IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, May 9-10, 1991

HYPERCYCLES: A STATUS REPORT

Nikitas J. Dimopoulos , R. Sivakumar, V. Dimakopoulos, M. Chowdhury and Don Radvan
Electrical and Computer Engineering Department
University of Victoria
Victoria, B.C. Canada

ABSTRACT†

In this work, we present the Hypercycles, a class of multidimensional graphs, which are generalizations of the n-cube. These graphs are obtained by allowing each dimension to incorporate more than two elements and a cyclic interconnection strategy. Hypercycles, offer simple routing, and the ability, given a fixed degree, to chose among a number of alternative size graphs. These graphs can be used in the design of interconnection networks for distributed systems tailored specifically to the topology of a particular application. We are also presenting a back-track-to-the-origin-and-retry routing, whereupon paths that block at intermediate nodes are abandoned, and a new attempt is made. Intermediate nodes are chosen at random at each point from among the ones that form the shortest paths from a source to a destination. Simulation results that establish the performance of a variety of configurations are presented. In addition our initial attempt of constructing a Hypercycle based router is discussed.

1.0 Introduction

Message passing concurrent computers such as the Hypercube[11, 17], Cosmic Cube[15], MAX[12, 13], consist of several processing nodes that interact via messages exchanged over communication channels linking these nodes into one functional entity.

There are many ways of interconnecting the computational nodes, the Hypercube, Cosmic Cube, and the Connection Machine[18] having adopted a regular interconnection pattern corresponding to a binary n-dimensional cube, while MAX adopts a less structured, yet unspecified topology.

Several recent studies attempt extensions and generalizations of the basic tenets of the n-cube. Broder et. al. [4] have proposed product graphs[14] of small "basic" graphs. Their prime concern is to synthesize fault tolerant networks with a given degree of coverage. In these multidimensional graphs, they define a single route from a source to a destination, as the product of routes in each of the constituent dimensions. Routing is exhausted in each dimension before another dimension is considered. Bhunyan and Agrawal [3] have introduced the generalized hypercubes (GHC) which are also graph products of fully connected "basic" graphs. The mixed radix system [2] is used to express the properties of these graphs and their routing. Wittie [19] gives a good overview and comparison of several interconnection networks including the spanning bus and dual bus hypercubes. These are essentially binary n-cubes with broadcast busses connecting the processors in each dimension.

The advantages of having a regularly structured interconnection are many-fold, and they have been proven time and again in their being incorporated in many recent designs [6,11,12,13,15,17,18]. In these structures, easy deadlock-free routing [7] can be accomplished by locally computing each successive intermediate node -for a path that originates at a source node and terminates at a destination node- as a function of the current position and the desired destination. Many regular problems (such as the ones found in image processing, physics etc.) have been mapped on such regular structures, and run on the corresponding machines exhibiting significant speedups.

In contrast, embedded real-time applications [12, 13], tend to exhibit variable structures that do not necessarily map optimally to an n-cube. In addition, since the size of a binary n-cube is given as 2^n , it means that a particular configuration cannot be expanded but in predefined quantum steps. For example, if a given embedded application requires a system comprised of 9 nodes, the next larger n-cube with 16 nodes must be chosen. This constitutes a significant increase in resource allocation, especially in power-mass limited environments.

Hypercycles[13] can be considered as products of "basic" graphs that allow, as compared to the Generalized Hypercubes (GHC) [3], a richer set of component "basic" graphs ranging in complexity from simple rings to the fully connected ones used in the GHC. Also, contrary to Broder et al [4], we define the component graphs and provide analytical expressions for routing, our aim being twofold:

(a) To provide computer interconnection networks that match the node requirements of a given embedded system.
(b) To increase throughput of a given network by providing routing expressions that can be computed analytically (and hence are candidates for VLSI implementation) and which provide a maximum number of alternate paths from a source to a destination. The existence of alternate paths guarantees that a message will not be blocked waiting for its single route to be freed, but it would in turn search for the availability of alternate paths. This strategy also provides for fault protection, since a faulty path can be marked permanently busy, and thus messages can be routed around it..

The Hypercycles, being regular graphs, retain the advantages of easy routing and regularity. Yet, since we are dealing with a class, rather than isolated graphs, we have the flexibility of adopting any particular graph (from the class) that closely matches the requirements of a given application.

This work is divided into these parts. Section 2.0 introduces the Mixed Radix System, Section 3.0 presents some basic graph terminology and notation, while Section 4.0 introduces the Hypercycles and discusses their properties. Sections 5.0 and 6.0 discuss routing and evaluate the performance of several example Hypercycles.

2.0 Mixed Radix Number System

The mixed radix representation [2], is a positional number representation, and it is a generalization of the the standard b-base representation, in that it allows each position to follow its own base independently of the other.

Given a decimal number M factored into r factors as $M = m_1 \times m_2 \times m_3 . \times \cdots \times m_r$ then any number $0 \le X \le M\text{-}1$ can be represented as

$(X)_{m_1 m_2 \ldots m_r} = x_1 x_2 \ldots x_r \mid_{m_1 m_2 \ldots m_r}$ where $0 \le x_i \le (m_i - 1)$; $i = 1,2,\ldots,r$ and the x_i's are chosen so that $X = \sum_{i=1}^{r} x_i w_i$ where $w_i = \dfrac{M}{m_1 m_2 \cdots m_i}$

3.0 Hypercycles.

An r- dimensional Hypercycle, is the following regular undirected graph: $\mathcal{G}_{\mathbf{m}}^{\rho} = \left\{ \mathcal{N}_{\mathbf{m}}^{\rho} . \mathcal{E}_{\mathbf{m}}^{\rho} \right\}$ where $\mathcal{N}_{\mathbf{m}}^{\rho}$ is the set of nodes

† This work has been supported by the Natural Sciences and Engineering Research Council Canada, under grant #OGP0001337 and by IRIS

$\mathcal{E}^{\rho}_{\boldsymbol{m}}$ the set of edges, $\boldsymbol{m} = m_1, m_2, m_3, \ldots, m_r$ a mixed radix, $\rho = \rho_1, \rho_2, \ldots, \rho_r$; $\rho_l \le m_l / 2$ the connectivity vector, determining the connectivity in each dimension which ranges from a ring ($\rho_l = 1$) to fully connected ($\rho_l = \lfloor m_l / 2 \rfloor$). $\mathcal{N}^{\rho}_{\boldsymbol{m}} = \{0, 1, 2, \ldots, M-1\}$. Given $\alpha, \beta \in \mathcal{N}^{\rho}_{\boldsymbol{m}}$ then $(\alpha, \beta) \in \mathcal{E}^{\rho}_{\boldsymbol{m}}$ if and only if there exists $1 \le j \le r$ such that $\beta_j = (\alpha_j \pm \xi_j) \,\mathbf{mod}\, m_j$ with $1 \le \xi_j \le \rho_j$ and $\alpha_l = \beta_l$; $l \ne j$

Hypercycles, have degrees [8] $d = \sum_{l=1}^{r} f(m_l, \rho_l)$ where

$$f(m_l, \rho_l) = \begin{cases} 2\rho_l & \text{if} \quad 2\rho_l < m_l \\ m_l - 1 & \text{if} \quad 2\rho_l = m_l \end{cases}$$

and diameter k ¶.

$$k = \sum_{l=1}^{r} \left\lceil \frac{\lfloor m_l / 2 \rfloor}{\rho_l} \right\rceil$$

The n-cube is a Hypercycle, with $M = 2 \times 2 \times \cdots \times 2 = 2^n$ and $\rho = 1, 1, 1, \ldots, 1$.

3.1 Routing

Hypercycles, have routing properties that are similar to those of the n-cube. Given nodes $(\alpha)_{m_1 m_2 \ldots m_l \ldots m_r} = \alpha_1 \alpha_2 \ldots \alpha_l \ldots \alpha_r$ and $(\alpha^*)_{m_1 m_2 \ldots m_l \ldots m_r} = \alpha_1 \alpha_2 \ldots \xi \ldots \alpha_r$, a walk, from node α to node α^*, can be constructed as follows:
$\alpha_1 \alpha_2 \ldots \alpha_l \ldots \alpha_r$, $\alpha_1 \alpha_2 \ldots \xi_1 \ldots \alpha_r$, $\alpha_1 \alpha_2 \ldots \xi_2 \ldots \alpha_r$, ..., $\alpha_1 \alpha_2 \ldots \xi \ldots \alpha_r$, such that§

$$\xi_{j_i+1} = \begin{cases} \left(\xi_{j_i} + \rho_i\right) \bmod m_i & \text{if } \left[\left(\xi - \xi_{j_i}\right) \bmod m_i = \left|\xi_{j_i}, \xi\right|\right] > \rho_i & \text{(a)} \\ \left(\xi_{j_i} + \left|\xi_{j_i}, \xi\right| \bmod \rho_i\right) \bmod m_i & \text{if } \left[\left(\xi - \xi_{j_i}\right) \bmod m_i = \left|\xi_{j_i}, \xi\right|\right] > \rho_i \text{ and } \left|\xi_{j_i}, \xi\right| \bmod \rho_i \ne 0 & \text{(b)} \\ \left(\xi_{j_i} - \rho_i\right) \bmod m_i & \text{if } \left[\left(\xi_{j_i} - \xi\right) \bmod m_i = \left|\xi_{j_i}, \xi\right|\right] > \rho_i & \text{(c)} \\ \left(\xi_{j_i} - \left|\xi_{j_i}, \xi\right| \bmod \rho_i\right) \bmod m_i & \text{if } \left[\left(\xi_{j_i} - \xi\right) \bmod m_i = \left|\xi_{j_i}, \xi\right|\right] > \rho_i \text{ and } \left|\xi_{j_i}, \xi\right| \bmod \rho_i \ne 0 & \text{(d)} \\ \xi & \text{if } \left|\xi_{j_i}, \xi\right| \le \rho_i & \text{(e)} \end{cases}$$

$$\xi_0 = \alpha_l \qquad \xi_{max} = \xi \qquad \text{Eqn. 3.1.1}$$

Equation 3.1.1 defines all the minimum-length paths from a source to a destination in a single dimension. Parts (a), and (c) constitute a greedy strategy where the maximum step towards the destination is taken. Parts (b) and (d) form alternate paths by allowing the step described in part (e) to be taken earlier. Observe that there is only one step of length smaller than the maximum, and when it is taken it is guaranteed that the remaining steps will be maximal. This is because

$$\left|\left(\xi_{j_i} \pm \left|\xi_{j_i}, \xi\right| \bmod \rho_i\right) \bmod m_i, \xi\right| \bmod \rho_i = 0$$

Given an origin $(\alpha)_{m_1 m_2 \ldots m_r} = \alpha_1 \alpha_2 \ldots \alpha_r$ and a destination $(\beta)_{m_1 m_2 \ldots m_r} = \beta_1 \beta_2 \ldots \beta_r$ then distinct walks of minimum length that connect them are constructed according by sequentially modifying the source address, each time substituting a source digit by an intermediate walk digit determined according to equation 3.1.3, until the destination is formed. The following walk connects **source** to **destination.**
source $= \alpha_1 \alpha_2 \alpha_3 \ldots \alpha_r$; $\alpha_1 \xi_1 \alpha_3 \ldots \alpha_r$; $\alpha_1 \xi_1 \psi_1 \ldots \alpha_r$; $\alpha_1 \xi_2 \psi_1 \ldots \alpha_r$; $\alpha_1 \xi_2 \psi_2 \ldots \alpha_r$; ...; $\alpha_1 \xi_2 \beta_3 \ldots \alpha_r$;...; $\beta_1 \beta_2 \beta_3 \ldots \beta_r$ = **destination**

If only the greedy strategy is followed, it results to a total of ‡

$$I = \binom{q}{q_1, q_2, \ldots, q_r} = \frac{q!}{q_1!\, q_2! \cdots q_r!}$$

paths of minimum length that connect them.
$\mathbf{dis}(\alpha, \beta) = q = \sum_{l=1}^{r} q_l$, is the distance between the origin and the destination. We shall call the so formed paths *greedy paths*.

Figure 1a., gives an example of two distinct walks of equal minimum length that connect a source to a destination, for a Hypercycle.

4.0 Deadlock Avoidance in Routing.

In section 3.1, we presented a method that establishes at least one path of minimum length from a source to a destination node. In this part, we are concerned with optimally choosing one of the paths. Routing must be efficient and deadlock free. Deadlock occurs when resources (in this case node to node communication segments) are allocated so that the completion of a partial path requires a segment already allocated to a different partial path which in turn waits for a segment in the first partial path. It is obvious that no messages can propagate over the deadlocked paths, and the only remedy is to break the already established and deadlocked partial paths and try again.

Deadlock may occur easily in cases where the segments that form the paths are chosen at random. Certain routing algorithms (e.g. virtual channels, e-cube routing [7]) prevent deadlocks by ordering the resources (channels) to be allocated. Thus a lower order resource cannot be committed if a needed higher order resource cannot be obtained. The disadvantage of this approach in an interconnection network is that it limits the number of paths connecting a source to a destination to exactly one, even though several alternate free paths may exist at a particular moment. We are proposing to adopt a strategy where deadlocks are avoided by requiring a blocked partial path to backtrack to its origin and retry.

5.0 Backtrack-to-the-origin-and-retry routing

For Hypercycle-based interconnection networks, because of the existence of cycles in each dimension, the use of an e-cube type routing that prevents deadlocks, is impractical. We are proposing instead a deadlock avoiding routing strategy. According to our backtrack-to-the-origin-and-retry routing we identify, at each node, all nodes that can be used for the continuation of the path. For all such identified nodes, we also identify the corresponding ports that can be used in order to continue the path. Since several paths may be forming in parallel, some of these ports may already be allocated to some other path. After excluding all the allocated ports, we select one of the remaining free ports at random. The subsequent link in the path is established is then established through the selected port, and the procedure repeats itself until the destination is reached, or no free ports could be found. If no free ports are to be found at an intermediate node in the path, then a break is returned to the origin (through the already established partial path to the blocking node), the partial path is dissolved, and a new attempt for the creation of the required path is initiated. This routing strategy avoids deadlocks through backtracking, and also guarantees that the formed path will be of a minimum length, since each subsequent link is selected according to equation 3.1.1. The backtrack-to-the-origin-and-retry routing is a type of two-phase locking [16], where

¶ The function $\lfloor x \rfloor$ denotes the largest integer smaller than or equal to x, and $\lceil y \rceil$ denotes the smallest integer larger than or equal to y.
§ We define $|a, b| = \mathbf{min}((a-b)\,\mathbf{mod}\,m_l, (b-a)\,\mathbf{mod}\,m_l)$

‡ For the definition of a multinomial number, see [1] pp 32.

as resources we consider the various links necessary for the completion of the source-to-destination circuit.

We have used Extend™† to construct a simulator capable of simulating any Hypercycle based network. For this simulator, we implemented both the backtrack-to-the-origin-and-retry as well as the e-cube routing strategies. The e-cube routing can only be used for binary cube networks. For each node, we assumed a Poisson message generator which generates packets with uniform distribution of destinations. Each packet carries the destination address which is used for routing. Links are assigned priorities, so that collisions can be resolved. We assumed a packet transmission time (over an established source to destination path) of 100 simulation-clock ticks. We used the simulator to obtain the throughput and delay characteristics of several networks for both e-cube and backtrack-to-the-origin-and-retry in terms to the offered load. Both the offered load and the throughput were normalized in terms to the maximum capacity of each network taken to be proportional to the number of links in the corresponding graph. The average delay was expressed in actual time units necessary to establish a source-to-destination circuit. Simulation results are depicted in fig. 2.

As it was expected, the performance of the backtrack-to-the-origin-and-retry for both binary cubes and hypercycles of similar sizes, is clearly superior to that of the e-cube. This is attributed to the fact that the backtrack-to-the-origin-and-retry can use alternative paths to the destination instead of the single path allotted by the e-cube routing. The additional advantage of the backtrack-to-the-origin and-retry is its inherent fault tolerance. Indeed, if one of links in the network failed, it could be marked as permanently busy, and packets would be routed around it. This obviously is not the case for the e-cube routing.

Generally, system throughput and delay are functions of both average distance and the average number of alternate paths between any two nodes.

6.0 Router Implementation Status

The backtrack-to-the-origin-and-retry with greedy routing, as discussed above, is currently being implemented in hardware. Figure 3 gives a block diagram of an r-dimensional Hypercycle router.

As it can be seen in Figure 3. we are implementing our routing as a system having four modules. The destination address is used in the Next-Port Generator to generate all possible ports that can be used in forming the path to the required destination address. Subsequently, the Port Validator masks out the ports which are currently used by other paths. Finally, The Port Selector, selects at random one of the validated ports which is then used to continue the circuit towards the required destination.

For the random number generator, we use a mixed congruential random number generator to generate a 16 bit random numbers, which we use to obtain its $\mathbf{mod}k$ where k is the number of valid ports incoming to the Port Selector. It is worth noting that the system is programmable, in the sense that it needs the parameters m, ρ as defined earlier and which define the structure of the network, as well as ξ which the address of the current node.

We have completed the implementation of a 16 port 4-dimensional routing engine in 1.2μ technology. Our design is currently under fabrication by CMC. Simulation results yielded propagation delays of less than 50 ns in choosing an available next port. A micrograph of the designed engine is given in Figure 4.

7.0 Conclusions and Discussion

In this work, we presented the Hypercycle, a class of multidimensional graphs, which are essentially generalizations of the n-cube.

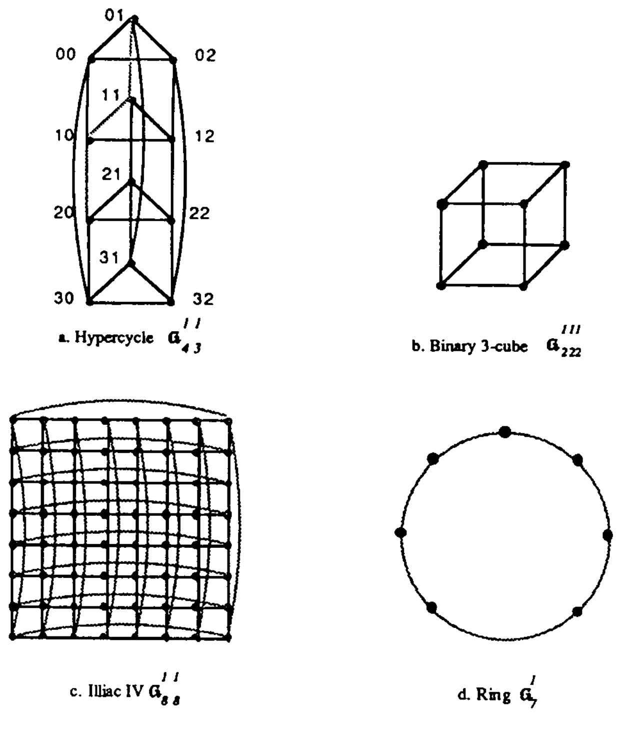
Although these graphs are not the densest possible, they are attractive, because of their simple routing. Similarly to the n-cube, the destination address is used to sequentially route a message through intermediate nodes as outlined in section 3.1. Also, since the node addresses are represented in a mixed radix as a sequence of r-digits, each one of these digits is processed independently and in parallel with the remaining digits. Thus the hardware involved in the routing can be made fast (because of the parallelism) and simple (since each module need only handle arithmetic $\mathbf{mod}m_i$, as compared to arithmetic $\mathbf{mod}m_1m_2...m_r$ needed when all the address digits are necessary as is the case with such networks as the chordal rings [10], or the cube connected cycles [5]).

The graphs presented in this study, are generalizations of some well known graphs such as the binary n-cube, 2- and 3-dimensional meshes, and rings, which are included as special cases. Examples of some special cases are depicted in Figure 1.

† Extend is a trademark of Imagine That Inc.

REFERENCES

1. Berge C., *Principles of Combinatorics* Academic Press, New York and London, 1971.
2. Bhuyan, L. N., and D. P. Agrawal, "Design and Performance of Generalized Interconnection Networks" *IEEE Trans. Comput.* Vol. C-32, No. 12, pp. 1081-1090, Dec. 1983.
3. Bhuyan, L. N., and D. P. Agrawal, "Generalized Hypercube and Hyperbus Structures for a Computer Network" *IEEE Trans. Computers.* Vol. C-33, No. 4. pp. 323-333, (Apr. 1984).
4. A. Broder, D. Dolev, M. Fischer, B. Simons "Efficient Fault Tolerant Routings in Networks" *Proceedings of the 16th Annual ACM Symposium on the Theory of Computing* pp. 536-541, (May 1984).
5. Carlsson, G. E. , J. E. Cruthirds, H. B. Sexton, and C. G. Wright "Interconnection Networks Based on a Generalization of Cube-Connected Cycles" *IEEE Trans. Comput.*, Vol. C-34, No. 8, pp. 769-772, Aug. 1985.
6. E. Chow, H. Madan, J. Peterson "A Real-Time Adaptive Message Routing Network for the Hypercube Computer" *Proceedings of the Real-Time Systems Symposium,* pp. 88-96, San Jose CA., (Dec. 1987)
7. Dally, W.J., and C. L. Seitz "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks" *IEEE Trans. Comput.* Vol. C-36, No. 5, pp. 547-553, May 1987.
8. Dimopoulos, N., R.D. Rasmussen, G.S. Bolotin, B.F. Lewis, R. M. Manning "Hypercycles, Interconnection Networks with simple Routing Strategies" *Proceedings of the Canadian Conference on Electrical and Computer Engineering,* pp. 577-580, Vancouver, B.C., Canada, Nov. 1988.
9. Hortensius, P.D., R. D. McLeod, H.C. Card " Parallel Random Number Generation for VLSI Systems using Cellular Automata" *IEEE Trans. Computers* Vol. 38, pp. 1466-1473, October 1989.
10. Imase, M., T. Soneoka, and K. Okada, "Connectivity of Regular Directed Graphs with Small Diameters" *IEEE Trans. Comput.*, Vol. C-34, No. 3, pp. 267-273, Mar. 1985.
11. Peterson, J.C., J. O. Tuazon, D. Lieberman, M. Pniel "The MARK III Hypercube -Ensemble Concurrent Computer" *Proceedings of the 1985 International Conference on Parallel Processing* pp. 71-73, Aug. 20-23 1985.
12. Rasmussen, R. D., G. S. Bolotin, N. J. Dimopoulos, B. F. Lewis, and R. M. Manning "Advanced General Purpose Multicomputer for Space Applications" *Proceedings of the 1987 International Conference on Parallel Processing,* pp. 54-57. (Aug. 1987)
13. Rasmussen, R. D., N. J. Dimopoulos, G. S. Bolotin, B. F. Lewis, and R. M. Manning "MAX: Advanced General Purpose Real-Time Multicomputer for Space Applications" *Proceedings of the IEEE Real Time Systems Symposium* pp. 70-78, San Jose, CA.,(Dec. 1987).
14. G. Sabidussi "Graph Multiplication" *Math. Zeitschr.* Vol. 72, pp. 446-457 (1960).
15. Seitz, C. L. "The cosmic cube" *CACM* vol. 28, pp.22-33, Jan. 1985
16. Tanenbaum, A. .S., *Operating Systems: Design and Implementation* Prentice Hall, 1987.
17. Tuazon, J. O., J. C. Peterson, M. Pniel, and D. Lieberman "Caltech/JPL MARK II Hypercube Concurrent Processor"*Proceedings of the 1985 International Conference on Parallel Processing* pp. 666-673, Aug. 20-23 1985.
18. Waltz, D. L. "Applications of the Connection Machine" *IEEE Computer* January 1987, pp.85-97
19. L. D. Wittie "Communication Structures for Large Networks of Microcomputers" *IEEE Trans. on Computers* Vol. C-30, No. 4, pp.264-273. (Apr. 1981).



01
00
02
11
10
12
21
20
22
31
30
32
a. Hypercycle $\mathfrak{G}^{\prime\prime}_{43}$
b. Binary 3-cube $\mathfrak{G}^{\prime\prime\prime}_{222}$
c. Illiac IV $\mathfrak{G}^{\prime\prime}_{88}$
d. Ring $\mathfrak{G}^{\prime}_{7}$

Figure 1. Examples of Hypercycles.

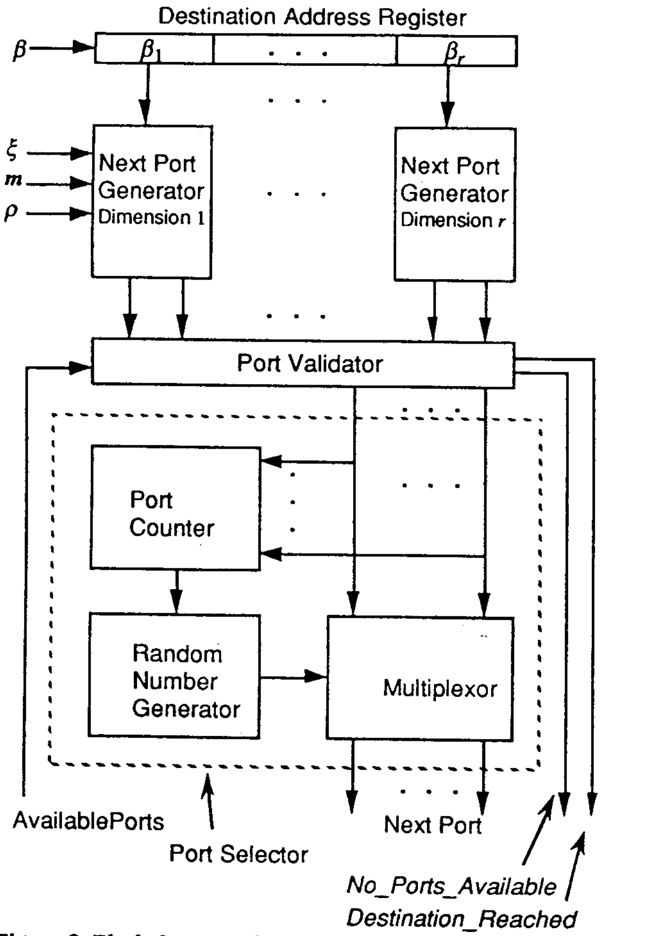


Figure 3. Block diagram of the Hypercycle routing engine. Routing in each dimension is done in parallel.

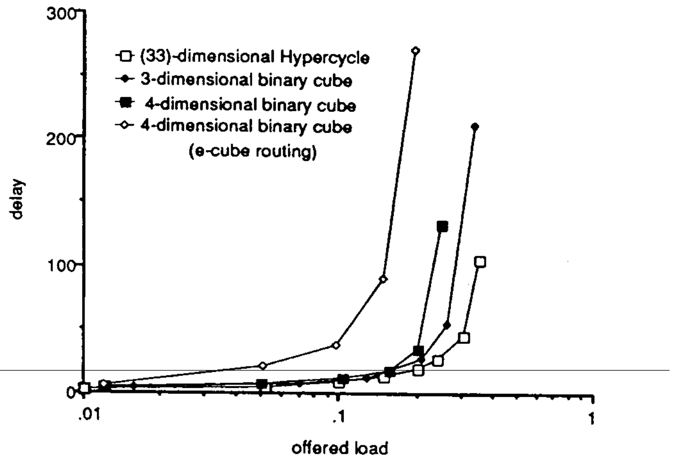


Figure 2. Delay vs. offered load for the 4-cube using backtrack-to-the-origin and e-cube routing. The offered load is normalized to the capacity of the interconnection network. The delay is normalized to the data transmission time.

Figure 4. The implementation of a 16-port 4-dimensional Hypercycle Routing Engine in 1.2 μ NT CMOS4S technology.