A SIMPLE MATHEMATICAL MODEL OF ADAPTIVE ROUTING IN WORMHOLE *k*-ARY *n*-CUBES

H. Sarbazi-Azad, M. Ould-Khaoua
Computing Science Department, Glasgow University, Glasgow, UK
Tel: +44 141 339 8855 ext. 0914, 6056 Fax: +44 141 330 4913
Email: {hsa,mohamed}@dcs.gla.ac.uk

Keywords
Interconnection Networks, *k*-Ary *n*-Cubes, Wormhole Switching, Adaptive Routing, Performance Modelling.

ABSTRACT
Many fully-adaptive algorithms have been proposed for *k*-ary *n*-cubes over the past decade. The performance characteristics of most of these algorithms have been analysed by means of software simulation only. This paper proposes a simple yet reasonably accurate analytical model to predict message latency in wormhole-routed *k*-ary *n*-cubes with fully adaptive routing. This model requires a running time of $O(1)$ which is the fastest model yet reported in the literature while maintaining reasonable accuracy.

1. INTRODUCTION
Network *topology* defines the way nodes are interconnected. Most current multicomputers, e.g. iWarp [17], J-machine [15], Cray T3D [12] and Cray T3E [4], employ *k*-ary *n*-cubes for low-latency and high-bandwidth inter-processor communication. The *k*-ary *n*-cube has an *n*-dimensional grid structure with *k* nodes in each dimension such that every node is connected to its neighbouring nodes in each dimension by direct channels. The two most popular instances of *k*-ary *n*-cubes are the hypercube (where $k=2$) and the torus (where $n=2$).

Wormhole switching (also known as "wormhole routing") have widely been used by current multicomputers due to its low buffering requirements and, more importantly, it makes latency almost independent of the message distance in the absence of blocking. In this switching method, a message is divided into a sequence of fixed-size units, called *flits*, each of a few bytes for transmission and flow control. The *header* flit (containing routing information) establishes the path through the network while the remaining data flits follow it in a pipelined fashion.

SAC 2002, Madrid, Spain
© 2002 ACM 1-58113-445-2/02/03...$5.00

Most practical multicomputers have adopted deterministic routing where messages with the same source and destination addresses always take the same network path [11]. This form of routing has been popular because it requires a simple deadlock-avoidance algorithm, resulting in a simple router implementation [11]. However, a message cannot use alternative paths to avoid congested channels along its route. Fully-adaptive routing has often been suggested to overcome this limitation by enabling messages to explore all available paths [11]. Several authors like Duato [10], Lin et al [14], and Su and Shin [19] have proposed fully-adaptive routing algorithms, that can achieve deadlock-freedom with a minimal requirement for hardware resources.

Analytical models are cost-effective and versatile tools for evaluating system performance under different design alternatives. Analytical models of deterministic routing in wormhole-routed k-ary n-cubes have been widely reported in the literature [1,2,3,5,9]. However, only two analytical models [16,18] of fully-adaptive routing have been proposed for general k-ary n-cubes, to our best knowledge. Ould-Khaoua [16] was the first developing a model for high radix k-ary n-cubes. His model exhibits a good degree of accuracy under light and moderate traffic loads. More recently an accurate analytical model has been introduced in [18]. However, this model is very complex and time consuming with a running time of $O(N=k^n)$.

In this paper, we propose a simple analytical model (with a running time of $O(1)$) for adaptive routing in wormhole *k*-ary *n*-cubes. It is much simpler than the model presented in [18] while the results of comparing these model reveals that the proposed model here has still an acceptable accuracy. The simplicity of this model makes it a practical evaluation tool that can be used to gain insight into the performance behavior of fully-adaptive routing in wormhole-routed *k*-ary *n*-cubes. As in previous similar studies [16,18], the present analysis uses Duato's fully adaptive routing algorithm [14]. This form of routing is widely accepted as one of the most general fully-adaptive routing algorithm for wormhole-routed networks, leading to an efficient router implementation. The Cray T3E [4] and the reliable router [8] are two examples of recent practical systems that have adopted Duato's routing algorithm.

2. PRELIMINARIES
The *k*-ary *n*-cube, where *k* is referred to as the *radix* and *n* as the *dimension*, has $N=k^n$ nodes, arranged in *n* dimensions, with *k* nodes per dimension. Each node can be identified by an *n*-digit

radix k address $(a_1, a_2, \ldots, a_n)$. The i^{th} digit of the address vector, a_i, represents the node position in the i^{th} dimension. Nodes with address $(a_1, a_2, \ldots, a_n)$ and $(b_1, b_2, \ldots, b_n)$ are connected if and only if there exists i, $1 \le i \le n$, such that $a_i = (b_i+1) \bmod k$ and $a_j = b_j$ for $1 \le j \le n$; $i \ne j$. Thus, each node is connected to two neighbouring nodes in each dimension. Each node consists of a processing element (PE) and router. The PE contains a processor and some local memory. The router has n+1 input and n+1 output channels. A node is connected to its neighbouring nodes through n inputs and n output. The remaining channels are used by the PE to inject/eject messages to/from the network respectively. Messages generated by the PE are transferred to the router through the injection channel. Messages at the destination are transferred to the local PE through the ejection channel. Each physical channel is associated with some, say V, *virtual channels*. A *virtual channel* has its own flit queue, but shares the bandwidth of the physical channel with other virtual channels in a time-multiplexed fashion [6]. The router contains flit buffers for any incoming virtual channel. An $(n+1)V$-way crossbar switch direct message flits from any input virtual channel to any output virtual channel. Such a switch can simultaneously connect multiple input to multiple output virtual channels while there is no conflicts.

Deadlock-free fully-adaptive routing algorithms that require only one extra virtual channel compared to deterministic routing have been discussed in [10,14, 19]. For instance, Duato's algorithm [10] divides the virtual channels into two classes a and b making two virtual sub-networks. At each Deadlock-free fully-adaptive routing algorithms that require only one extra virtual channel compared to deterministic routing have been discussed in [10,14, 19]. For instance, Duato's algorithm [10] divides the virtual channels into two classes a and b making two virtual sub-networks. At each routing step, a message visits adaptively any available virtual channel from class a. If all the virtual channels belonging to class a are busy it visits a virtual channel from class b using deterministic routing. The virtual channels of class b define a complete deadlock-free virtual sub-network, which acts like a "drain" for the virtual sub-network built from virtual channels belonging to class a. The routing in the deadlock-free virtual sub-network operates as described in [7]. In k-ary n-cubes at least three virtual channels per physical channel are required to ensure deadlock-freedom where the class a contains one virtual channel and class b owns two virtual channels. When there are more than three virtual channels network performance is maximised when the extra virtual channels are added to adaptive virtual channels in class a [10,11]. Thus, with V virtual channels per physical channel the best performance is achieved when class a and b contain $(V-2)$ and 2 virtual channels respectively.

3. THE ANALYTICAL MODEL

The model uses assumptions that are widely used in the literature [1,2,3,5,6,9,16, 18].

a) Nodes generate traffic independently of each other, and which follows a Poisson process with a mean rate of λ messages per cycle.

b) The arrival process at a given channel is approximated by an independent Poisson process. This approximation has often been invoked to determine the arrival process at channels in store-and-forward networks [13]. Although wormhole routing differs from store-and-forward in various aspects, simulation experiments from previous studies have revealed that it is still a reasonable approach to determine the arrival process [1,3,5,9].

c) Message destinations are uniformly distributed across network nodes.

d) Message length is fixed and equal to M flits, each of which is transmitted in one cycle from one router to the next.

e) The local queue at the injection channel in the source node has infinite capacity. Moreover, messages are transferred to the local PE as soon as they arrive at their destinations through the ejection channel.

f) V virtual channels are used per physical channel. Class a contains V-2 virtual channels, that are crossed adaptively. On the other hand, class b contains two virtual channels that are crossed deterministically. Let the virtual channels belonging to class a and b be called the adaptive and deterministic virtual channels respectively. When there is more than one adaptive virtual channel available a message chooses one at random. To simplify the model derivation no distinction is made between the deterministic and adaptive virtual channels when computing virtual channels occupancy probabilities [16,18].

The model computes the mean message latency as follows. First, the mean network latency, S, that is the time to cross the network is determined. Then, the mean waiting time seen by a message in the source node, W_s, is evaluated. Finally, to model the effects of virtual channels multiplexing, the mean message latency is scaled by a factor, V_{avg}, representing the average degree of virtual channels multiplexing that takes place at a given physical channel. Therefore, the mean message latency can be written as

$$Latency = (S+W_s)V_{avg}$$

The average number of hops that a message makes across one dimension and across the network, k_{avg} and d, are given by [1]

$$k_{avg}=(k-1)/2 \quad (2)$$

$$d=nk_{avg} \quad (3)$$

Fully-adaptive routing allows a message to use any available channel that brings it closer to its destination resulting in an evenly distributed traffic rate on all network channels. A router in the k-ary n-cube has n output channels and the PE generates, on average, λ messages in a cycle. Since each message makes, on average, d hops to cross the network the rate of messages received by each channel, λ_c, can be written as [1]

$$\lambda_c = \lambda d/n \quad (4)$$

Let us follow a typical message which makes d hops to reach its destination. The average network latency, S, seen by the message crossing from the source to the destination node consists of two parts: one is the delay due to the actual message transmission time, and the other is due to the blocking time in the network.

Therefore, S can be written as

$$S=M+d+dB \quad (5)$$

where M is the message length, and B is the average blocking time seen by the message at each hop. The term B is given by

$$B=P_{block}W_c \quad (6)$$

with P_{block} being the probability that a message is blocked at the current channel and W_c is the mean waiting time to acquire a channel in the event of blocking. Let us now calculate the blocking probability P_{block}. A message is blocked at a given channel when all the adaptive virtual channels of the remaining dimensions to be visited and also the deterministic virtual channels of the lowest dimension still to be visited are busy [10]. When a message has entirely crossed z dimensions it can select any available $(n\text{-}z)(V\text{-}2)$ adaptive virtual channels associated with $n\text{-}z$ remaining physical channels and one deterministic virtual channel at the lowest dimension to make its next hop. Let P_a be the probability that all adaptive virtual channels of a physical channel are busy and $P_{a\&d}$ denote the probability that all adaptive and deterministic virtual channels of a physical channel are busy. The message has finished traversing a dimension with probability $1/k_{avg}$. If all the adaptive virtual channels of $n\text{-}z\text{-}1$ remaining physical channels are busy (with probability $(P_a)^{n-z-1}$) and all adaptive and deterministic virtual channels at the lowest dimension physical channel are busy (with probability $P_{a\&d}$). The probability of blocking, P_{block}, can therefore be written as

$$P_{block} = \sum_{z=0}^{n-1} \binom{n}{z} (1/k_{avg})^z (1-1/k_{avg})^{n-z} (P_a)^{n-z-1} P_{a\&d} \quad (7)$$

The probabilities P_a and $P_{a\&d}$ are given by [18]

$$P_a = P_V + 2P_{V-1}/\binom{V}{V-1} + P_{V-2}/\binom{V}{V-2} \quad (8)$$

$$P_{a\&d} = P_V + 2P_{V-1}/\binom{V}{V-1} \quad (9)$$

To determine the mean waiting time, W_c, to acquire a virtual channel a physical channel is treated as an M/G/1 queue with a mean waiting time of [13]

$$W_c = \rho S(1+C_S^2)/[2(1-\rho)] \quad (10)$$

where

$$\rho = \lambda_c S \quad (11)$$

$$C_S^2 = \sigma_S^2 / S^2 \quad (12)$$

where λ_c (given by Equation 4) is the traffic rate on the channel, S (given by Equation 5) is its service time, and σ_S^2 is the variance of the service time distribution. Since the minimum service time at a channel is equal to the message length, M, following a suggestion proposed in [9], the variance of the service time distribution can be approximated as

$$\sigma_S^2 = (S-M)^2 \quad (13)$$

Hence, the mean waiting time becomes

$$W_c = \lambda_c S^2 (1+(S-M)^2/S^2)/(2(1-\lambda_c S)) \quad (14)$$

A message originating from a given source node sees a network latency of S. Modelling the local queue in the source node as an M/G/1 queue, with the mean arrival rate λ/V (as a message in the source node can enter the network through any of the V virtual channels) and service time S with an approximated variance $(S\text{-}M)^2$ yields the mean waiting time seen by a message at thesource node as [13]

$$W_s = \lambda S^2 (1+(S-M)^2/S^2)/(2(V-\lambda S)) \quad (15)$$

The probability, P_v, that $v=0,1,\ldots,V$ virtual channels are busy at a physical channel, can be determined using a Markovian model. State π_v corresponds to v virtual channels being requested. The transition rate out of state π_v to state π_{v+1} is the traffic rate λ_c (given by Equation 4) while the rate out of state π_v to state π_{v-1} is $1/S$ (S is given by Equation 5). The probability that v virtual channels are busy, when $v=0,1,\ldots,V\text{-}1$, is the probability of being in state π_v, i.e. $P_v=\Pr(\pi_v)$. However, the probability that V virtual channels are busy is the summation of the probabilities of being in states π_v, $V \le v < \infty$, i.e. $P_V = \sum_{l=V}^{\infty} \Pr(\pi_l)$. The steady-state solution of the Markovian model yields the probability P_v to be

$$P_v = \begin{cases} (1-\lambda_c S)(\lambda_c S)^v, & 0 \le v < V \\ (\lambda_c S)^v, & v = V \end{cases} \quad (16)$$

When multiple virtual channels are used per physical channel they share the bandwidth in a time-multiplexed manner. The average degree of multiplexing of virtual channels, that takes place at a given physical channel, can be estimated by [6]

$$V_{avg} = \sum_{v=1}^{V} v^2 P_v / \sum_{v=1}^{V} v P_v \quad (17)$$

The above equations reveal that there are several inter-dependencies between the different variables of the model. For instance, Equations 5 and 6 reveal that S is a function of W_c while Equation 14 shows that W_c is a function of S. Given that closed-form solutions to such inter-dependencies are very difficult to determine the different variables of the model are computed using iterative techniques for solving equations.

In Fig.1, we have compared the proposed model to the accurate model proposed in [18] for three different network, namely the 20-ary 2-ary, the 10-ary 3-cube and the 6-ary 4-cube with message lengths M=32 nad 64 flits and V=3 and 5 virtual channels per physical channel. As can be seen in the figure, the proposed model here is matching the accurate model under light and moderate traffic loads. However, it is slightly overestimating the mean message latency which causes an earlier saturation in turn compared to the accurate model proposed in [18]. Although the model proposed is slightly less accurate than the model in [18]

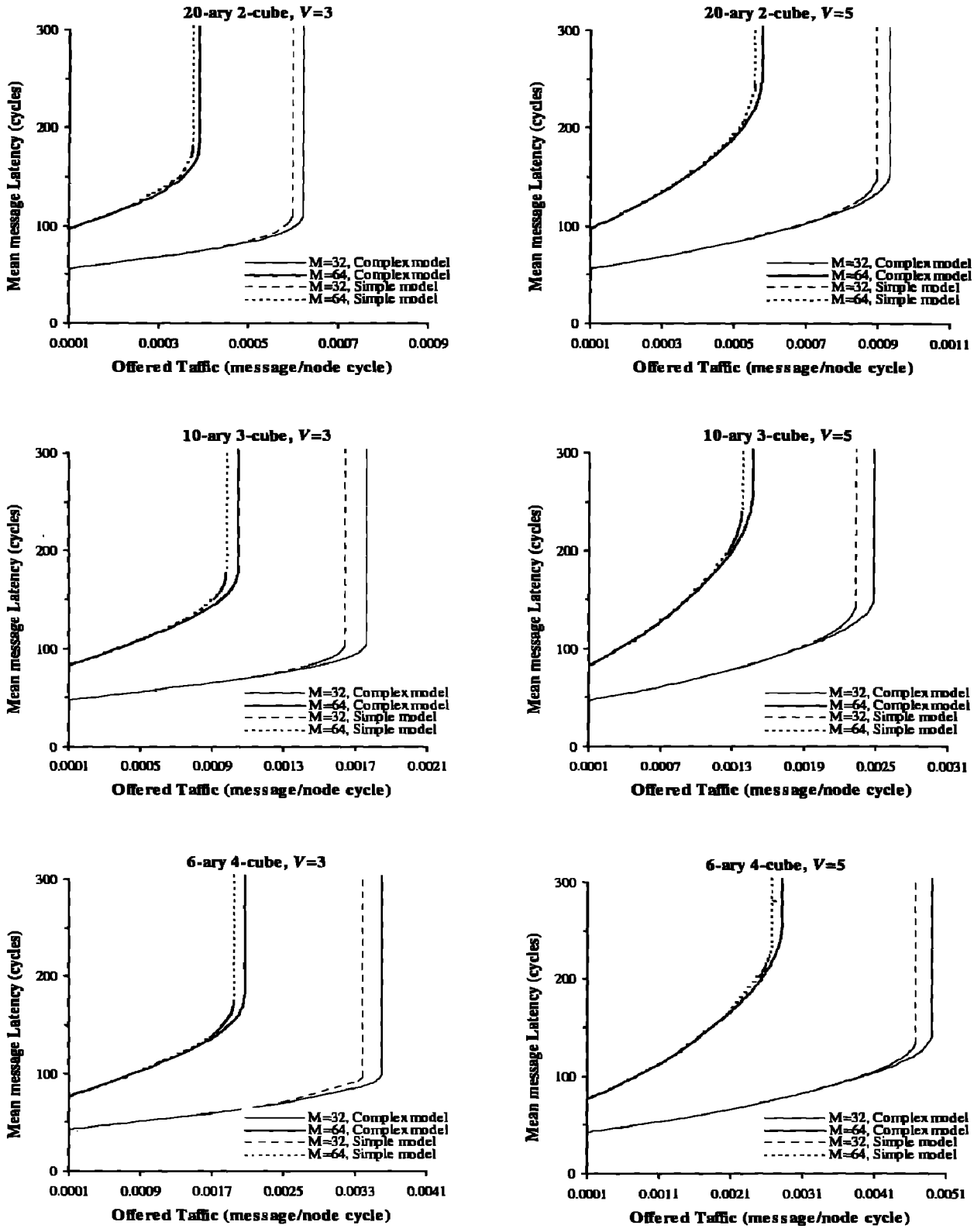

Fig. 1- Average message latency predicted by the simple model proposed here and the complex model reported in [18] versus offered traffic rate in a 400-node (20-ary 2-cube), a 1000-node (10-ary 3-cube), and a 1296-node (6-ary 4-cube) network with message length M=32 and 64 flits and number of virtual channel V=3 and 5 per physical channel.

under heavy traffic loads and near the network saturation region, its simplicity makes it an attractive tool for studying performance merits of k-ary n-cubes under different working conditions.

4. CONCLUSION

This paper has described a simple analytical model to compute the mean message latency in wormhole-routed k-ary n-cubes with Duato's fully-adaptive routing algorithm. The model manages to achieve a good degree of accuracy wile maintaining simplicity, making it a practical evaluation tool that can be used to gain insight into the performance behavior of fully-adaptive routing in wormhole-routed k-ary n-cubes. Our next objective is to use this model to compare performance merits of tori and hypercubes under different implementation constraints.

5. REFERENCES

[1] Agarwal A., Limits on interconnection network performance, *IEEE Trans. Parallel & Distributed Systems* **2(4)** (1991) pp. 398-412.

[2] Anderson J.R. and Abraham S., Performance-based constraints for Multi-dimensional networks, *IEEE Trans. Para. Distr. Syst.* **11(1)** (2000) pp. 21-35.

[3] Ciciani B., Colajanni M. and Paolucci C., Performance evaluation of deterministic wormhole routing in k-ary n-cubes, *Parallel Computing* **24** (1998) pp. 2053-2075.

[4] Cray Research Inc., The Cray T3E scalable parallel processing system, on Cray's web page at http://www.cray.com/PUBLIC/product- info/T3E.

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

[6] Dally W.J., Virtual channel flow control, *IEEE Trans. Parallel & Distributed Systems* **3(2)** (1992) pp. 194-205.

[7] Dally W.J. and Seitz C.L., Deadlock-free message routing in multiprocessor interconnection networks, *IEEE Trans. Computers* **C36(5)** (1987) pp. 547-553.

[8] Dally W.J., Dennison L.R., Harris D., Kan K.and Xanthopoulos T., The reliable router: A reliable and high-performance communication substrate for parallel computers, (*Proc. Int'l Workshop Parallel. Computer Routing and Communication*, May 1994) pp. 241-255.

[9] Draper J.T. and Ghosh J., A Comprehensive analytical model for wormhole routing in multicomputer systems, *Journal Parallel & Distributed Computing* **32** (1994) pp. 202-214.

[10] Duato J., A New theory of deadlock-free adaptive routing in wormhole routing networks, *IEEE Trans. Paral. Distr. Syst.* **4(12)** (1993) pp. 320-1331.

[11] Duato J., Yalamanchili S. and Ni L., *Interconnection networks: An engineering approach* (IEEE Computer Society Press, 1997).

[12] Kessler R.E. and Schwarzmeier J.L., CRAY T3D: A new dimension for Cray Research, (*CompCon*, Spring 1993) pp. 176-182.

[13] Kleinrock L., Queueing Systems, (John Wiley, New York, 1975).

[14] Lin X., Mckinley P.K. and Lin L.M., The message flow model for routing in wormhole-routed networks, (*Proc. Int'l Conference on Parallel Processing*, 1993) pp. 294-297.

[15] Noakes M. and Dally W.J., System design of the J-machine, (*Proc. Advanced Research in VLSI* , MIT Press, 1990) pp. 179-192.

[16] Ould-Khaoua M., A performance model of Duato's adaptive routing algorithm in k-ary n-cubes, *IEEE Transactions on Computers*, **48** (1999) pp. 1297-1304.

[17] Peterson C. *et al*, iWarp: a 100-MOPS VLIW microprocessor for multicomputers, *IEEE Micro*, **11(2)** (1991) pp. 26-37.

[18] Sarbazi-Azad H., Ould-Khaoua M. and Mackenzie L.M., An accurate analytical model of adaptive wormhole routing in k-ary n-cube interconnection networks, *Performance Evaluation* **43(2-3)** (2001) pp. 165-179.

[19] Su C. and Shin K.G., Adaptive deadlock-free routing in multicomputers using one extra channel, (*Proc. Int'l Conf. Parallel Processing*, 1993) pp. 175-182.