Analytical Performance Modelling of Partially Adaptive Routing in Wormhole Hypercubes

Ahmad Patooghy, Hamid Sarbazi-Azad

School of Computer Science, IPM & Sharif University of Technology, Tehran, Iran
{patooghy,azad}@ipm.ir

Abstract

Although several analytical models have been proposed in the literature for different interconnection networks with different routing algorithms, there is only one work dealing with partially adaptive routing algorithms. This paper proposes an accurate analytical model to predict message latency in wormhole-routed hypercube based networks using the partially adaptive routing algorithm. The results obtained from simulation experiments confirm that the proposed model exhibits a good accuracy for various network sizes and under different operating conditions.

1. Introduction

The hypercube, has been one of the most common multicomputer networks due to its desirable properties, such as high connectivity to deal with fault tolerance, ability to exploit communication locality to reduce message latency, recursive structure to solve important and popular problems such as FFT and matrix multiplication, symmetry in vertex and edge that makes it suitable for VLSI implementation and ability to simulate and embed many other important topologies. The hypercube has been used in several machines such as n-Cube, iPSC/2, Cosmic Cube [8].

Modern parallel routers significantly reduce average latency by using wormhole switching. Wormhole is a switching strategy that divides each packet in elementary units called *flit*, each of a few bytes for transmission and flow control, and advances each flit as soon as it arrives at a node. The *header* flit (containing routing information) governs the route and the remaining data flits follow it in a pipelined fashion. If a channel transmits the header of a message, it must transmit all the remaining flits of the message before transmitting flits of another message. Once the header is blocked, the data flits are blocked in situ. Wormhole switching is attractive because it reduces the latency of message delivery compared to the store and forward switching. Network throughput of wormhole routed networks can be increased by organizing the flit buffers associated with each physical channel into several virtual channels. These virtual channels are allocated independently to different packets and compete with each other for using the bandwidth of the physical channel. This decoupling allows active messages to pass blocked messages using network bandwidth that would otherwise be wasted [8].

The routing algorithm indicates the path a message should take to reach to its destination by selecting the proper output channel. This channel can be selected from a set of possible choices and according to the size of this set, routing algorithms are divided into three categories. *Deterministic routing,* assigns a single path to each source and destination (i.e., size of the mentioned set is one in this category.) [8]. This form of routing has been popular due to its simple deadlock-avoidance algorithm, resulting in a simple router implementation [1]. However, in deterministic routing a message cannot use alternative paths to avoid congested channels along its route and therefore the network performance is low. Several multicomputers have used deterministic routing [8]. *Fully adaptive* routing algorithm has often been suggested [8] to overcome this limitation by enabling messages to explore all available paths (i.e., the above mentioned set has maximum size here) and consequently offers the potential for making better use of network resources; but these algorithms imply more router complexity for deadlock-freedom. *Partially adaptive* routing algorithms try to combine the advantages of the two other categories to produce a routing with limited adaptivity and establish a balance between performance and router complexity [2, 3]. They allow selecting output channel from a subset of all possible channels; in fact these algorithms limit the size of the set of possible choices. Turn model based algorithms [3] and planar adaptive routing algorithm [2] are the most important partially adaptive routing algorithms for the mesh, torus, and hypercube networks.

Mathematical models are cost-effective and versatile tools for evaluating performance of a network under different design alternatives. The significant advantage of analytical models over simulation is that they can be used to obtain performance results for large systems and behaviour under different network configurations and working conditions which may not be feasible to study

1-4244-0054-6/06/$20.00

using simulation on conventional computers due to the excessive computation demands. Although several researchers have proposed analytical models of deterministic and fully adaptive routing in wormhole-routed hypercube networks [6, 13], there is only one model [10] dealing with the partially adaptive routing algorithm that has low accuracy. This paper proposes an accurate analytical model for P-cube routing as the most-known partially adaptive routing algorithm in hypercubes.

The rest of the paper is organized as follows. A brief description on P-cube routing algorithm is introduced in section 2. Section 3 proposes a mathematical model for P-cube routing in wormhole-switched hypercubes. The proposed model is validated in section 4. Finally, some concluding remarks and suggestions for future work in this line are made in section 5.

2. The P-cube Routing Algorithm

The P-cube is a routing algorithm based on turn model for designing deadlock-free routing algorithms with maximal adaptivity. The turn model involves analyzing the directions in which packets can turn in the network and the cycles that the turns can form and then prohibiting just enough number of turns to break all of the cycles preventing network from deadlock. The steps of the turn model algorithm are as follows:

Step 1. Partition the channels of the network into sets according their direction.

Step 2. Identify the possible turns from one direction to another, ignoring 0-degree turns. A 0-degree turn is only possible when there are multiple virtual channels in one direction.

Step 3. Identify the cycles that may form through these turns. Generally, identifying the simplest cycles in each plane of the topology is adequate.

Step 4. Prohibit one turn in each cycle so as to prevent deadlocks. The turns must be chosen carefully in order to break every possible cycle, including complex cycles not identified in step 3.

Routing algorithms that route packets along the sets of channels identified in step 1 and use only the turns from one set to another allowed by step 4 are deadlock free because breaking all the cycles prevents circular waits. In other words, preventing circular wait in this way means that it is possible to number the channels in the network so that the algorithm routes every packet along channels in strictly decreasing (or increasing) order. This, together with the fact that a network contains a finite number of channels, means that a packet will reach its destination after limited number of hops. Thus, the turn model based algorithms are livelock free [3] and maximally adaptive, as the model prohibits just the minimum required number of turns. The most important partially adaptive routings algorithms based on the turn model are *West first*, *North last*, and *Negative first* routing algorithms for the mesh and *P-cube* routing algorithm for the hypercube.

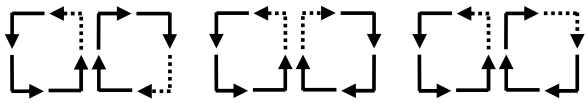

Figure 1. Prohibited turns in West first (a), North last (b), and Negative first (c) routing algorithms.

For instance figure 1.*c* shows a way to prohibit two turns in a 2D mesh. The prohibited turns are the two from a positive direction to a negative direction. To travel in a negative direction, a packet has to start out in a negative direction consequently *Negative first* routing algorithm routes a packet first adaptively west and south, and then adaptively east and north.

P-cube routing algorithm is a special case of the Negative first routing algorithm that has a particularly compact expression. Let C be the binary address of the node the header flits currently occupy, and D be the binary address of the destination node. The P-cube routing algorithm has two phases which in the first phase, starts and algorithm routes the packet along a dimension i for which $c_i=1$ and $d_i=0$. When there is no such a dimension, the second phase routes the packet along dimension i for which $c_i=0$ and $d_i=1$. These steps are easily computed using bitwise logic operation as shown in Figure 2. The only input transmitted in the header flits is D. C is a unique constant for each router.

Algorithm P-cube routing (n-D hypercube);
Input: Current address, C, and destination address, D.
{ If $C = D$ then route packet to local processor and exit;
$R = C \wedge \overline{D}$;
If $R = 0$ then $R = \overline{C} \wedge D$;
Route the packet adaptively along any available channel in dimension i for which $r_i = 1$;
}

Figure 2. The Pseudo code for the P-cube routing

More precisely it is obvious that, in the first phase of the P-cube routing, packets are routed towards a pivot node and the second phase will start from this node. For example, if source and destination nodes be 10101010 and 10010011 respectively, then the pivot node will be 10000010. Note that the pivot node tends to have 0s in its address patterns for all dimensions. That is, the probability of being a pivot node for a node with more 0's in its address is higher than a node with 1's in its address pattern. This causes more traffic over channels connected to the nodes with more 0's in their address (the most crowded channels are those connected to node 0=(000..0) in the hypercube. This means even with a uniform traffic pattern for the destination of messages, the P-cube routing algorithm results in an unbalanced traffic rate over network channels. Figure 3 shows the message arrival rate over network channels in a 10-D hypercube when P-cube routing algorithm is used and nodes are generating

message with an average $\lambda = 0.01$ messages per cycle. For the sake of comparison, the white bar shows the rate when a traffic-balanced routing algorithm (e.g. e-cube or Duato's fully adaptive routing algorithm) is used for which the traffic rate over network channels is distributed evenly. The traffic rate over channels with small Hamming weights (with a small number of 1's in the address pattern of nodes indicating the channel) is high and gradually decreases when the channel Hamming weight increases.

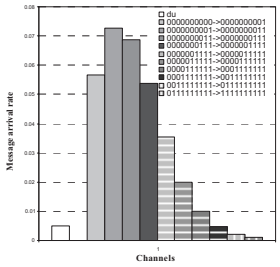

Figure 3. The message arrival rate on different channels in a 10-cube using P-cube routing; (top) negative channels, (down) positive channels.

Message arrival rate over positive and negative channels respectively can be calculated by [12]

$$\lambda_{<a,b>} = \frac{\lambda}{2^n - 1} \sum_{i=0}^{n-m-1} \sum_{j=0}^{m} \frac{\binom{n-1}{i} \binom{m}{j} 2^{m-j}}{(i+j+1) \binom{i+j}{i}} \quad (1)$$

$$\lambda_{<a,b>} = \frac{\lambda}{2^n - 1} \sum_{i=0}^{n-m} \sum_{j=0}^{m-1} \frac{\binom{n}{i} \binom{m-1}{j} 2^{m-j-1}}{(i+j+1) \binom{i+j}{i}}. \quad (2)$$

3. The Model

In this section, we propose an analytical performance model for P-cube routing in the hypercube. The modelling approach used here can be equally applied for other routing schemes after minor changes in the model. The parameter studied in our model is the average message latency as a criterion for network performance.

3.1. Model Assumptions

The following assumptions are made when developing the proposed performance model. These assumptions have been widely used in similar modelling studies [4, 5, 6, 7, 9, 10, 11, 13]:

a) Messages are of fixed length and equal to *M* flits. The flit transfer time between any two adjacent routers is assumed to be one cycle.

b) Message destinations are uniformly distributed across the network nodes.

c) Nodes generate traffic independently of each other, and follow a Poisson process with a mean rate of λ_g messages/cycle.

d) Messages are transferred to the local processor through the ejection channel once they arrive at their destination.

e) Each physical channel is shared between *V* virtual channels.

3.2. Model Description

The model computes the mean message latency as follows. First, the mean network latency which is the required time to cross the network, $\overline{S}$, is determined. Then, the mean waiting time seen by a message in the source node to be injected into the network, $\overline{W_s}$, is evaluated. To model the effect of virtual channels multiplexing, the mean message latency is scaled by the average virtual channels multiplexing degree, $\overline{V}$. Therefore, the mean message latency can be given as

$$Latency = (\overline{S} + \overline{W_s})\overline{V}. \quad (3)$$

Let source and destination node addresses be $s = s_{n-1}s_{n-2}s_{n-3}...s_1s_0$ and $d = d_{n-1}d_{n-2}d_{n-3}...d_1d_0$. The parameters h_n and h_p are defined as the number of the required hops in the first and second phases of P-cube routing algorithm, respectively, and can be given by

$$\begin{cases} h_p = \sum_{i=0}^{n-1} \overline{s_i}.d_i \\ h_n = \sum_{i=0}^{n-1} s_i.\overline{d_i} \end{cases}. \quad (4)$$

Thus, the total number of hops needed to deliver a message to the destination is $H = h_n + h_p$.

The number of different nodes, K_i, that a message may be located at, after making *i* hops from the source node towards its destination, can be given by

$$K_i = \begin{cases} \binom{h_p}{i} & \text{if } i \le h_p \\ \binom{h_n}{i - h_p} & \text{if } i > h_p \end{cases}. \quad (5)$$

3.2.1. Calculation of the Average Multiplexing Degree

Due to the unbalanced traffic rate over network channels, we must calculate the average degree of multiplexing, $\overline{V}_{(s,d)}$, for all possible physical channels which can be used by a message communicated between each pair of source and destination nodes (*s, d*). Using $\overline{V}_{(s,d)}$ we can calculate the total average degree of multiplexing as

$$\overline{V} = \frac{1}{N}\left(\sum_{s\in G}\frac{1}{N-1}\sum_{d\in G-\{s\}}\overline{V}_{(s,d)}\right) \quad (6)$$

where G is the set of all nodes in the network and $N = 2^n$ is the number of nodes in the network. $\overline{V}_{(s,d)}$ may be evaluated by averaging the average multiplexing degrees in all H hops from the source node s to the destination node d. Also, the average degree of multiplexing of channels at the i-th hop is itself the average of multiplexing degree of all possible output channels in all K_{i-1} reachable nodes by the (i-1)-th hop from the source node towards the destination node. Thus, we can write

$$\overline{V}_{(s,d)} = \frac{1}{h_p+h_n}\sum_{i=1}^{h_p+h_n}\left(\sum_{j=1}^{K_{i-1}}\overline{V}_{\langle a_i,b_i\rangle(s,d),j}\Big/K_{i-1}\right) \quad (7)$$

where $\overline{V}_{\langle a_i,b_i\rangle(s,d),j}$ is the average of the multiplexing degree of all reachable output channels at the i-th hop in the j-th node among K_{i-1} nodes reachable at the (i-1)-th hop from the source node to the destination node. To calculate $\overline{V}_{\langle a_i,b_i\rangle(s,d),j}$, it should be noted that the current node address and possible output channels for the next hop are known. This simplifies the problem to calculating the multiplexing degree of some specific channel at a specific node. As discussed in [14], $\overline{V}_{\langle a,b\rangle}$, i.e. the multiplexing degree of the channel connecting nodes a and b, can be expressed as

$$\overline{V}_{\langle a,b\rangle} = \frac{\sum_{v=1}^{V} v^2 P_{\langle a,b\rangle,v}}{\sum_{v=1}^{V} vP_{\langle a,b\rangle,v}} \quad (8)$$

where $P_{\langle a,b\rangle,v}$, is the probability that v virtual channels are busy at physical channel $\langle a,b\rangle$.

3.2.2. Calculation of the Average Message Blocking Time

Different traffic rates over network channels force us to calculate mean traversal latency, $S_{(s,d)}$, i.e. the mean time to cross the network from a specific source s to a specific destination d for all possible source/destination pairs $(s,d), s\in G,\ d\in G-\{s\}$. The mean network latency, $\overline{S}$, is calculated as the average of these values as

$$\overline{S}_s = \frac{1}{N-1}\sum_{d\in G-\{s\}} S_{(s,d)} \quad (9)$$

$$\overline{S} = \frac{1}{N}\sum_{s\in G}\overline{S}_s \quad (10)$$

where $S_{(s,d)}$ consists of two parts. One is the delay due to the actual message transmission time, $H+M$, where M is the message length, and the other delay is due to the message blocking in the network, $T_{blocking(s,d)}$. Therefore, $S_{(s,d)}$ can be written as

$$S_{(s,d)} = H + M + T_{blocking(s,d)} + \overline{W}_{eject}. \quad (11)$$

where $\overline{W}_{eject}$ is the waiting time for a message arrived at its destination to pass through the ejection channel. With respect to P-cube routing algorithm, $T_{blocking(s,d)}$, can be divided to blocking time in the first and second phases of routing. Thus, we can write

$$T_{blocking(s,d)} = T_{blocking(s,pivot)} + T_{blocking(pivot,d)}. \quad (12)$$

Mean blocking time in the first phase of routing for any pair of source/destination, (s,d), is equal to the probability of blocking multiplied by the mean waiting time in all h_n hops. Thus,

$$T_{blocking(s,pivot)} = \sum_{i=1}^{h_n} P_{(s,pivot),i}\times W_{mean,(s,pivot),i}. \quad (13)$$

Mean blocking time for the second phase can be calculated in a similar way as

$$T_{blocking(pivot,d)} = \sum_{i=1}^{h_p} P_{(pivot,d),i}\times W_{mean,(pivot,d),i}. \quad (14)$$

Blocking probability of a message from s towards d can be calculated by averaging this probability in all K_{i-1} nodes that the message can reach after i-1 hops. Thus,

$$P_{(s,pivot),i} = \frac{1}{K_{i-1}}\sum_{j=1}^{K_{i-1}} PB^{j}{}_{(s,pivot),i} \quad (15)$$

Similarly, for the second phase we can write

$$P_{(pivot,d),i} = \frac{1}{K_{i-1}}\sum_{j=1}^{K_{i-1}} PB^{j}{}_{(pivot,d),i} \quad (16)$$

where $PB^{j}{}_{(a,b),i}$ is the blocking probability of the message in the j-th node among K_{i-1} nodes that the message can reach after i-1 hops. In such a node the message can select one of the $h_n - i + 1$ (or $h_p - i + 1$ for the second phase of routing) unselected channels to continue its way towards its destination and consequently all of these channels must be simultaneously occupied in order for blocking to occur. Thus,

$$PB^{j}{}_{(s,pivot),i} = \prod_{k=1}^{h_n-i+1} P^{j,k}_{used,(s,pivot),i} \quad (17)$$

$$PB^{j}{}_{(pivot,d),i} = \prod_{k=1}^{h_p-i+1} P^{j,k}_{used,(pivot,d),i} \quad (18)$$

where $P^{j,k}_{used,(s,pivot),i}$ is the probability that the k-th physical channel of the j-th node among K_{i-1} nodes reachable after i-1 hops from the source node to the pivot node is busy. $P^{j,k}_{used,(s,pivot),i}$ is the same probability for the second phase.

Now, the problem has been simplified to calculating the probability that V virtual channels of a specific

physical channel are busy at the same time. This new simplified problem can be solved using the Markov model as shown in figure 4 [11]. As shown in the figure, state π_v, $1 \le v \le V$, corresponds to v virtual channels being busy at a physical channel. The transition rate out of state π_v to state π_{v+1} is the channel traffic rate $\lambda_{\langle a,b \rangle}$ while the rate out of state π_v to state π_{v-1} is $1/\overline{S}_{\langle a,b \rangle}$. The transition rate out of state π_v is reduced by $\lambda_{\langle a,b \rangle}$ to account for the arrival of the message while a channel is in this state. The steady-state solution of the Markov model yields the probability $P_{\langle a,b \rangle,v}$, $1 \le v \le V$, for the channel $\langle a,b \rangle$.

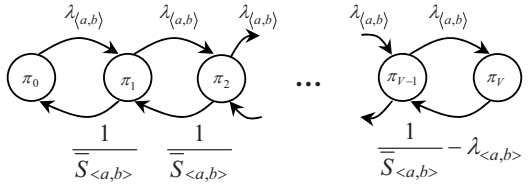

Figure 4. Markov model of a physical channel with *V* virtual channels. State S_i corresponds to *i* virtual channels being busy.

$$Q_{\langle a,b \rangle,v} = \begin{cases} \left(\lambda_{\langle a,b \rangle}\overline{S}_{\langle a,b \rangle}\right)^v, & if \quad 0 \le v \le V-1 \\ \dfrac{\left(\lambda_{\langle a,b \rangle}\overline{S}_{\langle a,b \rangle}\right)^v}{1-\lambda_{\langle a,b \rangle}\overline{S}_{\langle a,b \rangle}}, & if \quad v = V \end{cases} \tag{19}$$

$$P_{\langle a,b \rangle,v} = \begin{cases} \dfrac{1}{\sum_{i=0}^{V} Q_{\langle a,b \rangle,i}}, & if \quad v = 0 \\ P_{\langle a,b \rangle,0} Q_{\langle a,b \rangle,v}, & if \quad 1 \le v \le V \end{cases}. \tag{20}$$

Service time of channel $\langle a,b \rangle$, $\overline{S}_{\langle a,b \rangle}$, is approximated by averaging $S_{(s,d)}$ for all pairs of source/destination that can use this channel in at least one of their minimal paths, and can be expressed as

$$\overline{S}_{\langle a,b \rangle} = \frac{\sum_{(s,d) \in G_{\langle a,b \rangle}} S_{(s,d)} m_{(s,d),\langle a,b \rangle}}{\sum_{(s,d) \in G_{\langle a,b \rangle}} m_{(s,d),\langle a,b \rangle}} \tag{21}$$

where $m_{(s,d),\langle a,b \rangle}$ is the number of different minimal paths from the source node *s* to the destination node *d* that can use channel $\langle a,b \rangle$ in at least one of the possible paths between *s* and *d*.

The average waiting time to acquire a channel $\langle a,b \rangle$ when a message is blocked at this channel, $w_{\langle a,b \rangle}$, can be computed using an *M/G/*1 queue with arrival rate $\lambda_{\langle a,b \rangle}$ and service time $\overline{S}_{\langle a,b \rangle}$ can be given by [11]

$$w_{\langle a,b \rangle} = \frac{\lambda_{\langle a,b \rangle}\overline{S}^2_{\langle a,b \rangle}\left(1 + \left(\overline{S}_{\langle a,b \rangle} - M\right)^2 \Big/ \overline{S}^2_{\langle a,b \rangle}\right)}{2\left(1 - \lambda_{\langle a,b \rangle}\overline{S}_{\langle a,b \rangle}\right)} \tag{22}$$

Note that, the variance of the service time of channel $\langle a,b \rangle$ is approximated by $\left(\overline{S}_{\langle a,b \rangle} - M\right)^2$ as suggested in [5, 11].

3.2.3. Calculating the Average Waiting Time at the Source Node

A message originating from a given source node, *s*, sees a network latency of $\overline{S}_s$. Modeling the local queue in the source node as an *M/G/*1 queue, with the mean arrival rate of λ_g / V (recalling that a message in the source node can enter the network through any of the V virtual channels) and service time $\overline{S}_s$ with an approximate variance of $\left(\overline{S}_s - M\right)^2$ yields the mean waiting time seen by a message at the source node *s* as [11]

$$\overline{W}_s = \frac{\lambda_g / V \,\overline{S}^2_s \left(1 + \left(\overline{S}_s - M\right)^2 / \overline{S}^2_s\right)}{2\left(1 - \lambda_g / V \,\overline{S}_s\right)} \tag{23}$$

and consequently the average waiting time at the source node is given by

$$\overline{W} = \frac{1}{N}\sum_{s \in G} \overline{W}_s . \tag{24}$$

3.2.4. Calculating the Average Waiting Time at Ejection Channel

In the steady state, the rate at which messages exit the network through ejection channel is equal to the injection rate of messages, which is in turn equal to the generation rate λ_g. Utilization of the ejection channel (in each node) is therefore equal to $M\lambda_g$. Given that messages are of fixed length, there is no variance in service time. Using an *M/D/1* queuing model [5, 9], we can calculate the waiting time at the ejection channel as

$$W_{ejection} = M^2 \lambda_g / 2(1 - M\lambda_g). \tag{25}$$

3.3. Solving the Model Equations

The above equations reveal that there are several inter-dependencies between the different variables of the model. For instance, Equations 11 reveals that $S_{(s,d)}$ is a function of $T_{blocking(s,d)}$ while equation 13 shows that $T_{blocking(s,d)}$ is a function of $P_{(s,pivot),i}$. Equations 15 to 18 tell us that $P_{(s,pivot),i}$ depends on $P_{\langle a,b\rangle,v}$ and equations 19 and 20 show that $P_{\langle a,b\rangle,v}$ is itself a function of $\overline{S}_{\langle a,b\rangle}$ while we calculate $\overline{S}_{\langle a,b\rangle}$ by averaging the average network crossing times for each arbitrary source/destination pair, i.e. $S_{(s,d)}$.

As the closed-form solutions to such inter-dependencies are very difficult to determine, the different variables of the model are computed using an iterative technique as discussed in [11].

4. Model Validation

The proposed analytical model has been validated through a discrete-event simulator that mimics the behaviour of the described routing algorithms in the network at the flit level. In each simulation experiment, a minimum of 200000 messages are delivered. Statistics gathering was inhibited for the first 20000 messages to avoid distortions due to the initial start-up conditions. The simulator uses the same assumptions as the analysis, some detailed here with the aim of making the network operation clearer. The network cycle time is defined as the transmission time of a single flit from one router to the next. Messages are generated at each node according to a Poisson process with a mean inter-arrival rate of λ_g messages/cycle. Message length is fixed at M flits. Destination nodes are determined using a uniform random number generator. The mean message latency is defined as the mean amount of time elapsed between the generation of a message and the last data flit reaching the local processor at the destination node. Several validation experiments have been performed for several combinations of network sizes, message lengths, and number of virtual channels to validate the model.

The accuracy of the proposed model has been validated through a large set of simulation experiments for different scenarios defining different working conditions. However, for the sake of brevity, we report a few of them here. Figures 5, 6, 7, 8, and 9 show the results predicted by the proposed model against those obtained through simulation experiments, for the 6-, 8-, and 9-dimesnional hypercubes, with V=3 and 6 virtual channels, and different message lengths M=32, 64, and 128 flits. As can be seen in the figures, the proposed model has predicted the average message latency with good accuracy in low and moderate traffic loads. However, Approximations made for some parameters has resulted in some underestimation for the saturation point of the network, and reducing the accuracy of the model for heavy traffic regions slightly. Assuming that a network must work in the realistic working conditions (under low and moderate traffic loads) and not near the saturation region, we can conclude that the proposed model can predict the behaviour of the network with a good accuracy.

To exhibit the good accuracy of the proposed model, let us compare its accuracy against the only model [10] reported in the literature for P-cube routing. Ould-Khaoua [10], introduced a model for P-cube routing that suffers from low accuracy. This model is applicable only for very low traffic loads. It assumes a balanced traffic rate over network channels which we previously investigated it and saw that it is unbalanced over network channels. Figure 9 compares the results predicted by Ould-Khaoua's model and the proposed model here for an 8-dimensional hypercube with 3 virtual channels per physical channel and message length M=32, 64 and 128 flits. The inaccuracy of the old model [10] compared to the proposed model here can be easily seen in the figure 8.

5. Conclusions and Future Work

This paper has described an analytical model to compute the mean message latency in wormhole-routed hypercube networks using P-cube partially adaptive routing algorithm. Simulation experiments have revealed that the latency results predicted by the model are in good agreement with those obtained through simulation experiments. This model achieves a good degree of accuracy under different operation conditions as it computes the exact expression for the traffic rate on each channel and the probability of message blocking in any given channel of dimensions for all possible pairs of source- destination. Furthermore, the good degree of accuracy while maintaining acceptable simplicity, makes the proposed model a practical evaluation tool that can be used to gain insight into the performance behavior of partially adaptive routing in wormhole hypercubes. Our next objective is to extend our above modeling approach to deal with other partially adaptive routing algorithms such as planar routing and different traffic patterns, such as hotspots. Proposing models for partially adaptive routing in other popular networks, such as tori, meshes, and k-ary n-cubes, can also be a challenging future work in this line.

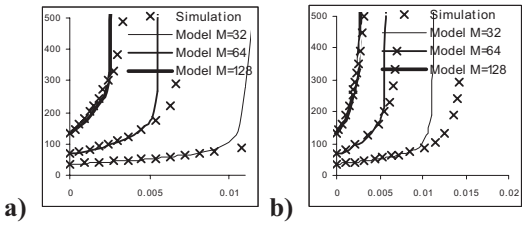

Figure 5. Average message latency predicted by model against simulation results for a) 3 b) 6 virtual channels per physical channel and M=32, 64, 128 flits in a 6 dimensional hypercube.

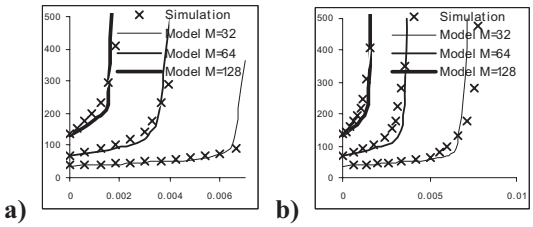

Figure 6. Average message latency predicted by model against simulation results for a) 3 b) 6 virtual channels per physical channel and M=32, 64, 128 flits in an 8 dimensional hypercube.

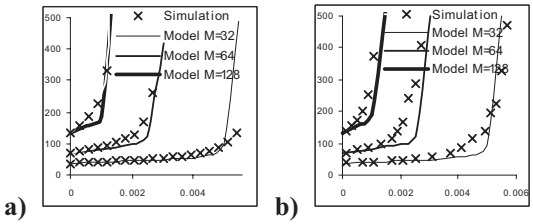

Figure 7. Average message latency predicted by model against simulation results for a) 3 b) 6 virtual channels per physical channel and M=32, 64, 128 flits in a 9 dimensional hypercube.

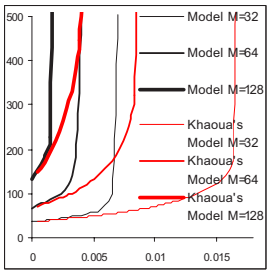

Figure 8. Comparison between the proposed model and Ould-Khaoua's model for an 8-dimensional hypercube, with 3 virtual channels per physical channel, and message lengths of *M*=32, 64, 128 flits.

References

[1] A. A. Chien, "A cost and performance model for k-ary n-cubes wormhole routers", IEEE TPDS 9(2) (1998) 150–162.

[2] A. A. Chien, J.H. Kim, "Planar-adaptive routing: low-cost adaptive networks for multiprocessors", proc. of international symp. on computer architecture, Journal of ACM 42(1), pp. 91-123, 1992.

[3] C. J. Glass and L. M. Ni, "The turn model for adaptive routing", Proc. 19th Int'l Symp. on Computer Architecture, pp. 278-287, 1992.

[4] B. Ciciani, M. Colajanni, C. Paolucci, "An accurate model for the performance analysis of deterministic wormhole routing", Proc. 11th Int. Parallel Processing Symp., pp. 353-359, 1997.

[5] J. T. Draper, J. Ghosh, "A Comprehensive analytical model for wormhole routing in multicomputer systems", J. Parallel & Distributed Computing 32, pp. 202-214, 1994.

[6] H. Sarbazi-Azad, "A mathematical model of deterministic wormhole routing in hypercube multicomputers using virtual channels", Journal of Applied Mathematical Modelling 27, 943-953, 2003.

[7] H. Sarbazi-Azad, A. Khonsari, M. Ould-Khaoua, "Analysis of k-ary n-cubes with dimension-ordered routing", Future Generation Computer Systems 19 (4), 493-502, 2003.

[8] J. Duato, S. Yalamanchili, L. Ni, Interconnection networks: an engineering approach, Morgan Kaufmann Publication, 2002.

[9] H. H. Najafabadi, H. Sarbazi-azad, P. Rajabzadeh, "Performance Modeling of Fully Adaptive Wormhole Routing in 2-D Mesh-Connected Multiprocessors", proc. of 12th International Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), 2004.

[10] M. Ould-Khaoua, "An approximate performance model for partially adaptive routing algorithm in hypercubes", Microprocessors & Microsystems 23 185-190, 1999.

[11] H. Sarbazi-Azad, M.Ould-Khaoua, L. M.Mackenzie, "An accurate analytical model of adaptive wormhole routing in k-ary n-cube interconnection networks", Performance Evaluation, Vol. 43, No. 2-3, pp. 165-179, 2001.

[12] H. Sarbazi-Azad, "Analytic Modeling of Channel Traffic in n-Cube Networks", Technical report, IPM School of computer Science, Tehran, Iran, 2005.

[13] M. Ould-Khaoua, H. Sarbazi-Azad, "An analytical model of adaptive wormhole routing in hypercubes in the presence of hot spot traffic", IEEE Trans. Parallel Distrib. Syst. 12(3), pp. 283-292, 2001.

[14] W. J. Dally, *"Virtual channel flow control"*, IEEE Trans. Parallel & Distributed Systems, Vol. 3, No. 2, pp. 194-205, 1992