# A Comparative Performance Analysis of *n*-Cubes and Star Graphs

Abbas Eslami Kiasari[1,2], Hamid Sarbazi-Azad[1,2]

[1]IPM School of Computer Science
Tehran, Iran
{kiasari,azad}@ipm.ir

[2]Sharif University of Technology
Dept. of Computer Engineering
Tehran, Iran

## Abstract

*Many theoretical-based comparison studies, relying on the graph theoretical viewpoints with using structural and algorithmic properties, have been conducted for the hypercube and the star graph. None of these studies, however, considered real working conditions and implementation limits. We have compared the performance of the star and hypercube networks for different message length and virtual channels and considered two implementation constraints, namely the constant bisection bandwidth and constant node pin-out. We use two accurate analytical models already proposed for the star graph and hypercube and implement the parameter changes imposed by technological implementation constraints. The comparison results reveal that the star graph has a better performance compared to the equivalent hypercube under light traffic loads while the opposite conclusion is reached for heavy traffic loads. The hypercube with more channels compared to its equivalent star graph saturates later showing that it can bear heavier traffic loads.*

## 1. Introduction

One of the most popular and well-known network topologies is indeed the hypercube. Its popularity stems from the fact that a large number of processors ($2^n$ processors) can be interconnected using a small number of communication links ($n$ links per processor) while at the same time keeping the communication delay between processors at a minimum. In addition, the hypercube is a completely symmetric topology and, consequently, minimizes congestion problems. It also permits the use of identical processors since every vertex plays an identical role in the topology. Additionally a number of algorithms have been designed to run on the hypercube. These algorithms have demonstrated the usefulness of the structure and symmetry properties inherent in the graph and how they may be exploited. The simplicity of these routing algorithms allows us to design low cost routing hardware for the hypercube. With a rich connectivity, the hypercube can efficiently tolerate most traffic workloads especially global traffic loads.

Considering this list of attractive properties of the hypercube, its popularity is not surprising. However, many of these properties of the hypercube are, in fact, group theoretic properties possessed by a large class of networks called cayley graphs [5]. In fact, it was shown that some cayley graphs not only possess all these properties but even offer a better degree and diameter than the hypercube. The most important candidate of such a family of graphs is the star graph. The star graph has been introduced in [4] as an attractive alternative for the hypercube. It has many desirable properties such as topological symmetry, low diameter, low degree, and recursive structure [4][5].

Comparative studies, e.g. [10], of the star graph and the hypercube were mainly realized based on topological and algorithmic properties of these two networks. However, besides these structural properties, implementation constraints and working conditions in real world have great impacts on the overall performance of machines. Ignoring such important issues, when comparing two network topologies, may lead us to wrong conclusions [15].

In this paper, we compare stars and hypercubes taking into account some more practical issues including implementation constraints from a performance point of view. To the best of our knowledge, no implementation-based comparison has been reported in the literature to conduct such a performance comparison between stars and hypercubes at the various operating conditions. The rest of paper is organized as follows. Section 2 describes the structure and some fully adaptive wormhole routing algorithms in the two networks. In Section 3, the analytical models, for the star graph and hypercube, are generally introduced (exact equa-

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

tions are reported in the appendix). Section 4 presents and star graph under technological implementation constraints. Finally, Section 5 concludes the paper.

## 2. Preliminaries

### 2.1. The Hypercube and Star Graph

The two networks considered in this study are the hypercube and the star graph. Let us first introduce them briefly. A $n$-dimensional hypercube can be modelled as a graph $H_n$ ($V$, $E$), with the node set $V_n$ and edge set $E_n$, where $|V_n|=2^n$ and $|E_n|=n2^n$ nodes. The $2^n$ nodes are distinctly addressed by $n$-bit binary numbers, with values from 0 to $2^n - 1$. Each node has link at $n$ dimensions, ranging from 1 (lowest dimension) to $n$ (highest dimension), connecting to $n$ neighbours. An edge connecting nodes $X= x_n x_{n-1}\ldots x_1$ and $Y= y_n y_{n-1}\ldots y_1$ is said to be at dimension $j$ or to the $j$th dimensional edge if their binary addresses $x_n x_{n-1}\ldots x_1$ and $y_n y_{n-1}\ldots y_1$ differ at bit position $j$ only, i.e. $x_j \neq y_j$. An edge in $H_n$ can also be represented by an $n$-character string with one hyphen (-) and $n$-1 binary symbols {0, 1}. For example in a $H_4$, the string 00-1 denotes the edge connecting nodes 0001 and 0011.

Star graph has more complex structure rather binary $n$-cube. Let $V_n$ be the set of all $n!$ permutations of symbols 1, 2, 3,… , $n$. For any permutation $v \in V_n$ if we denote the $i^{th}$ symbol of $v$ by $v_i$, $v$ can be written as $v_1 v_2 \ldots v_n$. A *star graph* defined on $n$ symbols, $S_n$ = ($V_n$,$E_n$), is an undirected graph with $n$! nodes, where each node $v$ is connected to $n - 1$ nodes which can be obtained by interchanging the first and $i^{th}$ symbols of $v$, i.e. $[v_1 v_2 \ldots v_i v_{i+1} \ldots v_n, v_i v_2 \ldots v_1 v_{i+1} \ldots v_n] \in E_n$, for $2 \leq i \leq n$. We call these $n - 1$ connections as *dimensions*. Thus each node is connected to $n - 1$ nodes through dimensions 2, 3, … ,$n$. The $S_n$ is also called an $n$-star.

Figure 1 shows the 4-dimensional star and hypercube, $S_4$ and $H_4$. The star graph is an attractive alternative to the hypercube, and compares favourably with it in several aspects [4][5]. For example, the degree of $S_n$ is $n - 1$, i.e. sub-logarithmic in the number of nodes of $S_n$ while a hypercube with $\Theta(n!)$ nodes has degree $\Theta(\log n!) = \Theta(n \log n)$, i.e. logarithmic in the number of nodes. The same can be said about the diameter of $S_n$. Much work has been done to study both the topological properties and parallel algorithms of the star graph lately.

### 2.2. Adaptive wormhole routing

In this research, we use fully routing algorithms for hypercube and star interconnection networks. According to this methodology, virtual channels are divided into *adaptive* (class $a$ with $V_1$ virtual channel) and *deadlock free* (class $b$ with $V_2$ virtual channel) virtual sub-networks. At each 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 a deadlock-free routing algorithm. 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 $a$. Network performance is maximized when the extra virtual channels are added to adaptive virtual channels in class $a$, the best performance is achieved when class $b$ contains minimum required virtual channels and extra virtual channels are allocated to class $a$.

In hypercube network, class $b$ virtual channels can be traversed according to the dimension order routing algorithm [3][9]. Therefore, messages traverse class $b$ virtual channels one dimension at a time in an increasing order. When more than 2 virtual channels per physical channel are used, network performance is optimized when 1 virtual channel is classified in class $b$ and the remaining virtual channels are classified in class $a$.

Several fully adaptive routing algorithms for the star graph have been evaluated in [12] of which the one using negative hop-based (*NHop*) [7] deadlock free routing, augmented with a new idea called bonus card has shown to have the best performance.

In NHop algorithm, the network is partitioned into several subsets, such that no subset contains adjacent nodes (this is equivalent to the well-known graph colouring problem). If $C$ is the number of subsets, then the subsets are labelled as 0, 1, …, $C$-1, and nodes in subset $i$ are labelled (or coloured) as $i$. A hop is a negative hop if it is from a node with a higher label to a node with a lower label; otherwise, it is a positive hop. A message occupies a buffer of virtual channel $i$ at an intermediate node if and only if the message has taken exactly $i$ negative hops to reach that intermediate node. If $H$ is the diameter of the network and $C$ is the number of colours, then the maximum number of negative hops that can be taken by a message is $H_N = \lceil H(C-1)/C \rceil$ [6][7].

The star graph, $S_n$, is a bipartite graph, and its nodes can be partitioned into two subsets; therefore, it can be coloured using only two colours [7]. Because adjacent nodes are in distinct partitions, the maximum number of negative hops a message may take is at most half the diameter of $S_n$, which equals $\lceil H/2 \rceil = \lceil \lfloor 3(n-1)/2 \rfloor / 2 \rceil$. Hence, negative-hop schemes with $\lceil \lfloor 3(n-1)/2 \rfloor / 2 \rceil$

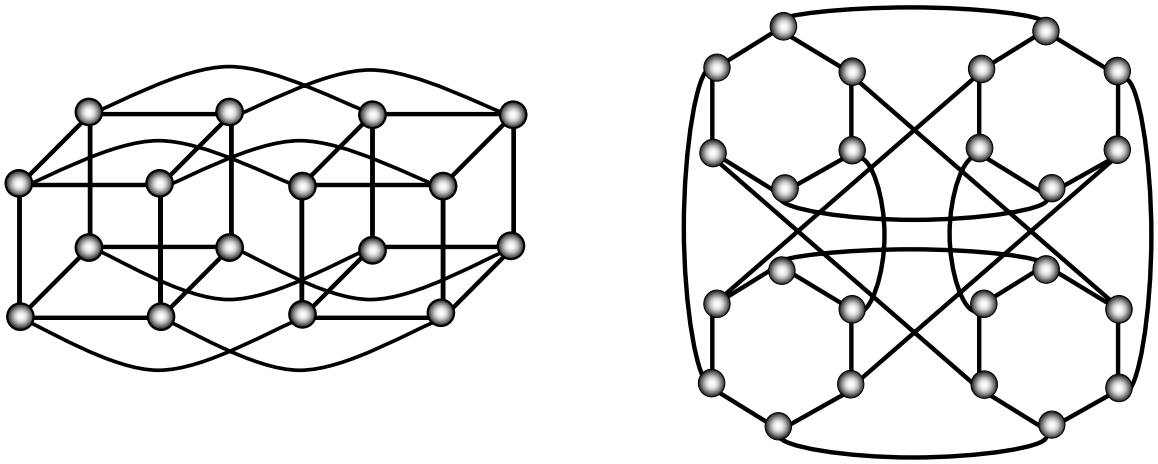

**Figure 1. The 4-dimensional hypercube (left) and 4-dimensional star graph (right).**

virtual channels per physical channel can be designed for $S_n$. The NHop algorithm has an unbalanced use of virtual channels because messages start their journey starting from virtual channel 0. However, very few messages take the maximum number of hops and use all the virtual channel $0...\lceil \lfloor 3(n-1)/2 \rfloor / 2 \rceil$, and thus virtual channels with high numbers will be used rarely. The NHop scheme can be improved by giving each header flit a *number bonus card* [6]. For negative-hop scheme it is equal to the number of virtual channel level minus the number of required negative hops to reach the destination node. At each node, the header flit has some flexibility in the selection of virtual channels. The range of virtual channels that can be selected for each physical channel is equal to the number of bonus cards available plus one [7]. The resulting deadlock-free routing algorithm using negative-hop routing scheme and the bonus card is named *Nbc* which has been evaluated and investigated against other routing algorithms for the star graphs in [12]. The Nbc routing scheme has been used and a routing algorithm, named *Enhanced-Nbc* with high performance and minimum virtual channel requirements was resulted [12]. Investigations showed that Enhanced-Nbc has a better performance [12] compared to other algorithms reported in the literature and the other algorithms proposed in [13]. Thus, in our analysis in the next sections, we use Enhanced-Nbc routing for the star graph and the best fully adaptive routing algorithm for the hypercube (used in many studies), as the two best available fully adaptive routing algorithms for the candidate networks.

## 3. Analytical models

In this section, we highlight important parts and main common formulas for the 2 analytical performance models for the hypercube and star graph, already proposed in [8] and [11] respectively. Theses models capture the behaviour of adaptive routing algorithms (described in previous section) and virtual channel flow control that multiplexes a number of virtual channels over physical channels. The main equations of the analytical models are derived as follows.

First, the mean network latency, $\overline{S}$, that is the time to cross the network 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 using an M/G/1 queuing system. Finally, the average degree of multiplexing, $\overline{V}$, is determined using a Markov model, and the computed average latency is inflated by the multiplexing degree in order to obtain the effective average latency. Therefore, the mean message latency can be written as

$$Latency = (\overline{S} + \overline{W_s})\overline{V} . \quad (1)$$

Since the hypercube and star graph are symmetric, averaging the network latencies seen by the messages generated by only one node for all other nodes gives the mean message latency in the network. Let $S$ be the source node with linear address 0 and $i$ denotes linear address of the destination node, where $1 \le i \le N\text{-}1$ ($N$ is the number of nodes in the network). The network latency, $S_i$, seen by the message crossing from node 0 to node $i$ 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, $S_i$ can be written as

$$S_i = (M + d_i)t_w + \sum_{k=1}^{d_i} wP_{block_{i,k}} , \quad (2)$$

where $M$ is the message length, $d_i$ is the distance between the node 0 and node $i$, $t_w$ is the transfer time of a flit across a physical channel, $P_{block_{i,k}}$ is the average probability blocking seen by a message form node 0 to node $i$ on its $k^{th}$ hop, and $w$ is the mean waiting time when blocking occurs. Averaging over all the possible destination nodes for a typical message yields the mean network latency as

$$\overline{S} = \frac{\sum_{i=1}^{N-1} S_i}{N-1}. \quad (3)$$

Details of the analytical models used in this study are not reported here for the sake of brevity. However, interested reader can find the details of the very-recently proposed star graph model in [11], while derivation of the hypercube model can be found in [8]. Equations of both models are listed in the Appendix.

## 4. Performance comparison

Extensive examination of interconnection networks has been conducted over the last decade, both with a view to studying fundamental graph-theoretic properties and feasibility of implementation in various technologies. The latter consideration is of crucial importance since in practice implementation technology puts bandwidth constraints on network channels, and these are important factors in determining how well the theoretical properties of a particular network topology can be exploited. When systems are implemented on a single VLSI-chip, the wiring density of the network mainly determines the overall system cost and performance [9]. For instance, Dally [9] has used the bisection width, i.e. the number of wires that cross the middle of the network, as a rough measure of the network wiring density in a pure VLSI implementation.

Other researchers, including Abraham [2] and Agrawal [3], have conducted similar studies to Dally's and arrived at the same conclusion. However, they have also argued that while the wiring density constraint is certainly applicable where an entire network is implemented on a single VLSI-chip, this is not the case in the currently more realistic situation where a network has to be partitioned over many chips. In such circumstances, they have identified that the most critical bandwidth constraint is imposed by the chip's I/O pins through which any data entering or leaving the chip must travel.

### 4.1. The effect of implementation constraints

Due to the limited channel bandwidth imposed by implementation technology, a message is broken into channel words (or phits), each of which is transferred in one cycle. If the channel width (i.e. number of wires) is $C_w$ bits, a message of $B$ bits is divided into $M=B/C_w$ phits [9]. Let us define $n_1$ and $n_2$ to be, respectively, the dimensions of the equivalent hypercube and star graph, i.e. $N = 2^{n_1} = n_2!$. The bisection width of the hypercube and star graph, $B_{hypercube}$ and $B_{star}$, with a channel width, $C_{w_{hypercube}}$ and $C_{w_{star}}$, can be expressed as [9] [10] [14]

$$B_{hypercube} = 2 \times \frac{N}{2} \times C_{w_{hypercube}} = N \times C_{w_{hypercube}} \quad (4)$$

and

$$B_{star} = \begin{cases} 2 \times N \times \frac{n_2}{4(n_2-1)} \times C_{w_{star}}, & \text{if } n_2 \text{ is even} \\ 2 \times N \times \frac{n_2^2-3}{4n_2(n_2-2)} \times C_{w_{star}}, & \text{if } n_2 \text{ is odd} \end{cases} \quad (5)$$

If the bisection width is held fixed, the relationship between channel widths in the hypercube and star graph, is given by

$$\mu_{\text{bisection width}} = \frac{C_{w_{hypercube}}}{C_{w_{star}}} = \begin{cases} \frac{n_2}{2(n_2-1)}, & \text{if } n_2 \text{ is even} \\ \frac{n_2^2-3}{2n_2(n_2-2)}, & \text{if } n_2 \text{ is odd} \end{cases} \quad (6)$$

Therefore, $\mu_{\text{bisection width}}$ is the factor by which the channel cycle of the hypercube grows compared to that in an equivalent star, when bisection width is held fixed for both networks.

Similarly, in multiple-chip implementation, where a complete node is fabricated on a chip, pin-out, which is the number of I/O pins (i.e. node degree × channel width), is a more suitable metric. The node pin-out for the hypercube and star graph, $P_{hypercube}$ and $P_{star}$, can be written as

$$P_{hypercube} = n_1 C_{w_{hypercube}} \quad (7)$$

and

$$P_{star} = (n_2 - 1) C_{w_{star}} \quad (8)$$

Assuming a constraint of constant node pin-out, the channel cycle ratio of the hypercube and its equivalent star graph, when node pin-out is fixed in two networks, can be given as

$$\mu_{\text{pin-out}} = \frac{C_{w_{hypercube}}}{C_{w_{star}}} = \frac{n_2 - 1}{n_1} \quad (9)$$

### 4.2. The comparative analysis

The measure used in our comparison is the average message latency given by equation (1). To achieve it, we assume that the channel width of the star graph is

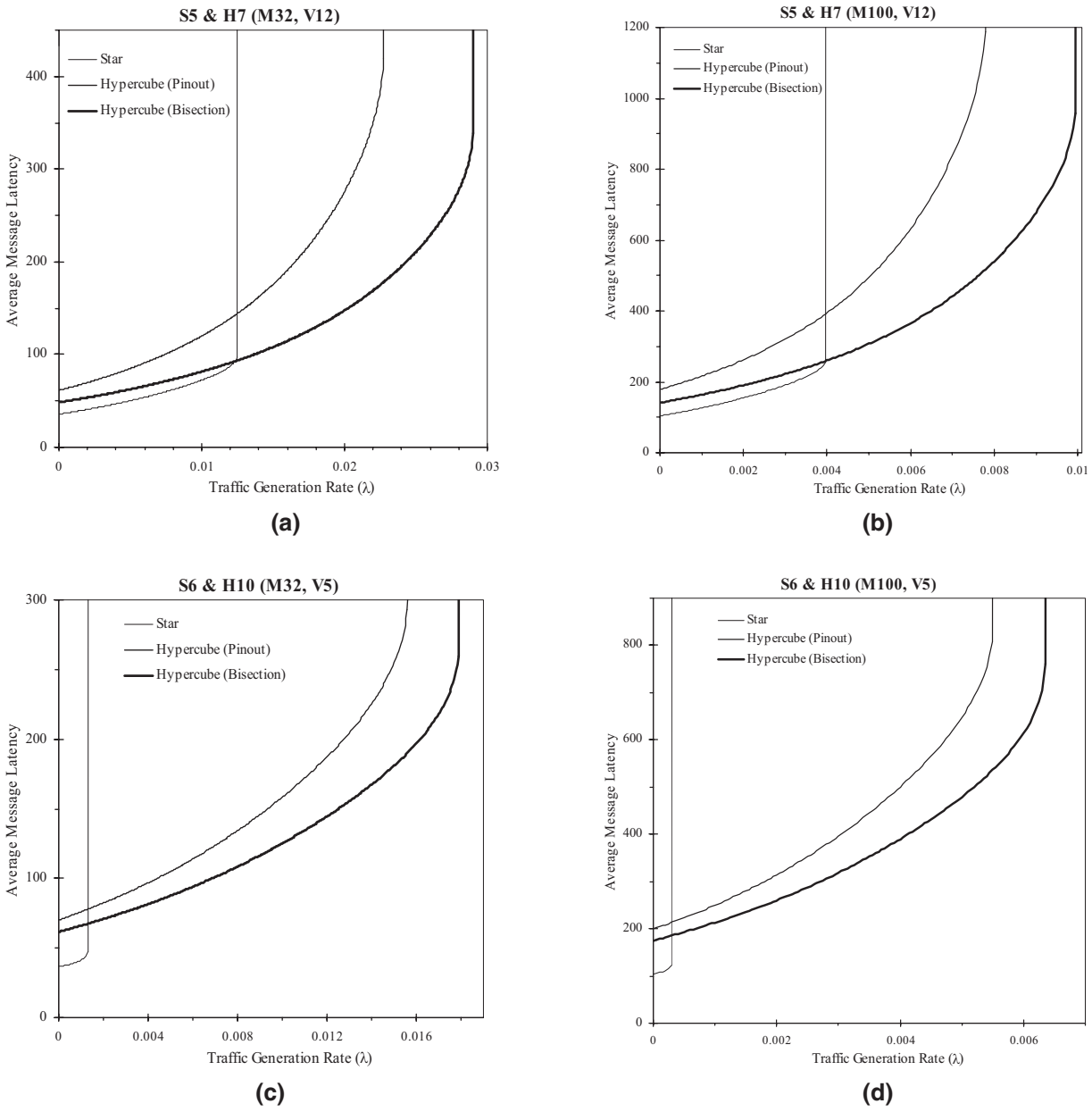


**Figure 2. The average message latency in the star and hypercube for different network sizes and topologies with fixed bisection width and constant node pin-out (a) for $S_5$ and $H_7$ with message length 32 and 12 virtual channels, (b) for $S_5$ and $H_7$ with message length 100 and 12 virtual channels, (c) for $S_6$ and $H_{10}$ with message length 32 and 5 virtual channels, (d) for $S_6$ and $H_{10}$ with message length 100 and 5 virtual channels.**

fixed at flit length and find the relative channel width of equivalent hypercube using equations 6 and 9.

Figure 2 illustrates the results obtained for the two networks of different sizes, number of virtual channels and message lengths. As the channel cycle of the star graph is assumed to be fixed at the unit time and the channel cycle of the hypercube is normalized to that in the star, the average message latency curve for the star graph will be the same for both implementation constraints, the constant bisection bandwidth and constant pin-out. Therefore, for brevity, we have shown the average message latency curves for the hypercube under different implementation constraints and the curve for

the star network in one graph. In each graph, the hori zontal axis denotes the traffic generation rate and the vertical axis shows the calculated average message latency.

It can be seen in the figures that the average message latency in the star graph is less that in the hypercube only for low traffic rates. When traffic load becomes heavier and under both implementation constraints the hypercube network exhibits a better performance compared to the equivalent star graph. Note that when a constant bisection width constraint is used, the difference is even more noticeable. Such a performance superiority of the star network is mainly because of its lower diameter and channel bandwidth compared to the equivalent hypercube.

## 5. Conclusion

The star graph has been proposed as an attractive alternative to the hypercube. Most comparison studies between hypercubes and stars used fundamental graph-theoretic properties and none of them, to our best knowledge, considered feasibility of implementation in various technologies when comparing these two networks. Two important implementation constraints are used in this study to conduct a fair comparison between the two networks.

We used fully adaptive routing for both networks and examined their relative performance for different network sizes, number of virtual channels, and message lengths, with both constant bisection bandwidth constraint and constant pin-out constraint. The performance measure of use in our comparison was the average message latency. The results for both implementation constraints revealed that the hypercube exhibits a better performance than the star graph under heavy traffic loads while the conclusion achieved in light traffic regions is on opposite.

## 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] S. Abraham and K. Padmanabhan. Performance of Multicomputer Networks under Pin-out Constraints. *Journal of Parallel and Distributed Computing*, 12(13):237-248, 1991.

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

[4] S. B. Akers, D. Harel and B. Krishnamurthy. The Star Graph: An Attractive Alternative to the *n*-cube. *Proceedings of International Conference on Parallel Processing*, pages 393-400, 1987.

[5] S. B. Akers and B. Krishnamurthy. A Group-Theoretic Model for Symmetric Interconnection Network. *IEEE Transactions on Computers*, 38(4): 555–565, 1989.

[6] 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.

[7] 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.

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

[9] W. J. Dally. Performance Analysis of *k*-arry *n*-cubes Interconnection Networks. *IEEE Transactions on Computers*, C-39(6):775-785, 1990.

[10] 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.

[11] A. E. Kiasari, H. Sarbazi-Azad and M. Ould-Khaoua. Analytical Performance Modelling of Adaptive Wormhole Routing in the Star Interconnection Network. *Proceedings of the 20th IEEE International Parallel and Distributed Processing Symposium* (*IPDPS'06*)*,* 2006.

[12] A. E. Kiasari, H. Sarbazi-Azad and M. Rezazad. Performance Comparison of Adaptive Routing Algorithms in the Star Interconnection Network. *Proceedings of the 8th International Conference on High Performance Computing in Asia Pacific Region* (*HPC-Asia'05*)*,* pages 257-264, 2005

[13] J. V. Misic and Z. Jovanovic. Routing Function and Deadlock Avoidance in a Star Graph Interconnection Network. *Journal of Parallel and Distributed Computing,* 22(2):216-228, 1994.

[14] S. Ponnuswamy and V. Chaudhary. A Comparative Study of Star graph and Rotator Graphs. *International Conference on Parallel Processing*, pages 46-50, 1994.

[15] H. Sarbazi-Azad. Performance analysis of wormhole routing in multicomputer interconnection networks. PhD Thesis, Department of Computing Science, University of Glasgow, U.K., 2002.

## **Appendix:** *Equations for the analytical models*

(a) The hypercube model

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

$$\overline{S} = \sum_{i=1}^{n} p_i S_i$$

$$p_i = \frac{\binom{n}{i}}{2^n - 1}$$

$$S_i = (M + i)t_w + \sum_{j=1}^{i} B_j$$

$$B_i = P_{block_i} w$$

$$P_{block_i} = P_V (P_V + \frac{P_{V-1}}{V})^{i-1}$$

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

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

(b) The star graph model

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

$$\overline{S} = \frac{\sum_{i=1}^{n!-1} S_i}{n!-1}$$

$$S_i = (M + d_i)t_w + \sum_{k=1}^{d_i} B_{i,k}$$

$$B_{i,k} = P_{block_{i,k}} w$$

$$P_{block_{i,k}} = \frac{1}{N_{set_i}} \sum_{j=1}^{N_{set_i}} P_{block_{i,j,k}}$$

$$P_{block_{i,j,k}} = \left( P_{block_A} + \frac{1}{2} P_{block_{B^-}} + \frac{1}{2} P_{block_{B^+}} \right)^{f(i,j,k)}$$

$$P_{block} = \sum_{v=V_1+1}^{V} P_v \frac{\binom{V_2 - 1}{v - V_1 - 1}}{\binom{V}{v}}$$

$$P_{block_A} = \sum_{v=V-\lceil d/2 \rceil + 1}^{V} P_{vc_0} P_v \frac{\binom{\lceil d/2 \rceil - 1}{v - V + \lceil d/2 \rceil - 1}}{\binom{V}{v}}$$

$$P_{block_{B^-}} = \sum_{l=1}^{V_2 - \lceil d/2 \rceil} \sum_{v=V_1+n_l}^{V} P_{vc_l} P_v \frac{\binom{V_2 - n_l}{v - V_1 - n_l}}{\binom{V}{v}}$$

$$P_{block_{B^+}} = \sum_{l=1}^{V_2 - \lceil d/2 \rceil + 1} \sum_{v=V_1+n_l+1}^{V} P_{vc_l} P_v \frac{\binom{V_2 - n_l - 1}{v - V_1 - n_l - 1}}{\binom{V}{v}}$$

$$P_{VC_l} = \begin{cases} \dfrac{V_1}{V - \lceil d/2 \rceil + 1}, & l = 0, \\ \dfrac{1}{V - \lceil d/2 \rceil + 1}, & 1 \le l \le V_2 - \lceil d/2 \rceil + 1. \end{cases}$$

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

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