An Analysis of Connectivity of k-ary n-cube m-diag Interconnection Networks *

Didier Badouel, Charles A. Wüthrich, and Eugene L. Fiume
Computer System Research Institute
University of Toronto

May 22, 1992

Abstract

The difference among different topologies in interconnection networks is a debated issue among highly parallel machine designers, since opposing parameters influence the network performance. This paper presents a new model, k-ary n-cube m-diags, which generalizes existing k-ary n-cube topologies to include hexagonal networks and their n-dimensional extensions. A performance analysis based on this model is conducted, showing that the increase of connectivity in low-dimensional networks by means of diagonal links results in better latency and throughput in the network. An algorithm for routing messages in k-ary n-cube m-diags is proposed, showing the relative ease of performing this task on such topologies.

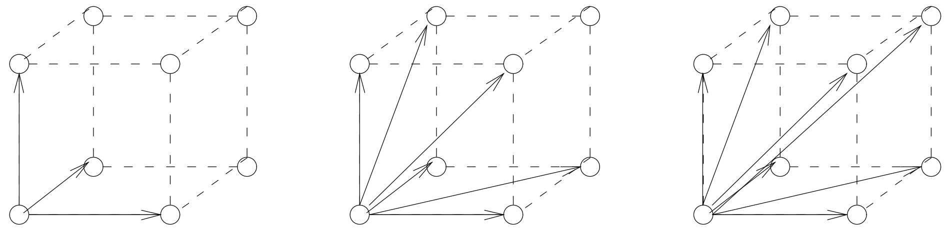
Index Terms Interconnection networks, highly parallel systems, topological comparisons, performance analysis, routing algorithms.

1 Introduction

Parallel architectures are receiving increased attention because scalar and vector supercomputers are reaching their performance limits. While it is clear that massively parallel systems with today's technology allow high technology to achieve high performances, to date it is still not clear which topologies are more suited for the interconnection network. The topology of an interconnection network is defined by the graph of the physical links between its nodes, and greatly influences the communication performance.

The performance of an interconnection network is measured by its latency and throughput. Throughput measures the number of messages the network can deliver in a unit time interval, and therefore represents the 'communication power' of the network, while latency measures the average delivery time for a message, and thus the network 'communication

*This work has been supported by the *Information Technology Research Centre* of Ontario, by the French *Institut National de Recherche en Informatique et Automatique*, and by the Swiss *Schweizerischer Nationalfonds zur Förderung der wissenschaftlichen Forschung.*



a) 2-ary 3-cube b) 2-ary 3-cube 2-diag c) 2-ary 3-cube 3-diag

Figure 1: Extending the neighbourhood relationship in a 2-ary 3-cube.

speed'. In order to increase communication performance, the best compromise between the wiring constraints and the average internode distance has to be determined.

For this purpose, different topologies have been proposed and compared. At first advocated for the logarithmic diameter of the network, binary hypercubes have been progressively abandoned in favour of low-dimensional topologies for reasons of *wire efficiency* [4]. The comparison of networks having the same hardware complexity, i.e. maintaining the same number of wires crossing the bisection, shows that by using channels of larger width networks of small connectivity can outperform highly connected topologies.

The model used for the comparison, k-ary n-cubes, is able to represent a large range of network configurations, from 2D tori to binary hypercubes, but it cannot be used for important regular topologies such as hexagonal meshes.

This paper presents a generalization of this model, which shall be called k-ary n-cube m-diags, that can include hexagonal networks and their n-dimensional extensions. K-ary n-cube m-diags allow diagonal connections in a k-ary n-cube without modifying the positions of the nodes, thus preserving the same node addressing in the network, and hence simplifying the model presented in [3, 7]. The new model allows one to theoretically compare the network performances which are presented in Section 3. The analysis shows that increasing the connectivity of low-dimensional networks improves both their latency and their throughput, for a given bisection width.

Moreover, this model permits the use of a simple message routing algorithm which is derived directly from the widely used e-cube algorithm [6], and which will be presented in Section 4. In the case of two-dimensional networks, this algorithm allows an elegant and natural routing mechanism on hexagonal meshes.

2 Extending k-ary n-cube Interconnection Networks

In order to compare interconnection networks, the k-ary n-cube model has been extensively used because it models a wide range of interconnection configurations among the processing elements. In fact, most of the existing concurrent architectures are either based on such a network topology or one that is isomorphic to it [4]. In this notation, n is the *dimension* of the network, and k is the width of the network in each direction. The number

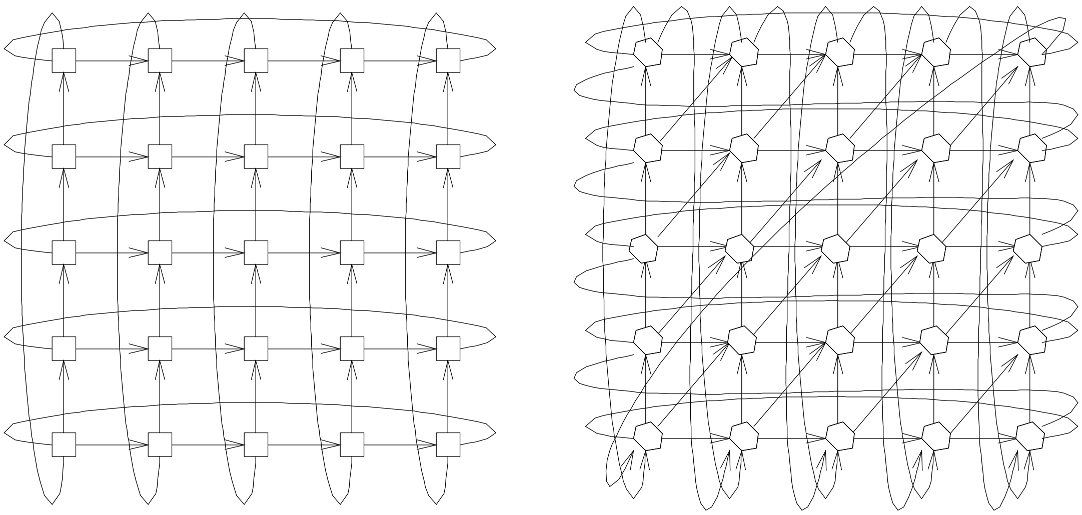


Figure 2: A 5-ary 2-cube and a 5-ary 2-cube 2-diag.

of nodes in the network is given by $N = k^n$. However, since the purpose of this paper is to introduce and evaluate alternative node interconnection configurations, an extension to this model is needed. The extension, which shall be denoted k-ary n-cube m-diag where $1 \leq m \leq n$, modifies the neighbourhood relation by allowing diagonal connections. Since only the neighbourhood relationship is modified, k and n will still represent the width and the dimension of the network. The number m in k-ary n-cube m-diag represents the maximum number of the dimensions in which diagonal connections are allowed, as illustrated in Figure 1. For example, in a k-ary n-cube 2-diag, only nodes lying in the same plane perpendicular to the axes can be diagonally connected. In fact, a k-ary n-cube 1-diag coincides with a k-ary n-cube. Moreover, a binary hypercube n-diag is a fully connected network.

Any node in a k-ary n-cube m-diag is identified by a n-uple $(a_0, a_1, \ldots, a_{n-1})$, where each a_i represents the position of the node in the i-th dimension $(0 \leq a_i \leq k-1)$. While in a k-ary n-cube, the neighbourhood of the node $(a_0, a_1, \ldots, a_i, \ldots, a_{n-1})$ is given by the nodes $(a_0, a_1, \ldots, a_i \oplus 1, \ldots, a_{n-1})$ $(\forall i)$, where $\oplus$ indicates the sum modulo k and the coordinates are numbers modulo k, in a k-ary n-cube 2-diag, the neighbourhood of the same node is given by the nodes $(a_0, a_1, \ldots, a_i \oplus 1, \ldots, a_j \oplus 1, \ldots, a_{n-1})$ $(\forall i, j)$. A bi-dimensional example is given in Figure 2: on the left, a 2D torus, and on the right, a wrapped hexagonal mesh. For the general k-ary n-cube m-diag two nodes $A = (a_0, a_1, \ldots, a_{n-1})$ and $B = (b_0, b_1, \ldots, b_{n-1})$ are neighbours *iff* $b_i \in \{a_i, a_i \oplus 1\} \wedge card(\{b_i : b_i = a_i \oplus 1\}) \leq m$. The links in k-ary n-cubes will be called *cartesian* links and the extra links in k-ary n-cube m-diag *diagonal* links.

This paper will focus on k-ary n-cube 2-diags since in the general model the hardware complexity grows very rapidly when both n and m increase. However, k-ary n-cube 2-diags provide significant performance improvements for low-dimensional interconnection networks under the assumption of constant wire bisection.

3 Performance Analysis

While several topology proposals for interconnection networks were compared by means of their abstract topological properties [2, 3, 8], Dally introduced a more realistic comparison method based on *constant hardware complexity* [4]. Since VLSI computing systems are wire-limited, the different networks are compared under the condition that a constant wire density is maintained, allowing wider channels in low-dimensional networks. As a result, low-dimensional interconnection networks, such as tori, outperform the high-dimensional ones, such as binary hypercubes.

The network performance is measured in terms of the average latency for delivering a message. If the *wormhole* switching technique is used, the mean latency of the network is

$$T_{net} = T_c \left(D + \frac{L}{W} \right) , \tag{1}$$

where T_c is the channel cycle time, D is the mean distance between two nodes, L is the average message length, and W is the channel width. Dally used a constant value for L ($L = 150$ bits) and computed T_{net} under different assumptions for T_c. First, he assumes *constant* wire delay, i.e. the number of cycles required for delivering a message is independent of the channel length. For a more realistic hardware representation, *logarithmic* and *linear* wire delays are then considered. When the wires are short, the logarithmic delay is an appropriate model, whereas linear delay is expected for longer channels. T_c is therefore

$$T_c(l) \;\alpha \begin{cases} 1 & \text{for constant delays} \\ 1 + \log_e l & \text{for logarithmic delays} \\ l & \text{for linear delays,} \end{cases} \tag{2}$$

where l is the longest wire in the system.

Thus, in order to compare the performances of k-ary n-cube 2-diags with k-ary n-cubes, $W(k, n)$, $l(k, n)$, and $D(k, n)$ have to be computed. While W, l, and D will refer to k-ary n-cubes, W_2, l_2, and D_2 will refer to k-ary n-cube 2-diags.

Channel Width. Channel width is a scale factor used to compare different interconnection networks. In order to maintain the same bisection width B, i.e. the same number of wires crossing the midline of the network, $\frac{B(k,n)}{W(k,n)}$ should be a constant. The constant used in [4] is N, and provides the normalization to a binary n-cube with unit width channels, $W = 1$.

To systematically lay out an n-dimensional network in two physical dimensions, $n/2$ of the logical dimensions are embedded into each physical dimension. Figure 3 illustrates such a planar projection: the $k^{\frac{n}{2}}$ nodes belonging to each of the $\sqrt{N}$ $(= k^{\frac{n}{2}})$ rows can be grouped to form k subsets each of which contains $k^{\frac{n}{2}-1}$ nodes. In other words, each row can be viewed as a k-ary 1-cube whose elements are themselves k-ary $(\frac{n}{2} - 1)$-cubes. Therefore only the channels of one of the logical dimensions, say the highest dimension, cross the midline of the network, the vertical bisection line in Figure 3. Due to *wraparound* in a toroidal topology, in each row of the physical projection twice the number of channels linking two k-ary $(\frac{n}{2} - 1)$-cube subnetworks cross its midpoint. For each node of a k-ary n-cube, there is only one channel in the highest dimension, and thus $2k^{\frac{n}{2}-1}$ channels cross the midline of

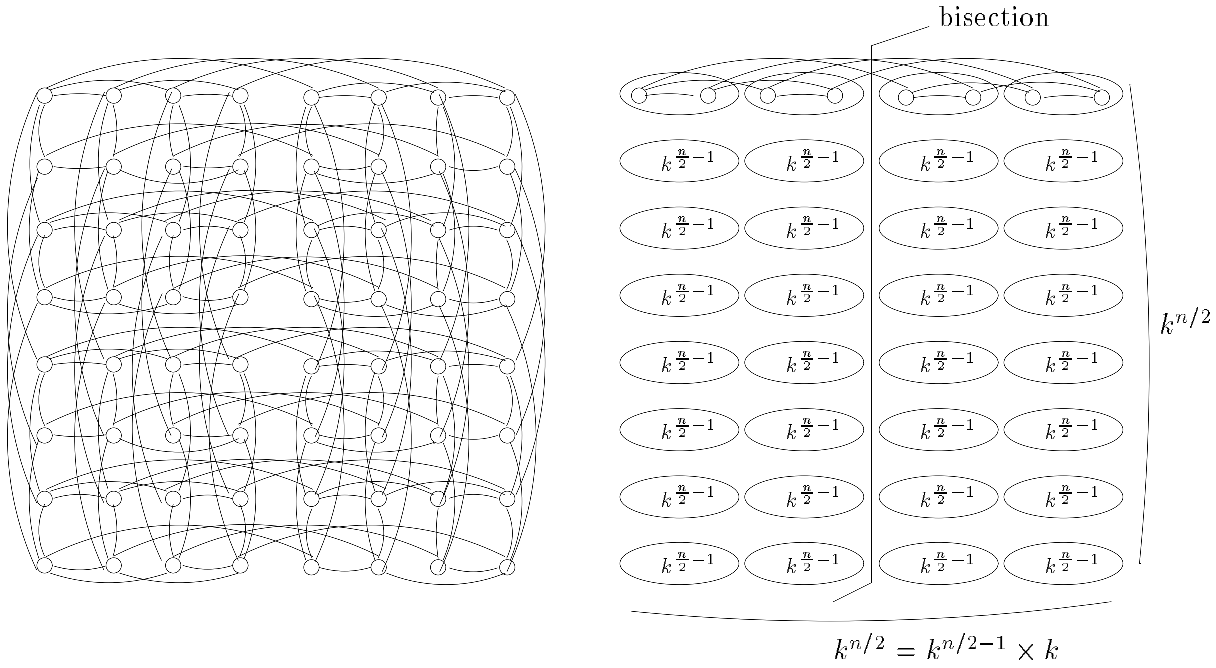


Figure 3: Computing the network bisection: a 4-ary 3-cube example.

the network for each row. On the other hand, for each node of a k-ary n-cube 2-diag, there are $(n-1)$ extra channels in the highest dimension, one for each plane passing at the node and containing a diagonal link. Therefore the number of channels crossing the midpoint of one row is $2nk^{\frac{n}{2}-1}$. To obtain the bisection width, this number has to be multiplied by the number of rows in the physical projection, i.e. $k^{\frac{n}{2}}$. Thus, the number of channels for the bisection is

$$B(k,n) = 2k^{\frac{n}{2}-1}k^{\frac{n}{2}} = 2k^{n-1}$$

for k-ary n-cubes, and

$$B_2(k,n) = 2nk^{\frac{n}{2}-1}k^{\frac{n}{2}} = 2nk^{n-1}$$

for k-ary n-cube 2-diags.

The channel width, using the normalization relation $\frac{B(k,n)}{W(k,n)} = k^n$, is

$$W(k,n) = \frac{k}{2} \tag{3}$$

for k-ary n-cubes, and

$$W_2(k,n) = \frac{k}{2n} \tag{3'}$$

for k-ary n-cube 2-diags.

Longest Wire. A similar procedure is used to compute the length of the longest wire. For k-ary n-cubes, the longest wires in the layout are the wires corresponding to the highest

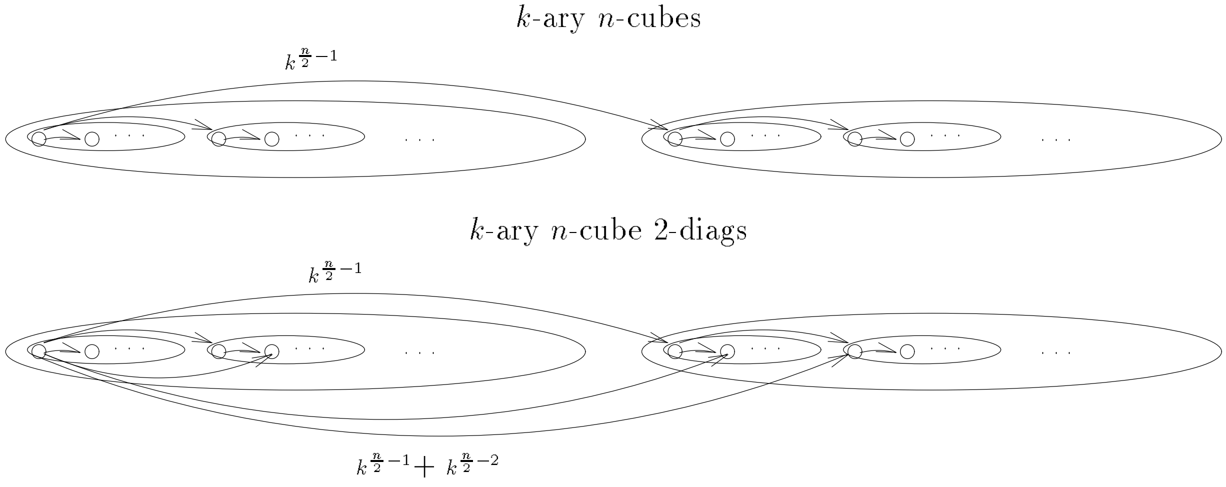


Figure 4: Computing the longest wire in k-ary n-cubes and k-ary n-cube 2-diags.

logical dimensions. In each of the two physical dimensions, the neighbour of a node is the node with the same position in the neighbouring subnetwork. Therefore the link between these two nodes skips the width of a subnetwork. For k-ary n-cubes the longest wire is therefore

$$l(k,n) = k^{\frac{n}{2}-1} \,. \tag{4}$$

Since in a k-ary n-cube 2-diag diagonal links are added, the longest wire in the network can be deduced from the definition of the longest wire in a k-ary n-cube(see Figure 4). If A and B are two nodes connected by a longest wire in a k-ary n-cube, in a k-ary n-cube 2-diag A will be diagonally connected with all neighbours of B except the one linked to B in the same logical direction linking A and B. Among these nodes the most distant from B is the one in the highest dimension of the k-ary $(\frac{n}{2}-1)$-cube subnetwork. The longest wire is the diagonal link between A and this neighbour of B, which skips a k-ary $(\frac{n}{2}-1)$-cube subnetwork and a k-ary $(\frac{n}{2}-2)$-cube subnetwork. Its length is thus

$$l_2(k,n) \;=\begin{cases} k^{\frac{n}{2}-1} + k^{\frac{n}{2}-2} = \left(1+\frac{1}{k}\right) k^{\frac{n}{2}-1} & n > 2 \\ l(k,n) & n = 2\,. \end{cases} \tag{4'}$$

It should be noted that the analysis assumes that in the bi-dimensional projection, all the channels in the same dimension are of the same length. Such a disposition is called a *folded* system [4].

Mean Distance. In a k-ary n-cube, internode communications require on average to travel one half of the network width in each dimension. The mean internode distance in a k-ary n-cube is therefore

$$D(k,n) = \left(\frac{k-1}{2}\right) n \tag{5}$$

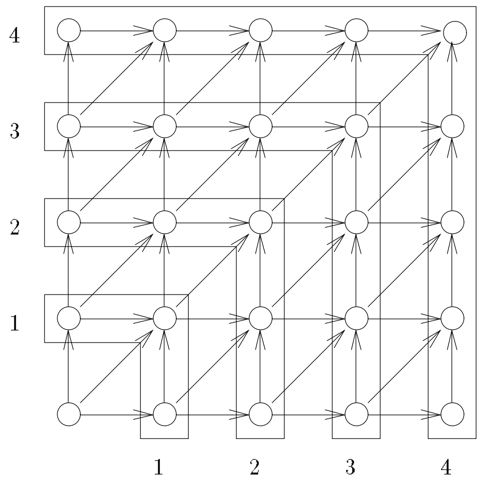


Figure 5: The distance graph: a 5-ary 2-cube 2-diag example.

For a k-ary n-cube 2-diag, the computation of the mean distance is less trivial and in fact remains an open problem. We will only give upper and lower bounds for D_2. A first approximation is obtained by observing that D_2 is less than D but greater than $D/2$ since, at best, cartesian hops can be grouped in pairs to be substituted by diagonal hops. A first coarse bracket for D_2 is therefore

$$\left(\frac{k-1}{4}\right) n < D_2(k,n) < \left(\frac{k-1}{2}\right) n \, . \tag{5'}$$

However, it is possible to obtain better bounds. K-ary n-cube m-diags are homogeneous interconnection networks, i.e. there is an isomorphism that maps any node of the network onto any other node[1] [8]. The mean distance between the nodes of the network is the sum of the distance $d_*(O,X)$ of each node X from the origin node O, divided by the total number of nodes minus one, as a node does not communicate with itself:

$$D_*(k,n) = \frac{\sum_X d_*(O,X)}{k^n - 1} \, . \tag{6}$$

Let $(x_0, x_1, \ldots, x_{n-1})$, where $0 \le x_i \le k-1$, be the coordinates of a node X, and order them in a descending fashion. The result is the sequence $y_0 \ge y_1 \ge \ldots \ge y_{n-1}$. For the same reason as in equation 5', the number of hops between O and X is bounded by

$$\frac{\sum x_i}{2} = \frac{\sum y_i}{2} = \le d_2(O,X) \le \sum x_i = \frac{\sum y_i}{2} \, . \tag{7}$$

In order to obtain a lower bound for D_2 in a k-ary n-cube 2-diag, D_n, the mean distance in a k-ary n-cube n-diag, has to be computed.

[1] A graph having this property is called a *regular* graph.

Proposition 3.1 *The mean distance on a k-ary n-cube n-diag is*

$$D_n(k,n) = \frac{\sum_{i=1}^{k-1} i((i+1)^n - i^n)}{k^n - 1} .$$

Proof. In a n-diagonal connected system, the distance of a node X to the origin is its largest coordinate value y_0 since the steps in the directions corresponding to y_i $(i \neq 0)$ can be combined with the steps in the direction corresponding to y_0. A node X has a distance d from O if it belongs to the hypercube of width d having O as the origin withou belonging to the hypercube of width $d-1$. Therefore there are $(d+1)^n - d^n$ nodes at a distance d from O, where $1 \leq d \leq k-1$. Figure 5 illustrates a two-dimensional example. Since the mean distance between the nodes of a network is given by the sum of the distances between a fixed node, for example O, and all other nodes of the network, divided by the number of possible pairs of nodes containing the origin, i.e. $k^n - 1$, the proposition is true. □

Note that

$$\sum_X y_0 = \sum_{i=1}^{k-1} i((i+1)^n - i^n) . \tag{8}$$

When a message travels from O to X in a k-ary n-cube 2-diag, the steps in any cartesian direction can always be combined by means of diagonal steps; only the direction corresponding to y_0 cannot be systematically combined with another direction. A new bracket for $d_2(O,X)$ is therefore

$$y_0 \leq d_2(O,X) \leq y_0 + \frac{\sum_{i=1}^{n-1} y_i}{2} = \frac{y_0 + \sum_{i=0}^{n-1} y_i}{2} . \tag{9}$$

Averaging for the whole network, and using Equations 5 and 8, we obtain

$$D_n(k,n) \leq D_2(k,n) \leq \frac{D_n(k,n) + D(k,n)}{2} . \tag{5''}$$

The curves in Figure 6 illustrate the mean distance in k-ary n-cubes and k-ary n-cube 2-diags and show that this distance is almost halved. It should be noted that the results shown in the Figure are given using the exact values for D_2, obtained by running the routing algorithm described in Section 4.

Latency. Given the channel width, the longest wire, and the mean distance for k-ary n-cubes and k-ary n-cube 2-diags, latency can now be evaluated using its definition from Equation 4. The curves in Figures 7 and 8 show the latency on k-ary n-cubes and k-ary n-cube 2-diags with respect to the dimension of the network. The results are given only for logarithmic and linear delays of the communication links since constant delays are not a realistic assumption. At first sight, the advantage given by the diagonal links is not evident since in k-ary n-cube 2-diags the latency quickly grows to infinity as the number of dimensions increases. These curves show that the hardware complexity introduced by the diagonal connections is feasible only for low dimensional interconnection networks. In fact, the diagonal links emphasize the behaviour of the latency curves in k-ary n-cubes: they further decrease the latency where it is low but increase it dramatically where it is already high.

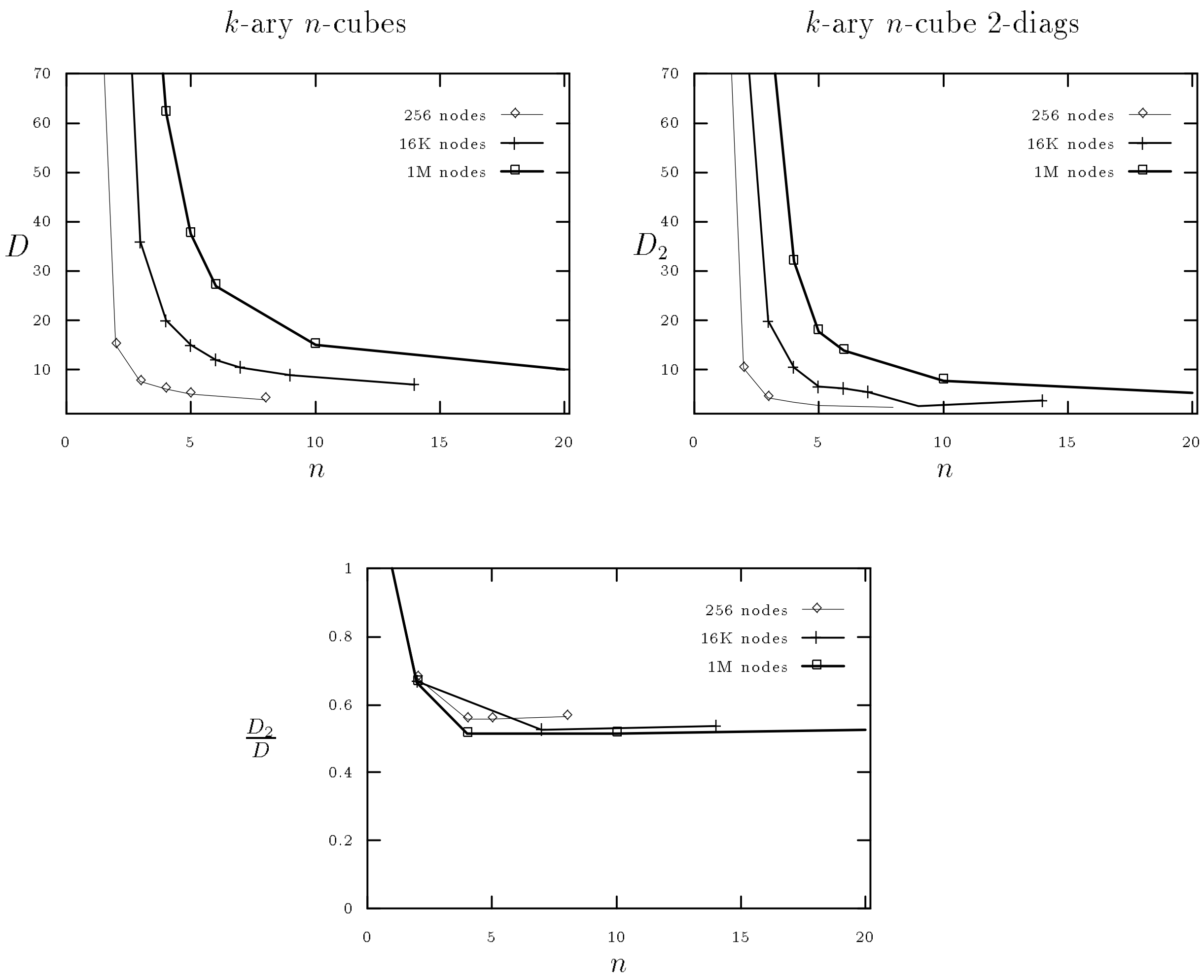


Figure 6: Mean distance (number of hops) between two nodes.

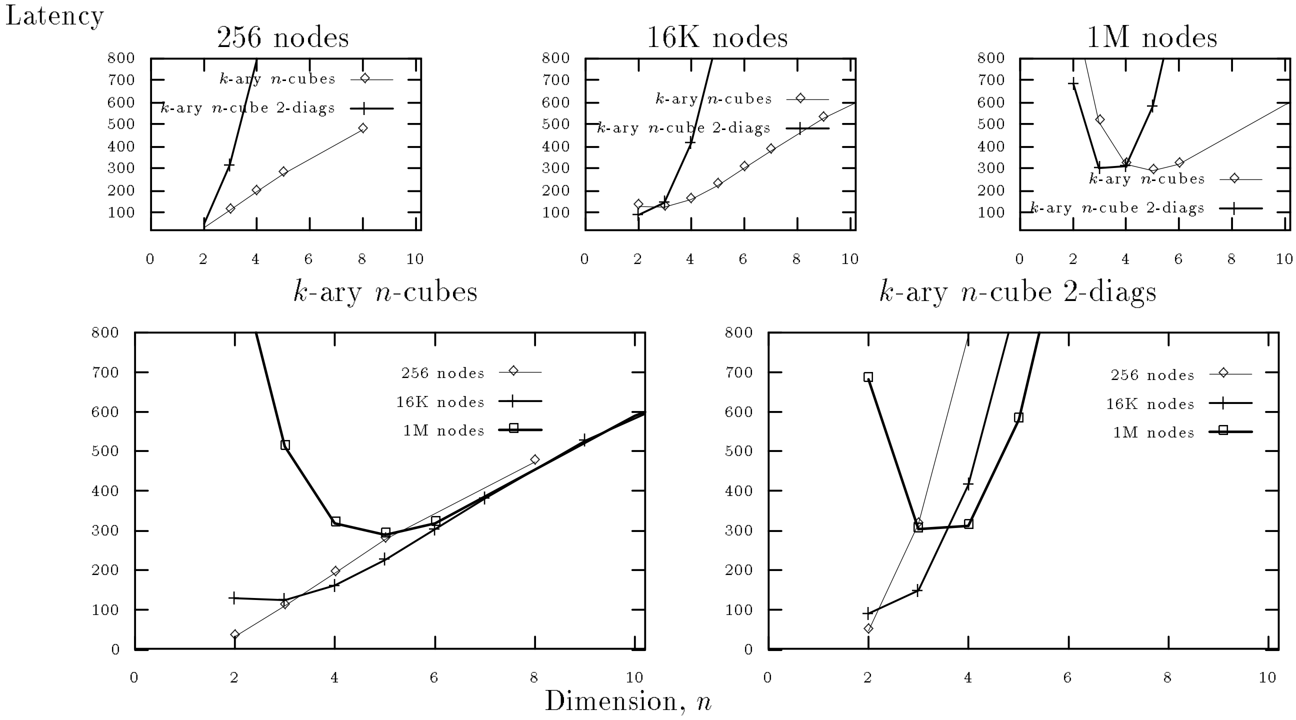


Figure 7: Latency versus dimension for 256, 16K, and 1M nodes: logarithmic wire delay.

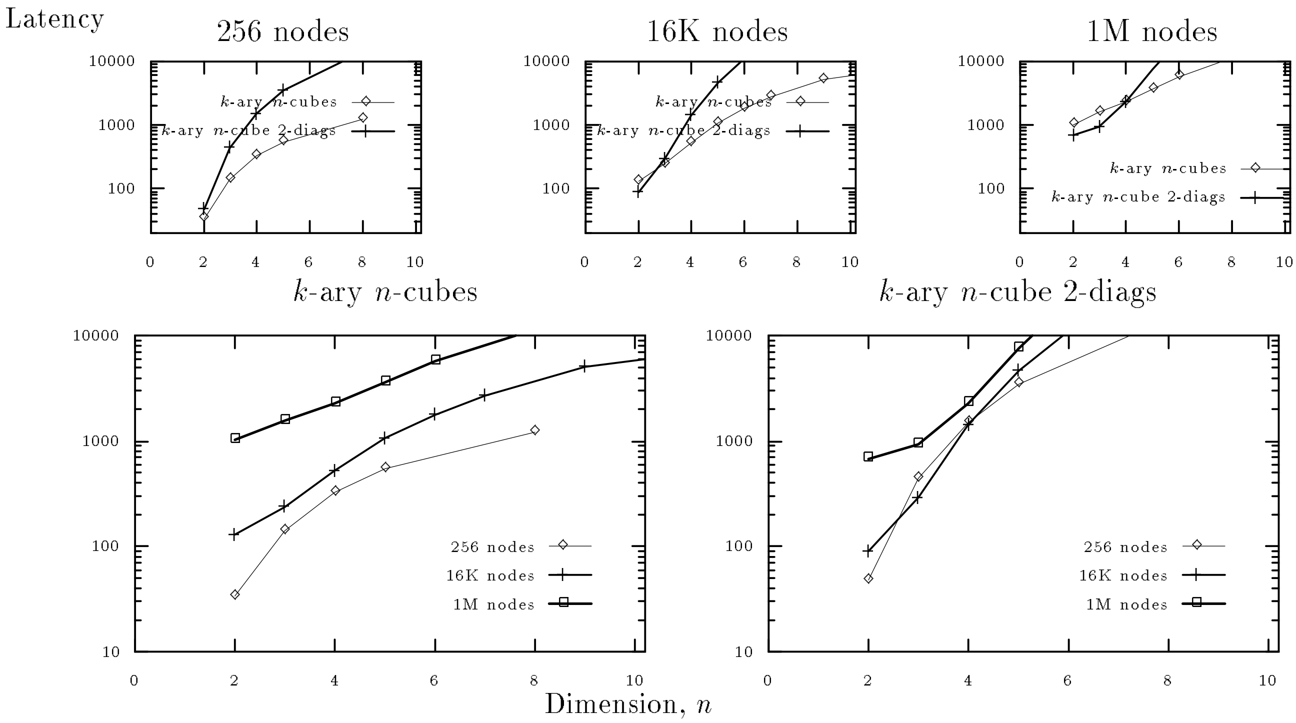


Figure 8: Latency versus dimension for 256, 16K, and 1M nodes: linear wire delay.

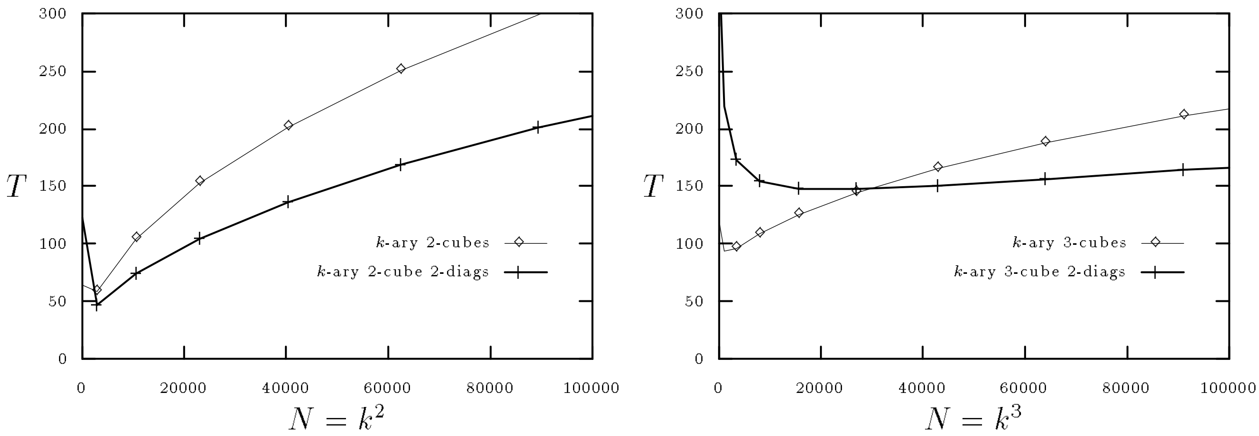


Figure 9: Two- and three-dimensional latency comparison: logarithmic wire delay.

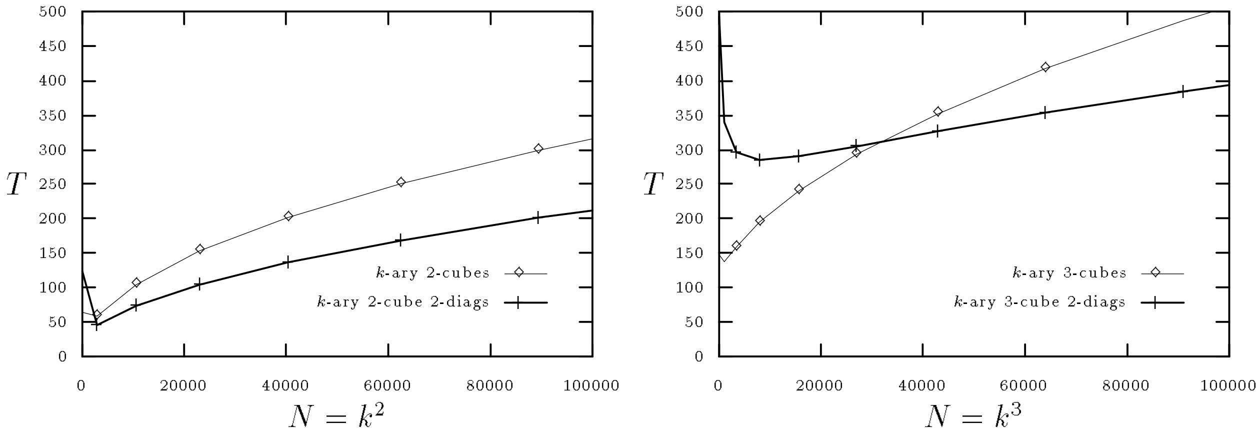


Figure 10: Two- and three-dimensional latency comparison: linear wire delay.

Figures 9 and 10 compare the latency results between k-ary n-cubes and k-ary n-cube 2-diags in two- and three-dimensional interconnection networks and illustrate the interest of increasing the connectivity on low dimensional networks. Diagonal connections improve the latency significantly when a certain number of nodes is reached. While this number is only 1089 (33^2) in two-dimensional networks, in three-dimensional networks improvements are obtained beyond 32768 (32^3) nodes. The implementation of diagonal links in three dimensions is therefore convenient for very large parallel systems. Also, for very large systems, the latency values are better in 3D under the assumption of logarithmic wire delay. This assumption is applicable when the wires are short, which is the case in low-dimensional networks.

Pin Density. Low-dimensional networks have the disadvantage of requiring a high pin density per node [5]. In a k-ary n-cube, each node has $2n$ channels, corresponding to n input

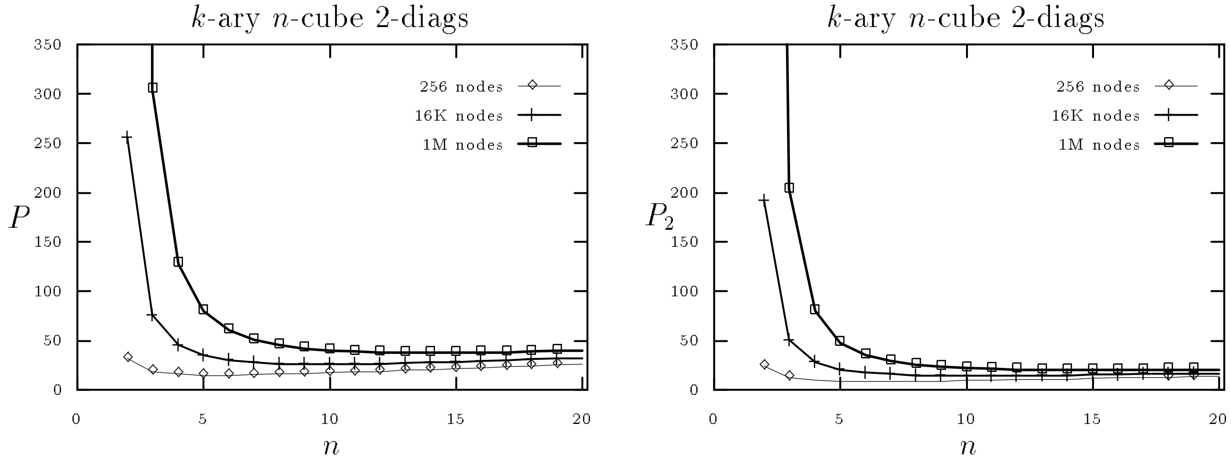


Figure 11: Pin density versus dimension for 256, 16K and 1M nodes.

and n output channels. The number of pins used for the exchange of data is therefore

$$P(n,k) = 2nW(n,k) = nk \; , \tag{10}$$

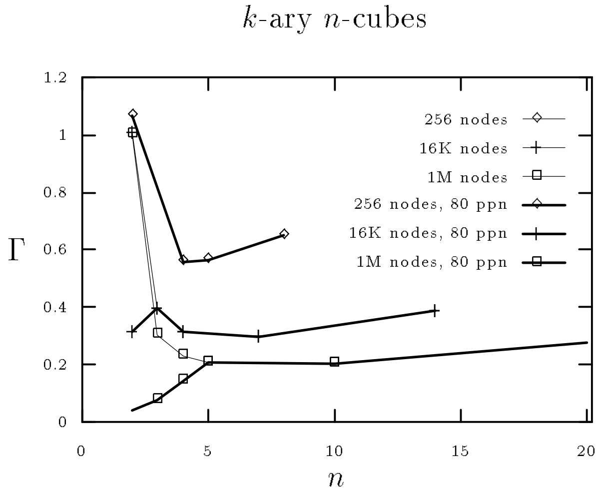
for each node, while in k-ary n-cube 2-diags, there are $2C_{n:2}$ extra channels in the diagonal directions, and the number of pins is therefore

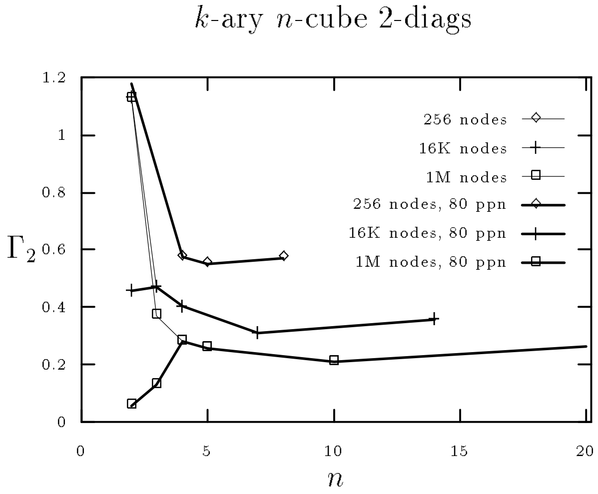
$$P_2(n,k) = 2(n + C_{n:2})W(k,n) = \left(1 + \frac{(n-1)}{2}\right) k \; . \tag{10$'$}$$

The corresponding curves are plotted in Figure 11, and show that the realization of a wire-efficient low-dimensional network is restricted by the pin density factor. Increasing the connectivity in low-dimensional networks moderates this drawback, but even if a 1024-ary 2-cube 2-diag (1Mega nodes) requires fewer pins per node than a 1024-ary 2-cube, 1536 pins and 2048 pins respectively, a processing node with such characteristics is still unrealizable. However, it should be noted that the physical restriction on the pin density does not have a great influence in the latency results when using the wormhole communication switching technique. In fact, recalling Equation 1 which defined this latency, the most influential parameters in it are the length of the longest wire and the mean distance between nodes in the network. On the other hand, the capacity of the network is determined directly by the number of pins, and this fact is taken into account in the following analysis.

Network throughput. While latency represents the average time to deliver a message, throughput measures the number of messages that are delivered through the network in a given time interval. The maximal throughput, called network capacity, is dependent on the number of links outgoing from each node and from the time of occupation of the network channels by a message. Formally, the network capacity per node is

$$\Gamma(k,n) \;\; \alpha \;\; \frac{nW(k,n)}{D(k,n)T_c(l)} \, . \tag{11}$$





256 nodes

p \ n	2	4	5	8
∞	1.07	0.56	0.57	0.65
80	**1.07**	**0.56**	**0.57**	**0.65**

p \ n	2	4	5	8
∞	1.18	0.57	0.55	0.57
80	**1.18**	**0.57**	**0.55**	**0.57**

16K nodes

p \ n	2	3	4	7	14
∞	1.01	0.40	0.31	0.30	0.39
80	**0.31**	**0.40**	**0.31**	**0.30**	**0.39**

p \ n	2	3	4	7	14
∞	1.13	0.47	0.40	0.31	0.36
80	**0.46**	**0.47**	**0.40**	**0.31**	**0.36**

1M nodes

p \ n	2	3	4	5	10	20
∞	1.00	0.30	0.23	0.21	0.20	0.28
80	**0.04**	**0.08**	**0.14**	**0.21**	**0.20**	**0.28**

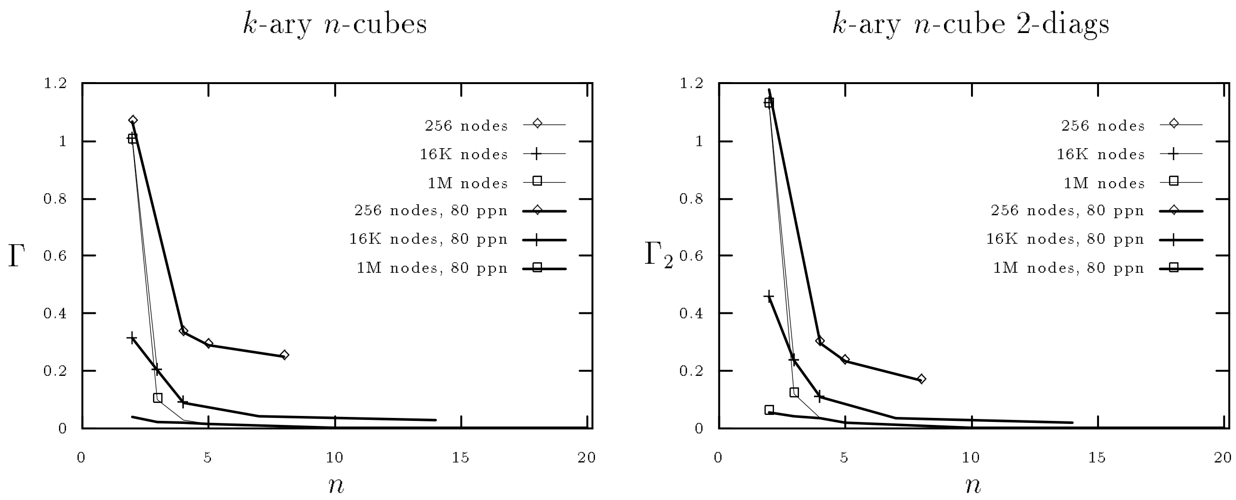
n	2	3	4	5	10	20
∞	1.13	0.37	0.28	0.26	0.21	0.26
80	**0.06**	**0.13**	**0.28**	**0.26**	**0.21**	**0.26**

Figure 12: Network Capacity per node in k-ary n-cubes and k-ary n-cube 2-diags: logarithmic wire delay. Curves in bold are obtained limiting the pins per node (ppn) to 80.

This expression differs from the one given in [4] by the term T_c, the channel cycle time. This allows one to account for the hardware constraints when the physical channel latency is dependent on the wire length. In fact, for constant delays, the two expressions of the capacity coincide.

Figures 12 and 13 illustrate the network capacity per node in k-ary n-cubes and k-ary n-cube 2-diags. Bold lines represent the impact of the limitations on the number of pins on it. Using the wire efficiency principle, the width of the channels is computed by using Equations 3 and 3' until the resulting number of pins is smaller than the constant p determined by hardware constraints, and by inverting Equations 10 and 10', fixing $P = p$, and solving with respect to W otherwise[2]. Note that although theoretically low-dimensional networks can sustain a high throughput, real throughput is in fact limited by pin density.

[2] In the figures, p, the number of pins for the communication links, is fixed to 80.



256 nodes

		n = 2	4	5	8
p	∞	1.07	0.33	0.29	0.25
	80	**1.07**	**0.33**	**0.29**	**0.25**

		n = 2	4	5	8
p	∞	1.18	0.30	0.23	0.17
	80	**1.18**	**0.30**	**0.23**	**0.17**

16K nodes

		n = 2	3	4	7	14
p	∞	1.01	0.20	0.09	0.04	0.03
	80	**0.31**	**0.20**	**0.09**	**0.04**	**0.03**

		n = 2	3	4	7	14
p	∞	1.13	0.24	0.11	0.04	0.02
	80	**0.46**	**0.24**	**0.11**	**0.04**	**0.02**

1M nodes

		n = 2	3	4	5	10	20
p	∞	1.00	0.10	0.03	0.02	0.005	0.004
	80	**0.04**	**0.03**	**0.02**	**0.02**	**0.005**	**0.004**

	n = 2	3	4	5	10	20
∞	1.13	0.12	0.04	0.02	0.004	0.003
80	**0.06**	**0.04**	**0.04**	**0.02**	**0.004**	**0.003**

Figure 13: Network Capacity per node in k-ary n-cubes and k-ary n-cube 2-diags: linear wire delay. Curves in bold are obtained limiting the pins per node to (ppn) 80.

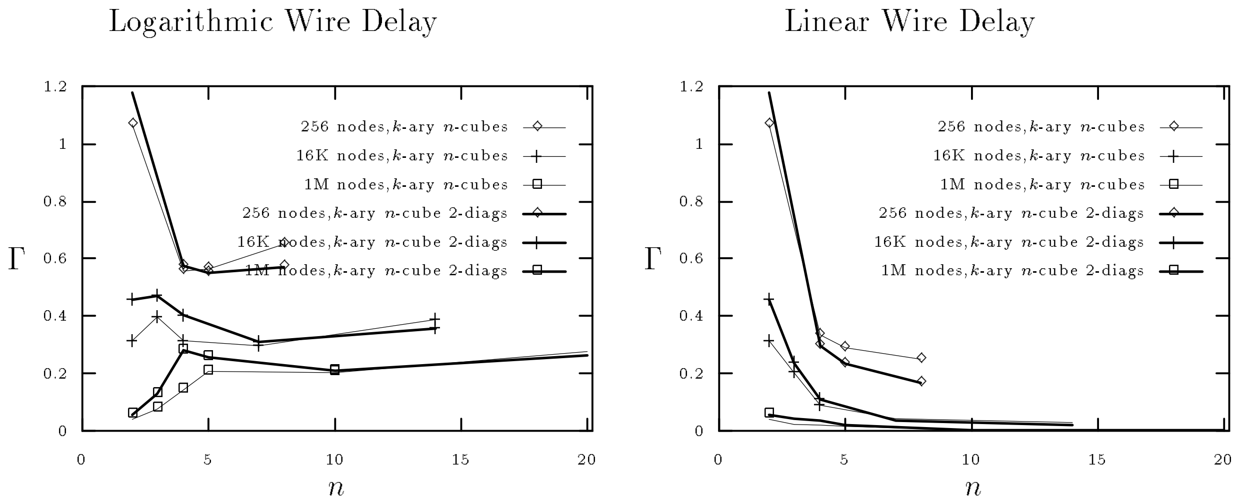


Figure 14: Comparing the Network Capacity per node in k-ary n-cubes (thin lines) and in k-ary n-cube 2-diags (bold lines) limiting the pins per node to 80: linear wire delay.

Figure 14 compares the network capacity per node, constrained by pin density, between k-ary n-cubes and k-ary n-cube 2-diags. The curves show that by increasing connectivity, k-ary n-cube 2-diags allow better throughput on low-dimensional interconnection networks. Again, this is particularly true when logarithmic wire delay is assumed.

4 The Routing Algorithm

Diagonal links in k-ary n-cube m-diags allow up to m cartesian hops to be performed in a single step. To ensure that the path generated is the shortest possible, the routing algorithm must take maximum advantage of hop grouping. The e-cube algorithm has traditionally been used to route in k-ary n-cubes [6]. If a message has to travel from node $P = (p_0, p_1, \ldots, p_{n-1})$ to node $Q = (q_0, q_1, \ldots, q_{n-1})$, then $h_i = (q_i - p_i) \bmod k$ is the number of times that the i-th cartesian direction has to be chosen ($i = 0, 1, \ldots, n$). The e-cube algorithm routes the message first h_{n-1} times in the direction $n-1$, so that node $(p_0, p_1, \ldots, q_{n-1})$ is reached, then routes it h_{n-2} times in the direction $n-2$ and so on until the node Q is reached. Keeping this strict ordering of the directions is unsatisfactory on k-ary n-cube m-diags, since direction grouping cannot be made independently of the position of the path endnodes, while this can be done in k-ary n-cubes. Such a choice would lead to the generation of paths that are not always the shortest path between the emitting node and the receiver, and the adaptation of the e-cube algorithm so that it generates the shortest path requires rethinking.

The shortest path between two nodes is generated only if the maximal number of diagonal steps are performed. Since the directions that have to be mostly employed, and therefore combined, are the ones corresponding to the maximal values of the h_i, at each step, the m greatest values of the h_i are decremented by one and the corresponding diagonal direction is chosen.

This allows the possibility of combining the directions to be maintained as long as possible. In other words, starting from P, the m greatest values of the n-tuple $(h_0, h_1, \ldots, h_{n-1})$ are selected and decremented, and the message with the new tuple is routed through the corresponding link. If the number of non-zero h_i is $m_0 < m$, then only the m_0 non-zero directions are combined together for the hop. This process is repeated until Q is reached.

It is interesting to observe that the drawback of the e-cube algorithm, the unbalanced load of the links when the message distribution is non-uniform [1], is reduced in the *sorted e-cube* algorithm described above, since no fixed direction is favoured by the routing strategy. Moreover, for k-ary n-cube n-diags, and therefore for hexagonal meshes, it is possible to derive an algorithm that generates a path having uniformly distributed directions from Computer Graphics straight line rasterization algorithms. Such an algorithm, which is similar to the graphical algorithm defined in [1], allows higher throughputs to be sustained in presence of hot spots.

5 Conclusions

This paper has introduced a new model for interconnection networks that generalizes k-ary n-cubes to k-ary n-cube m-diags that include wrapped hexagonal meshes and their n-dimensional extensions. A study of the theoretical performance of the different k-ary n-cube m-diag networks has been conducted, showing that significant improvements can be obtained by increasing the connectivity of low-dimensional networks. The paper focused on k-ary n-cube 2-diags and presented a detailed analysis for this case, since hardware complexity grows very rapidly in k-ary n-cube m-diags when m increases. Data presented in Section 3 depict the optimal organization of nodes in a network, and show that hexagonal meshes represent a serious alternative to existing planar topologies. Routing algorithms for k-ary n-cube m-diags have been presented and discussed, showing the relative ease of performing this task on such topologies.

The evolution from highly connected networks, e.g. binary hypercubes, to lowly connected topologies, e.g. meshes and tori, has been founded on pragmatic considerations, based on the embedding of the architecture in a physical layout. It is known that hexagonal meshes build an optimal disposition of a lattice of points in the plane [9], and therefore allow an optimal planar physical layout of the processing elements. The optimality of the disposition of the points in the plane explains intuitively the performance increase obtained for hexagonal meshes. The performance analysis presented in this paper confirms these assumptions and offers a model for comparing a wide range of topologies which takes into account the technological constraints.

Acknowledgment

The authors would like to thank M. Stumm of the University of Toronto, for his helpful comments in the revision process of the paper.

References

[1] Badouel, D., Wüthrich, C., and Fiume, E. Routing Strategies and Message Contention on Low-dimensional Interconnection Networks. Tech. Rep. 258, CSRI, University of Toronto, Dec. 1991.

[2] Beivide, R., Herrada, E., Balcázar, J., and Arruabarrena, A. Optimal Distance Networks of Low Degree for Parallel Computers. *IEEE Transactions on Computers 40*, 10 (Oct. 1991), 1109–1123.

[3] Chen, M.-S., Shin, K., and Kandlur, D. Addressing, Routing, and Broadcasting in Hexagonal Mesh Multiprocessors. *IEEE Transactions on Computers 39*, 1 (Jan. 1990), 10–18.

[4] Dally, W. Performance Analysis of k-ary n-cube Interconnection Networks. *IEEE Transactions on Computers 39*, 6 (June 1990).

[5] Dally, W. Express Cubes: Improving the Performance of k-ary n-cube Interconnection Networks. *IEEE Transactions on Computers 40*, 9 (Sept. 1991).

[6] Dally, W., and Seitz, C. Deadlock-Free Message Routing in Multiprocessor Interconnection Networks. *IEEE Transactions on Computers 36*, 5 (May 1987), 547–553.

[7] Dolter, J., Ramanathan, P., and Shin, K. Performance Analysis of Virtual Cut-Through Switching in HARTS: A Hexagonal Mesh Multicomputer. *IEEE Transactions on Computers 40*, 6 (June 1991), 669–680.

[8] Reed, D., and Grunwald, D. The Performance of Multicomputer Interconnection Networks. *IEEE Computer 20*, 6 (June 1987), 63–73.

[9] Rogers, C.A. *Packing and Covering.* Cambridge University Press, Cambridge, 1964.