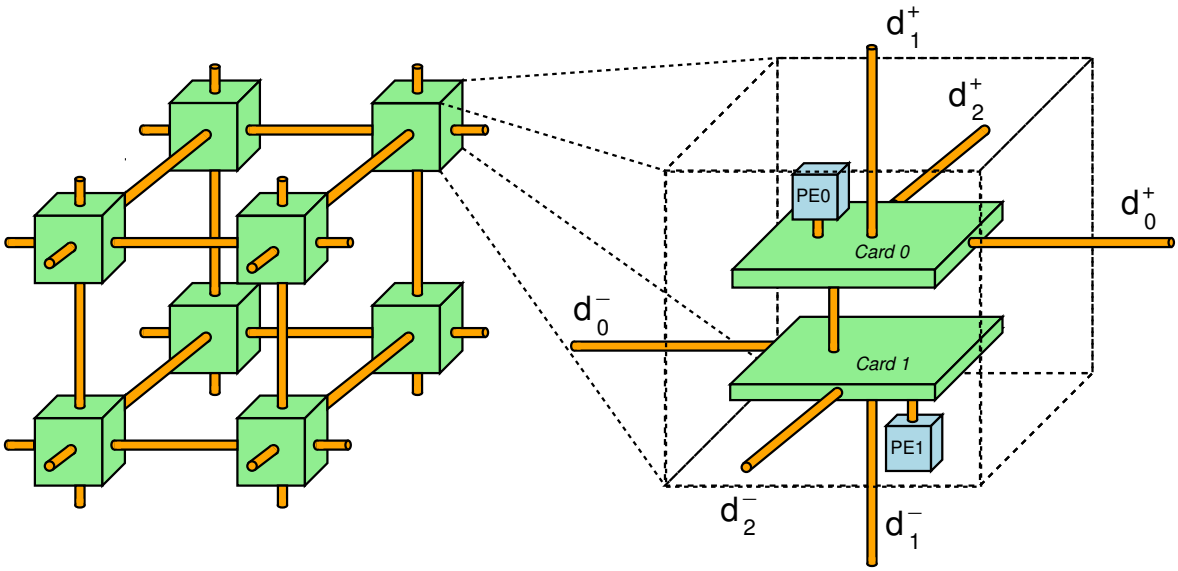


Figure 1: Fragment of a 3DT torus and detail of the communications hardware circuit, based on two 4-port cards.

3 Analysis of the port configuration of the nDT torus.

According to the nDT torus definition, there are multiple ways of assigning card ports to dimensions and therefore different port configurations can be formed. Note that the best configuration that minimizing the number of paths that go across a node using the two cards. Thus, the objective of this paper is to find the optimal configuration, where we consider as.

In order to determine the optimal configuration, for nDT torus, as was show in the previous study for the specific case of 3DT torus [4], it is necessary to consider the routing algorithm and the traffic pattern. In this study, we consider the routing algorithm DOR (**D**imension **O**rder **R**outing) [11] and an uniform traffic pattern. To simplify the analysis, we only consider odd values of k, because, given a node, there is the same number of nodes in both directions of a dimension. A similar study can be performed to even values of k. For the same reason explained in [4], the routing has been considered at node level, not a processing element level.

This study has been performed based on the methodology explained in [4], but it has been modified since the number of possible configurations for building a nDT torus is too high[3]. In this case, the methodology consists of the following steps:

1. To define the sets $N_s^P(\langle o_0 \dots o_{n-1}\rangle)$ and $N_d^P(\langle o_0 \dots o_{n-1}\rangle)$ and obtain their cardinals, $D_s^P(\langle o_0 \dots o_{n-1}\rangle)$ and $D_d^P(\langle o_0 \dots o_{n-1}\rangle)$, respectively, where $0 \leq o_i < k$, $0 \leq i < n$ and $P \in \mathcal{P}$ (Section 3.1).

2. To calculate the number of paths that pass through the node $\langle o_0 \dots o_{n-1}\rangle$ using the internal link[4] for each input pair of ports (Section 3.2).

3. To define the configuration we consider the optimal configuration for a nDT torus and to calculate the number of paths that cross the node using the internal link (Section 3.3).

4. Finally, to demonstrate that any other configuration is not better than the one previously defined (Section 3.5).

3.1 Sets N_s^P and N_d^P for the node $\langle o_0 \dots o_{n-1}\rangle$

Based on any n-cube k-ary and the DOR routing algorithm, it is easy to determine the nodes that belong to the sets N_s^P and N_d^P. Next, we indicate the members of these sets using set terminology. Since k is odd, the number of reachable nodes from a specific node that are located in the same dimension is $\frac{k-1}{2}$, regardless the direction.

Definition 3.1 *Let $N_s^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle)$ be a set of nodes that send messages to the node $\langle o_0 \dots o_{n-1}\rangle$ and reach it through the port d_i^- ($0 \leq o_i < k$, $0 \leq i < n$). Hence:*

$$N_s^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle) = \{\langle o'_0 \dots o'_{n-1}\rangle : \begin{cases} o'_j \in [0, k-1] & \text{if } j < i \\ o'_j \in \left[o_j - \frac{k-1}{2}, o_j - 1\right]^k & \text{if } j = i \\ o'_j = o_j & \text{if } j > i \end{cases} , \ 0 \leq j < n \ \}$$

Definition 3.2 *Let $N_s^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle)$ be a set of nodes that send messages to the node $\langle o_0 \dots o_{n-1}\rangle$ and reach it through the port d_i^+ ($0 \leq o_i < k$, $0 \leq i < n$). Hence:*

$$N_s^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle) = \{\langle o'_0 \dots o'_{n-1}\rangle : \begin{cases} o'_j \in [0, k-1] & \text{if } j < i \\ o'_j \in \left[o_j + 1, o_j + \frac{k-1}{2}\right]^k & \text{if } j = i \\ o'_j = o_j & \text{if } j > i \end{cases} , \ 0 \leq j < n \ \}$$

[3]In the $3DT$ torus case, there are only ten different port configurations and this fact allows us to analyze and compare all of them.

[4]Henceforth, we will use internal link to refer to the connection between the two cards in a node.

Example 3.1 *Given a 3D torus ($n = 3$), where $k = 5$, the set of nodes that send messages to the node $\langle 3,1,0\rangle$ and reach it through its port d_0^+ is:*

$$N_s^{d_0^+}(\langle 3,1,0\rangle) = \{\langle 4,1,0\rangle,\ \langle 0,1,0\rangle\}$$

Figure 2 shows graphically this subset of nodes.

Definition 3.3 *Let $N_d^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle)$ be a set of nodes to which the node $\langle o_0 \dots o_{n-1}\rangle$ sends messages from its port d_i^- ($0 \le o_i < k$, $0 \le i < n$). Hence:*

$$N_d^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle) = \{\langle o'_0 \dots o'_{n-1}\rangle : \begin{cases} o'_j = o_j & \text{if } j < i \\ o'_j \in \left[o_j - \frac{k-1}{2}, o_j - 1\right]^k & \text{if } j = i \\ o'_j \in [0, k-1] & \text{if } j > i \end{cases} ,\ 0 \le j < n\ \}$$

Definition 3.4 *Let $N_d^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle)$ be a set of nodes to which the node $\langle o_0 \dots o_{n-1}\rangle$ sends messages from its port d_i^+ ($0 \le o_i < k$, $0 \le i < n$). Hence:*

$$N_d^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle) = \{\langle o'_0 \dots o'_{n-1}\rangle : \begin{cases} o'_j = o_j & \text{if } j < i \\ o'_j \in \left[o_j + 1, o_j + \frac{k-1}{2}\right]^k & \text{if } j = i \\ o'_j \in [0, k-1] & \text{if } j > i \end{cases} ,\ 0 \le j < n\ \}$$

Example 3.2 *Given a 3D torus ($n = 3$), where $k = 5$, the set of nodes to which the node $\langle 3,1,0\rangle$ sends messages from its port d_1^- is:*

$$\begin{aligned} N_d^{d_1^-}(\langle 3,1,0\rangle) \ = \ & \{\langle 3,0,0\rangle,\ \langle 3,0,1\rangle,\ \langle 3,0,2\rangle,\ \langle 3,0,3\rangle,\ \langle 3,0,4\rangle, \\ & \langle 3,4,0\rangle,\ \langle 3,4,1\rangle,\ \langle 3,4,2\rangle,\ \langle 3,4,3\rangle,\ \langle 3,4,4\rangle\} \end{aligned}$$

Figure 2 shows graphically this subset of nodes.

3.1.1 D_s^P and D_d^P values for the node $\langle o_0 \dots o_{n-1}\rangle$

Applying the same criteria used in [4], we can calculate easily the values of D_s^P and D_d^P. Basically, given a port, if we only consider one dimension d_i, the number of source or destination nodes in this dimension is $\frac{k-1}{2}$. If we consider all dimensions, this value is multiplied by k raised to the number of dimensions that are routed before or after d_i by DOR routing, for D_s^P and D_d^P, respectively. That is:

$$D_s^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle) = D_s^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle) = \frac{k-1}{2}k^i \tag{1}$$

$$D_d^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle) = D_d^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle) = \frac{k-1}{2}k^{n-1-i} \tag{2}$$

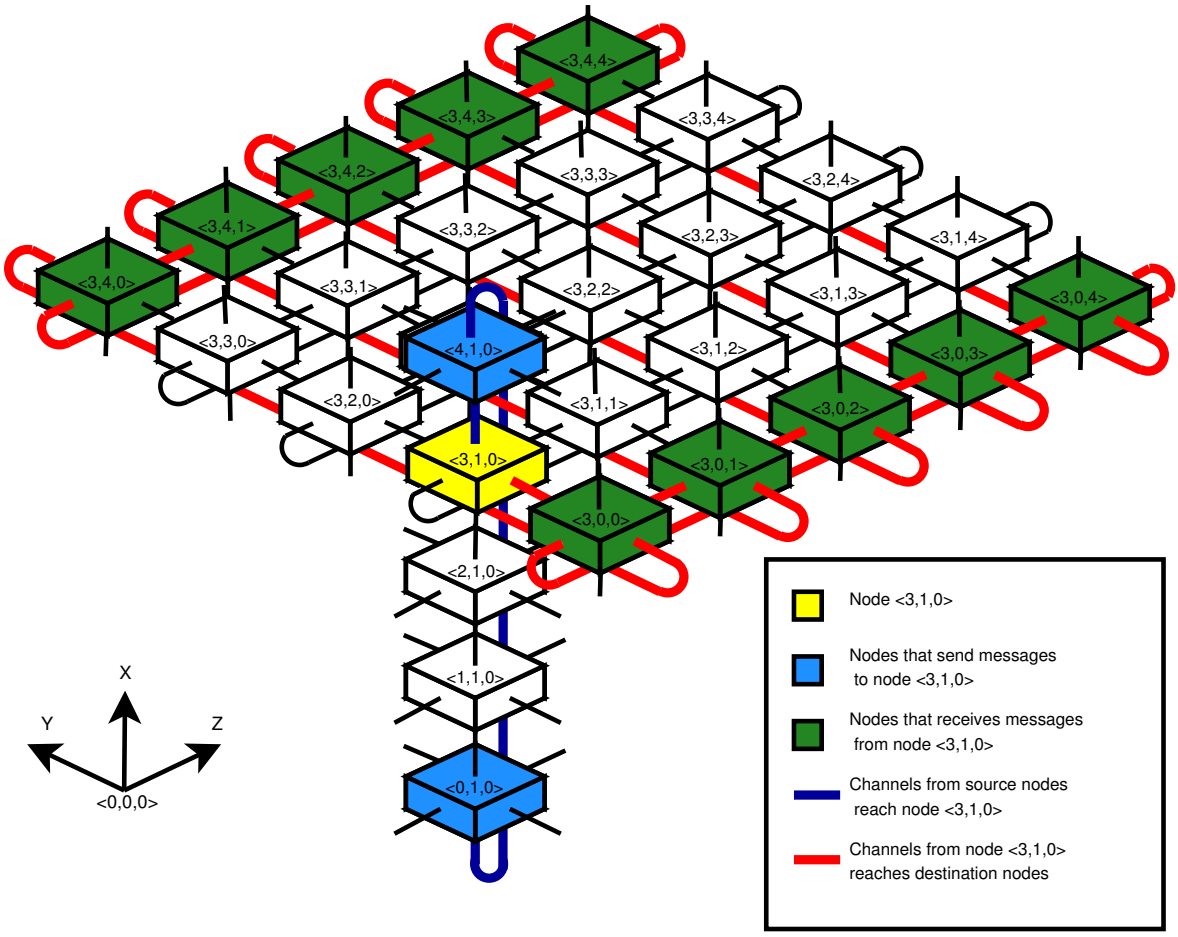


Figure 2: Nodes that send messages to the node $\langle 3,1,0\rangle$ and reach it through its port d_0^+ and nodes to which the node $\langle 3,1,0\rangle$ sends messages from its port d_1^-.

Sometimes, we will refer to the number of source or destination nodes in one dimension, regardless of the direction. In this case, we do not include the + or − sign in the superscript of D_s or D_d.

$$\begin{aligned} D_s^{d_i}(\langle o_0 \dots o_{n-1}\rangle) &= D_s^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle) + D_s^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle) = \\ &= 2\times\frac{k-1}{2}k^i = (k-1)k^i \qquad (3) \\ D_d^{d_i}(\langle o_0 \dots o_{n-1}\rangle) &= D_d^{d_i^-}(\langle o_0 \dots o_{n-1}\rangle) + D_d^{d_i^+}(\langle o_0 \dots o_{n-1}\rangle) = \\ &= 2\times\frac{k-1}{2}k^{n-1-i} = (k-1)k^{n-1-i} \qquad (4) \end{aligned}$$

In other cases we will need to calculate the number of source or destination nodes for a subset of ports $d_{[0,i]}$.

Given a subset of ports $d_{[0,i]}$, where $0 < i < n$, the subset of nodes that send messages to the node $\langle o_0 \dots o_{n-1}\rangle$ from this subset of ports, $N_s^{d_{[0,i]}}(\langle o_0 \dots o_{n-1}\rangle)$, is:

$$\begin{aligned} N_s^{d_{[0,i]}}(\langle o_0 \dots o_{n-1}\rangle) &= N_s^{d_0}(\langle o_0 \dots o_{n-1}\rangle) \cup N_s^{d_1}(\langle o_0 \dots o_{n-1}\rangle) \cup \dots \\ &\cup N_s^{d_{i-1}}(\langle o_0 \dots o_{n-1}\rangle) \cup N_s^{d_i}(\langle o_0 \dots o_{n-1}\rangle) \end{aligned}$$

Then, the number of nodes that send messages to the node $\langle o_0 \dots o_{n-1} \rangle$ from the subset of ports $d_{[0,i]}$ is:

$$\begin{aligned} D_s^{d_{[0,i]}}(\langle o_0 \dots o_{n-1} \rangle) &= D_s^{d_0}(\langle o_0 \dots o_{n-1} \rangle) + D_s^{d_1}(\langle o_0 \dots o_{n-1} \rangle) + \dots \\ &+ D_s^{d_{i-1}}(\langle o_0 \dots o_{n-1} \rangle) + D_s^{d_i}(\langle o_0 \dots o_{n-1} \rangle) = \\ &= (k-1)k^0 + (k-1)k^1 + \dots + (k-1)k^{i-1} + (k-1)k^i = \\ &= (k-1)\sum_{j=0}^{i} k^j = \left(\sum_{j=0}^{i} k^{j+1} - \sum_{j=0}^{i} k^j \right) = k^{i+1} - 1 \end{aligned} \tag{5}$$

In a similar way, we can calculate the number of nodes to which the node $\langle o_0 \dots o_{n-1} \rangle$ sends messages from a subset of ports $d_{[i,n-1]}$ where $0 < i < n$, i.e. the cardinal of $N_d^{d_{[i,n-1]}}(\langle o_0 \dots o_{n-1} \rangle)$:

$$\begin{aligned} D_d^{d_{[i,n-1]}}(\langle o_0 \dots o_{n-1} \rangle) &= D_d^{d_i}(\langle o_0 \dots o_{n-1} \rangle) + D_d^{d_{i+1}}(\langle o_0 \dots o_{n-1} \rangle) + \dots \\ &+ D_d^{d_{n-1}}(\langle o_0 \dots o_{n-1} \rangle) = k^{n-i} - 1 \end{aligned} \tag{6}$$

For a subset of ports $d_{[i,j]}$, where $0 < i, j < n$ and $i < j$, the subset of nodes that send messages to the node $\langle o_0 \dots o_{n-1} \rangle$ from this subset of ports, $N_s^{d_{[i,j]}}(\langle o_0 \dots o_{n-1} \rangle)$, is:

$$N_s^{d_{[i,j]}}(\langle o_0 \dots o_{n-1} \rangle) = N_s^{d_{[0,j]}}(\langle o_0 \dots o_{n-1} \rangle) - N_s^{d_{[0,i-1]}}(\langle o_0 \dots o_{n-1} \rangle)$$

Then, the number of nodes that send messages to the node $\langle o_0 \dots o_{n-1} \rangle$ from the subset of ports $d_{[i,j]}$ is:

$$\begin{aligned} D_s^{d_{[i,j]}}(\langle o_0 \dots o_{n-1} \rangle) &= D_s^{d_{[0,j]}}(\langle o_0 \dots o_{n-1} \rangle) - D_s^{d_{[0,i-1]}}(\langle o_0 \dots o_{n-1} \rangle) \\ &= (k^{j+1} - 1) - (k^{(i-1)+1} - 1) = k^{j+1} - k^i \end{aligned} \tag{7}$$

On the other hand, the subset of nodes to which the node $\langle o_0 \dots o_{n-1} \rangle$ sends messages from the subset of ports $d_{[i,j]}$ is:

$$N_d^{d_{[i,j]}}(\langle o_0 \dots o_{n-1} \rangle) = N_d^{d_{[i,n-1]}}(\langle o_0 \dots o_{n-1} \rangle) - N_d^{d_{[j+1,n-1]}}(\langle o_0 \dots o_{n-1} \rangle)$$

Then, the number of nodes to which the node $\langle o_0 \dots o_{n-1} \rangle$ sends messages from this subset of ports is:

$$\begin{aligned} D_d^{d_{[i,j]}}(\langle o_0 \dots o_{n-1} \rangle) &= D_d^{d_{[i,n-1]}}(\langle o_0 \dots o_{n-1} \rangle) - D_d^{d_{[j+1,n-1]}}(\langle o_0 \dots o_{n-1} \rangle) \\ &= (k^{n-i} - 1) - (k^{n-(j+1)} - 1) = k^{n-i} - k^{n-(j+1)} \end{aligned} \tag{8}$$

3.2 Paths that pass through the node $\langle o_0 \dots o_{n-1} \rangle$

In this section we calculate the number of paths that pass through the node $\langle o_0 \dots o_{n-1} \rangle$ for each input–output pair of ports.

Proposition 3.1 *Given a node $\langle o_0 \dots o_{n-1} \rangle$, the paths that cross this node from its input port P to its output port P', where $P \in d_i$ and $P' \in d_j$, $0 \leq i, j < n$ are:*

$$R_{P \in d_i \rightarrow P' \in d_j}(\langle o_0 \dots o_{n-1} \rangle) = \begin{cases} \frac{(k-1)^2}{4} k^{n-1+i-j} & \text{if } i < j \\ \frac{(k-1)(k-3)}{8} k^{n-1} & \text{if } i = j \\ 0 & \text{if } i > j \end{cases}$$

Proof: If $i < j$, the routing algorithm DOR permits $P \rightarrow P'$ transitions. Then, the source node of the path could be any member of the set $N_s^P(\langle o_0 \dots o_{n-1} \rangle)$, and the destination node of the path could be any member of the set $N_d^{P'}(\langle o_0 \dots o_{n-1} \rangle)$. Therefore, the total number of paths that go across the node $\langle o_0 \dots o_{n-1} \rangle$ from its input port P to its output port P' is obtained by the following product:

$$R_{P \in d_i \rightarrow P' \in d_j}(\langle o_0 \dots o_{n-1} \rangle) = D_s^{P \in d_i} \times D_d^{P' \in d_j} = \frac{k-1}{2} k^i \times \frac{k-1}{2} k^{n-1-j} = \frac{(k-1)^2}{4} k^{n-1+i-j}$$

On the other hand, for the case $i = j$, in [4] we show that the number of paths that cross a node $\langle o_0 \dots o_{n-1} \rangle$ from P to P', if both ports belong to the same dimension d_i and we only consider the source or destination nodes that can be reached using d_i, and k is odd, is:

$$R_{P \in d_i \rightarrow P' \in d_j}(\langle o_0 \dots o_{n-1} \rangle) = \sum_{i=1}^{\frac{k-1}{2}-1} i = \frac{(k-1)(k-3)}{8}$$

If we consider the remaining $n-1$ dimensions, the number of paths that cross the node $\langle o_0 \dots o_{n-1} \rangle$ from $P \in d_i$ to $P' \in d_j$, when $i = j$, is:

$$R_{P \in d_i \rightarrow P' \in d_j}(\langle o_0 \dots o_{n-1} \rangle) = \frac{(k-1)(k-3)}{8} k^{n-1}$$

Remember that we are assuming that k is odd and $k \geq 3$ and therefore $R_{P \in d_i \rightarrow P' \in d_j}(\langle o_0 \dots o_{n-1} \rangle)$ cannot be less than zero. Note that if $k = 3$, $R_{P \in d_i \rightarrow P' \in d_j}(\langle o_0 \dots o_{n-1} \rangle) = 0$ and there are not paths between P and P' crossing the node $\langle o_0 \dots o_{n-1} \rangle$.

Finally, if $i > j$, the routing algorithm does not permit $P \rightarrow P'$ transitions and there are not paths that cross the node from P to P'. □

Sometimes, we need to calculate the number of paths that pass through the node between the two ports of the same dimension.

Proposition 3.2 *Given a node $\langle o_0 \dots o_{n-1} \rangle$, the number of paths that cross this node from the input port P to the output port P', where $P \neq P'$ and they both belong to the same dimension, that is, $P, P' \in d_i$ are:*

$$R_{d_i \leftrightarrow d_i}(\langle o_0 \dots o_{n-1} \rangle) = \frac{(k-1)(k-3)}{4} k^{n-1}$$

Proof: The demonstration is trivial in this case. From Proposition 3.1, we obtain:

$$\begin{aligned} R_{d_i \leftrightarrow d_i}(\langle o_0 \dots o_{n-1} \rangle) &= R_{d_i^+ \rightarrow d_i^-}(\langle o_0 \dots o_{n-1} \rangle) + R_{d_i^- \rightarrow d_i^+}(\langle o_0 \dots o_{n-1} \rangle) = \\ &= 2 \times \frac{(k-1)(k-3)}{8} k^{n-1} = \frac{(k-1)(k-3)}{4} k^{n-1} \end{aligned}$$

□

Proposition 3.3 *Given a node $\langle o_0 \dots o_{n-1} \rangle$, the number of paths that cross this node using the two ports P and P', where $P \in d_j$, $P' \in d_k$, $k = j + i$, $0 \leq j, k < n$ and $i \geq 0$, is:*

$$PATH(i) = \begin{cases} \frac{1}{4}(k-1)(k-3)k^{n-1} & \text{if } i = 0 \\ \frac{1}{4}(k-1)^2 \ k^{n-1-i} & \text{if } 0 < i < n \\ 0 & \text{if } i \geq n \end{cases}$$

Proof: If $i = 0$, $j = k$ and the two ports belong to the same dimension. The number of paths is obtained from Proposition 3.2:

$$PATH(0) = R_{d_j \leftrightarrow d_j}(\langle o_0 \dots o_{n-1} \rangle) = \frac{(k-1)(k-3)}{4} k^{n-1}$$

If $0 < i < n$, $PATH(i)$ is equal to $R_{P \in d_j \rightarrow P' \in d_k}(\langle o_0 \dots o_{n-1} \rangle)$. Then, from Proposition 3.1:

$$PATH(i) = \frac{(k-1)^2}{4} k^{n-1+j-k} = \frac{(k-1)^2}{4} k^{n-1+j-(j-i)} = \frac{(k-1)^2}{4} k^{n-1-i}$$

Finally, remember that $k = i + j$ and $0 \leq j < k < n$. Then, if $i \geq n$, either j or k is out of range and it is not possible that there are paths between d_j and d_k. In this case, $PATH(i) = 0$. □

3.3 Optimal port configuration in a nDT torus

As mentioned before, for a nDT torus, there are many ways to configure the ports of the two cards and therefore the number of possible configurations becomes higher and higher when the number of dimensions is increased. For example, in a $3DT$ torus (two 4–port cards) there are only 10 configurations, whereas in a $5DT$ torus (two 6–port cards) and a $7DT$ torus (two 8–port cards) there are 126 and 1716 configurations, respectively. Due to this, analyzing all configurations for finding the best is an unaffordable problem.

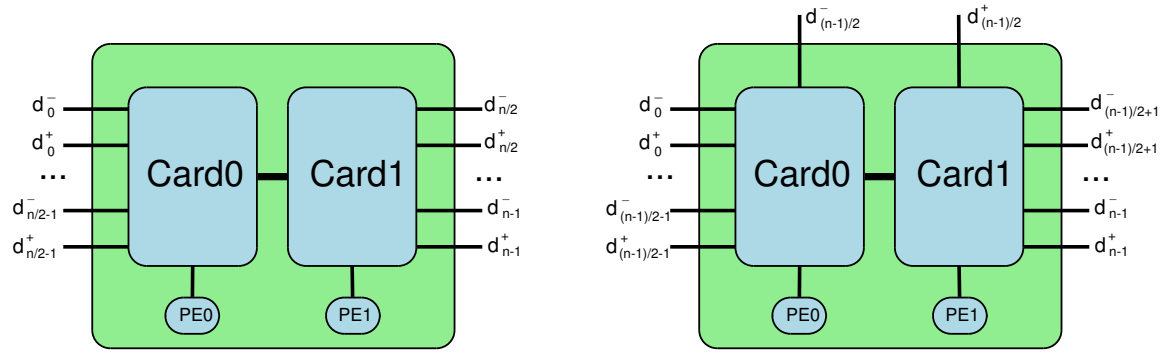
Hence, we use another method which consists in presenting the configuration we claim is the optimal and demonstrating that. This configuration is based on that

obtained for the particular case of the $3DT$ torus topology [4]. In that case, the X^+, X^- and Y^+ (or Y^- if k is odd) ports are connected to one card, and the ports Z^+, Z^- and Y^- (or Y^+ if k is odd) ports are connected to the other card. If we replace X, Y and Z with d_0, d_1 and d_2, respectively, we can see that the first ports, sorted by dimension, are connected to one card, whereas the last ports are connected to the other card. We can generalize this idea for a torus of any number of dimensions.

3.3.1 Defining the optimal configuration $\mathcal{C}_{Best}$

Definition 3.5 *Given two cards with $(n+1)$–ports each, the configuration $\mathcal{C}_{Best}$ allowing to build a nDT torus is defined as follows:*

- *If n is even:*
  - *The ports corresponding to the first $\frac{n}{2}$ dimensions, i.e., the ports belonging to dimensions from d_0 to $d_{\frac{n}{2}-1}$, are connected to Card0.*
  - *The ports corresponding to the last $\frac{n}{2}$ dimensions, i.e., the ports belonging to dimensions from $d_{\frac{n}{2}}$ to d_{n-1},are connected to Card1.*
- *If n is odd:*
  - *The ports corresponding to the first $\frac{n-1}{2}$ dimensions, i.e., the ports belonging to dimensions from d_0 to $d_{\frac{n-1}{2}-1}$, are connected to Card0.*
  - *The ports corresponding to the last $\frac{n-1}{2}$ dimensions, i.e., the ports belonging to dimensions from $d_{\frac{n-1}{2}+1}$ to d_{n-1}, are connected to Card1.*
  - *The two ports of the dimension $d_{\frac{n-1}{2}}$ are separated in the two cards. Since k is odd, the number of paths that cross the internal link is the same, independently in which card each port is connected. From now on, we assume that the port $d^-_{\frac{n-1}{2}}$ is connected to Card0 and the port $d^+_{\frac{n-1}{2}}$ is connected to Card1.*



(a) n is even. (b) n is odd.

Figure 3: Configuration $\mathcal{C}_{Best}$ for a nDT torus

Figure 3 shows graphically the configuration $\mathcal{C}_{Best}$. Note that the idea is to place the ports of the same dimension in the same card in order to avoid the crossing of the internal link connecting both cards. In the case when n is odd, the central dimension ($d_{\frac{n-1}{2}}$) is divided placing each one of the directions in a different card.

3.3.2 Calculating the usage of the internal link considering $\mathcal{C}_{Best}$

Given a node, using the configuration $\mathcal{C}_{Best}$, the internal link connecting both cards is used in the following cases:

- If n is odd:
  - When a path crosses the node using the dimension $d_{\frac{n-1}{2}}$.
- For all values of n:
  - When a path enters in the node using a dimension whose ports are connected to *Card0* and leaves the node using a dimension whose ports are connected to *Card1*.
  - When the source PE is connected to the node, but the source PE and the first dimensions crossed by the path are connected to different cards.
  - When the destination PE is connected to the node, but the destination PE and the last dimensions crossed by the path are connected to different cards.

Just as in [4], we do not consider paths whose source or destination is a PE of the node. If a port P is connected to *Card0*, the paths from or to $PE1$ cross the internal link, while if P is connected to *Card1* the paths from or to $PE0$ cross the internal link. In the two cases, the number of paths is the same. Then, the total number of paths that start or end in the node and use the internal link is constant, independently the configuration used. Moreover, the paths are considered at node level, not at PE level, since considering the paths at PE level only multiplies by two the number of reachable nodes from a port P.

Proposition 3.4 *The number of paths that cross a node $\langle o_0 \dots o_{n-1} \rangle$ using the internal link considering the configuration $\mathcal{C}_{Best}$ is:*

$$R_{\mathcal{C}_{Best}} = \begin{cases} (k^{\frac{n}{2}} - 1)^2 & \text{if } n \text{ is even} \\ (k^{\frac{n-1}{2}} - 1)(k^{\frac{n+1}{2}} - 1) + \frac{(k-1)(k-3)}{4} k^{n-1} & \text{if } n \text{ is odd} \end{cases}$$

Proof: We can distinguish the following cases:

- If n is even:
  - $d_{\left[0,\frac{n}{2}-1\right]} \rightarrow d_{\left[\frac{n}{2},n-1\right]}$: the input port belongs to the first $\frac{n}{2}$ dimensions (from d_0 to $d_{\frac{n}{2}-1}$ in *Card0*) and the output port belongs to the last $\frac{n}{2}$ dimensions (from $d_{\frac{n}{2}+1}$ to d_{n-1} in *Card1*). The number of paths that cross the internal link is:

$$R_{d_{[0,\frac{n}{2}-1]} \rightarrow d_{[\frac{n}{2},n-1]}}(\langle o_0 \dots o_{n-1} \rangle) = D_s^{d_{[0,\frac{n}{2}-1]}} \times D_d^{d_{[\frac{n}{2},n-1]}}$$

    Applying the expressions (5) and (6):

$$R_{d_{[0,\frac{n}{2}-1]} \rightarrow d_{[\frac{n}{2},n-1]}}(\langle o_0 \dots o_{n-1} \rangle) = (k^{\frac{n}{2}} - 1) \times (k^{\frac{n}{2}} - 1) = (k^{\frac{n}{2}} - 1)^2$$

- If n is odd:

  A) $d_{\left[0,\frac{n-1}{2}-1\right]} \rightarrow d^+_{\frac{n-1}{2}}$: the input port belongs to the first $\frac{n-1}{2}$ dimensions (from d_0 to $d_{\frac{n-1}{2}-1}$ in *Card0*) and the output port is $d^+_{\frac{n-1}{2}}$ (*Card1*). The number of paths that cross the internal link is:

$$R_{d_{\left[0,\frac{n-1}{2}-1\right]} \rightarrow d^+_{\frac{n-1}{2}}}(\langle o_0 \dots o_{n-1} \rangle) = D_s^{d_{\left[0,\frac{n-1}{2}-1\right]}} \times D_d^{d^+_{\frac{n-1}{2}}}$$

  Applying the expressions (5) and (2):

$$R_{d_{\left[0,\frac{n-1}{2}-1\right]} \rightarrow d^+_{\frac{n-1}{2}}}(\langle o_0 \dots o_{n-1} \rangle) = (k^{\frac{n-1}{2}} - 1) \times \frac{k-1}{2} k^{n-1-\frac{n-1}{2}} = \\ = (k^{\frac{n-1}{2}} - 1) \times \frac{k-1}{2} k^{\frac{n-1}{2}}$$

  B) $d_{\left[0,\frac{n-1}{2}-1\right]} \rightarrow d_{\left[\frac{n-1}{2}+1,n-1\right]}$: the input port belongs to the first $\frac{n-1}{2}$ dimensions (from d_0 to $d_{\frac{n-1}{2}-1}$ in *Card0*) and the output port belongs to the last $\frac{n-1}{2}$ dimensions (from $d_{\frac{n-1}{2}+1}$ to d_{n-1} in *Card1*). The number of paths that cross the internal link is:

$$R_{d_{\left[0,\frac{n-1}{2}-1\right]} \rightarrow d_{\left[\frac{n-1}{2}+1,n-1\right]}}(\langle o_0 \dots o_{n-1} \rangle) = D_s^{d_{\left[0,\frac{n-1}{2}-1\right]}} \times D_d^{d_{\left[\frac{n-1}{2}+1,n-1\right]}}$$

  Applying the expressions (5) and (6):

$$R_{d_{\left[0,\frac{n-1}{2}-1\right]} \rightarrow d_{\left[\frac{n-1}{2}+1,n-1\right]}}(\langle o_0 \dots o_{n-1} \rangle) = (k^{\frac{n-1}{2}} - 1) \times (k^{\frac{n-1}{2}} - 1) = \\ = (k^{\frac{n-1}{2}} - 1)^2$$

  C) $d^-_{\frac{n-1}{2}} \rightarrow d_{\left[\frac{n-1}{2}+1,n-1\right]}$: the input port is $d^-_{\frac{n-1}{2}}$ (*Card0*) and the output port belongs to the last $\frac{n-1}{2}$ dimensions (from $d_{\frac{n-1}{2}+1}$ to d_{n-1} in *Card1*). The number of paths that cross the internal link is:

$$R_{d^-_{\frac{n-1}{2}} \rightarrow d_{\left[\frac{n-1}{2}+1,n-1\right]}}(\langle o_0 \dots o_{n-1} \rangle) = D_s^{d^-_{\frac{n-1}{2}}} \times D_d^{d_{\left[\frac{n-1}{2}+1,n-1\right]}}$$

Applying the expressions (1) and (6):

$$R_{d^-_{\frac{n-1}{2}} \to d_{\left[\frac{n-1}{2}+1,n-1\right]}}(\langle o_0 \dots o_{n-1}\rangle) = \frac{k-1}{2}k^{\frac{n-1}{2}} \times (k^{\frac{n-1}{2}} - 1)$$

D) $d^-_{\frac{n-1}{2}} \leftrightarrow d^+_{\frac{n-1}{2}}$: input and output ports are $d^-_{\frac{n-1}{2}}$ (*Card0*) and $d^+_{\frac{n-1}{2}}$ (*Card1*). From Proposition 3.2, we obtain the number of paths that cross the internal link:

$$R_{d_{\frac{n-1}{2}} \leftrightarrow d_{\frac{n-1}{2}}}(\langle o_0 \dots o_{n-1}\rangle) = \frac{(k-1)(k-3)}{4}k^{n-1}$$

Therefore, the total number of paths using the internal link considering the configuration $\mathcal{C}_{Best}$ is:

$$\begin{aligned}
R_{\mathcal{C}_{Best}} &= R_{d_{\left[0,\frac{n-1}{2}-1\right]} \to d^+_{\frac{n-1}{2}}}(\langle o_0 \dots o_{n-1}\rangle) + R_{d_{\left[0,\frac{n-1}{2}-1\right]} \to d_{\left[\frac{n-1}{2}+1,n-1\right]}}(\langle o_0 \dots o_{n-1}\rangle) + \\
&+ R_{d^-_{\frac{n-1}{2}} \to d_{\left[\frac{n-1}{2}+1,n-1\right]}}(\langle o_0 \dots o_{n-1}\rangle) + R_{d_{\frac{n-1}{2}} \leftrightarrow d_{\frac{n-1}{2}}}(\langle o_0 \dots o_{n-1}\rangle) = \\
&= (k^{\frac{n-1}{2}} - 1) \times \frac{k-1}{2}k^{\frac{n-1}{2}} + (k^{\frac{n-1}{2}} - 1)^2 + \\
&+ \frac{k-1}{2}k^{\frac{n-1}{2}} \times (k^{\frac{n-1}{2}} - 1) + \frac{(k-1)(k-3)}{4}k^{n-1} = \\
&= (k^{\frac{n-1}{2}} - 1) \times \left(\frac{k-1}{2}k^{\frac{n-1}{2}} + (k^{\frac{n-1}{2}} - 1) + \frac{k-1}{2}k^{\frac{n-1}{2}}\right) + \\
&+ \frac{(k-1)(k-3)}{4}k^{n-1} = \\
&= (k^{\frac{n-1}{2}} - 1)(k^{\frac{n+1}{2}} - 1) + \frac{(k-1)(k-3)}{4}k^{n-1} \qquad (9)
\end{aligned}$$

We do not simplify the term $\frac{(k-1)(k-3)}{4}k^{n-1}$ to facilitate the calculations in the next section. □

3.4 Some properties related to $PATH(i)$

In this section, we show a set of properties related to $PATH(i)$. These properties will be used in the next section to demonstrate the configuration $\mathcal{C}_{Best}$ minimizes the use of the internal link.

Proposition 3.5 *Given a node $\langle o_0 \dots o_{n-1}\rangle$ of a nDT torus and two dimensions d_i and d_{i+k}, where $0 \leq i < i+k < n$, and considering that the four ports of the dimensions d_i and d_{i+k} are not connected to the same internal card, then:*

- *If the two ports of d_i are connected to the same card, and the two ports of d_{i+k} are connected to the other card, there are $4 \times PATH(k)$ paths between the ports of the dimension d_i and the ports of the dimension d_{i+k} crossing the internal link.*

- *In other case, there are* $2{\times}PATH(k)$ *paths between the ports of the dimension* d_i *and the ports of the dimension* d_{i+k} *crossing the internal link.*

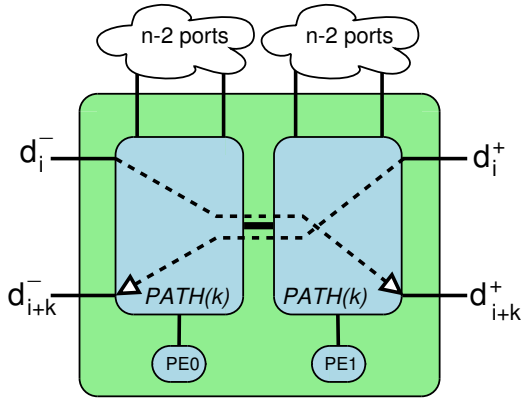
Proof: If the four ports of d_i and d_{i+k} are not connected to only one card, there are four possible ways of connecting them to the two cards. The Figure 4 shows graphically the number of paths that cross the internal link in each case.

a) The ports of d_i are separated in the two cards and the ports of d_{i+k} are also separated in the two cards (Figure 4(a)).

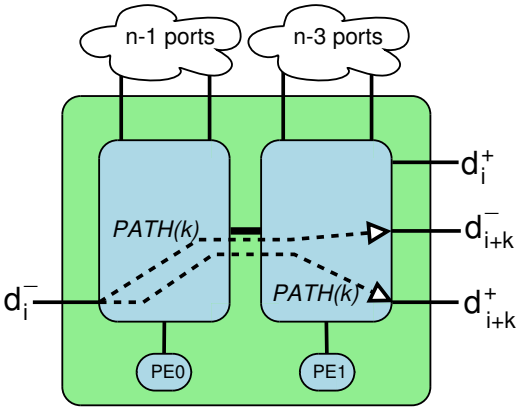
b) The ports of d_i are separated in the two cards, but the ports of d_{i+k} are connected to the same card (Figure 4(b)).

c) The ports of d_{i+k} are separated in the two cards, but the ports of d_i are connected to the same card (Figure 4(c)).

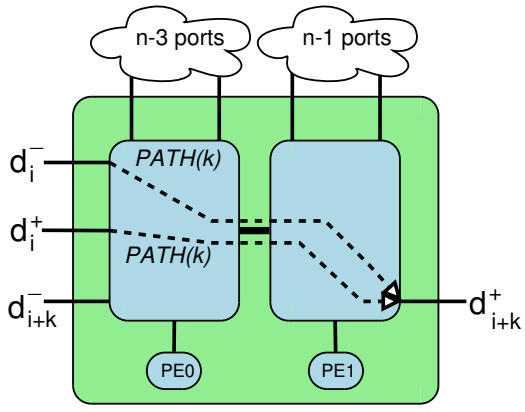
d) The ports of d_i are connected to one card and the ports of d_{i+k} are connected to the other card (Figure 4(d)).



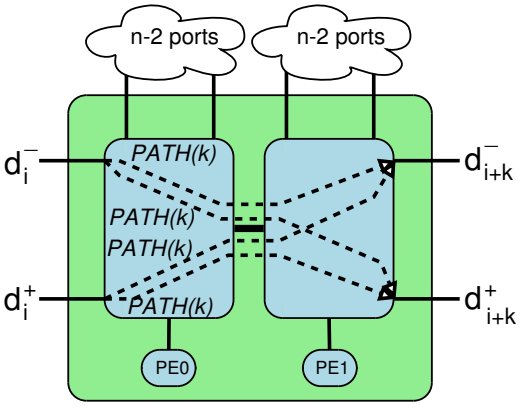
(a) The ports of d_i and d_{i+k} are separated in the two cards.



(b) Only the ports of d_i are separated in the two cards.



(c) Only the ports of d_{i+k} are separated in the two cards.



(d) The ports of d_i are in the same card and the ports of d_{i+k} are in the other card.

Figure 4: Number of paths using the internal link from dimension d_i to dimension d_{i+k}.

As can be seen, in the last case there are $4\times PATH(k)$ paths crossing the node using the internal link, and in the other three cases there are $2\times PATH(k)$ paths crossing the internal link. □

Proposition 3.6 *Let $\langle o_0 \dots o_{n-1}\rangle$ be a node of a nDT torus, built using a port configuration $\mathcal{C}$, $n \geq 4$, n even and $\mathcal{C} \neq \mathcal{C}_{Best}$. If no dimension has its ports separated in the two cards, there are at least $8\times PATH(1)$ paths crossing the node $\langle o_0 \dots o_{n-1}\rangle$ using the internal link.*

Proof: If n is even and no dimension has its ports separated in the two cards, the ports of $n/2$ dimensions are connected to each card. In the configuration $\mathcal{C}_{Best}$ the dimensions are distributed as follow:

$$\begin{aligned} d_0, d_1, \dots, d_{\frac{n}{2}-2}, d_{\frac{n}{2}-1} &\quad \text{are conneted to} \quad Card0 \\ d_{\frac{n}{2}}, d_{\frac{n}{2}+1}, \dots, d_{n-2}, d_{n-1} &\quad \text{are conneted to} \quad Card1 \end{aligned}$$

and therefore there are only two consecutive dimensions ($d_{\frac{n}{2}-1}$ and $d_{\frac{n}{2}}$) that are connected to different cards. However, any other distribution of the dimensions causes that there are at least two dimensions d_i and d_j, such that the dimensions d_{i+1} and d_{j+1} are connected to different cards than d_i and d_j. This happens in $\mathcal{C}$, due to $\mathcal{C} \neq \mathcal{C}_{Best}$. Then, from Proposition 3.5, there are at least $8\times PATH(1)$ paths using the internal link when crossing the node $\langle o_0 \dots o_{n-1}\rangle$ configured with port configuration $\mathcal{C}$.

□

Proposition 3.7 *Let $\langle o_0 \dots o_{n-1}\rangle$ be a node of a nDT torus, built using a port configuration $\mathcal{C}$, $n \geq 4$ and $\mathcal{C} \neq \mathcal{C}_{Best}$. If there is one dimension or more whose two ports are separated in the two cards, there are at least $6\times PATH(1)$ paths using the internal link for crossing the node.*

Proof: Let d_s be a dimension whose ports are separated in the two cards. Taking into account the possible values of s, we distinguish the following cases:

- $s = 0$. From Proposition 3.5, there are $2\times PATH(1)$ paths using the internal link between $d_s = d_0$ and d_1. Besides, depending on the distribution of the remaining ports:
  - If d_s is the only dimension whose ports are separated, there are $n-1$ consecutive dimensions to be distributed in two sets of $\frac{n-1}{2}$ dimensions. Then, there is at least one dimension d_i, such that d_i is connected to $Card0$ and d_{i+1} is connected to $Card1$ or vice versa. Therefore, from Proposition 3.5 there are other $4\times PATH(1)$ paths using the internal link, and so we have a total of $6\times PATH(1)$ paths crossing the internal link.
  - There is at least another dimension d_r whose ports are separated in the two cards:

* If $r = 1$, there are $2\times PATH(1)$ paths using the internal link between $d_r = d_1$ and d_2. The remaining $n-2$ dimensions are consecutive (from d_2 to d_{n-1}) and whatever configuration has at least one dimension d_i such that some of its ports are connected to different card than the ports of d_{i+1}. Therefore, there are other $2\times PATH(1)$ paths using the internal link, and so we have a total of $6\times PATH(1)$ paths crossing the internal link.
* If $1 < r < n-1$ there are $2\times PATH(1)$ paths using the internal link between d_{r-1} and d_r and other $2\times PATH(1)$ paths between d_r and d_{r+1}. Thus, we have a total of $6\times PATH(1)$ paths crossing the internal link.
* If $r = n-1$, there are at least $6\times PATH(1)$ using the internal link, considering the paths between d_{r-1} and d_r, and the paths between two consecutive dimensions of the remaining $n-2$ dimensions, as it occurs when $r = 1$.

- $0 < s < n-1$. From Proposition 3.5, there are $2\times PATH(1)$ paths using the internal link between d_{s-1} and d_s and other $2\times PATH(1)$ paths between d_s and d_{s+1}. Then, we must distribute the remaining $n-1$ dimensions in two sets of $\frac{n-1}{2}$ dimensions. We consider two subsets of ports: the first set contains the ports of s consecutive dimensions (from d_0 to d_{s-1}) and the second set contains the ports of $(n-1)-s$ consecutive dimensions (from d_{s+1} to d_{n-1}). Taking into account the value of s we can distinguish the following cases:

  – $s < \frac{n-1}{2}$. In this case, $(n-1)-s$ is greater than $\frac{n-1}{2}$, so there is at least one dimension d_i, when $s < i < n-1$, that some of its ports are connected to different card than the ports of d_{i+1}. Therefore, from Proposition 3.5 there are at least another $2\times PATH(1)$ paths using the internal link.

  – $s > \frac{n-1}{2}$. In this case, s is greater than $\frac{n-1}{2}$, so there is at least one dimension d_i, when $0 \leq i < s$, that some of its ports are connected to different card than the ports of d_{i+1}. Therefore, from Proposition 3.5 there are other $2\times PATH(1)$ paths using the internal link.

  – $s = \frac{n-1}{2}$. In this case, the two subsets of ports contain $\frac{n-1}{2}$ dimensions. We can distribute each subset on each card, but this is the same distribution than $\mathcal{C}_{Best}$, and $\mathcal{C} \neq \mathcal{C}_{Best}$. In any other distribution there are at least two consecutive dimension whose ports are connected to different cards, adding at least other $2\times PATH(1)$ paths using the internal link from Proposition 3.5.

  Hence, if $0 < s < n-1$ there are at least $6\times PATH(1)$ paths using the internal link.

- If $s = n-1$, there are at least $6\times PATH(1)$ paths using the internal link, for the same reasons than when $s = 0$.

Then, for all possible values of s, there are at least $6\times PATH(1)$ paths using the internal link when crossing the node $\langle o_0 \dots o_{n-1} \rangle$ using port configuration $\mathcal{C}$. □

Proposition 3.8 *Let $\langle o_0 \dots o_{n-1} \rangle$ be a node of a nDT torus, built using a port configuration $\mathcal{C}$ and $n \geq 4$. If there is one dimension or more whose two ports are separated in the two cards, there are at least $2{\times}PATH(2)$ paths using the internal link for crossing a node.*

Proof: Let d_s be a dimension whose ports are separated in the two cards. On one hand, if $2 \leq s \leq n-1$, the dimension d_{s-2} always exists. On the other hand, the dimension d_{s+2} always exists if $0 \leq s \leq n-3$. Since $n \geq 4$, any value of s satisfies at least one of the previous conditions. Hence, from Proposition 3.5 there are at least $2{\times}PATH(2)$ paths using the internal link between d_{s-2} and d_s or between d_s and d_{s+2}. □

3.5 Demonstrating that $\mathcal{C}_{Best}$ is the optimal configuration

In order to demonstrate that $\mathcal{C}_{Best}$ is the optimal configuration, we show that for any other configuration the number of paths using the internal link is greater than for the configuration $\mathcal{C}_{Best}$.

Theorem 3.1 *Given a nDT torus, with $n \geq 3$ and $k \geq 5$, the configuration $\mathcal{C}_{Best}$ minimizes the number of paths that cross the internal link.*

Proof: We distinguish three different cases:

$3DT$ torus

We studied the use of the internal link in a $3DT$ torus in [4]. In this paper, we calculated the number of paths that cross the internal link for all the different 10 configurations, and we demonstrated that configuration $\mathcal{C}_{Best}$ minimizes the use of the internal link, except when $k = 3$.

nDT torus, when n is odd and $n \geq 5$

We demonstrate that $\mathcal{C}_{Best}$ is the optimal configuration by reductio ad absurdum. Let $\mathcal{C}_\alpha$ be a configuration that minimizes the use of the internal link and $\mathcal{C} \neq \mathcal{C}_{Best}$. Then, from Propositions 3.7 and 3.8, there are at least $6{\times}PATH(1)$ and $2{\times}PATH(2)$ paths using the internal link. Moreover, because n is odd, there is at least one dimension whose ports are separated in the two cards and there are at least $PATH(0)$ paths using the internal link. Hence, the minimum number of paths that cross the internal link in $\mathcal{C}_\alpha$ is:

$$\begin{aligned} R_{\mathcal{C}_\alpha} \leq R_{\mathcal{C}_{\alpha min}} &= PATH(0) + 6{\times}PATH(1) + 2{\times}PATH(2) = \\ &= \frac{(k-1)(k-3)}{4}k^{n-1} + 6{\times}\frac{(k-1)^2}{4}k^{n-2} + 2{\times}\frac{(k-1)^2}{4}k^{n-3} = \\ &= \frac{(k-1)(k-3)}{4}k^{n-1} + (6k+2){\times}\frac{(k-1)^2}{4}k^{n-3} \end{aligned}$$

If $\mathcal{C}_\alpha$ minimizes the use of the internal link, then:

$$R_{\mathcal{C}_\alpha} < R_{\mathcal{C}_{Best}}$$

Therefore:

$$\begin{aligned}
R_{\mathcal{C}_{\alpha min}} \leq R_{\mathcal{C}_\alpha} &< R_{\mathcal{C}_{Best}} \\
R_{\mathcal{C}_{\alpha min}} &< R_{\mathcal{C}_{Best}} \\
R_{\mathcal{C}_{\alpha min}} - R_{\mathcal{C}_{Best}} &< 0 \\
\frac{(k-1)(k-3)}{4}k^{n-1} + (6k+2)\times\frac{(k-1)^2}{4}k^{n-3} &- \\
-\left((k^{\frac{n-1}{2}}-1)(k^{\frac{n+1}{2}}-1) + \frac{(k-1)(k-3)}{4}k^{n-1}\right) &< 0 \\
(6k+2)\times\frac{(k-1)^2}{4}k^{n-3} - (k^{\frac{n-1}{2}}-1)(k^{\frac{n+1}{2}}-1) &< 0 \\
\frac{k^n - 5k^{n-1} + k^{n-2} + k^{n-3}}{2} + k^{\frac{n+1}{2}} + k^{\frac{n-1}{2}} - 1 &< 0 \\
(k^3 - 5k^2 + k + 1)\frac{k^{n-3}}{2} + (k+1)k^{\frac{n-1}{2}} &< 1 \\
((k-5)k^2 + k + 1)\frac{k^{n-3}}{2} + (k+1)k^{\frac{n-1}{2}} &< 1
\end{aligned}$$

Since $k \geq 5$, $((k-5)k^2+k+1)\frac{k^{n-3}}{2}$ and $(k+1)k^{\frac{n-1}{2}}$ are always positive. The minimum value for this sum is obtained when $k=5$ and $n=5$. Then, the minimum value is 225 that is greater than 1. Therefore, we reach a contradiction and the configuration $\mathcal{C}_\alpha$ does not minimize the use of the internal link.

nDT torus, when n is even and $n \geq 4$

Let $\mathcal{C}_\beta$ be a configuration that minimizes the use of the internal link and $\mathcal{C} \neq \mathcal{C}_{Best}$. Taking into account the number of dimensions whose ports are separated in the two cards, we can distinguish two cases:

- $\mathcal{C}_{\beta_1}$**: There is no dimension whose ports are separated in the two cards.** From Proposition 3.6, there are at least $8\times PATH(1)$ paths using the internal link. Hence, the minimum number of paths that cross the internal link in $\mathcal{C}_{\beta_1}$ is:

$$R_{\mathcal{C}_{\beta_1}} \leq R_{\mathcal{C}_{\beta_1 min}} = 8\times PATH(1) = 2\times(k-1)^2k^{n-2}$$

If $\mathcal{C}_{\beta_1}$ minimizes the use of the internal link, then:

$$R_{\mathcal{C}_{\beta_1}} < R_{\mathcal{C}_{Best}}$$

Therefore:

$$\begin{aligned} R_{\mathcal{C}_{\beta_1 min}} \leq R_{\mathcal{C}_{\beta_1}} &< R_{\mathcal{C}_{Best}} \\ R_{\mathcal{C}_{\beta_1 min}} &< R_{\mathcal{C}_{Best}} \\ R_{\mathcal{C}_{\beta_1 min}} - R_{\mathcal{C}_{Best}} &< 0 \\ 2\times(k-1)^2 k^{n-2} - (k^{\frac{n}{2}}-1)^2 &< 0 \\ k^n - 4k^{n-1} + 2k^{n-2} + 2k^{\frac{n}{2}} - 1 &< 0 \\ (k^2-4k+2)k^{n-2} + 2k^{\frac{n}{2}} - 1 &< 0 \\ ((k-4)k+2)k^{n-2} + 2k^{\frac{n}{2}} - 1 &< 0 \\ ((k-4)k+2)k^{n-2} + 2k^{\frac{n}{2}} &< 1 \end{aligned}$$

Since $k \geq 5$, $((k-4)k+2)k^{n-2}$ and $2k^{\frac{n}{2}}$ are always positive. The minimum value for this sum is obtained when $k = 5$ and $n = 4$. Then, the minimum value is 225 that is greater than 1. Therefore, we reach a contradiction and the configuration $\mathcal{C}_{\beta_1}$ does not minimize the use of the internal link.

- $\mathcal{C}_{\beta_2}$**: There are two or more dimensions whose ports are separated in the two cards.** From Propositions 3.7 and 3.8, there are at least $6\times PATH(1)$ and $2\times PATH(2)$ paths using the internal link. Hence, the minimum number of paths that cross the internal link in $\mathcal{C}_{\beta_2}$ is:

$$\begin{aligned} R_{\mathcal{C}_{\beta_2}} \leq R_{\mathcal{C}_{\beta_2 min}} &= 6\times PATH(1) + 2\times PATH(2) = \\ &= 6\times\frac{(k-1)^2}{4}k^{n-2} + 2\times\frac{(k-1)^2}{4}k^{n-3} = \\ &= (6k+2)\times\frac{(k-1)^2}{4}k^{n-3} \end{aligned}$$

If $\mathcal{C}_{\beta_2}$ minimizes the use of the internal link, then:

$$R_{\mathcal{C}_{\beta_2}} < R_{\mathcal{C}_{Best}}$$

Therefore:

$$\begin{aligned} R_{\mathcal{C}_{\beta_2 min}} \leq R_{\mathcal{C}_{\beta_2}} &< R_{\mathcal{C}_{Best}} \\ R_{\mathcal{C}_{\beta_2 min}} &< R_{\mathcal{C}_{Best}} \\ R_{\mathcal{C}_{\beta_2 min}} - R_{\mathcal{C}_{Best}} &< 0 \\ (6k+2)\times\frac{(k-1)^2}{4}k^{n-3} - (k^{\frac{n}{2}}-1)^2 &< 0 \\ \frac{k^n - 5k^{n-1} + k^{n-2} + k^{n-3}}{2} + 2k^{\frac{n}{2}} - 1 &< 0 \\ (k^3-5k^2+k+1)\frac{k^{n-3}}{2} + 2k^{\frac{n}{2}} - 1 &< 0 \\ ((k-5)k^2+k+1)\frac{k^{n-3}}{2} + 2k^{\frac{n}{2}} - 1 &< 0 \\ ((k-5)k^2+k+1)\frac{k^{n-3}}{2} + 2k^{\frac{n}{2}} &< 1 \end{aligned}$$

Since $k \geq 5$, $((k-5)k^2+k+1)\frac{k^{n-3}}{2}$ and $2k^{\frac{n}{2}}$ are always positive. The minimum value for this sum is obtained when $k = 5$ and $n = 4$. Then, the minimum value is 65 that is greater than 1. Therefore, we reach a contradiction and the configuration $\mathcal{C}_{\beta_2}$ does not minimize the use of the internal link.

Neither $\mathcal{C}_\alpha$, nor $\mathcal{C}_{\beta_1}$ nor $\mathcal{C}_{\beta_2}$ minimize the use of internal link. Then, $\mathcal{C}_{Best}$ minimizes the number of paths crossing the internal link and $\mathcal{C}_{Best}$ is the optimal configuration when $k \geq 5$. A similar study can be developed to demonstrate that configuration $\mathcal{C}_{Best}$ is the best configuration starting from the initial hypothesis that $k \geq 3$ instead $k \geq 5$, but we have preferred to not show this study because is larger and more tedious than the present study. In any case, configuration $\mathcal{C}_{Best}$ is the optimal configuration for nDT torus when $k = 3$, except when $n = 3$, as we were shown in [4]. □

4 Routing in nDT torus

The routing algorithm is the mechanism that determines the path selected by a message to reach its destination. In many cases, some situations can difficult the routing, like *deadlock*, *livelock* or *starvation*. Specifically, the *deadlock* is an inherent problem in k-ary n-cubes. This problem is even more important in a nDT torus because the internal link is shared by all dimensions of the nDT torus.

In [4], we presented a study about how the *deadlock* appears in the $3DT$ torus topology and explained how to avoid it using virtual channels [9] or the bubble flow control mechanism [6]. Basically, in a nDT torus that uses the configuration $\mathcal{C}_{Best}$, the *deadlock* occurs for the same reasons that in the $3DT$ torus. It is easy to extend the DOR routing algorithm presented in [4] to route the packets and to avoid *deadlock* in nDT torus.

In this section, we present the DOR routing algorithm (**D**imension **O**rder **R**outing) adapted for nDT topology (Section 4.1). Then, we show a brief description of *deadlock* in the nDT torus (Section 4.2) and finally, we explain how to avoid *deadlock* in Section 4.3.

4.1 DOR routing algorithm adapted for nDT torus topology

DOR routing is commonly used in a k-ary n-cube because it is a very simple routing algorithm. Basically, a message is routed by the n dimensions following an ascending (or descending) strict order. If a node is identified by a n-tuple $\langle o_0 \dots o_{n-1} \rangle$, a message needing to use all the dimensions is first routed through dimension 0, after that it is routed through dimension 1, and so on until reaching the dimension $n-1$.

For a nDT torus topology, each PE needs an identifier composed of n digits, one digit for each dimension, and another digit to identify the PE inside the node.

First, a message crosses the n dimensions as needed. In this case, if the message has reached the destination node, it checks if the message is destined to the current PE or the neighbor PE, routing the message to the NIC or the internal link, respectively. Finally, we check if the output link belongs to the current card. Otherwise, the selected output port will be the internal link.

In Algorithm 1 we can see the pseudo-code of the DOR routing algorithm adapted for the nDT torus. The function *ringDirection()* (Algorithm 2) is used for DOR routing to determine the output direction in any ring.

Algorithm 1 DOR routing algorithm for a nDT torus.

Require: current PE $\langle o_0 \dots o_{n-1} | pe \rangle$, destination PE $\langle o'_0 \dots o'_{n-1} | pe' \rangle$
Return: output port p

```
1: if o_0 ≠ o'_0 then
2:    p = ringDirection(o_0, o'_0)
3: else if o_1 ≠ o'_1 then
4:    p = ringDirection(o_1, o'_1)
5:    ...
6: else if o_{n-1} ≠ o_{n-1} then
7:    p = ringDirection(o_{n-1}, o'_{n-1})
8: else if pe ≠ pe' then
9:    p = internal_link
10: else
11:    p = NIC
12: end if
13: if p ∈ LINKS(pe) then
14:    return p
15: else
16:    return internal_link
17: end if
```

Algorithm 2 *ringDirection()* function.

Require: current digit o_i, destination digit o'_i
Return: output port $(d_i^+,\ d_i^-)$ // $0 \leq i < n$

```
1: aux = (o'_i − o_i) mod k_i
2: if aux > k_i/2 then
3:    aux = aux − k_i
4: end if
5: if aux ≥ 0 then
6:    return d_i^+
7: else
8:    return d_i^-
9: end if
```

4.2 Analysing the *deadlock* in nDT torus

Most of the deterministic routing algorithms base their deadlock-freedom on the channel dependency graph. A routing algorithm is deadlock free if there are no cycles in its channel dependency graph [9]. As it happens in the $3DT$ torus, new cycles appear on the nDT torus network because a message can use the internal link regardless of the dimension where it is traveling.

If we analyze in detail the use of the internal link, we can distinguish 3 cases (Figure 5), depending on the destination of the message after using the internal link:

1.- The message uses the internal link to be injected in a dimension d_i, and $i \neq \frac{n-1}{2}$ when n is odd (d_i is not the dimension whose ports are separated in the two cards). In Figure 5 we can see how a message that arrives from port d_0^- and another message that arrives from port $d_{\frac{n-1}{2}}^-$, and both messages must use the internal link to be injected in dimension $d_{\frac{n-1}{2}+1}$ (red dotted line).

2.- The destination of the message is the PE connected to the other card in the node. In this case, the message can arrive at the node from any link of the current card (blue dotted line).

3.- The message uses the internal link as a part of the dimension $d_{\frac{n-1}{2}}$. The message can cross the $d_{\frac{n-1}{2}}$ before using the internal link or can be injected from another dimension. Note that when n is even, no dimension has its ports separated in the two cards, and there are not messages if this type crossing the internal link. In Figure 5 we can see a message that arrives from the port d_0^+, as well as a message that crosses the dimension $d_{\frac{n-1}{2}}$, must use the internal link to exit the node from the $d_{\frac{n-1}{2}}^-$ port (yellow dotted line).

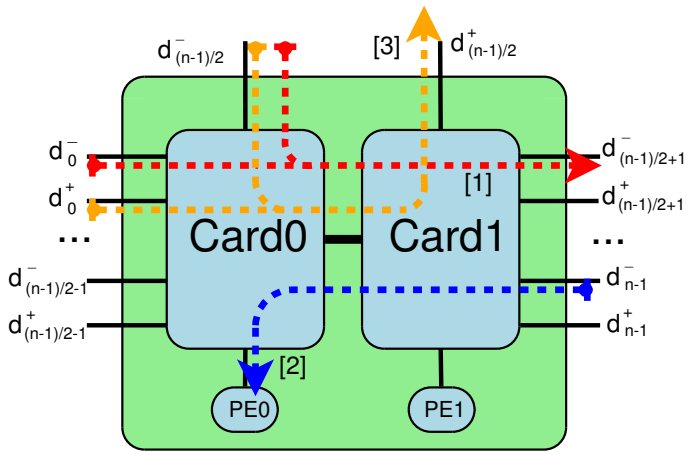


Figure 5: Possible uses of the internal link.

Then, we have identified two possible types of cycles in which the internal link is involved:

A.- Several messages use the internal link as part of the ring of dimension $d_{\frac{n-1}{2}}$. Without mechanisms to avoid *deadlock*, it can appear in any ring of a k-ary n-cube. This type of cycle is caused by the traffic of type 3. Example 4.1 shows in more detail a situation in which a *deadlock* appears due to this reason.

B.- Several messages use along their paths several internal links to be injected in a new dimension and also to reach the destination PE. This type of cycle appears due to the type of traffic 1 and 2. Example 4.2 shows in more detail a situation in which a *deadlock* appears due to this reason.

Example 4.1 *Given a nDT torus network, with two nodes in dimension* $d_{\frac{n-1}{2}}$ *and the nodes using the configuration* $\mathcal{C}_{Best}$*, and considering that:*

- *The PE* $\langle o_1, \ldots, o_{\frac{n-1}{2}-1}, 0, o_{\frac{n-1}{2}+1}, \ldots, o_{n-1} |\ 0\rangle$ *sends a message to the PE* $\langle o_1, \ldots o_{\frac{n-1}{2}-1}, 1, o_{\frac{n-1}{2}+1}, \ldots, o_{n-1} |\ 0\rangle$*, and vice versa.*
- *The PE* $\langle o_1, \ldots, o_{\frac{n-1}{2}-1}, 0, o_{\frac{n-1}{2}+1}, \ldots, o_{n-1} |\ 1\rangle$ *sends a message to the PE* $\langle o_1, \ldots, o_{\frac{n-1}{2}-1}, 1, o_{\frac{n-1}{2}+1}, \ldots, o_{n-1} |\ 1\rangle$*, and vice versa.*

there exist cycles and deadlock can appear in the network. In Figure 6 we can see graphically this situation.

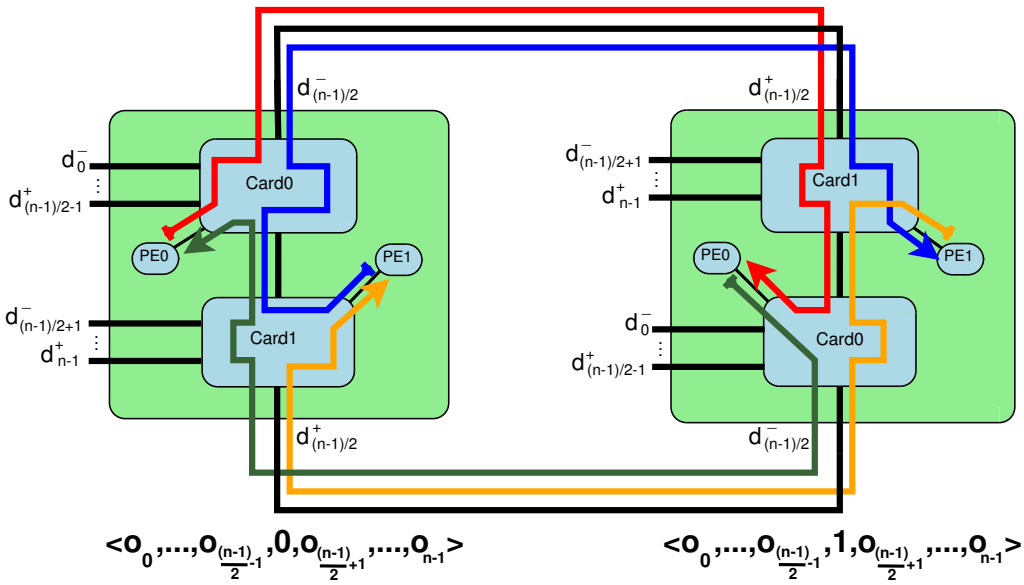


Figure 6: Possible deadlock due to the use of the internal link as a part of the Y-dimension ring.

Example 4.2 *Given a nDT torus network of any size and nodes using configuration D, and considering that:*

- *The PE* $\langle o_0, \ldots, o_{n-1} |\ 1\rangle$ *sends a message to the PE* $\langle o_0 + 1, \ldots, o_{n-1} + 1 |\ 0\rangle$.
- *The PE* $\langle o_0 + 1, \ldots, o_{n-1} + 1 |\ 1\rangle$ *sends a message to the PE* $\langle o_0, \ldots, o_{n-1} |\ 0\rangle$.

there exist cycles and deadlock can appear in the network. In Figure 7 we can see graphically this situation.

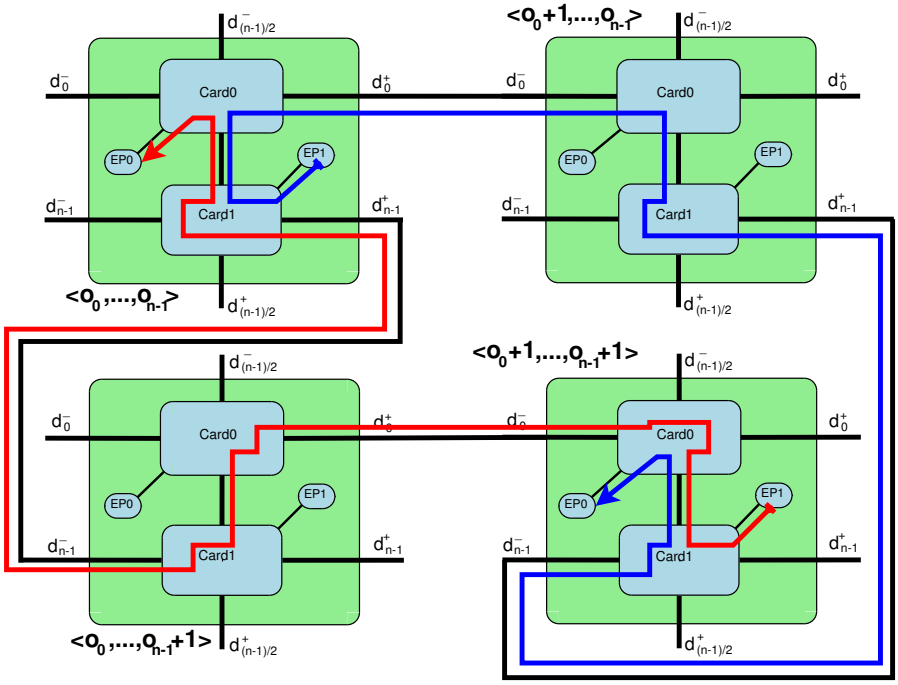


Figure 7: Possible deadlock due to the use of the internal link to change between dimensions and to reach the destination PE.

4.3 Deadlock-avoidance in nDT torus topologies

Once we know how the cycles are produced, now we proceed to their elimination. As mentioned above, new cycles appear in the network because all the dimensions use the internal link. To avoid *deadlock*, it is necessary to separate in different channels the three types of messages that uses the internal link. Then, we need at least three virtual channels. One virtual channel is used to route the messages destined to the neighbor PE. A second virtual channel is used to route the messages whose input and output ports belong to different dimension and different cards. Finally, when n is odd, a third virtual channel is used for the messages that are traveling in dimension $d_{\frac{n-1}{2}}$ if the network employs the bubble flow control mechanism [6] to avoid *deadlock*. If virtual channels [9] are employed, two virtual channels are necessary instead of one for this type of messages.

Algorithm 3 Modified DOR to avoid deadlock using virtual channels and conf. $\mathcal{C}_{Best}$.

Require: current $PE\ \langle o_0 \dots o_{n-1} | pe \rangle$, destination $PE\ \langle o'_0 \dots o'_{n-1} | pe' \rangle$
Return: output port p, virtual channel vc

1: **if** $o_0 \neq o'_0$ **then**
2: $p = ringDirection(o_0, o'_0)$
3: **else if** $o_1 \neq o'_1$ **then**
4: $p = ringDirection(o_1, o'_1)$
5: ...
6: **else if** $o_{n-1} \neq o_{n-1}$ **then**
7: $p = ringDirection(o_{n-1}, o'_{n-1})$
8: **else if** $pe \neq pe'$ **then**
9: $p = internal\_link$
10: $vc = 3$ [5] *// type 2.*
11: **else**
12: $p = NIC$
13: **end if**
14: **if** $p \notin LINKS(pe)$ [6] **then**
15: **if** $p \notin d_{\frac{n-1}{2}}$ **then**
16: $vc = 0$ *// type 1.*
17: **else if** $vc = Up\_Links$ **then**
18: $vc = 1$ *// type 3.*
19: **else**
20: $vc = 2$ *// type 3.*
21: **end if**
22: $p = internal\_link$
23: **end if**

So two, three or four virtual channels, depending on the chosen mechanism and the number of dimensions, are required in internal link to avoid *deadlock*. Also, if the nDT torus uses the configuration $\mathcal{C}_{Best}$, the number of required virtual channels are always the same, independently the values of n and k. If we use another configuration to build the nDT torus, new cycles always appear in the channel dependency graph, increasing the number of virtual channels to avoid *deadlock*.

Adding the virtual channel selection implemented in both versions of DOR algorithm for $3DT$ torus to the DOR algorithm presented in Section 4.1, we obtain a routing algorithm that ensures the *deadlock* freedom in the network. We only show the DOR algorithm that uses virtual channels to avoid deadlock, but we can implement in a similar way the algorithm to avoid *deadlock* using the bubble flow control mechanism.

Algorithm 3 and 4 show the modifications realized in the routing algorithm and *ringDirection()* function, respectively, when n is odd.

[5] If n is even, $vc = Up\_Links$.

[6] If n is even, it is not necessary to check the port p. In this case, when $p \notin LINKS(pe)$, $vc = Low\_Links$.

Algorithm 4 Modified *ringDirection()* function to avoid deadlock using virtual channels and configuration $\mathcal{C}_{Best}$.

Require: current digit o_i, destination digit o'_i
Return: output port $(d_i^+,\ d_i^-)$, virtual channel vc // $0 \leq i < n$

1: $aux = (o_i - o'_i) mod\ k_i$
2: **if** $aux > k/2$ **then**
3: $\quad aux = aux - k_i$
4: **end if**
5: **if** $aux \geq 0$ **then**
6: $\quad p = d_i^+$
7: **else**
8: $\quad p = d_i^-$
9: **end if**
10: **if** $o'_i > o_i$ **then**
11: $\quad vc = Up\_Links$
12: **else**
13: $\quad vc = Low\_Links$
14: **end if**

References

[1] N.R. Adiga, G. Almasi, Y. Aridor, R. Barik, D. Beece, R. Bellofatto, G. Bhanot, R. Bickford, M. Blumrich, and A. A. Bright. An overview of the Blue Gene/L supercomputer. In *Supercomputing 2002 Technical Papers*, 2002.

[2] Y. Ajima, Y. Takagi, T. Inoue, S. Hiramoto, and T. Shimizu. The Tofu interconnect. In *High Performance Interconnects (HOTI), 2011 IEEE 19th Annual Symposium on*, pages 87–94, 2011.

[3] R. Alverson, D. Roweth, and L. Kaplan. The Gemini system interconnect. In *High Performance Interconnects (HOTI), 2010 IEEE 18th Annual Symposium on*, pages 83–87, 2010.

[4] Francisco J. Andújar, Juan A. Villar, Francisco J. Alfaro, Jose L. Sánchez, and José Duato. Building 3D torus using low–profile expansion cards. Technical Report DIAB-11-02-3, Department of Computing Systems. University of Castilla-La Mancha, 2011. `http://www.dsi.uclm.es/descargas/thecnicalreports/DIAB-11-02-3/diab-11-02-3.pdf`.

[5] Francisco J. Andújar, Juan A. Villar, Francisco J. Alfaro, José L. Sánchez, and José Duato. Building 3D torus using low-profile expansion cards. *IEEE Transactions on Computers*, 2013.

[6] C. Carrion, R. Beivide, J.A. Gregorio, and F. Vallejo. A flow control mechanism to avoid message deadlock in k-ary n-cube networks. In *High-Performance Computing, 1997. Proceedings. Fourth International Conference on*, pages 322–329, dec 1997.

[7] Dong Chen, N.A. Eisley, P. Heidelberger, R.M. Senger, Y. Sugawara, S. Kumar, V. Salapura, D.L. Satterfield, B. Steinmacher-Burow, and J.J. Parker. The IBM Blue Gene/Q interconnection network and message unit. In *High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for*, pages 1–10, 2011.

[8] Cray Inc. Cray XT specifications. `http://www.cray.com/Products/XT/Specifications.aspx`, 2009.

[9] W.J. Dally and C.L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *Computers, IEEE Transactions on*, C-36(5):547–553, may 1987.

[10] J. J. Dongarra, H. W. Meuer, and E. Strohmaier. TOP500 supercomputer sites. *j-SUPERCOMPUTER*, 11(2–3):133–163, June 1995.

[11] José Duato, Sudhakar Yalamanchili, and Lionel Ni. *Interconnection networks. An engineering approach.* Morgan Kaufmann Publishers Inc., 2003.

[12] Lawrence Livermore National Laboratory. Sequoia supercomputer. `https://asc.llnl.gov/computing_resources/sequoia/`, 2012.

[13] C. E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Transactions on Computers*, 34(10):892–901, 1985.

[14] Oak Ridge National Laboratory. Introducing Titan. Advancing the era of accelerated computing. `http://www.olcf.ornl.gov/titan/`, 2012.

[15] M. Yokokawa, F. Shoji, A. Uno, M. Kurokawa, and T. Watanabe. The K-Computer: Japanese next-generation supercomputer development project. In *Low Power Electronics and Design (ISLPED) 2011 International Symposium on*, pages 371–372, 2011.