COMMUNICATION LOCALITY IN HYPERMESHES AND TORI

MOHAMED OULD-KHAOUA and REZA SOTUDEH

Division of Electronics & Computer Engineering
University of Teesside
Middlesbrough, TS1 3BA, UK.

ABSTRACT The torus has been a popular topology for multicomputers due to its ease of implementation. Existing networks, such as the torus, are graph topologies, where a channel connects exactly two nodes. This paper argues that hypergraph topologies, where a channel connects any number of nodes, are potential candidates for future high-performance multicomputer networks. The paper assesses the support of a multi-dimensional hypergraph, referred to as the Distributed Crossbar Switch Hypermesh (DCSH), and the torus for communication locality. The results show that DCSH is a more general structure as it supports more efficiently a wide range of traffic patterns.

Keywords: Interconnection Topology, Graph, Hypermesh, Hypergraph, Torus, Latency, Performance Analysis.

I. INTRODUCTION

Multicomputers are commonly organised as a set of nodes that communicate over an interconnection network. Each node contains a processing element (PE), and switching element (SE) responsible for routing. The success of these systems is highly dependent on the efficiency of their underlying networks [3, 20, 23].

The network *topology*, which defines the way nodes are connected, has a great impact on performance. Implementation feasibility of topologies in various technologies has been researched extensively in the past [1, 2, 7, 22]. Feasibility studies are of crucial significance since any practical implementation will be subject to technology-specific limitations that may exclude certain desirable graphs. For example, implementation technology places bandwidth constraints on network channels, and these are an important factor in determining how well the theoretical properties of a given topology can be exploited.

On a single VLSI-chip, the network *wiring density* determines its overall system cost and performance. For example, Dally [7] has shown that for a fixed wiring density, the 2-dimensional torus (or torus for short) outperforms the high-dimensional hypercube because of its wider channels. His results have greatly influenced the design of current multicomputers. The low-diameter hypercube, that was used in the iPSC/2 [17] and Cosmic Cube [23], has been replaced by the high-diameter torus (and its variation the mesh) in recent machines, such as the J-Machine [16], iWarp [19], Stanford Dash Multiprocessor [13].

It has been argued however [1], that this move towards low-dimensional topologies is not fully justifiable. In practice, technology that allows an entire system to be accommodated on a single chip will not be achievable for many years, except where the nodes have very low transistor counts. Abraham & Padmanabhan [1] have identified *pin-out* (reflecting the limited chip pin-out count) as the constraint applicable to current multi-chip technology, and reported that under such constraint, it is the hypercube which exhibits the superior performance. Both studies, however, have ignored switching delays through the SEs despite the fact that these remain important in current technology [6, 22]. Furthermore, both have considered a uniform traffic pattern only, disfavouring the torus as it is inherently unsuitable for such communication patterns.

The torus has been popular because of its efficient support for local communication, and its perceived *modularity;* it can be expanded simply by adding nodes and channels without any change to the existing node structure. Since torus nodes can be used as elementary building blocks, they are potentially marketable components. Unfortunately, this modularity is at the expense of performance; for in a fixed-degree network, as the network size grows the channels must be increased in bandwidth to maintain the same performance [20]. The number of pins on a torus node must, therefore, be increased with the system size, a fact that is obvious in a topology like the hypercube whose degree increases as the number of nodes grows, but less apparent for the torus.

Common multicomputer network topologies, such as the torus and cube, can be formally modelled as ***graphs*** of the form $G=(V,E)$, defined over a set of vertices V and a set of edges E. Each vertex typically represents a node, and

0-7803-3529-5/96/$5.00 © 1996 IEEE

each edge between two vertices represents a channel connecting the two nodes. A fundamental constraint of the graph model is that each edge joins *exactly* two vertices. If a network channel can connect any number of nodes, a new class of topologies emerge. Using a graph-theoretical framework, members of this class can be modelled as *hypergraphs* [4], which are generalisations of the conventional graph in which individual edges are able to join an arbitrary number of vertices.

In previous work, we have shown that a particular class of regular multi-dimensional hypergraph topologies, known as the Distributed Crossbar Switch Hypermesh (DCSH), has several desirable topological properties that encompass those of the torus, mesh, and hypercube [14, 18]. This paper compares the DCSH to the most common graph topology, namely the torus, when implemented in VLSI and multiple-chip technology. Although many existing network evaluation studies assume the uniform reference model, it is not always true in practice as there are several real-world applications that exhibit non-uniform patterns [2, 10, 20]. Typical non-uniform traffic patterns include *communication locality* where the likelihood of communication to different nodes decreases with distance. This study assesses the support of the DCSH and torus for various degrees of communication locality. The comparison for the uniform case can be found in [18]. The analysis makes realistic assumptions, ignored by previous studies [1,7], such as including delays due to decision time and the use of pipelined-bit transmission to lower the effects of long wires.

The remainder of the paper is organised as follows. Section 2 introduces the hypermesh and the DCSH. Section 3 outlines queueing models for the DCSH and torus under both uniform and non-uniform traffic distributions, and compares their performance. Finally, Section 4 concludes this study.

II. DISTRIBUTED CROSSBAR SWITCH HYPERMESH (DCSH)

The *hypermesh* is a regular k-ary n-dimensional hypergraph which has $N = k^n$ nodes, and is a *Cartesian n-product* of a fundamental *cluster*, consisting of k nodes that are directly connected [4]. A network of this kind has extremely desirable properties [14, 18, 25, 26]. Firstly, their diameter grows much more slowly as the system size is scaled up, compared to most graph networks. Secondly, they can host applications that map naturally on tori, hypercubes, and binary trees. Thirdly, they can emulate all SIMD permutations of the Omega and inverse Omega multi-stage networks. Finally, they are effective at broadcast and multicast operations.

A typical example of the hypermesh is the spanning-bus hypercube [27], where nodes in a cluster are connected by means of a single shared-bus. Other hypermeshes are based on either crossbar switches [11, 26] or complete-connections [6]. However, all these implementations suffer from bandwidth limitations as the system size is scaled up [11].

The hypermesh structure is ideally suited for optical implementations given a sufficiently advanced transceiver technology bandwidth [9, 21, 25]. Nonetheless, optical systems also have problematical aspects. As integrated single-chip nodes become feasible, it will be necessary to develop opto-electronic transmitters and receivers, that are capable of operating at a rate of hundreds of GHz, and sufficiently small so as not to cause a substantial increase in the site area occupied by a node. Commercial technologies with such parameters appear to be some way off [15].

A k^n node DCSH is an n-dimensional hypermesh where the k nodes in a given cluster are connected by means of a *distributed crossbar switch*; the multiplexing/demultiplexing functions of the conventional "centralised" crossbar switch are performed in the SEs. Fig. 1 depicts the basic structure of a cluster. Every node possesses a uniquely owned channel that connects it to the other $(k-1)$ nodes in the cluster. At each of these k-1 destinations, there is a $(k-1)$-to-1 multiplexer with buffered inputs.

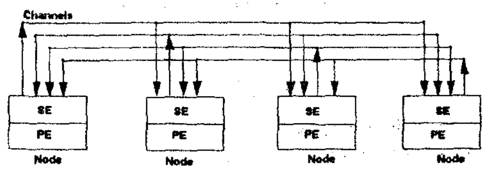


Fig. 1: A DCSH cluster.

Let dimensions be numbered n to 1. A node, v, can then be labelled by an $n \times 1$ address vector with v_i being the node's position in its dimension i cluster. Each node is connected to $n(k-1)$ other nodes with which it differs in only one address digit, i.e. $v = v_n \ldots v_{i+1} v_i v_{i-1} \ldots v_1$ is connected to $v' = v_n \ldots v_{i+1} v'_i v_{i-1} \ldots v_1$ for all $1 \le i \le n$, $1 \le v_i, v'_i \le k$ and $v_i \ne v'_i$.

The most interesting cases of the DCSH are those of low dimension (n=2 or 3), which not only have a low diameter, but map naturally into the physical space. Furthermore, low-dimensional DCSHs can exploit an efficient layered implementation scheme that alleviates the most critical bandwidth constraints that the torus, hypercube, and other hypermesh implementations suffer from [14]. Instead of aiming to integrate a node on a single chip, the DCSH implementation employ a functional partitioning to separate processing and switching functions into physical layers, thereby increasing the available wiring density and reducing the pin-out requirements.

III. ANALYSIS

Although the torus can have either uni-directional or bi-directional channels, we will concentrate on the latter case since it is more popular in current multicomputers. Detailed derivations of the models, presented below, can be found in [18].

The models assume that messages are transmitted between SEs using *wormhole routing* [24]. A message is broken into *flits* (a few bytes each) for transmission and flow control. The *header* flit, containing routing information, governs the route. As the header advances along a specified route, the remaining flits follow in a pipeline fashion. If the header encounters a busy channel, it is blocked until the channel is freed; the flow control within the network blocks the trailing flits. Wormhole routing has been widely used in multicomputers as it considerably reduces buffering requirements, and allows the implementation of simpler and faster SEs [6, 16].

Nodes generate traffic independently of each other at a rate which follows a Poisson process, with a mean of m messages/cycle. Messages are B flits long, each flit requiring one-cycle transmission time.

Deadlock during message routing is avoided by using the *virtual channel* algorithm, proposed by Dally & Seitz [8]. Restricted routing, where messages visit network dimensions in a pre-defined order, is a special case of this algorithm, and ensure deadlock-free routing in the DCSH. In the torus, however, in addition to restricted routing, each physical channel is divided into two virtual channels to avoid deadlock. Restricted routing has been popular in multicomputers [13, 16, 19, 24] because it requires a minimal number of virtual channels, and thus allows the design of faster SEs [6, 16]. Let dimensions in the DCSH and torus be numbered from 1 to n, with the destination node at the fictive dimension 0, and messages visit higher-numbered dimensions first.

The SE's decision time takes D_t cycles. Previous studies of k-ary n-cubes have ignored the effects of decision time on performance and stressed those of long wires. This is unrealistic given current and foreseeable technology. Delays through SEs due to decision time are still significant and much higher than wire delays. Delays due to long wires can be made less of an issue by using bit-pipeline transmission [22].

The following analysis uses the *sphere of locality* model (the conclusions drawn apply equally to the other models [20]). A k^n node network is divided into disjoint groups of $N_s = k^s$ nodes each. All nodes within a group share the same sphere of locality (composed of the nodes within that group). A message is destined for a node within the same sphere of locality as the source node (a *sphere message*) with a probability, β, and to a node in a different sphere (a *non-sphere message*) with probability (1-β). Destinations of sphere messages are equiprobable, as are destinations of non-sphere messages. if $\beta = (N_s - 1)/(N-1)$ the traditional uniform traffic model is obtained.

Let the channels be divided into two classes: sphere channels and non-sphere channels. Sphere channels are associated with the s lower dimensions while non-sphere channels are associated with the (n-s) higher dimensions. Sphere messages visit sphere channels only. Non-sphere messages, on the other hand, visit both sphere and non-sphere channels.

A. The DCSH Model

The average number of sphere and non-sphere channels visited by sphere and non-sphere messages respectively are given by

$$a_{sc} = s\frac{(k-1)}{k} \tag{1}$$

$$a_{nsc} = (n-s)\frac{(k-1)}{k}\frac{N}{(N-N_s)} \tag{2}$$

The factor $(N/N-N_s)$ accounts for the fact that non-sphere message have to be destined to a different sphere from that of the source.

Under light traffic, the mean latency as latency, L, in a k-ary n-dimensional DCSH can be approximated by

$$L = a_{sc}D_t + (1-\beta)a_{nsc}D_t + B \tag{3}$$

The first term in the above equation accounts for the average number of SEs that a header flit visits while the second is the transmission time of the data flits.

Under increased traffic however, message encounters blocking over network channels. All the dimensions are considered when determining the mean latency since the latency at dimension i $(1 \le i \le n)$ depends on dimensions j $(j < i)$. Latency is determined first at the destination (dimension 0), and then propagated back to tot the source (dimension n).

Since both sphere and non-sphere messages visit dimension i $(1 \le i \le s)$, the traffic rate (m_{sc}) at dimension i is given

$$m_{sc} = m \tag{4}$$

Only non-sphere messages visit dimension i $(s < i \le n)$, the traffic rate (m_{nsc})at dimension i is therefore given by

$$m_{nsc} = (1-\beta)m\frac{N}{N-N_s} \quad (5)$$

Since flits are serviced as soon as they arrive at the destination, the latency at dimension 0 is given by

$$L_0 = B \quad (6)$$

Let us now determine the message latency seen by entering dimension i $(1 \le i \le s)$ belonging to the sphere of locality. The probability that a message skips a dimension is $\alpha = 1/k$. Given that a message has either passed through skipped dimension i+1, the rates of traffic that either pass through or skip dimension i are (the first subscript is for dimension i+1 and the second for i)

$$m_{sc_{pp}} = (1-\alpha)^2 m_{sc} \quad (7)$$

$$m_{sc_{ps}} = m_{sc_{sp}} = \alpha(1-\alpha)m_{sc} \quad (8)$$

$$m_{sc_{ss}} = \alpha^2 m_{sc} \quad (9)$$

Only messages that pass through dimension i encounter blocking at multiplexer i. The increase in latency seen by a message at multiplexer i is found to be

$$L^M_{sc_i} = \frac{(k-2)(1-\alpha)m_{sc}L^2_{sc_i}}{2(k-1)} \quad (10)$$

The latency seen by a message entering dimension i $(1 \le i \le s)$, taking into account the various traffic components that pass and skip the dimension, can be written as

$$\begin{aligned} L_{sc_{i+1}} = {} & D_t + L_{sc_i} + (1-\alpha)L^M_{sc_i} + \\ & \alpha^3(1-\alpha)m_{sc}L^2_{nsc_i} + \\ & \alpha(1-\alpha)^3 m_{sc}(L_{sc_i} + L^M_{sc_i})^2 \end{aligned} \quad (11)$$

Similarly, for dimension i $(s < i \le n)$, equations 10 and 11 become

$$L^M_{nsc_i} = \frac{(k-2)(1-\alpha)m_{nsc}L^2_{nsc_i}}{2(k-1)} \quad (12)$$

$$\begin{aligned} L_{nsc_{i+1}} = {} & D_t + L_{nsc_i} + (1-\alpha)L^M_{nsc_i} + \\ & \alpha(1-\alpha)^3 m_{nsc}(L_{nsc_i} + L^M_{nsc_i})^2 \\ & + \alpha^3(1-\alpha)m_{nsc}L^2_{nsc_i} \end{aligned} \quad (13)$$

A sphere message sees a latency of L_{sc_s} while non-sphere one sees L_{nsc_n}. Assuming that the message service time at the source node follows an exponential distribution, with a mean $\beta L_{sc_s} + (1-\beta)L_{nsc_n}$, M/M/1 queueing theory [12] gives the mean latency as

$$L = \frac{[\beta L_{sc_s} + (1-\beta)L_{nsc_n}]}{1 - m[\beta L_{sc_s} + (1-\beta)L_{nsc_n}]} \quad (14)$$

B. The Torus Model

Although the torus can have either uni-directional or bi-directional channels, the discussion here will concentrate on the latter case since it is more popular in multicomputers.

For simplicity, the model presented here does not take into account the effects of multiplexing the virtual channels onto the physical channels, and therefore underestimates the mean message latency. Nonetheless, it is sufficient for our comparative study because it provides an upper bound to the torus performance. A more detailed model would be more complicated to manage. Furthermore, it has been found that the general conclusions drawn from this study using the torus model are identical to those provided through simulation.

The derivation which follows for wormhole latency in the torus is similar to that in Section 3.1 and, for brevity, only differences are shown. A sphere of locality in a k^n node torus is a s-dimensional torus with $k_s{}^s$ along a dimension. The average message distances for sphere and non-sphere messages are given by

$$a_{sc} = s\frac{(k_s-1)}{4} \quad (15)$$

$$a_{nsc} = (n-s)\frac{(k-1)}{4}\frac{N}{(N-N_s)} \quad (16)$$

The traffic rates (m_{sc} and m_{nsc}) arriving at a given direction at dimension i $(1 \le i \le s)$ and $(s < i \le n)$ respectively can be written as

$$m = \frac{m}{2}\frac{N}{(N-1)} \quad (17)$$

$$m_{nsc} = (1-\beta)\frac{m}{2}\frac{N}{N-N_s} \quad (18)$$

The traffic arriving at a direction at dimension i $(1 \le i \le s)$ is composed of eight streams, four streams coming from each direction of the previous dimension. Given that the probability that a message skips dimension i $(1 \le i \le s)$ is $\alpha_{sc} = 1/k_s$, the latency seen by a message entering dimension i through a given direction is found to be

$$L_{sc_{i+1}} = D_t + L_{sc_i} + (1-a_{sc})L^R_{sc_i} + a_{sc}^2(1+2a_{sc}-2a_{sc}^2)m_{sc}\frac{L_{sc_i}^2}{2} + (1-a_{sc})^3(1+3a_{sc})m_{sc}\frac{(L_{sc_i}+L^R_{sc_i})^2}{2} \quad (19)$$

The routing latency seen by a message that passes through dimension i is found to be

$$L^R_{sc_i} = L_{sc_{i0}} + \frac{m_{sc_C}}{2}L^2_{sc_{i0}} - L_{sc_i} \quad (20)$$

$$m_{sc_C} = \frac{(k_s-2)}{4}m_{sc} \quad (22)$$

$$L_{sc_{i0}} = \frac{1-\sqrt{1-2m_{sc_C}((k_s-2)/4)D_t + L_{sc_i})}}{m_{sc_C}} \quad (21)$$

Given that a non-sphere message skips dimension i $(s < i \le n)$ with probability $\alpha = 1/k$, equations 20, 22, 23 for dimension i $(s < i \le n)$ can be written as

$$L_{nsc_{i+1}} = D_t + L_{nsc_i} + (1-a)L^R_{nsc_i} + a^2(1+2a-2a^2)m_{nsc}\frac{L_{nsc_i}^2}{2} + (1-a)^3(1+3a)m_{nsc}\frac{(L_{nsc_i}+L^R_{nsc_i})^2}{2} \quad (24)$$

$$m_{nsc_C} = \frac{(k-2)}{4}m_{nsc} \quad (25)$$

$$L_{nsc_{i0}} = \frac{1-\sqrt{1-2m_{nsc_C}((k-2)/4)D_t + L_{nsc_i})}}{m_{nsc_C}} \quad (26)$$

C. Results and Discussions

Due to the limited channel width, a message is broken into *phits* (i.e., channel words), each of which is transferred in one cycle. If the channel width is W bits, a message length M bits is broken up into $B = M/W$ phits, each containing W bits. A flit, which is a unit of transfer for wormhole routing, can be composed of one or more phits. The results presented below assume that a flit is equal to one phit.

This section compares the 2-dimensional DCSH to the torus with fixed network size N and implementation cost. In a pure VLSI implementation, the *bisection width* [7], that is the number of wires that cross the middle of the network, is an approximate measure of wiring density. Assuming that a network is implemented in the 2-dimensional space with $\sqrt{N}$ nodes in a dimension, the bisection widths of the DCSH and torus, with channel widths of W_{DCSH} and W_{Torus} respectively, are found to be

$$\mathcal{B}_{DCSH} = k^2W_{DCSH} = NW_{DCSH} \quad (27)$$

$$\mathcal{B}_{Torus} = \frac{4k^2}{k} = \frac{4N}{k}W_{Torus} \quad (28)$$

In multi-chip implementations, where a node is implemented on a single chip, the ***node pin-out*** [1], i.e. node degree × channel width, is a more suitable cost metric. The node pin-out of the DCSH and torus can be written as

$$\mathrm{P}_{DCSH} = 2kW_{DCSH} \quad (29)$$

$$\mathrm{P}_{Torus} = 8W_{Torus} \quad (30)$$

If the bisection width is fixed, the channel width of the torus in terms of that of the DCSH is given by

$$W_{Torus} = \left\lceil \frac{k}{4} \right\rceil W_{DCSH} \quad (31)$$

In multiple-chip technology, assuming constant node pin-out, the same channel width relationship as equation 31 is obtained. Therefore, under both constant bisection width and node pin-out constraints, the torus has $k/4$ wider channels than the DCSH.

To investigate whether the torus can take advantage of its wider channels to outperform the DCSH, let the network size be $N = 1024$ nodes, which is a moderately large system. Furthermore, let us set W_{DCSH} to 1 (DCSH) phit (or phit for short), and consider a message length of M = 64 (DCSH) phits.

Fig. 2 depicts latency results that reveal the relative performance of DCSH and torus when traffic patterns contains an element of locality. The decision time is set as low as 1 cycle, which is an optimistic figure given current implementation technology [6, 16, 22]. The x-axis in the figure represents the traffic rate; the rate at which a node injects messages into the network in a cycle. The y-axis gives the mean message latency to cross from the source to destination.

Performance improves when communication locality is exploited [10]. This is because sphere messages go through fewer channels, and thus go through fewer blocking stages, and non-sphere messages experience less blocking because the traffic at non-sphere channels is reduced. Due to its interconnection structure, the torus relies on local communication to achieve good performance. This section investigates to what extent the torus needs to exploit locality in order to take advantage if its wider channels.

Figs. 2-*a*, *b*, and *c* show latency for three degrees of locality; high (β=75%), moderate (β=50%), and low (β=20%). The sphere size in the DCSH is one cluster (i.e., the whole row); smaller sphere sizes than one cluster does other hand, a sphere contains eight nodes along the row. This means that a local message travels, on average, two hops only in either direction.

When applications generate traffic with strong degrees of locality, the torus uses its wider channels effectively, and provides better performance than the DCSH. As the degree of locality decreases, the torus performance degrades; when 20% of messages are local, the torus latency is higher at moderate to heavy traffic loads.

Figs. 3-*a* and *b* reveal that when the decision time is increased to 2 cycles, the torus is outperformed by the DCSH at early traffic loads, even when communication locality is as high as 75%. The torus is extremely sensitive to the effects of decision time, especially as the sphere size increases or the locality decreases since the message distance increases. Therefore, decision time in torus-based multicomputers has to be carefully engineered, and kept as low as possible to take advantage of any locality exhibited by the application.

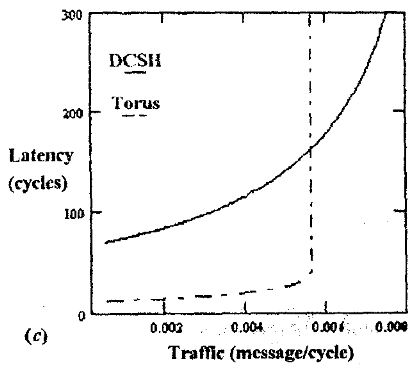
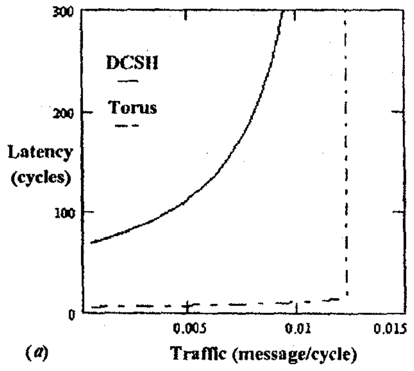
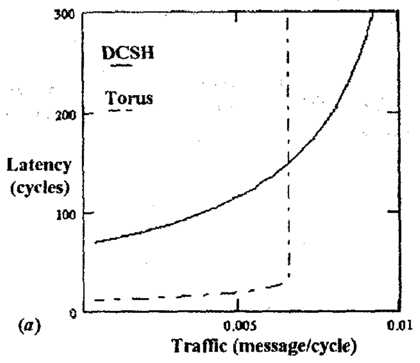
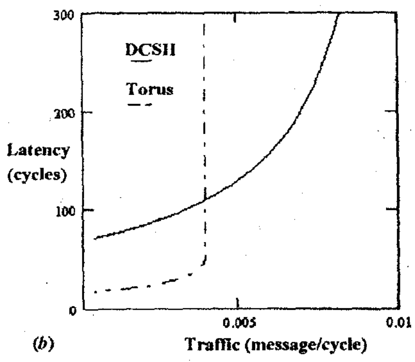


Fig. 2: Performance of the DCSH & Torus.
a) Locality=75%, *b*) Locality=50%.
c) Locality=20%.
Decision time=1 cycle.







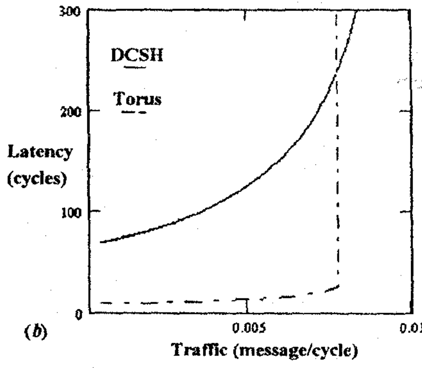


Fig. 3: Performance of the DCSH & Torus.
a) Locality=75%. *b*) Locality=50%.
Decision time=2 cycles..

IV. CONCLUSIONS

When compared to the torus, the DCSH is functionally a more general topology as it supports more efficiently a wider range of traffic patterns. Although the torus has wider channels than the DCSH, for equal implementation cost in VLSI and multiple-chip technology, its high sensitivity to decision time offsets this advantage. The performance of the torus degrades sharply even with optimistic figures for decision time. This is because a message in the torus visits, on average, a greater number of nodes to reach its destination, and thus encounters higher blocking in the network.

The results, presented here, have revealed the decision time in the torus must be kept as low as possible (at 1 cycle) to take any performance advantage from exploiting strong communication locality. Moderate and low degrees (<=50%) of locality do not favour the torus. When the decision time is increased to 2 cycles, which is still a realistic figure given current technology, the torus has no performance advantages over the DCSH even when locality is as strong as 75%.

REFERENCES

[1] S. Abraham & K. Padmanabhan, Performance of multicomputer networks under pin-out constraints, *JPDC* 12, 1992, 237-248.

[2] A. Agrawal, Limits on interconnection network performance, *IEEE Trans. Par. & Dist. Syst.* 2, 1991, 398-412.

[3] W.C. Athas & C.L. Seitz, Multicomputers: Message-passing concurrent computers, *IEEE Computer* 21(8), 1988, 9-24.

[4] C. Berge, *Graphs and hypergraphs*, North-Holland, 1977.

[5] L.M. Bhuyan & P.D. Agarwal, Generalised hypercube and hyperbus structures for a computer network, *IEEE Trans. Comp.* C33 (4), 1984, 323-333.

[6] A.A. Chien, A Cost and performance model for *k*-ary *n*-cubes wormhole routers, *Proc. Hot Interconnects Workshop*, August 1993.

[7] W.J. Dally, Performance analysis of *k*-ary *n*-cubes interconnection networks, *IEEE Trans. Comp.* C39(6), 1990, 775-785.

[8] W.J. Dally, Deadlock-free message routing in multiprocessor interconnection Networks, *IEEE Trans. Comp.* C36(5), 1987, 547-553.

[9] P.W. Dowd, High-performance interprocessor communication through optical wavelength division multiple access channels, *Int. Symp. Comp. Arch.*, 1991, 96-105.

[10] G.C. Fox *et al*, Solving problems on concurrent processors, Prentice Hall, 1988.

[11] W.K. Giloi & S. Montenegro, Choosing the interconnect of distributed-memory systems by cost and blocking behaviour, *IEEE Parallel Processing Symp.*, 1991, 438-444.

[12] L. Kleinrock, Queueing Systems 1, John Wiley, New York, 1975.

[13] D. Lenoski, J. Laudon, K. Gharacholoo, A. Gupta, J. Hennessy, M. Horowitz, and M. Lam, The Stanford DASH Multiprocessor, *IEEE Computer*, March 1992, 63-79.

[14] L. Mackenzie *et al*, *COBRA:* A High-Performance Interconnection Network for Large Multicomputers, Tech-Rep. 119/R19, Comp. Sci. Dept., Glasgow Univ., 1991.

[15] L.M. Ni & D.K. Panda, Sea of interconnection networks: What's your choice, *A report of the ICPP '95 Pannel, Proc. ICPP*, 1994.

[16] M. Noakes & W.J. Dally, System design of the J-machine, *Advanced Research in VLSI*, MIT Press, 1990, 179-192

[17] S.F. Nugent, The iPSC/2 direct-connect communication technology, *Proc. Conf. Hypercube Concurrent Computers & Applications*, 1989, 51-60

[18] M. Ould-Khaoua, Hypergraph-based interconnection networks for large multicomputers, Ph.D. dissertation, Comp. Sci. Dept., Glasgow University, 1994.

[19] C. Peterson *et al*, iWarp: a 100-MOPS VLIW microprocessor for multicomputers, *IEEE Micro* 11(13), 1991, 26-37.

[20] D.A. Reed & R.M. Fujimoto, *Multicomputer networks: Message-based parallel processing*,

[21] A.B. Ruighaver, A decoupled multicomputer with full interconnection, *Proc. CONPAR-VIPP 92*, Lyon, 1992, 250-257.

[22] S.L. Scott & J.R. Goodman, The impact of pipelined channel on *k*-ary *n*-cube networks, *IEEE Trans. Par. & Dist. Syst.* 5(1), 1994, 2-16.

[23] C.L. Seitz, The Cosmic Cube, *Comm. ACM* 28, Jan. 1985, 22-33.

[24] C.L. Seitz *et al*, The hypercube communication chip, Dept. Comp. Sci., CalTech, Display File5182:DF:85, 1985.

[25] T. Szymanski, Hyper-meshes: Optical Interconnection Networks for Parallel Processing, *JPDC* 26, Jan. 1995.

[26] N. Tanabe *et al*, Base-*m* *n*-cube: high performance interconnection networks for highly parallel computer PRODIGY, *Proc. ICPP*, 1991, 509-516.

[27] L.D. Wittie, Communication structures for large networks of microcomputers, *IEEE Trans. Comp.* C30 (4), 1981, 264-273.