Performance Modelling of Necklace Hypercubes

S. Meraji[1,2], H. Sarbazi-Azad[2,1], A. Patooghy[1,2]
[1] *IPM School of Computer Science &* [2] *Sharif University of Technology, Tehran, Iran*
{meraji, patooghy}@ce.sharif.edu, azad@ipm.ir

Abstract

The necklace hypercube has recently been introduced as an attractive alternative to the well-known hypercube. Previous research on this network topology has mainly focused on topological properties, VLSI and algorithmic aspects of this network. Several analytical models have been proposed in the literature for different interconnection networks, as the most cost-effective tools to evaluate the performance merits of such systems. This paper proposes an analytical performance model to predict message latency in wormhole-switched necklace hypercube interconnection networks with fully adaptive routing. The analysis focuses on a fully adaptive routing algorithm which has been shown to be the most effective for necklace hypercube networks. The results obtained from simulation experiments confirm that the proposed model exhibits a good accuracy under different operating conditions.

1. Introduction

A large number of interconnection networks have been proposed and studied for highly parallel distributed-memory multicomputers [3, 7, 8, 11, 14, 15, 17, 18, 26, 29, 30, 31]. Among them the hypercube has been one of the most famous ones which has many desirable properties such as logarithmic diameter and fault-tolerance. It is not, however, scalable from hardware cost point of view, i.e. when adding some few nodes to it, we have to duplicate the network size to reach the next specified network size. Other drawback with hypercubes was reported by Patel et al. [28] when considering VLSI layout. They showed that the minimum number of tracks for VLSI layout of an *n*-cube (*n*-dimensional hypercube) using a one-dimensional implementation has an order of network size. Their investigation revealed that for an example network of 1k processors (a 10-cube), at least 687 tracks are required for VLSI implementation.

The *necklace hypercube,* introduced in [23], is a new interconnection network based on the hypercube network. While preserving most of properties of the hypercube, it has some other desirable properties such as hardware scalability and efficient VLSI layout that make it more attractive than an equivalent hypercube network [23].

There are three main approaches for performance evaluation of interconnection networks. The first one is monitoring the behavior of the actual system; it can capture the effects of low-level design choices, but restricts experimentation with different router policies since it can be prohibitively expensive and time-consuming to change these features. Simulation is the second approach for performance evaluation of interconnection network. We may implement different routing algorithms, different switching methods, and different interconnection topologies with simulation environments, but simulation is time consuming especially when we study large networks. The last way is to using mathematical approaches for performance analysis of interconnection networks. Mathematical models are cost-effective and versatile tools for evaluating system performance 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 their behaviour under network configurations and working conditions which may not be feasible to study using simulation on conventional computers due to the excessive computation and memory demands.

Several researchers have recently proposed analytical models of popular interconnection networks, e.g. k-ary n-cubes, tori, hypercubes, and meshes [1][6][24]. The most difficult part in developing any analytical model of adaptive routing is the computation of the probability of message blocking at a given router due to the number of combinations that have to be considered when enumerating the number of paths that a message may have used to reach its current position in the network. Almost all studies on necklace hypercube interconnection networks focus on topological properties and algorithmic issues. There has been hardly any study on performance evaluation of such networks and no analytical model proposed for necklace hypercubes. In this paper, we discuss performance issues of necklace hypercube graphs by introducing a reasonably accurate mathematical model to predict the average message latency in wormhole necklace hypercubes using a high-performance routing algorithm proposed in [22].

The rest of this paper is organized as follows. In Section 2, the structure of the necklace hyeprcube is described. In Section 3, adaptive wormhole routing in the

1-4244-0910-1/07/$20.00 ©2007 IEEE

necklace hypercube is discussed. Section 4 proposes a mathematical performance model for adaptive routing in wormhole necklace hypercube. Validation of the proposed performance model is realized in Section 5 using results obtained from simulation experiments. Finally, Section 6 concludes the paper.

2. The necklace hypercube graph

The *necklace hypercube* is an undirected graph that is based on a hypercube by appending a necklace of processors (or nodes, interchangeably) to each edge. That is, besides connecting adjacent nodes (according to the base hypercube topology), we connect *i*-th dimension neighbors by an array of nodes, as a necklace. The necklace length may be fixed or variable for different edge necklaces. In the former, there are fixed number of nodes between each two adjacent neighbors on a necklace. The network is called *regular necklace hypercube*, and can be defined as (*n*, *k*)-*RNH*, where *n* is the number of dimensions and *k* is the necklace size. With fixed length necklaces, however, scaling up the network is limited to specified network sizes indicated by *n* and *k* as $2^{n-1}(nk+2)$. In the latter form of necklace hypercubes, named as *irregular necklace hypercube*, each necklace associated to channel *i* in the base hypercube, contains k_i nodes. Hence, the network can be defined by a vector of $n2^{n-1}$ elements, i.e. $k=(k_1,k_2,...,k_{n2^{n-1}})$. Such a network, denoted as $(k_1,k_2,...,k_{n2^{n-1}})-INH$, has excellent scalability with theoretically no limitations to scale. The network size for a $(k_1,k_2,...,k_{n2^{n-1}})-INH$ is given as $2^n+\sum_{i=1}^{n2^{n-1}} k_i$. The second factor in network size expression (the sigma part) implies the scalability property as we can have any desirable value for $k_i, 1\le i\le n2^{n-1}$, in the network.

Figures 1(a) and 1(b) show examples of a regular and irregular necklace hypercube. Dark nodes are the nodes in the base hypercube and the grey nodes are the necklace nodes. The index of each necklace node in its necklace is shown inside the node. Edge number of each edge in the base hypercube is shown inside a grey rectangle on it which is based on *i*-th dimension edge of smaller base vertex. Let us now give a more formal definition of the necklace hypercube.

Definition 1. A necklace hypercube is an undirected graph *G* = (*V, E*), where $u\in V$ is defined as *u* = (*b, d, i*) with *b* (denoting the *Base Vertex*) being the hypercube node address of the node with smaller address in its dimension, *d* (difference), $0\le d\le n$, is the dimension of the necklace (containing the node), and *i* (as the index) is the index of the node in its necklace. Note that *d* for the hypercube vertices (or base vertices) is zero. For simplicity, we index the nodes of a necklace from zero (zero for the base vertex) up to the number of nodes on the necklace. A (*n, k*)-RNH has $2^n+(n2^{n-1})\times k$ nodes (2^n nodes of degree $2n$ and $n2^{n-1}k$ nodes of degree 2), $n2^{n-1}+n2^{n-1}(k+1)$ edges, and a diameter of *n*+*k*. The bisection width of the necklace hypercube is twice as that of its base hypercube. As the bisection width of an *n*-dimensional hypercube is 2^{n-1}, the bisection width of the necklace hypercube is then 2^n [23].

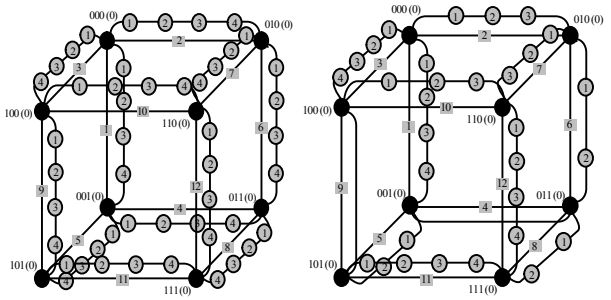

Fig. 1: a) A regular necklace hypercube with *n*=3 and *k*=4b) An irregular necklace hypercube.

One of the best features of this interconnection network is its efficient VLSI layout. According to the result given in [28] by Patel et al., the number of wiring tracks sufficient and necessary for the single-row wiring layout of the *n*-dimensional hypercube or *n*-cube, Q_n, when the numeric node order is used is given by [28]: $t_{num}(Q_n)=\lfloor(2/3)\times 2^n\rfloor+\lfloor n/2\rfloor$.

For a (*n, k*)-RNH, in general, we can extend this expression. Note that each necklace can be placed in just two extra tracks. So, the number of wiring tracks sufficient and necessary for the 2-dimensional wiring layout of the necklace hypercube, (*n, k*)-*RNH*, when the numeric node order is used is given by [23]: $t_{num}((n,k)-RNH)=2\times\left(\lfloor(2/3)\times 2^n\rfloor+\lfloor n/2\rfloor\right)$ that is independent of *k*. As can be seen, the VLSI layout for a (2,4)-RNH requires 6 tracks only. Note that (2,4)-RNH contains 20 nodes and the corresponding hypercube (a 4-cube) needs 12 tracks, i.e. is twice the number of tracks needed for its equivalent necklace network.

3. Adaptive wormhole routing in star graphs

In this section, we introduce a fully adaptive routing algorithm for the necklace hypercube already proposed in [22]. The algorithm can be used with both packet switching and wormhole switching techniques. To define a fully adaptive routing algorithm, we must first define a

deadlock-free routing (with any level of adaptivity). We can then use the deadlock-free routing algorithm as described in [22] to construct a fully adaptive routing algorithm.

In order to have a deadlock-free routing algorithm in the necklace hypercube, we use two classes of virtual channels A and B. Virtual channels of class A are used when the message is at the source necklace; we use virtual channels of this class until we reach the first base vertex of the source necklace. Virtual channels of class B are used in the destination necklace; it means that when we enter the destination necklace, we use this class of virtual channels until we reach to the destination node. Each of these classes can be used for a deadlock-free routing algorithm in the base hypercube, e.g. e-cube routing. The minimum number of virtual channels in each class is 1. Thus, we need at least 2 virtual channels per physical channel to implement a deadlock-free routing algorithm in necklace hypercubes.

According to Duato's methodology, since the base deadlock-free routing algorithm requires 2 virtual channels, we can have a fully adaptive deadlock-free routing algorithm in the necklace hypercube using at least 3 virtual channels, two of which used by the base routing algorithm and the remaining one used in any possible way that can brings the message closer to the destination node. We call the virtual channels used for base deadlock-free routing as the *base virtual channels* and the remaining virtual channels as the *adaptive virtual channels*.

When there are more than 3 virtual channels per physical channel, the network performance is maximized when the extra virtual channels are added to adaptive virtual channels. Thus, with V virtual channels per physical channel, the best performance is achieved when we have V-2 adaptive virtual channels and two base virtual channels. This is of course for routing in necklaces; in the base hypercube we can use Duato's fully adaptive routing algorithm [13] which uses V-1 adaptive virtual channels and 1 virtual channel for e-cube deterministic routing.

The proposed routing algorithm contains three main steps:

1. Move towards the nearest base neighbor using any of the V-2 adaptive virtual channels. If all V-2 virtual channels are busy use the virtual channel of class A from base virtual channels to move toward the nearest base neighbor in the source necklace.
2. Move towards the nearest neighbor of the destination using fully adaptive routing algorithm in hypercube with V-1 virtual channels [13]. If all V-1 virtual channels are busy use the remaining virtual channel with e-cube routing in the base hypercube.
3. Now the current node is one of the base vertices of the destination necklace. Move towards the destination node using any of the V-2 virtual channels. If all V-2 virtual channels are busy use the virtual channel of class B from base virtual channels to move toward the destination node in the destination necklace.

4. The analytical model

In this section, we derive an analytical performance model for wormhole adaptive routing in a necklace hypercube. Our analysis focuses on the routing algorithm which was introduced in previous section (described in [22]) but the modelling approach used here can be equally applied for other routing schemes after some few changes.

The measure of interest in our model is the *average message latency* as a representative for network performance. The following assumptions are made when developing the proposed performance model. These assumptions have been widely used in similar modelling studies [1, 6, 9, 12, 16, 20, 24, 27].

a) Messages are broken into some packet of fixed length of M flits which are the unit of switching. The flit transfer time between any two neighbouring nodes is assumed to one cycle.

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

c) Nodes generate traffic independently of each other, which 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) V virtual channels per physical channel are used. These virtual channels are used according to the routing algorithm described in the previous section.

According to the proposed fully adaptive routing algorithm, a message must cross three sub-networks from the source node to reach the destination node. At first, it moves from the source node to nearest base node, then moves to the nearest base node of the destination node in the base hypercube, and finally crosses the destination necklace to reach the destination. Therefore, we have 3 message latencies. First, the mean message latency at the source necklace, $Latency_N$, then, the mean message latency at the base hypercube, $Latency_H$, and finally the mean message latency at the destination necklace, $Latency_N$. Hence, the overall mean message latency can be written as:

$$Latency = Latency_H + 2 \times Latency_N. \quad (1)$$

In order to compute the mean message latency in each of the three sub-networks, we must consider three parameters: the mean network latency, $\overline{S}$, that is the time to cross the network, the mean waiting time seen by a message in the source node to be injected into the

network, $\overline{W_s}$. To model the effect of virtual channels multiplexing effects, the mean message latency is then scaled by a factor, $\overline{V}$, representing the average degree of virtual channels multiplexing that takes place at a given physical channel [10]. Therefore, the mean message latency in each sub-network can be approximated as

$$Latency = (\overline{S} + \overline{W_s})\overline{V}. \quad (2)$$

The average number of hops that a message makes across the necklace sub-network, $\overline{d_N}$, can be computed as follows:

$$\overline{d_N} = \frac{1}{k}\sum_{i=1}^{\lfloor k/2 \rfloor} 2i \quad (3)$$

a value of $\frac{1}{k}\lceil k/2 \rceil$ must be added to expression 3 if the necklace length is odd.

The average number of hops that a message makes across the hypercube sub-network, $\overline{d_H}$, is given by [6]

$$\overline{d_H} = \frac{n}{2} \times \frac{2^n}{2^n - 1}. \quad (4)$$

Figure 2 shows a necklace of (2,5)-RNH. In this figure all the messages that are generated in node 3 cross the edge (2,3) except those that their destination are nodes 1, 2 or 3. If we consider the messages that are generated in nodes 4 and 5 and their corresponding destinations are nodes 1, 2 and 5 (all of them most cross the edge (3,2)) the exact message traffic rate over this edge can be expressed as

$$\lambda_g \times \frac{N-4}{N-1} + \lambda_g \frac{2}{N-1} + \lambda_g \frac{1}{N-1} = \lambda_g. \quad (5)$$

We can compute the traffic over the edge (2,1) as follows. All messages generated in node 3 cross the edge (2,1) except those that their destination are nodes 2, 4 and 5. Also, all messages generated in node 2 and their destinations are not nodes 3, 4 or 5, cross this edge. If we consider the messages generated in node 4 to node 1, we can write the traffic rate over this channel as

$$\lambda_g \frac{N-4}{N-1} + \lambda_g \frac{N-4}{N-1} + \lambda_g \frac{1}{N-1} \cong 2\lambda_g. \quad (6)$$

Using a similar approach, the traffic arte over the channels of a necklace with k nodes can be written as $\lambda_g, 2\lambda_g, 3\lambda_g, \ldots, \frac{k}{2}\lambda_g$. We have similar results for the edges in other direction [32]. Therefore, we can compute the average traffic rate received by each necklace channel, λ_{c_N}, as

$$\lambda_{c_N} = \lambda_g \frac{\sum_{i=1}^{k/2} i}{k/2} = (k/2+1)\frac{\lambda_g}{2}. \quad (7)$$

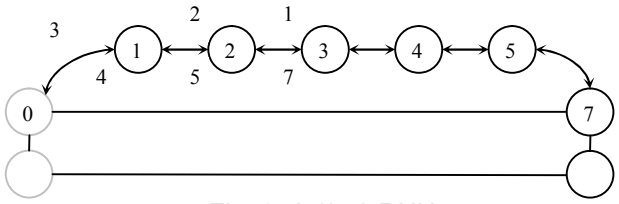

Fig. 2: A (2,5)-RNH

The traffic rate over the edge that connects the last necklace node to the base node is $\frac{k}{2}\lambda_g$. As each base node has n necklaces and the traffic rate generated in a base node itself is λ_g, we can compute the traffic rate injected in the base hypercube by a base node as

$$\lambda_b = \lambda_g + n\frac{k}{2}\lambda_g. \quad (8)$$

Fully adaptive routing in the base hypercube allows a message to use any available channel that brings it closer to its destination node, resulting in an evenly distributed traffic rate over hypercubic channels. A router in the base hypercube has n output channels. Since each message travels, on average, $\overline{d}$ hops to cross the network, the rate of messages received by each hypercubic channel, λ_{c_H}, can be expressed as [6]

$$\lambda_{c_H} = \frac{\lambda_b N}{2(N-1)}. \quad (9)$$

Let us follow a typical message which makes $\overline{d}$ hops to reach its destination. The average network latency, $\overline{S}$, 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, $\overline{S}$, can be expressed as

$$\overline{S} = M - 1 + \overline{d} + \overline{d}T_b \quad (10)$$

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

$$T_b = P_{block} w \quad (11)$$

with P_{Block} being the probability that a message is blocked at the current channel and w is the mean waiting time to acquire a channel in the event of blocking. A message is blocked at a given channel in the necklace sub-network when all adaptive and deterministic virtual channels of the current physical channel are busy. 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. In necklace sub-networks, we have only one path for the messages, so no adaptivity exists for messages moving across a necklace.

In order to compute $P_{a\&d\,N}$ (probability $P_{a\&d}$ for necklaces), we must consider two cases:
(a) the probability that all of V virtual channels of a physical channel are busy, P_V, and (b) the probability that V-1 virtual channels of the V virtual channels associated to a physical channel are busy. In this case only one combination of V-1 virtual channels of the total V virtual channels can result in blocking.
So the probability that all of the adaptive and deterministic virtual channels of a physical channel are busy can be expressed as

$$P_{a\&d_N} = P_V + \frac{P_{V-1}}{\binom{V}{V-1}}. \tag{12}$$

In order to compute the probability of blocking, P_{block}, we average over all the probabilities that a message may be blocked crossing $\overline{d}$ hops in the network as

$$P_{block_N} = \frac{1}{\overline{d_N}} \times \sum_{i=1}^{\overline{d_N}} P_{a\&d_N}. \tag{13}$$

A message is blocked at a given channel in the hypercube sub-network when all the adaptive virtual channels of the remaining dimensions to be visited and also the deterministic virtual channel of the current dimension are busy. We can write P_{a_H}, $P_{a\&d_H}$ and P_{block_H} as follows [6]:

$$P_{a_H} = P_V + \frac{P_{V-1}}{\binom{V}{V-1}} \tag{14}$$

$$P_{a\&d_H} = P_V \tag{15}$$

$$P_{block_H} = \frac{1}{\overline{d_H}} \sum_{i=0}^{\overline{d_H}-1} \left(P_{a_H}\right)^{\overline{d_H}-i-1} P_{a\&d_H} \tag{16}$$

where $\overline{d_H}$ is the average number of hops that a message might take in the hypercube sub-network.

To determine the mean waiting time, w, to acquire a virtual channel when a message is blocked, a physical channel is treated as an M/G/1 queue with a mean waiting time of [21]

$$w = \frac{\rho \overline{S}(1+C_{\overline{S}}^2)}{2(1-\rho)} \tag{17}$$

$$\rho = \lambda_c \overline{S} \tag{18}$$

$$C_{\overline{S}}^2 = \frac{\sigma_{\overline{S}}^2}{\overline{S}^2} \tag{19}$$

where λ_c is the traffic rate on the channel (given by equation 7 or 9 for necklace or hypercube channels), $\overline{S}$ is its service time calculated by equation 10, and $\sigma_{\overline{S}}^2$ 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 given in [12], the variance of the service time distribution can be approximated as $\sigma_{\overline{S}}^2 = (\overline{S} - M)^2$. Hence, the mean waiting time becomes

$$w = \frac{\lambda_c \overline{S}^2 (1 + (1 - M/\overline{S})^2)}{2(1 - \lambda_c \overline{S})} \tag{20}$$

Similarly, modelling the local queue in the source node as an M/G/1 queue, with the mean arrival rate λ_g / V and service time $\overline{S}$ with an approximated variance $(\overline{S} - M)^2$ yields the mean waiting time seen by a message at the source node as [21]

$$W_s = \frac{\frac{\lambda_g}{V} \overline{S}^2 (1 + (1 - M/\overline{S})^2)}{2(1 - \frac{\lambda_g}{V} \overline{S})} \tag{21}$$

the probability, P_v, that v virtual channels are busy at a physical channel can be determined using a Markovian model. State π_v ($0 \le v \le V$) corresponds to v virtual channels being busy. The transition rate out of state π_v to state π_{v+1} is the traffic rate λ_c (given by equations 7 and 9) while the rate out of state π_v to state π_{v-1} is $\frac{1}{\overline{S}}$ ($\overline{S}$ is given by equation 10). The transition rates out of state π_v are reduced by λ_c to account for the arrival of messages while a channel is in this state.

The Markovian model results in the following steady state probability [21], in which the service time of a channel has been approximated as the network latency of that channel, as

$$P_v = \begin{cases} (1 - \lambda_c \overline{S})(\lambda_c \overline{S})^v, & 0 \le v < V \\ (\lambda_c \overline{S})^v, & v = V. \end{cases} \tag{22}$$

when multiple virtual channels are used per physical channel, they share the physical bandwidth in a time-multiplexed manner. The average degree of multiplexing of virtual channels, that takes place at a given physical channel, can then be estimated by [10]

$$\bar{V} = \frac{\sum_{v=1}^{V} v^2 p_v}{\sum_{v=1}^{V} v p_v}. \quad (23)$$

The above equations reveal that there are several inter-dependencies between the different variables of the model. For instance, Equations 10 and 11 reveal that $\bar{S}$ is a function of w while equation 17 shows that w is a function of $\bar{S}$. Given that closed-form solutions to such inter-dependencies are very difficult to determine, the different variables of the model are computed using an iterative technique.

5. Validation of the model

The proposed analytical model has been validated through a discrete-event simulator (Xmulator [25]) that mimics the behaviour of the described routing algorithms in the necklace hypercube network at flit level. The simulator uses the same assumptions as the analysis, and some of these assumptions are detailed here with a view to 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 from the generation of a message until the last data flit reaches the local processor at the destination node. The other measures include the mean network latency, the time taken to cross the network, the mean queuing time at the source node, and the time spent at the local queue before entering the first network channel. Numerous validation experiments have been performed for several combinations of network sizes, message lengths, and number of virtual channels to validate the model.

Figures 3 depict latency results predicted by the model explained in the previous section, plotted against those provided by the simulator for different sized necklace hypercubes. The horizontal axis in the figure shows the traffic generation rate at each node while the vertical axis shows the mean message latency. In figures 3-a and 3-b, we consider two large networks (about 1000 nodes) with V=8 virtual channels per physical channel, and two different message lengths of M=32 and 64 flits. Figures 3-c and 3-d represent the same results for medium sized networks (about 256 nodes); here we have V=6 virtual channels per physical channel and message of length M=32, 64 and 128 flits. Finally in figures 3-e and 3-f, we have shown a comparison between the results given by the model and those gathered from the simulation experiments for small networks (about 100 nodes); here the number of virtual channels is V=4 per physical channel and message length is M=32, 64 and 128 flits.

The figures reveal that in all cases the analytical model can predict the mean message latency with a good degree of accuracy in the steady-state regions. Moreover, the model predictions are still good even when the network operates in the heavy traffic region, and when it starts to approach the saturation traffic region. However, some discrepancies around the saturation point are apparent. These can be accounted for by the approximations made to ease the derivation of different variables of the model, e.g. the approximation made to estimate the variance of the service time distribution at a channel. Such an approximation greatly simplifies the model as it allows us to avoid computing the exact distribution of the message service time at a given channel, which is not a straightforward task due to inter-dependencies between service times at successive channels as wormhole routing relies on a blocking mechanism for flow control.

6. Conclusion and future work

The necklace hypercube network has recently been introduced as an attractive alternative to the well-known hypercube. However, most of studies in this line have focused on topological properties and algorithmic aspects of these networks. In this paper, we introduced the first mathematical performance model of adaptive wormhole routing in necklace hypercubes and validated it through simulation experiments. We saw that the proposed model manages to achieve a good degree of accuracy while maintaining simplicity, making it a practical evaluation tool that can be used by the researchers in the field to gain insight into the performance behaviour of fully adaptive routing in wormhole-switched necklace hypercube.

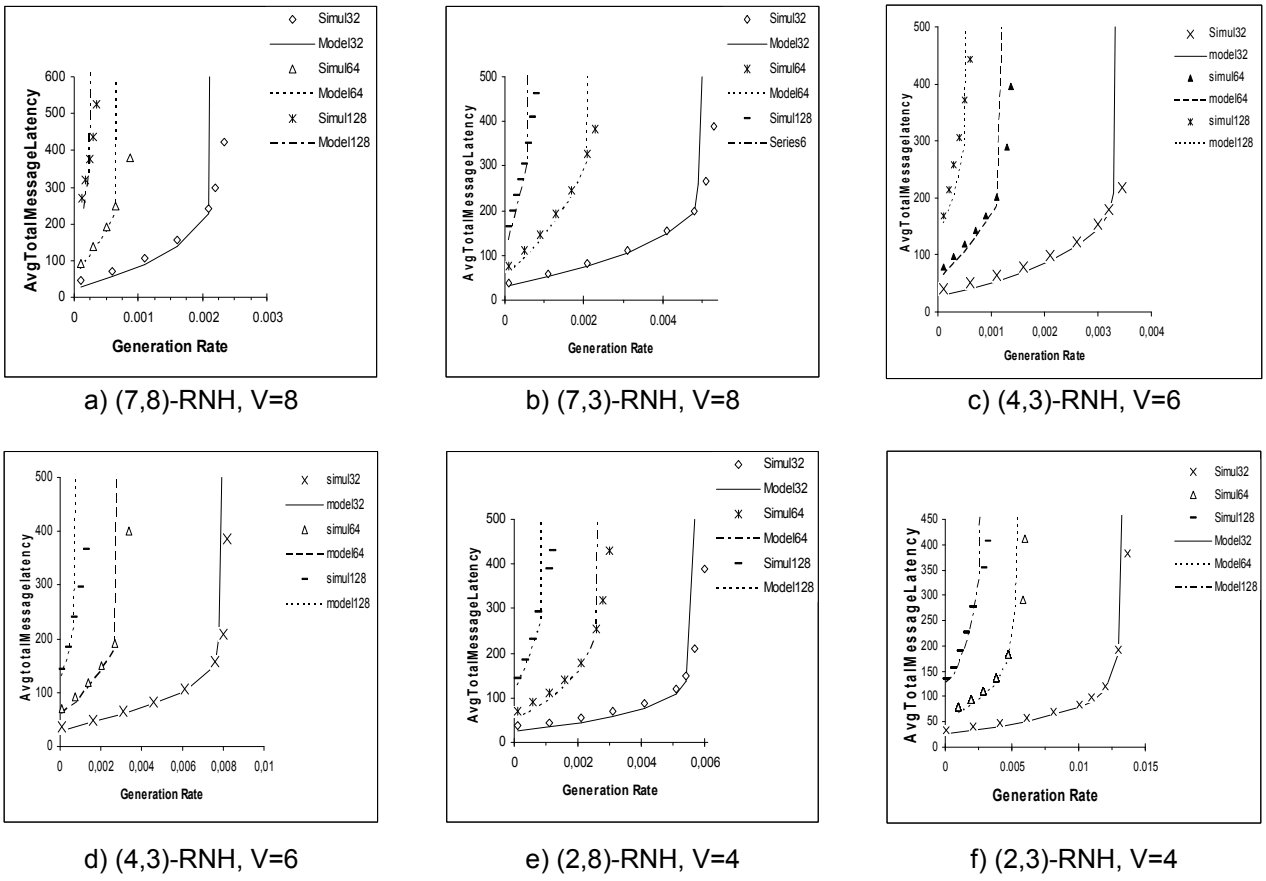

Fig. 3: The average message latency predicted by the model against simulation results for different sized networks.

7. References

[1] S. Abraham and K. Padmanabhan. Performance of the direct binary n-cube networks for multiprocessors. IEEE Transactions on Computers, 37(7):1000-1011, 1989.

[2] A. Agarwal. Limits on interconnection network performance. *IEEE Transactions on Parallel and Distributed Systems*, 2(4):398-412, 1991.

[3] Berthome P., Ferreira A. and Perennes S., ptimal Information Dissemination in Star and Pancake Networks IEEE Trans. on Parallel and Distributed Systems, vol. 7, no. 12, pp. 1292-1300, 1996.

[4] R. V. Boppana and S. Chalasani. A Comparison of Adaptive Wormhole Routing Algorithms. *Proceedings of the 20th Annual International Symposium on Computer Architecture (ISCA'93)*, pages 351-360, 1993.

[5] R. V. Boppana and S. Chalasani, A Framework for Designing Deadlock-Free Wormhole Routing Algorithms. *IEEE Transactions on Parallel and Distributed Systems*, 7(2):169-183, 1996.

[6] Y. Boura, C. R. Das and T. M. Jacob. A performance model for adaptive routing in hypercubes. *Proceedings of the International Workshop on Parallel Processing*, pages 11-16, 1994.

[7] Chen C., and Chen J., Optimal Parallel Routing in Star Networks, IEEE Trans. Computers, vol. 46, no. 12, pp. 1293-1303, 1997.

[8] Chen C., and Chen J., Nearly Optimal One-to-many Parallel Routing in Star Networks, IEEE Trans. on Parallel and Distributed Systems, vol. 8, no. 12, pp. 1196-1202, 1997.

[9] B. Ciciani, M. Colajanni and C. Paolucci. An accurate model for the performance analysis of deterministic wormhole routing. *Proceedings of the 11th International Parallel Processing Symposium*, pages 353-359, 1997.

[10] W. J. Dally. Virtual channel flow control. *IEEE Transactions on Parallel and Distributed Systems,* 3(2):194-205, 1992.

[11] K. Day and A. Tripathi. A Comparative Study of Topological Properties of Hypercubes and Star Graphs. *IEEE Transactions on Parallel and Distributed Systems*, 5(1):31–38, 1994.

[12] J. T. Draper and J. Ghosh. A Comprehensive analytical model for wormhole routing in multicomputer systems. *Journal of Parallel and Distributed Computing* 23(2): 202-214, 1994.

[13] Duato J., Yalamanchili C., Ni L., Interconnection Networks: an engineering approach", IEEE computer society press, 1997.

[14] Fujita, S., Neighborhood Information Dissemination in the Star Graph, IEEE Trans. on Computers, vol. 49, no. 12, pp. 1366-1370, 2000.

[15] Graham S. W., and Seidel S. R., The Cost of Broadcasting on Star Graphs and k-ary hypercubes, IEEE Trans. Computers, vol. 42, no. 6, pp. 756-759, 1993.

[16] R. Greenberg and L. Guan. Modelling and comparison of wormhole routed mesh and torus networks. *Proceedings of the 9th IASTED International Conference on Parallel and Distributed Computing and Systems*, pages 501-506, 1997.

[17] Jwo J. S., Lakshmivarahan S. and Dhall S. K., Embedding of Cycles and Grids in Star Graphs, Journal of Circuits, Systems and Computers, vol. 1, no. 1, pp. 43-74, 1991.

[18] Lee J. I., and Chang H., Embedding Complete Binary Trees in Star Graphs, Journal of the Korea Information Science Society, vol. 21, no. 2, pp. 407-415, 1994.

[19] R. E. Kessler and J. L. Schwarzmeier. CRAY T3D: A new dimension for Cray Research. *CompCon*, pages 176-182, 1993.

[20] J. Kim and C. R. Das. Hypercube communication delay with wormhole routing. *IEEE Transactions on Computers,* 43(7):806-814, 1994.

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

[22] Meraji S., Sarbazi-Azad H., Nayebi A., Deterministic routing in necklace hypercubes, Technical Report, School of Computer Science, IPM, Tehran, Iran, 2006.

[23] Monemizadeh M., and Sarbazi-Azad, H., The necklace hypercube: a well scalable hypercube-based interconnection network for multiprocessors, ACM SAC 2005, pp.729-733, 2005.

[24] H. H. Najafabadi, H. Sarbazi-Azad and P. Rajabzadeh. Performance Modeling of Fully Adaptive Wormhole Routing in 2-D Mesh-Connected Multiprocessors. *Proceedings of the 12th Annual Meeting of the IEEE / ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'04),* pages 528-534, 2004.

[25] A. Nayebi, S. Meraji, A. shamaei, Using listener-based integration to develop a simulation platform for interconnection networks – Xmulator, Sharif university of Technology, www.xmulator.com.

[26] Nigam M., Sahni S. and Krishnamurthy B., Embedding Hamiltonians and Hypercubes in Star Interconnection Graphs, Proc. Intl. Conf. Parallel Processing, vol. 3, pp. 340-343, 1990.

[27] H. Sarbazi-Azad, M. Ould-Khaoua and L. M. Mackenzie. An accurate analytical model of adaptive wormhole routing in *k*-ary *n*-cube interconnection networks. *Performanc*

[28] Patel A., Kusalik A. and McCrosky C., Area-Efficient VLSI Layout for Binary Hypercubes, IEEE Trans. on Computers, vol. 49, no. 2, 2000.

[29] Rezazad M., Sarbazi-Azad H., A Constraint-Based Performance Comparison of Hypercube and Star Multicomputers with Failures, 19th International Conference on Advanced Information Networking and Applications (AINA 2005), pp. 841-846, 2005.

[30] Sheu J. P., Liaw W. H. and Chen T. S., A Broadcasting Algorithm in Star Graph Interconnection Networks, Information Processing Letters, vol. 48, no. 5, pp. 237-241, 1993.

[31] Tseng Y. C., Chen Y. S., Juang T. Y. and Chang C. J., Congestion-Free, Dilation-2 Embedding of Complete Binary Tree in Star Graphs, Networks, vol. 33, no. 3, pp. 221-231, 1999.

[32] S. Meraji, Performance evaluation of Necklace Hypercube Interconnection Network, M.Sc. Thesis, Sharif University of Technology, October 2006.