HOT-SPOT CONTENTION IN BINARY HYPERCUBE NETWORKS

Sivarama P. Dandamudi and Derek L. Eager

SCR-TR-155, April 1989

School of Computer Science, Carleton University
Ottawa, Canada, KIS 5B6

Hot-Spot Contention in Binary Hypercube Networks

Sivarama P. Dandamudi
School of Computer Science
Carleton University
Ottawa, Ontario K1S 5B6
Canada

Derek L. Eager
Department of Computational Science
University of Saskatchewan
Saskatoon, Saskatchewan S7N 0W0
Canada

ABSTRACT

In large parallel processing systems there is a possibility of many processors requesting access to the same data item, or "service" by the same processor, at the same time. Such hot-spot contention creates congestion in the system. Hot-spot contention has been studied previously in the context of shared-memory multiprocessor systems. This paper considers the impact of hot-spot contention in distributed-memory multicomputer systems. Since the binary hypercube network offers a reasonable compromise between network cost and performance, two hypercube-based static interconnection networks are considered. One is a standard binary hypercube and the other is a binary hypercube-based hierarchical interconnection network. The impact of hot-spot contention in both these networks is studied. Two techniques have been proposed for shared-memory multiprocessor systems to increase the system capacity in handling hot-spot references: *hardware combining*, and *software combining*. The effectiveness of the two corresponding approaches for multicomputer systems is studied here.

Key words: Binary hypercubes, Hardware combining, Hot-spot contention, Multicomputers, Software combining.

1. Introduction

In large parallel processing systems there is a possibility of many processors requesting access to the same data item, or "service" by the same processor, at the same time. Such hot-spot contention creates congestion in the system. In shared-memory multiprocessor

systems, the memory module containing the "hot-spot" data item may become saturated. In distributed-memory multicomputer systems, communication links leading to the "hot-spot" processor may become saturated.

Hot-spot contention has been studied previously in the context of multiprocessor systems that utilize buffered multistage interconnection networks to connect processors to the shared memory modules ([Pfister & Norton 1985], [Yew *et al.* 1987], [Lee *et al.* 1986]). It has been observed that the access times for all memory references may be severely degraded, not just the references to a hot-spot location, owing to a phenomenon termed *tree saturation* [Pfister & Norton 1985]. In tree saturation, all switches within a "tree" rooted at the hot memory and extending to the switches connected to the processors become "saturated" in that buffer queues in these switches fill to capacity. This degrades overall performance, particularly in large systems. For example, in a system with 1000 processors, with only 0.125% hot spot traffic (i.e., with only 0.125% of all memory references directed at a hot-spot data item) the potential memory system throughput is reduced by half [Pfister & Norton 1985].

Two schemes have been previously proposed to increase the capacity of a system in handling concentrated references to a hot-spot -- *hardware combining* and *software combining*. (Alternative approaches that attempt only to limit the adverse effects of such references on other references also exist [Rettberg & Thomas 1986], [Ho & Eager 1989].) Both these schemes are based on essentially the same underlying principle of combining several hot-spot requests into a single request and forwarding this request to the hot-spot memory. These two schemes, however, differ in the way this combining is accomplished. In hardware combining, which was originally proposed for use in the IBM RP3 [Pfister *et al.* 1985], and which is used in the NYU Ultracomputer design [Gottlieb *et al.* 1983], when several requests directed at the same memory location meet at a switch, they are combined by the switch into a single request. When the response from memory returns, the switch hardware generates a response for each of the combined requests. It is estimated [Pfister & Norton 1985] that the extra hardware to support combining increases the switch

size and cost by factors of between 6 and 32 for combining networks consisting of 2×2 switches. For $k \times k$ switches (k>2), the hardware cost will be even greater.

Software combining [Yew *et al.* 1987] does not require any hardware modifications to the switch. A tree of data items is created with the original hot-spot item serving as the root of this tree. Each processor is assigned one of the leaf data items to operate on. When a certain condition is satisfied by a leaf data item, this fact is suitably reflected by one of the processors to which that data item was assigned by modifying the parent data item. This process repeats until the root data item is modified suitably. Communication may also proceed down the tree in the natural way. It has been shown that this technique is effective in decreasing memory contention and preventing tree saturation in a multistage interconnection network. Further details can be found in [Yew *et al.* 1987].

This paper is concerned with hot-spot contention in distributed-memory multicomputer systems. Processors in such systems communicate by explicit message passing. It is easy to see that hot-spots can also exist in multicomputer systems. For example, synchronization (such as barrier type synchronization) may be required in the applications that execute on these machines. In this situation, the processor which is coordinating the synchronization activity may become a "hot-spot" node. This paper studies the impact of hot-spot contention in multicomputer systems that use binary hypercube-based static interconnection networks. Such networks have been used in several commercial multicomputer systems such as the Intel iPSC [Rattner 1985] and the NCUBE/ten [Hayes *et al.* 1986].

Two types of hypercube-based static interconnection networks are considered; the binary hypercube, or BH and a type of binary hypercube-based "hierarchical interconnection network" (HIN), BH/BH. HINs can be informally described as follows [Dandamudi & Eager 1987]. Let N denote the total number of nodes in the network. These N nodes are grouped into K_1 clusters of $n_1 = N/K_1$ nodes each. It is assumed here, for convenience of analysis, that K_1 evenly divides N, and that each cluster is of identical size, although in practice clusters of varying sizes could be permitted. Each cluster of n_1 nodes is linked together by a level 1 interconnection network. One node from each cluster

is selected to act as an interface node, and these K_1 interface nodes are again grouped into K_2 clusters of $n_2 = K_1/K_2$ nodes each. (As before, it is assumed that K_2 evenly divides K_1, and that all clusters are of the same size.) A level 2 interconnection network is used to link each of these K_2 clusters of n_2 level 1 interface nodes. If desired, one node from each level 2 cluster could be selected as a level 2 interface node, and these nodes grouped into clusters, etc. Figure 1 shows an example of the type of HIN considered here; a two level "BH/BH" network in which both levels use a binary hypercube (BH) network.

An advantage of HINs is that they reduce the degree (i.e., the number of links connected to a node) of the majority of nodes, and this should allow greater packing densities and larger systems. The degree of the remaining nodes is the same as that of the corresponding non-hierarchical network. As an example, consider a binary hypercube (BH) network with $N = 2^D$ nodes. The degree of each node in this network is D. Now consider a two-level BH/BH HIN with a cluster size of $n = 2^d$ and the number of clusters $K = 2^{D-d}$. In this HIN, only $\frac{N}{2^d}$ nodes will have a degree of D (the interface nodes); the remaining nodes will have only a degree of d. If N = 1024 (i.e., D = 10) and n = 8 (i.e., d = 3), then the BH/BH HIN will have 896 nodes with degree 3 and 128 nodes with degree 10. This is in contrast to a degree of 10 for all 1024 nodes in the BH network.

These binary hypercube-based interconnection networks differ from the multistage interconnection networks, utilized in shared-memory multiprocessor systems, in several respects:

(i) The networks considered here are characterized by multiple paths between any pair of nodes. In contrast, the multistage interconnection networks of previous hot-spot studies have only one path between any source and destination. The results reported in this paper indicate that multiple paths allow improved routing algorithms to yield substantial performance improvements in a system suffering from hot-spot contention.

(ii) The distance between communicating entities is not fixed in the hypercube-based

interconnection networks. In contrast, multistage interconnection networks maintain a constant distance.

(iii) In shared-memory multiprocessor systems, the messages are memory references. Therefore, message combining can be achieved relatively easily. In multicomputer systems, message combining may be more involved, depending on the complexity of the semantics of the "combinable" message types. Certainly, the transport mechanism must be given some knowledge of these semantics.

The remainder of this paper is organized as follows. Section 2 gives details on the hot-spot and locality models used in this study. Sections 3 and 4 present results for networks with unbounded and bounded buffering capacity, respectively. Section 5 concludes the paper by summarizing the results.

2. The Workload and System Models

The two characteristics of communication in a distributed-memory multicomputer that are of interest in this study are the *communication locality* and the *hot-spot proportion* (the proportion of traffic that is directed towards a hot-spot). Each is modelled by a single parameter: α (defined below) is used to model locality, and a parameter h (defined below) is used to model the hot-spot proportion.

In our model of communication locality, nodes are conceptually divided into equal-sized clusters. (These conceptual clusters become physical clusters in the BH/BH network.) Locality is measured by the probability α (cluster-size dependent) that a message is a nonhot-spot message destined for a node within the same cluster as the source node. It is assumed that:

- Intra-cluster (except hot-spot) communication is uniformly random (i.e., an intra-cluster message is destined to each node within the cluster with equal probability).
- Inter-cluster (except hot-spot) communication is uniformly random (i.e., an inter-cluster message is destined to each cluster other than the source cluster with equal probability, and to each node within the destination cluster with equal probability).

Hot-spots can be modelled in several ways [Yew *et al.* 1987]. In this study, we assume that there is only a single hot-spot and use the hot-spot model of Pfister and Norton [1985]. In this model, each message has a probability h of being *hot-spot* message (a combinable message destined for the hot-spot node) and a probability (1-h) of being a *regular* message (assumed here to be a non-combinable). Here, the regular messages are further divided into *intra-cluster* and *inter-cluster* messages. When a message reaches its destination node, it is assumed that a reply is generated. Thus, there are hot-spot replies, intra-cluster replies, and inter-cluster replies. When each nonhot-spot node generates messages at rate λ, it generates hot-spot messages at rate $h\lambda$, intra-cluster messages at rate $\alpha\lambda$, inter-cluster messages at rate $(1-\alpha-h)\lambda$, intra-cluster replies at rate $\alpha\lambda$, and inter-cluster replies at rate $(1-\alpha-h)\lambda$. The hot-spot node generates messages and replies at the same rates, except that it also generates hot-spot replies at rate $Nh\lambda$. (Allowing a node to send messages to itself is for convenience in analysis only; since we model only link congestion, such messages have no effect on performance.) Note that h must be less than or equal to $(1-\alpha)$.

In the analysis presented in Section 3, each communication link is assumed to be full-duplex, and is conceptually replaced by two half-duplex links. Links are classified according to whether they join nodes in the same cluster ("cluster links") or in different clusters ("non-cluster links"). Each link is modelled as a queueing centre. The following assumptions are made about the network and its workload:

(*i*) The time between successive message generations at each node is exponentially distributed with mean $1/\lambda$.

(*ii*) The node message generation processes are independent of each other.

(*iii*) Message service times are exponentially distributed; each cluster link i processes messages at rate μ_{CL} while each non-cluster link processes messages at rate μ_{NCL}.

(*iv*) Packet-switching is used for message transmission. (It is assumed that a message fits within a single packet.)

(*v*) All messages are routed over a randomly chosen shortest path between the source and

destination nodes.

These assumptions allow each link to be modelled as an M/M/1 queueing centre [Kleinrock 1975].

3. Performance with Unbounded Buffering Capacity

In this section it is assumed that each node has unbounded message buffering capacity. The impact of hot-spot contention on the average message delivery time or "delay" R is derived analytically in Section 3.1. Sections 3.2 and 3.3 study the performance improvements that may be possible with hardware combining and software combining schemes, respectively. These three sections assume a routing algorithm that selects randomly among shortest paths. The impact of an improved routing algorithm is studied through simulation in Section 3.4.

3.1. Delay Analysis

To simplify the analysis Kleinrock's independence assumption [Kleinrock 1976] is used, in which it is assumed that the service time of a message on successive links from source to destination are statistically independent. Further, although a reply is generated, according to the workload model, every time a message reaches its destination node, it is assumed in the analysis that replies are generated independently by each node by a Poisson process at the same rate at which the node receives messages. The results of the simulation experiments presented in Section 3.1.3 indicate that these assumptions are reasonable.

3.1.1. Analysis of the BH network

This section considers the BH network with 2^D nodes. In order to model communication locality, the nodes are conceptually divided into clusters of size 2^d. All nodes whose addresses have identical most significant (D-d) bits belong to the same cluster. Each link is labelled with a number, ranging from 1 through D, that depends on its

distance from the hot-spot node. A cluster link is labelled j (referred to as a j-level cluster link) if the link connects nodes S and T such that node S is separated from the hot-spot node by j-1 links and node T is separated from the hot-spot node by j links. A j-level non-cluster link is defined similarly.

Let $\lambda_{CL}(j)$ and $\lambda_{NCL}(j)$ denote the effective arrival rates of messages and replies at a j-level CL link and j-level NCL link, respectively. The former is given by

$$\lambda_{CL}(j) = \lambda'_{CL}(j) + \lambda''_{CL}(j) + \lambda'''_{CL}(j) \quad (1)$$

where

$\lambda'_{CL}(j)$ = arrival rate (at a j-level CL link) due to intra-cluster messages and replies,

$\lambda''_{CL}(j)$ = arrival rate (at a j-level CL link) due to inter-cluster messages and replies, and

$\lambda'''_{CL}(j)$ = arrival rate (at a j-level CL link) due to hot-spot messages and replies.

Note that the assumptions in the workload model imply that the traffic due to regular messages and replies is uniformly distributed over the CL links. Thus, $\lambda'_{CL}(j)$ and $\lambda''_{CL}(j)$ are independent of j. An expression is now derived for each of these.

Consider a cluster of 2^d nodes. Each of these nodes generates intra-cluster messages at rate $\alpha\lambda$, and these messages travel an average distance of $d/2$ links. Intra-cluster replies are generated at the same rate and they also travel the same average distance. Therefore,

$$\lambda'_{CL}(j) = \frac{2^d \, \alpha\lambda d}{2d \, 2^{d-1}} = \alpha\lambda \quad (2)$$

Similarly $\lambda''_{CL}(j)$ can be derived as

$$\lambda''_{CL}(j) = (1 - \alpha - h)\,\lambda \quad (3)$$

Considering now $\lambda'''_{CL}(j)$, note that the links closer to the hot-spot node carry more hot-spot messages and replies than those that are farther away. Let $p_1(j)$ be the probability that a hot-spot message or reply gets routed through a particular j-level link on its way to or from the hot-spot node. A set of links J, all of whose element links are j-level links, act as intermediate links for all hot-spot messages generated by any source node that is at distance at least j from the hot-spot node. Therefore, the number of source nodes for which an element of J acts as an intermediate link is $2^D - \sum_{k=0}^{j-1}\binom{D}{k}$. Since there are $(D - j + 1)\binom{D}{j-1}$ such intermediate links (i.e., the number of elements in J),

$$p_1(j) = \frac{2^D - \sum_{k=0}^{j-1}\binom{D}{k}}{(D-j+1)\binom{D}{j-1}2^D} = \frac{\sum_{k=j}^{D}\binom{D}{k}}{(D-j+1)\binom{D}{j-1}2^D}$$

Then, $\lambda'''_{CL}(j)$ can be computed as

$$\lambda''_{CL}(j) = 2^D h\lambda p_1(j) \tag{4}$$

Combining Eqs. (1), (2), (3) and (4) yields

$$\lambda_{CL}(j) = (1-h)\lambda + 2^D h\lambda p_1(j) \tag{5}$$

The arrival rate of messages and replies at a j-level NCL link can be expressed as

$$\lambda_{NCL}(j) = \lambda'_{NCL}(j) + \lambda''_{NCL}(j) \tag{6}$$

where

$\lambda'_{NCL}(j)$ = arrival rate (at a j-level NCL link) due to inter-cluster messages and replies, and

$\lambda''_{NCL}(j)$ = arrival rate (at a j-level NCL link) due to hot-spot messages and replies.

The traffic due to inter-cluster messages and replies is uniformly distributed over all the NCL links. Each inter-cluster message or reply travels, on average, P_{icl} links, which is given by

$$P_{icl} = \frac{\sum_{i=1}^{D-d}\sum_{j=0}^{d}\left\{\binom{D-d}{i}\binom{d}{j}(i+j)\right\}}{2^D - 2^d}$$
$$= \frac{D\,2^{D-1} - d\,2^{d-1}}{2^D - 2^d}$$

The number of NCL links used by an inter-cluster message or reply, on average, is (P_{icl} - $d/2$). Since inter-cluster messages and their replies are each generated at rate $(1-\alpha-h)\lambda$ by each node and since there are $2[D\,2^{D-1} - 2^{D-d}\,d\,2^{d-1}] = 2(D-d)2^{D-1}$ NCL links, $\lambda'_{NCL}(j)$ is given as

$$\lambda'_{NCL}(j) = \frac{2(1-\alpha-h)\lambda 2^D\,[P_{icl} - d/2]}{2(D-d)2^{D-1}}$$
$$= (1-\alpha-h)\,\lambda\frac{2^D}{2^D - 2^d} \qquad (7)$$

Since the derivation of $\lambda'''_{CL}(j)$ is independent of whether the j-level link is a CL or NCL link,

$$\lambda''_{NCL}(j) = \lambda'''_{CL}(j) \qquad (8)$$

Eqs. (6), (7), and (8) yield

$$\lambda_{NCL}(j) = (1-\alpha-h)\,\lambda\frac{2^D}{2^D - 2^d} + 2^D h\lambda p_1(j) \qquad (9)$$

Since each link is modelled as an M/M/1 queueing centre, the average delays encountered by messages and replies at j-level CL and NCL links are given by:

$$\Delta_{CL}(j) = \frac{1}{\mu_{CL} - \lambda_{CL}(j)} \qquad (10)$$

and

$$\Delta_{NCL}(j) = \frac{1}{\mu_{NCL} - \lambda_{NCL}(j)} \qquad (11)$$

Denoting the average total delay of a regular message or reply by R_{reg}, the average total delay of a hot-spot message or reply by R_{hot}, and the overall total average delay by R_{avg}, one can derive:

$$R_{reg} = \frac{1}{(1-h)}\left[\alpha\left[\frac{d}{2}\right]\Delta^{avg}{}_{CL} + (1-\alpha-h)\left\{\left[\frac{d}{2}\right]\Delta^{avg}{}_{CL} + \left(P_{icl} - \frac{d}{2}\right)\Delta^{avg}{}_{NCL}\right\}\right]$$

$$= \frac{1}{(1-h)}\left[\alpha\left[\frac{d}{2}\right]\Delta^{avg}{}_{CL} + (1-\alpha-h)\left\{\left[\frac{d}{2}\right]\Delta^{avg}{}_{CL} + \left(\frac{(D-d)2^{D-1}}{2^D - 2^d}\right)\Delta^{avg}{}_{NCL}\right\}\right] \quad (12)$$

$$R_{hot} = \sum_{k=0}^{D-d}\left[\binom{D-d}{k}\left\{\sum_{l=1}^{d}(d-l+1)\binom{d}{l-1}\Delta_{CL}(k+l)\,p_1(k+l)\right\}\right]$$

$$+ \sum_{k=0}^{d}\left[\binom{d}{k}\left\{\sum_{l=1}^{D-d}(D-d-l+1)\binom{D-d}{l-1}\Delta_{NCL}(k+l)\,p_1(k+l)\right\}\right] \quad (13)$$

and

$$R_{avg} = (1-h)\,R_{reg} + hR_{hot} \quad (14)$$

where

$\Delta^{avg}{}_{CL}$ = average delay encountered on a cluster link

$$= \frac{1}{d\,2^{D-1}}\left[\sum_{k=0}^{D-d}\binom{D-d}{k}\left\{\sum_{l=1}^{d}(d-l+1)\binom{d}{l-1}\Delta_{CL}(k+l)\right\}\right]$$

and

$\Delta^{avg}{}_{NCL}$ = average delay encountered on a non − cluster link

$$= \frac{1}{(D-d)2^{D-1}}\left[\sum_{k=0}^{d}\binom{d}{k}\left\{\sum_{l=1}^{D-d}(D-d-l+1)\binom{D-d}{l-1}\Delta_{NCL}(k+l)\right\}\right]$$

3.1.2. Analysis of the BH/BH network

In this network, one node from each cluster acts as an interface node to which all non-cluster links connecting to the cluster attach. It is assumed that the hot-spot node is an interface node. As for the BH network, the analysis is based on finding expressions for $\lambda_{CL}(j)$ and $\lambda_{NCL}(j)$.

The effective arrival rate of messages and replies at a j-level CL link is given by

$$\lambda_{CL}(j) = \lambda'_{CL}(j) + \lambda''_{CL}(j) \quad (15)$$

where

$\lambda'_{CL}(j)$ = arrival rate (at a j-level CL link) due to intra-cluster messages and replies, and

$\lambda''_{CL}(j)$ = arrival rate (at a j-level CL link) due to inter-cluster and hot-spot messages and the corresponding replies.

It is easy to see that $\lambda'_{CL}(j)$ is given by Eq. (2). $\lambda''_{CL}(j)$ can be derived as

$$\lambda''_{CL}(j) = 2^d \lambda(2 - 2\alpha - h)p(j) \tag{16}$$

where $p(j)$ is the probability of an inter-cluster message or reply being routed through a j-level CL link, and is given by (similar to the derivation of $p_1(j)$ in Section 3.1.1)

$$p(j) = \frac{\sum_{k=j}^{d}\binom{d}{k}}{(d - j + 1)\binom{d}{j-1}2^d}$$

Therefore,

$$\lambda_{CL}(j) = \alpha\lambda + (2 - 2\alpha - h)2^d \lambda p(j) \tag{17}$$

$\lambda_{NCL}(j)$ can be expressed as

$$\lambda_{NCL}(j) = \lambda'_{NCL}(j) + \lambda''_{NCL}(j) \tag{18}$$

where

$\lambda'_{NCL}(j)$ = arrival rate (at a j-level NCL link) due to inter-cluster messages or replies, and

$\lambda''_{NCL}(j)$ = arrival rate (at a j-level NCL link) due to hot-spot messages or replies.

Since each inter-cluster message uses on average $\left[\frac{(D - d)2^{D-d-1}}{2^{D-d} - 1}\right]$ non-cluster links and each cluster generates these messages and their replies at rate $(1 - \alpha - h)\lambda 2^d$ each,

$$\lambda'_{NCL}(j) = \frac{2^{D-d} 2\, 2^d (1 - \alpha - h)\lambda\left[\frac{(D - d)2^{D-d-1}}{2^{D-d} - 1}\right]}{2(D - d)2^{D-d-1}}$$

$$= \frac{2^D(1 - \alpha - h)\lambda}{2^{D-d} - 1} \tag{19}$$

The derivation of $\lambda''_{NCL}(j)$ is similar to that of $\lambda'''_{CL}(j)$ for the BH network except that each "node" generates $2^d\ h\lambda$ hot-spot messages and there are 2^{D-d} "nodes". Let $p_1(j)$ be the probability that a hot-spot message (or reply) gets routed through a j-level NCL link on its way to (or from) the hot spot node. Then,

$$\begin{aligned}\lambda''_{NCL}(j) &= 2^{D-d}\,2^d\,h\lambda p_1(j)\\ &= 2^D\,h\lambda p_1(j)\end{aligned} \tag{20}$$

where

$$p_1(j) = \frac{\sum_{k=j}^{D-d}\binom{D-d}{k}}{(D-d-j+1)\binom{D-d}{j-1}2^{D-d}}$$

Eqs. (18), (19), and (20) yield

$$\lambda_{NCL}(j) = \frac{2^D(1-\alpha-h)\,\lambda}{2^{D-d}-1} + 2^D\,h\lambda p_1(j) \tag{21}$$

The average delays encountered by messages and replies at j-level CL and NCL links are given by:

$$\Delta_{CL}(j) = \frac{1}{\mu_{CL} - \lambda_{CL}(j)} \tag{22}$$

and

$$\Delta_{NCL}(j) = \frac{1}{\mu_{NCL} - \lambda_{NCL}(j)} \tag{23}$$

The average total message delays are given by:

$$R_{reg} = \frac{1}{(1-h)}\left[\alpha\left(\frac{d}{2}\right)\Delta_{avg} + (1-\alpha-h)\left\{2\Delta_{CL-avg} + \left(\frac{(D-d)2^{D-d-1}}{2^{D-d}-1}\right)\Delta^{reg}_{NCL-avg}\right\}\right] \tag{24}$$

$$R_{hot} = \Delta_{CL-avg} + \Delta^{hot}_{NCL-avg} \tag{25}$$

and

$$R_{avg} = (1-h)\,R_{reg} + hR_{hot} \tag{26}$$

where

Δ_{avg} = average delay encountered by intra – cluster messages on a cluster link

$$= \frac{1}{d2^{d-1}}\left[\sum_{j=1}^{d}\left\{(d-j+1)\binom{d}{j-1}\Delta_{CL}(j)\right\}\right]$$

Δ_{CL-avg} = average delay encountered by inter – cluster or hot – spot messages on a cluster link

$$= \sum_{j=1}^{d}\left\{(d-j+1)\binom{d}{j-1}\Delta_{CL}(j)\,p(j)\right\}$$

$\Delta^{reg}_{NCL-avg}$ = average delay encountered by an inter-cluster message on an intercluster link

$$= \frac{1}{(D-d)2^{D-d-1}}\left[\sum_{j=1}^{D-d}\left\{(D-d-j+1)\binom{D-d}{j-1}\Delta_{NCL}(j)\right\}\right]$$

and

$\Delta^{hot}_{NCL-avg}$ = average delay encountered by a hot-spot message on an intercluster link

$$= \sum_{j=1}^{D-d}\left\{(D-d-j+1)\binom{D-d}{j-1}\Delta_{NCL}(j)\,p_1(j)\right\}$$

3.1.3. Effect on accuracy of the analysis assumptions

Two assumptions were made in order to simplify the analysis. It was assumed that whenever a message arrives at a link on its path to its destination, a new service time is generated from the exponential service time distribution. In a real system, of course, the messages would retain their service times, which reflect the lengths of the messages, from the times they are generated to the times they arrive at their destinations. The other assumption is that replies are generated independently by Poisson processes. In real systems, a reply is generated whenever a message reaches its destination node. A simulation model was constructed in which these two assumptions were not made, and that, therefore, allowed for the study of their effect on accuracy. In all other respects, the simulation model matched the analytic model.

Both BH and BH/BH networks were used in the simulation experiments. For a

network of each type, the two components of delay -- R_{reg} and R_{hot} -- are plotted in Figure 2 as functions of α. The value of α is varied from 0.6 to 0.9 in steps of 0.1. The hot-spot proportion h is fixed at 8%. The other parameters used are: d=3, D=6, λ= 1, μ_{CL}= 3, μ_{NCL}= 3 for the BH network, and d=3, D=6, λ= 1, μ_{CL}= 3, μ_{NCL}= 6 for the BH/BH network. The results for the BH network are presented in Figure 2*a* and those for the BH/BH network are given in Figure 2*b*. It can be seen from these plots that the results obtained from the analysis match closely those obtained by simulation, thus justifying the two approximations used in the analysis.

3.1.4. Discussion of results

The impact of hot-spot contention on BH and BH/BH networks is shown in Figure 3. These graphs give R_{avg} as computed by Eqs. (14) and (26) for 256-node networks (D=8). The hot-spot proportion h is varied from 0% to 16%. The value of μ_{CL} is set at 1.4, for both cases, and μ_{NCL} to 1.4 for the BH network and to 2.8 for the BH/BH network. The other parameters used are d=3, and α= 0.75. (Only R_{avg} is shown in this and the succeeding graphs of this section since both hot-spot and regular messages exhibit similar behaviour

In the BH network, increasing the hot-spot proportion decreases λ_{sat} (the traffic intensity at which network saturation occurs) substantially. The BH/BH network is less sensitive to the hot-spot proportion. The BH/BH network behaves differently since CL and NCL links saturate at different relative traffic intensities. For h values in the range 0-4% in this example, CL links saturate first. For h values greater than about 4%, NCL links saturate first. The curves for h values ranging from 0 through 4% show that, surprisingly, the delay reduces with increasing h in this range. This is because the hot-spot node is assumed to be an interface node, and with increasing h values, a greater proportion of messages do not traverse any cluster links within the destination cluster, reducing the load on the "bottleneck" network resources (the CL links) and thus the average delay.

The impact of locality on hot-spot contention is shown in Figure 4. Here, the value of

α is varied from 0.6 to 0.9 while keeping the value of h constant at 8%. As shown in the figure, the changes in α have negligible impact on the BH network. This is an expected result because the message traffic due to intra- and inter-cluster messages is uniformly distributed over all the links within a cluster, and thus $\lambda_{CL}(j)$ is independent of α, as seen in Eq. (5). In the BH/BH network, on the other hand, increasing α results in a significant improvement in performance. As the value of α is increased from 0.6 to 0.9, λ_{sat} increases from 0.42 to 0.68. Note that, in this network, intra-cluster messages are uniformly distributed over all links within a cluster while the traffic due to inter-cluster messages tends to be highest on links that are closer to the interface node. Decreasing the proportion of inter-cluster messages decreases the load on these relatively heavily utilized links.

3.2. Hardware Combining

In hardware combining, the queues associated with links can be classified as either *forward* queues or *return* queues. A queue associated with a link l that carries messages from node a to node b is termed a forward queue if node b is closer to the hot-spot node than node a, and a return queue if node a is closer to the hot-spot node than node b. A "wait buffer" must be associated with each node. When several hot-spot messages are queued in a forward queue, they are combined into a single hot-spot message, called a *combined message*, which is forwarded toward the hot-spot node. A record of all messages combined is kept in the wait buffer. When the reply from the hot-spot node returns (routing must be such that a reply follows the same path, except in reverse as the corresponding message), the node, using the information in the wait buffer, generates replies to all combined messages and places them in return queues. Note that a message may participate in many combinings before reaching the hot-spot node. In this section, unbounded buffering space for forward queues, return queues, and wait buffers is assumed, and no restrictions are placed on the number of messages that may be combined at a single node.

The results of simulation experiments that were performed to assess the performance of hardware combining are shown in Figure 5. It is obvious from these graphs that hardware combining is effective in reducing the hot-spot contention. Interestingly, delays associated with regular and hot-spot messages reduce with an increasing hot-spot proportion h. This is because with higher h larger numbers of hot-spot messages get combined, effectively reducing the load and thus the contention within the network.

In practice, hardware combining may not perform as well as Figure 5 would suggest because of finite queue sizes and wait buffers, and limitations on the number of hot-spot messages that may be combined at a time. Finite queues are considered in Section 4. Unbounded queues, however, provide an upper bound on the achievable performance improvements.

3.3. Software Combining

Software combining has been studied in detail by Yew *et al.* [1987] in the context of shared-memory multiprocessors. Here only one type of application of this idea is described; specifically, the problem of recognizing the end of a computation participated by all nodes. One way to do this in shared-memory multiprocessor would be to have a single shared data item whose initial value is the number of nodes (N), and which is decremented by each node as it completes its portion of the computation, but this will result in the formation of a hot-spot.

With software combining, a tree of data items is created as shown in Figure 6. If N=1000 and assuming a fan-in of 10, there are 111 data items each with value 10. The nodes are partitioned into 100 groups of ten, with each group sharing one of the data items corresponding to the leaves of the tree. When the last node in each group decrements its data item to zero, this node then decrements the parent data item. This process is repeated until the data item corresponding to the root of the tree is finally decremented to zero. Software combining reduces hot-spot contention in this example because there are 111 data items each being accessed by only 10 nodes instead of one data item being accessed by

1000 nodes.

In a distributed-memory multicomputer, software combining can be applied to the problem of recognizing the end of a computation by logically constructing a tree of nodes that exchange messages so as to finally result in a "root" node being notified of the termination. This root node may then broadcast the fact that the computation has terminated back down the tree.

3.3.1. Application to the BH network

There are many ways a software combining tree can be constructed on a BH network. A simple mapping scheme that is applicable to both BH and BH/BH networks is considered here. The structure of the software combining tree is shown in Figure 7. This structure is a two-level, unbalanced tree. Nodes within a cluster are linked by a level 1 tree (shown in dashed lines). The root nodes of each of these subtrees (i.e., nodes 0, 8, 16, and 24 in Figure 7) are linked by a level 2 tree (shown in solid lines). At each level, the mapping is done as follows. A node is selected as the root node (for example, node 8 in Figure 7). (This root node selection may be made randomly for the networks considered here, except in the selection of the cluster root nodes in a BH/BH network for which the interface nodes should be chosen.) The children of this root node are the nodes that are one link away from the root node (e.g., nodes 9, 10, and 12), the grandchildren of this root node are all those nodes that are two links away (e.g., nodes 11, 14, and 13), and so on. In any part of the tree, the addresses of a child node and that of its parent differ in only one coordinate. Therefore, there is a link that directly connects each parent-child pairs. The members of a set of grandchildren nodes (e.g., 11, 14, 13) are distributed over a set of child nodes (e.g., 9, 10, 12) as uniformly as possible. For example, the grandchildren nodes 11, 14, and 13 are attached one each to the child nodes 9, 10 and 12. Further, if there is a choice in regard to which node a node can be attached in building the software combining tree, a node is randomly selected. For example, node 15 can be attached to either node 13 or node 14. It should be noted that, irrespective of the degree of a node, all

links in the tree process hot-spot messages at rate $h\lambda$.

The analysis of delay with a software combining tree is similar to that presented in Section 3.1.1. Each link is classified, as before, as either a CL or NCL link. Each link is further classified depending on whether it belongs to the software combining tree (a tree link) or does not belong to the tree (a non-tree link). Let λ_{CLt} and λ_{CL} denote the message arrival rates seen by tree CL links and non-tree CL links, respectively. λ_{NCLt} and λ_{NCL} are similarly defined for NCL links. Expressions for these arrival rates can be derived in a manner similar to that used in deriving Eqs. (5) and (9), yielding:

$$\lambda_{CL} = (1-h)\lambda, \qquad \lambda_{CLt} = (1-h)\lambda + h\lambda = \lambda \tag{27}$$

$$\lambda_{NCL} = (1-\alpha-h)\lambda\frac{2^D}{2^D-2^d}, \qquad \lambda_{NCLt} = (1-\alpha-h)\lambda\frac{2^D}{2^D-2^d} + h\lambda \tag{28}$$

The average delay encountered by messages in each type of link is given by

$$\Delta_{CL} = \frac{1}{\mu_{CL}-\lambda_{CL}}, \qquad \Delta_{CLt} = \frac{1}{\mu_{CL}-\lambda_{CLt}}$$

$$\Delta_{NCL} = \frac{1}{\mu_{NCL}-\lambda_{NCL}}, \qquad \Delta_{NCLt} = \frac{1}{\mu_{NCL}-\lambda_{NCLt}}$$

The average message delay for regular messages is given by Eq. (12), where Δ^{avg}_{CL} is given by.

$$\Delta^{avg}_{CL} = \frac{1}{d2^{d-1}}\left[(2^d-1)\Delta_{CLt} + (d2^{d-1}-2^d+1)\Delta_{CL}\right]$$

because there are (2^d-1) links in a tree of 2^d nodes, and Δ^{avg}_{NCL} is given by

$$\Delta^{avg}_{NCL} = \frac{1}{(D-d)2^{D-1}}\left[(2^{D-d}-1)\Delta_{NCLt} + ((D-d)2^{D-1}-2^{D-d}+1)\Delta_{NCL}\right]$$

With respect to R_{hot}, note that each hot-spot message traverses only a single link in the software combining tree. However, a hot-spot "reply" must be propagated all the way down the tree from the root node. Here, an expression is given for R_{hot} that represents in this case the average delay encountered by a hot-spot reply. It is easy to see that the average number of tree CL links traversed by hot-spot replies is given by

$$\frac{\sum_{i=1}^{d} i\binom{d}{i}}{2^d} = \frac{d}{2}$$

Similarly, the average number of tree NCL links traversed by hot-spot replies is $(D\text{-}d)/2$. Therefore,

$$R_{hot} = \frac{d}{2}\Delta_{CLt} + \frac{D-d}{2}\Delta_{NCLt}$$

R_{avg} is defined here as in Eq. (26), and can be interpreted as the average delay experienced by a reply message.

The impact of utilizing a software combining tree is presented in Figure 8*a* for a 256-node BH network. All parameters remain the same as those used in Figures 3 and 5. The figure shows that software combining is effective in reducing the adverse effect of hot-spot contention. Note that for all values of h, λ_{sat} remains constant. This is because λ_{CLt} is independent of h and these tree CL links saturate first, for these parameter values. In general, it can be shown from Eqs. (27) and (28) that λ_{sat} will never decrease as h increases, when the proposed software combining tree utilized.

3.3.2. Application to the BH/BH network

In this type of network, each interface node i chosen as the root node of the "level 1" portion of the software combining tree that is built within its cluster. Otherwise the same mapping scheme is used in this network as in the BH network. The analysis is similar to that presented in the last section. As in the last section, let λ_{CLt} and λ_{CL} be the message arrival rates seen by tree CL links and non-tree CL links, respectively. λ_{NCLt} and λ_{NCL} are similarly defined for NCL links. In a manner similar to that in which Eqs. (17) and (21) were obtained, these arrival rates can be derived as

$$\lambda_{CL} = \alpha\lambda + (1 - \alpha - h)\lambda 2^{d+1} p_o(j)$$

$$\lambda_{CLt} = (\alpha + h)\lambda + (1 - \alpha - h)\lambda 2^{d+1} p_o(j)$$

$$\lambda_{NCL} = \frac{2^D (1 - \alpha - h)\lambda}{2^{D-d} - 1}$$

and,

$$\lambda_{NCLt} = \frac{2^D (1 - \alpha - h)\,\lambda}{2^{D-d} - 1} + h\lambda$$

Δ_{CL}, Δ_{CLt}, Δ_{NCL}, and Δ_{NCLt} can be obtained as in Section 3.3.1. R_{reg} and R_{avg} can be derived using Eqs. (24) and (26), where Δ_{avg}, Δ_{CL-avg}, and $\Delta^{reg}_{NCL-avg}$ are given by

$$\Delta_{avg} = \frac{1}{d2^{d-1}}\left[(2^d - 1)\Delta_{CLt} + (d\,2^{d-1} - 2^d + 1)\Delta_{CL}\right]$$

$$\Delta_{CL-avg} = \sum_{j=1}^{d}\left\{\left[(d - j + 1)\binom{d}{j-1} - \binom{d}{j}\right]\Delta_{CL}(j)\,p_o(j) + \binom{d}{j}\Delta_{CLt}(j)\,p_o(j)\right\}$$

and

$$\Delta^{reg}_{NCL-avg} = \frac{1}{(D-d)2^{D-d-1}}\sum_{j=1}^{D-d}\left\{\left[(D - d - j + 1)\binom{D-d}{j-1} - \binom{D-d}{j}\right]\Delta_{NCL}(j) + \binom{D-d}{j}\Delta_{NCLt}(j)\right\}$$

Note that in the above expression for Δ_{CL-avg}, $(d - j + 1)\binom{d}{j-1}$ gives the number of j-level (from the interface node) CL links and $\binom{d}{j}$ gives the number of tree links in that group. R_{hot} is, again, chosen as the average delay experienced by a hot-spot reply, and is given by

$$R_{hot} = \left[\frac{D-d}{2}\right]\Delta_{NCLt} + \frac{1}{2^d}\sum_{j=1}^{d}\left[\sum_{k=j}^{d}\binom{d}{k}\right]\Delta_{CLt}$$

The delay encountered by hot-spot replies in the level 2 network is given by, as in a BH network, $\frac{(D-d)}{2}\Delta_{NCLt}$. Since the delay incurred at the tree CL links depends on how close they are to the interface node, the average delay encountered by hot-spot replies in the level 1 network is given as shown in the second term in the above expression.

A graph of R_{avg} versus λ is shown in Figure 8*b*. The network parameters are the same as those used in Figures 3 and 5. Again, as for the BH network, software combining works well. Note that for the BH/BH network, as with hardware combining, λ_{sat} increases with an increasing proportion of hot-spot traffic.

3.4. Impact of Improved Routing Algorithms

The analysis and the simulation experiments described in Sections 3.1 through 3.3 used a routing algorithm in which each of the shortest paths between the source and destination nodes is randomly selected with equal probability. However, the links that are used by hot-spot messages tend to saturate much earlier than the other links. Thus, if the routing algorithm can be improved such that it chooses that shortest path with the least traffic, the adverse impact of hot-spot contention on system performance can be reduced.

Several simulation experiments were conducted on the BH and BH/BH networks using an improved algorithm that works as follows. At each node, if a message can take one of l links, it is routed over the link that has the shortest queue of messages waiting to be forwarded. The results of these experiments are shown in Figures 9 and 10. (The network parameters are identical to those used in Figures 3, 5, and 8.) With this improved routing algorithm, regular messages can be directed away from links that saturate because of hot-spot traffic, thus lessening the impact of hot-spot contention on regular traffic. For this reason, the delays of regular and hot-spot messages are shown separately in these graphs. These graphs show that the improved routing algorithm reduces the adverse effects of hot-spot contention considerably. More specifically, regular message throughput is not affected as is the case with the random routing algorithm. As shown in Section 4, however, these

messages will experience increased delays because of hot-spot message interference in the case of bounded buffering capacity.

4. Performance with Bounded Buffering Capacity

This section presents simulation results for the case of bounded buffering capacity. When the number of buffers is finite, there is a possibility of store-and-forward deadlock [Merlin and Schweitzer 1980]. Store-and-forward deadlock refers to the situation in which there is a set of buffers, all of which hold messages waiting to be forwarded, and in which these messages can be forwarded only to other buffers of the set. The result is a deadlock. It can be shown that store-and-forward deadlock implies a cycle of buffer requests [Coffman *et al.* 1971, Holt 1972].

The simulation experiments implemented a scheme for deadlock avoidance proposed by Merlin and Schweitzer [1980]. Let M be the maximum number of hops a message can make in the network. At each node, suppose that there are $M+1$ buffers, say $[\, B_0, B_1, \ldots, B_M \,]$. When a message is generated, it is placed in buffer B_0 at the source node. For a given message, let t denote the total number of hops from source to destination. When a message has made $s<t$ hops, it can be placed in any buffer B_g at the next node on the message route such that

$$0 \le g \le M - (t - s) + 1.$$

It can be shown that if this rule is followed, there will not be store-and-forward deadlocks. For the BH network, $M=D$ and each node should have at least $(D+1)$ buffers. For the BH/BH network, $M=D+d$ and each node should have a minimum of $(D+d+1)$ buffers. If the actual number of buffers C is greater than M, the message can be placed in any buffer g such that

$$0 \le g \le C - (t - s) + 1. \tag{29}$$

The simulation experiments were conducted on a 256-node network with a cluster size of 8 (d=3). Thus, each node should have a minimum of 12 buffers. In the simulation, the improved routing algorithm described in the previous section is utilized (with the constraint

that a message can be forwarded only to a node with a buffer B_g that satisfies Eq. (29)). Also, when an empty buffer becomes available at a node, a node that has a message to be transmitted to this node that may use the buffer is randomly selected from among the neighbouring nodes, and is allowed to transmit and fill the buffer. Finally, the message generation process at a node is blocked whenever there are no buffers free at this node in which a newly generated packet could be placed.

Several simulation experiments were performed with different numbers of buffers. Figure 11 presents selected results that illustrate the effects of this parameter. Two performance measures are used -- the average delay R_{avg} and throughput. For the BH network, the parameters used are: $\mu_{CL} = \mu_{NCL} = 1.4$, $\lambda = 1.3$, $\alpha = 0.75$, and h=0%, and for the BH/BH network, the parameters used are: $\mu_{CL} = 1.4$, $\mu_{NCL} = 2.8$, $\lambda = 0.8$, $\alpha = 0.75$, and $h = 0\%$. These results show that the BH/BH network is more sensitive to the number of buffers.

For the remainder of the figures, a buffer capacity of 32 was used. Results for the BH network are shown in Figure 12. Note that these graphs use throughput rather than the message generation rate λ on the x-axis, and that the throughput may be less than the message generation rate because of buffer limitations. The results are somewhat different from those shown in Figure 9, which assumed unbounded buffering capacity. With finite buffering capacity, the hot-spot proportion greatly affects the performance of the regular messages, even with the improved routing algorithm. The hot-spot messages as well experience substantially more delay than in the case of unbounded buffering capacity. Figure 13 shows the results for the BH/BH network. These plots show a distinctive "fold-back" characteristic in the sense that increasing the message generation rate beyond the saturation point results not only in increased delays but also in *decreased* throughput. This effect is most pronounced in Figure 13*a* for h values of 8% and 16%. The explanation for this effect is that increasing the message generation rate results in more contention for buffer space, and may result in intra-cluster messages "hogging" the buffer space in interface nodes, causing the hot-spot and inter-cluster message throughput to drastically

decrease. This results in a drastic decrease in overall throughput. Such behaviour is not observed with the BH network because there are more alternate paths in the network.

The impact of hardware combining is shown in Figures 14 and 15. These graphs show that hardware combining is still effective in reducing the adverse effects of hot-spot contention. Hardware combining reduces the average delays associated with both regular and hot-spot messages substantially. Further, an increasing hot-spot proportion results in increased λ_{sat} for the BH/BH network as shown in Figure 15.

Figures 16 and 17 show the influence of software combining on the average delays of regular and hot-spot messages for the BH and BH/BH networks. The results show that software combining achieves message combining more effectively than hardware combining in this case. These results further demonstrate that the simple mapping scheme used here for the software combining tree is adequate, even with finite buffer space. For both networks, the hot-spot message delays are substantially less compared to the delays with hardware combining.

5. Summary

This paper has studied the impact of hot-spot contention in two types of static interconnection networks that are appropriate for use in distributed memory multicomputer systems. One was the standard binary hypercube (BH) network and the other was a hierarchical interconnection network -- the BH/BH network. An analysis for average delays was performed under an assumption of unbounded buffering capacity. The use of a routing algorithm in which each of the shortest paths between the source and destination nodes is randomly selected with equal probability resulted in extreme sensitivity to the hot-spot proportion h of traffic. With the random routing algorithm, increasing h reduced the message generation rate at which saturation occurs (λ_{sat}) substantially. The BH network tends to be more sensitive to h than does the BH/BH network.

The impacts of two schemes -- hardware combining and software combining -- that have been proposed for shared-memory multiprocessor systems to reduce the performance

degradation caused by hot-spot contention were studied. Both schemes were found to be effective in reducing the adverse effects of hot-spot contention in distributed-memory multicomputer systems as well.

The impact of improved routing algorithms was also studied, through simulations of a "shortest queue" routing algorithm. This improved routing algorithm resulted in much less sensitivity in the delays of regular messages to the proportion of hot-spot traffic, as regular messages could avoid links congested by hot-spot traffic.

Even with the improved routing algorithm, however, limitations on buffering capacity were found to induce significantly restricted network capacity as the hot-spot proportion h increased. The BH/BH network was found to exhibit a "fold-back" characteristic in the sense that increasing the message generation rate beyond the saturation point results not only in increased delays but also in *decreased* throughput. Such behaviour was not observed with the BH network because there are more alternate paths in the network. Again, both hardware and software combining schemes proved to be effective in reducing the adverse effects of hot-spot contention, software combining particularly so.

REFERENCES

[Coffman *et al.* 1971]

E. G. Coffman, Jr., M. J. Elphick, and A. Shoshani, "System Deadlocks," *ACM Comput. Surveys*, Vol. 3, No. 2, June 1971, pp. 67-78.

[Dandamudi & Eager 1987]

S. P. Dandamudi and D. L. Eager, "Hierarchical Interconnection Networks for Multicomputer Systems," to appear in *IEEE Trans. Computers* (Also Tech. Rep. 87-11, Department of Computational Science, University of Saskatchewan, Saskatoon).

[Gottlieb *et al.* 1983]

A. Gottlieb *et al.*, "The NYU Ultracomputer -- Designing a MIMD, Shared Memory Parallel Machine," *IEEE Trans. Computers*, Vol. C-32, No. 2, February 1983, pp. 175-189.

[Hayes *et al.* 1986]

J. P. Hayes, T. N. Mudge, Q. F. Stout, S. Colley, and J. Palmer, "Architecture of a Hypercube Supercomputer," *Proc. 1986 Int. Conf. Parallel Processing*, 1986, pp. 653-660.

[Ho & Eager 1989]

W. S. Ho and D. L. Eager, "A Novel Strategy for Controlling Hot Spot Congestion," to appear in *Proc. 1989 Int. Conf. Parallel Processing*, 1989.

[Holt 1972]

R. C. Holt, "Some Deadlock Properties of Computer Systems," *ACM Comput. Surveys*, Vol. 4, No. 3, September 1972, pp. 179-196.

[Kleinrock 1975]

L. Kleinrock, *Queueing Systems:* Volume 1, Wiley, New York, 1975.

[Kleinrock 1976]

L. Kleinrock, *Queueing Systems:* Volume 2, Wiley, New York, 1976.

[Lee *et al.* 1986]

G. Lee, C. Kruskal, and D. J. Kuck, "The Effectiveness of Combining in Shared Memory Parallel Computers in the Presence of 'Hot Spots'," *Proc. 1986 Int. Conf. Parallel Processing*, 1986, pp. 35-41.

[Merlin & Schweitzer 1980]

P. M. Merlin and P. J. Schweitzer, "Deadlock Avoidance in Store-and-Forward Networks -- I: Store-and-Forward Deadlock," *IEEE Trans. Communications*, Vol. COM-28, No. 3, March 1980, pp. 345-354.

[Pfister *et al.* 1985]

G. F. Pfister, W. C. Brantley, D. A. George, S. L. Harvey, K. P. Kleinfelder, E. A. Melton, V. A. Norton, and J. Wiess, "The IBM Research Parallel Processor Prototype (RP3): Introduction and Architecture," *Proc. 1985 Int. Conf. Parallel Processing*, 1985, pp. 764-771.

[Pfister & Norton 1985]

G. F. Pfister and V. A. Norton, " 'Hot Spot' Contention and Combining in Multistage Interconnection Networks," *IEEE Trans. Computers*, Vol. C-34, No. 10, October 1985, pp. 943-948.

[Rattner 1985]

J. Rattner, "Concurrent Processing: A New Approach in Scientific Computing," *Proc. AFIPS Conf.*, Vol. 54, 1985 NCC, pp. 157-166.

[Rettberg & Thomas 1986]

R. Rettberg and R. Thomas, "Contention is no Obstacle to Shared Memory Multiprocessing," *Comm. ACM*, Vol. 29, No. 12, December 1986, pp. 1202-1212.

[Yew *et al.* 1987]

P. -C. Yew, N. -F. Tzeng, and D. H. Lawrie, "Distributing Hot-Spot Addressing in Large-Scale Multiprocessors," *IEEE Trans. Computers*, Vol. C-36, No. 4, April 1987, pp. 388-395.

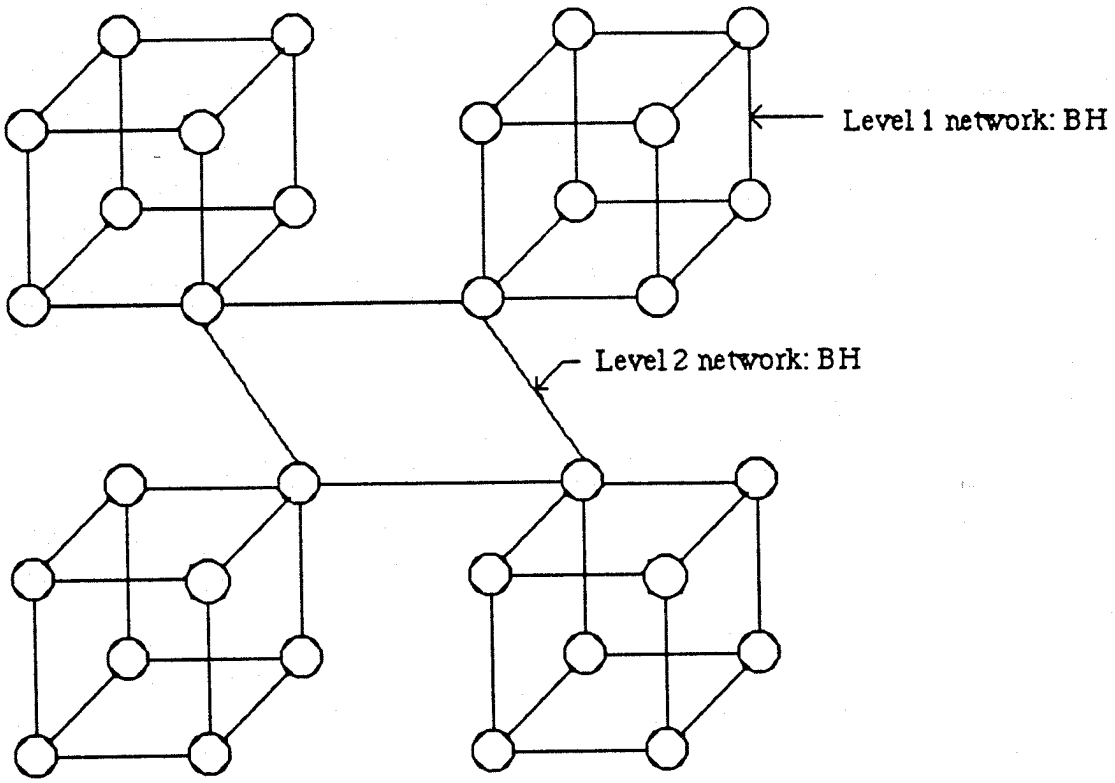
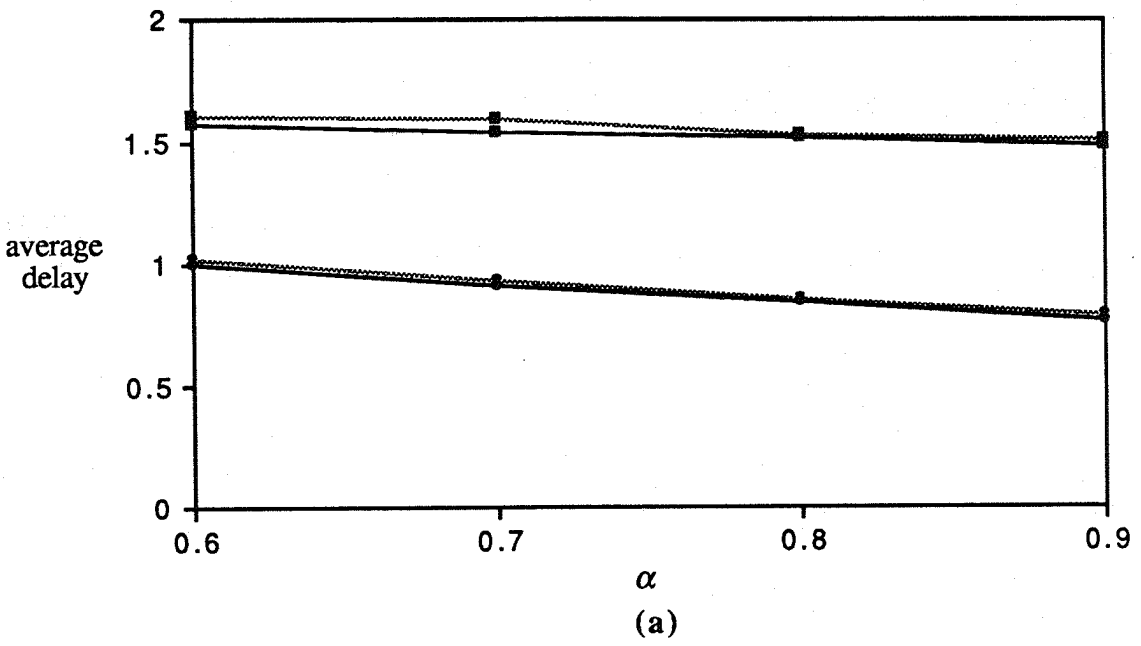
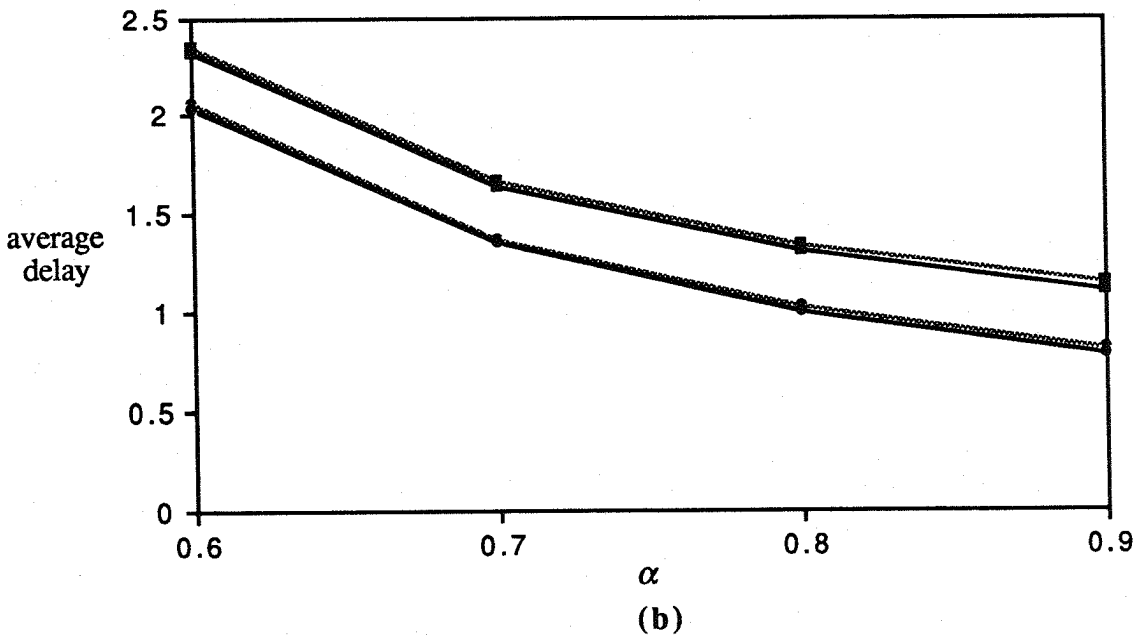


Figure 1 An example hierarchical interconnection network, BH/BH



(a)



(b)

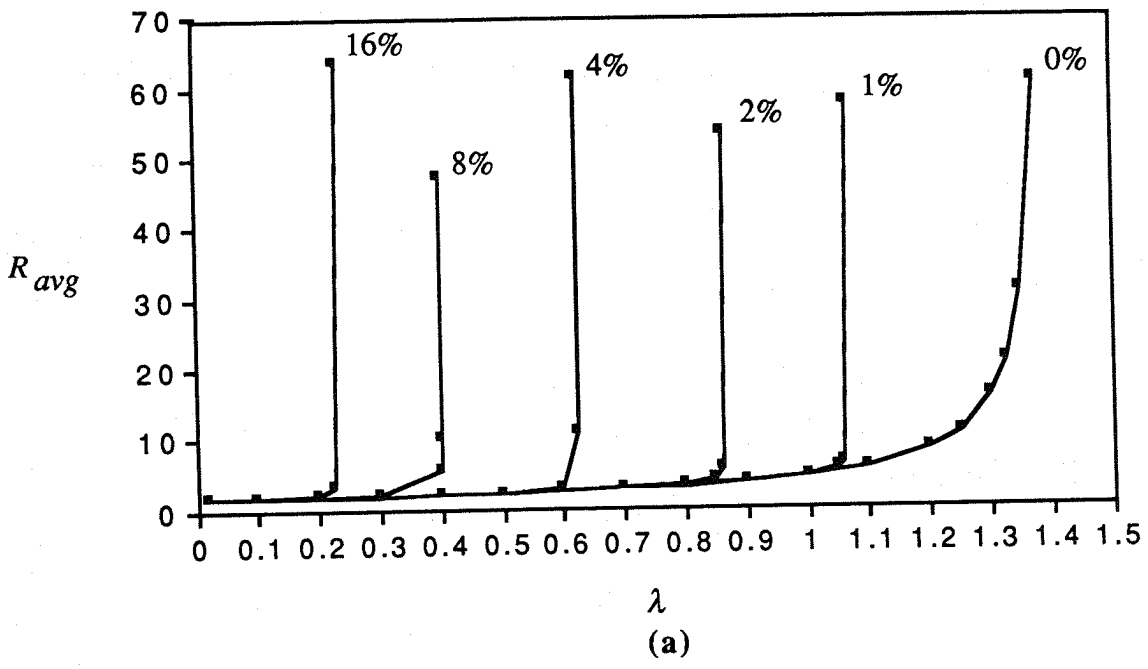
Figure 2 Comparison of analytical and simulation results

(a) BH network ($d = 3, D = 6, \lambda = 1, \mu_{CL} = \mu_{NCL} = 3, h = 8\%$)

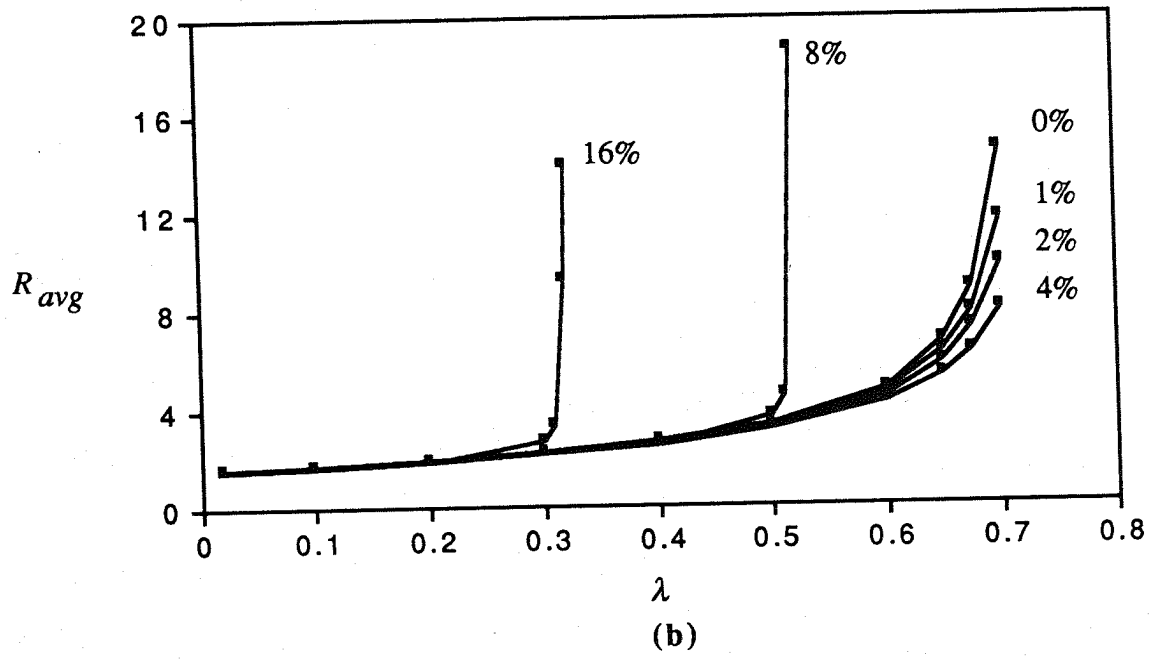
(b) BH/BH network ($d = 3, D = 6, \lambda = 1, \mu_{CL} = 3, \mu_{NCL} = 6, h = 8\%$)

• : R_{reg} ■ : R_{hot}

——— analysis ~~~~~ simulation



(a)

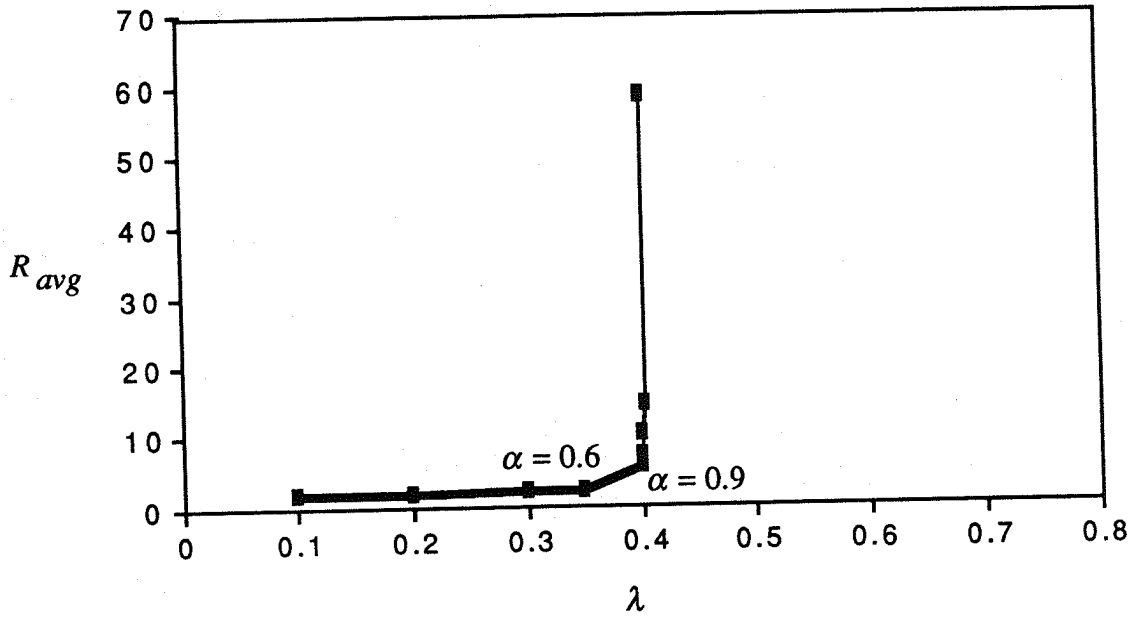


(b)

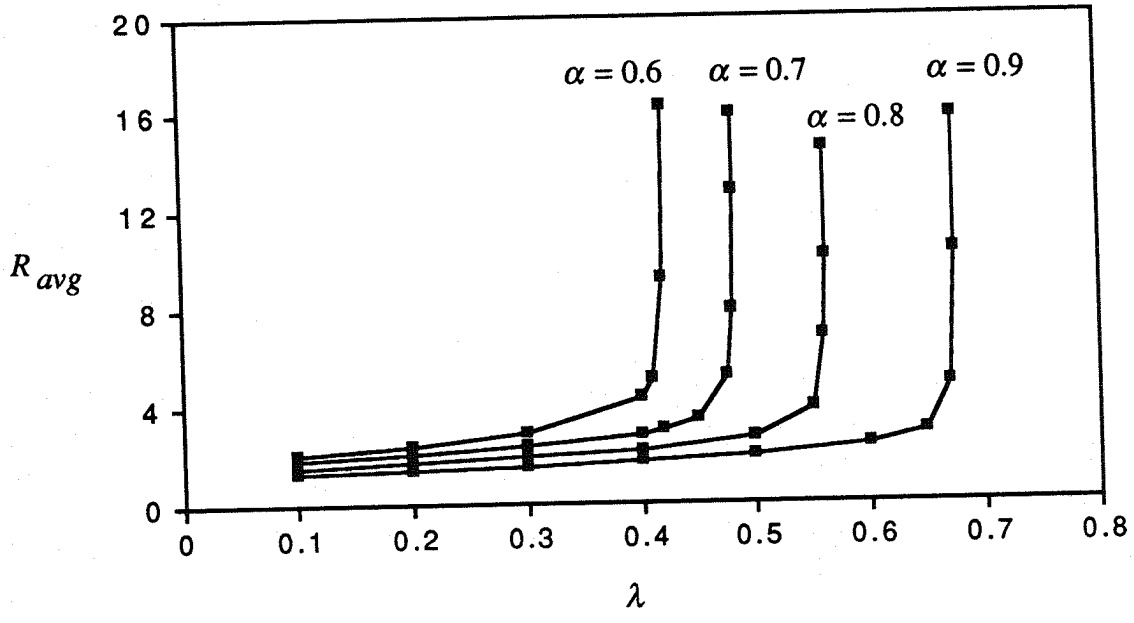
Figure 3 Impact of hot-spot contention on the BH and BH/BH networks (h is varied from 0% to 16%)

(a) BH network ($d = 3, D = 8, \mu_{CL} = \mu_{NCL} = 1.4, \alpha = 0.75$)

(b) BH/BH network ($d = 3, D = 8, \mu_{CL} = 1.4, \mu_{NCL} = 2.8, \alpha = 0.75$)

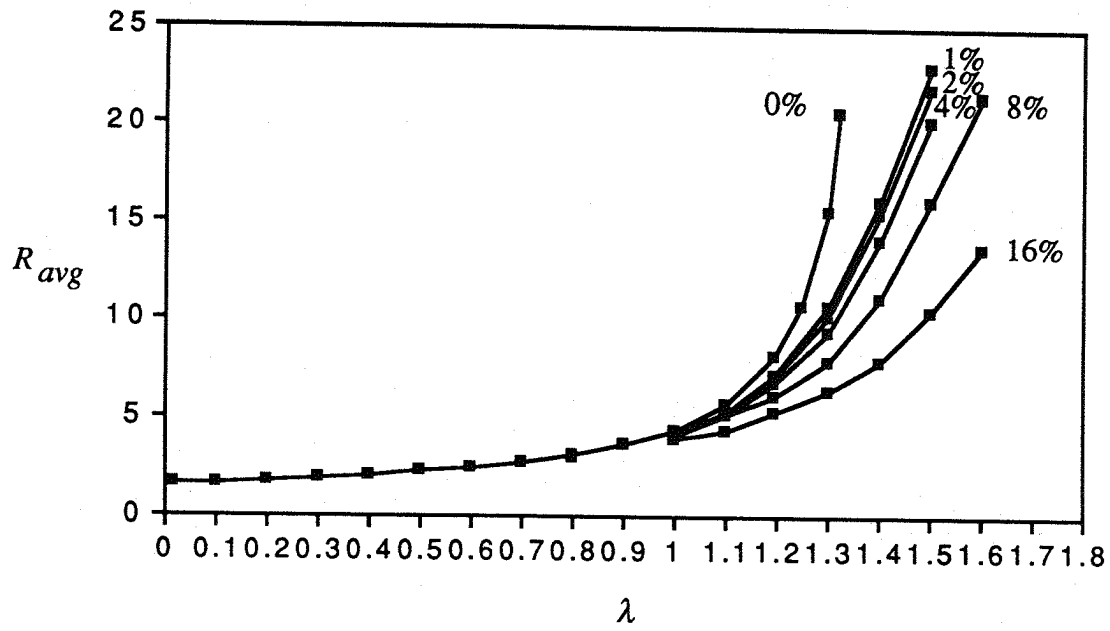


(a)

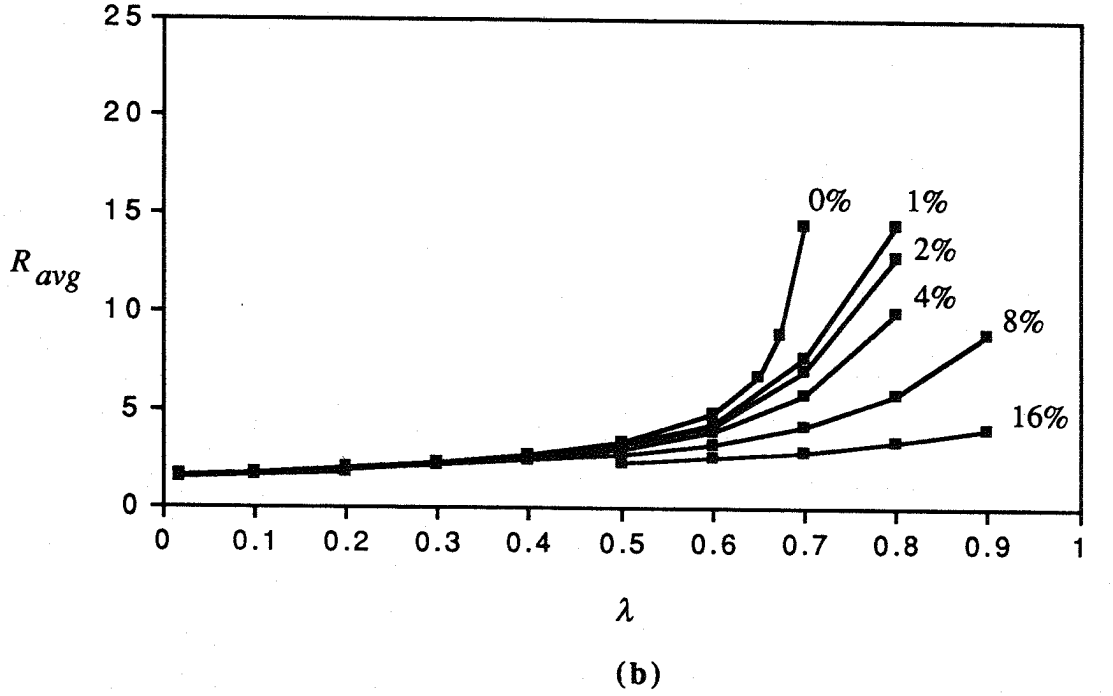


(b)

Figure 4 Impact of locality on hot-spot contention in the BH and BH/BH networks
(a) BH network ($d = 3, D = 8, \mu_{CL} = \mu_{NCL} = 1.4, h = 8\%$)
(b) BH/BH network ($d = 3, D = 8, \mu_{CL} = 1.4, \mu_{NCL} = 2.8, h = 8\%$)



(a)



(b)

Figure 5 Impact of hardware combining on hot-spot contention in the BH and BH/BH networks (h is varied from 0% to 16%)

(a) BH network ($d = 3, D = 8, \ \mu_{CL} = \mu_{NCL} = 1.4, \ \alpha = 0.75$)

(b) BH/BH network ($d = 3, D = 8, \ \mu_{CL} = 1.4, \ \mu_{NCL} = 2.8, \ \alpha = 0.75$)

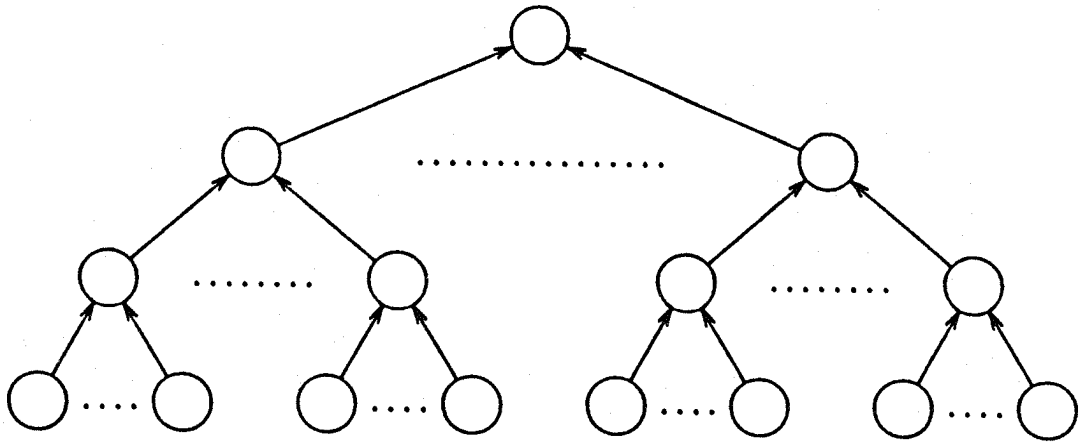


Figure 6 Software combining tree with a fan-in of 10

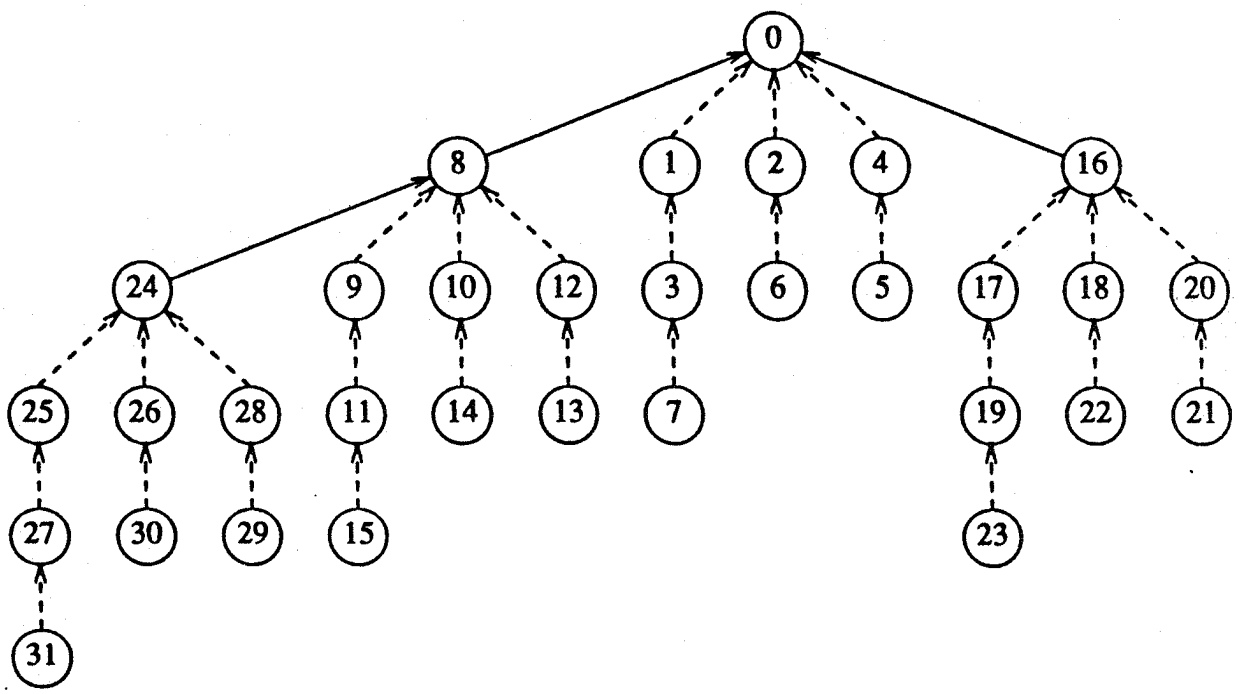
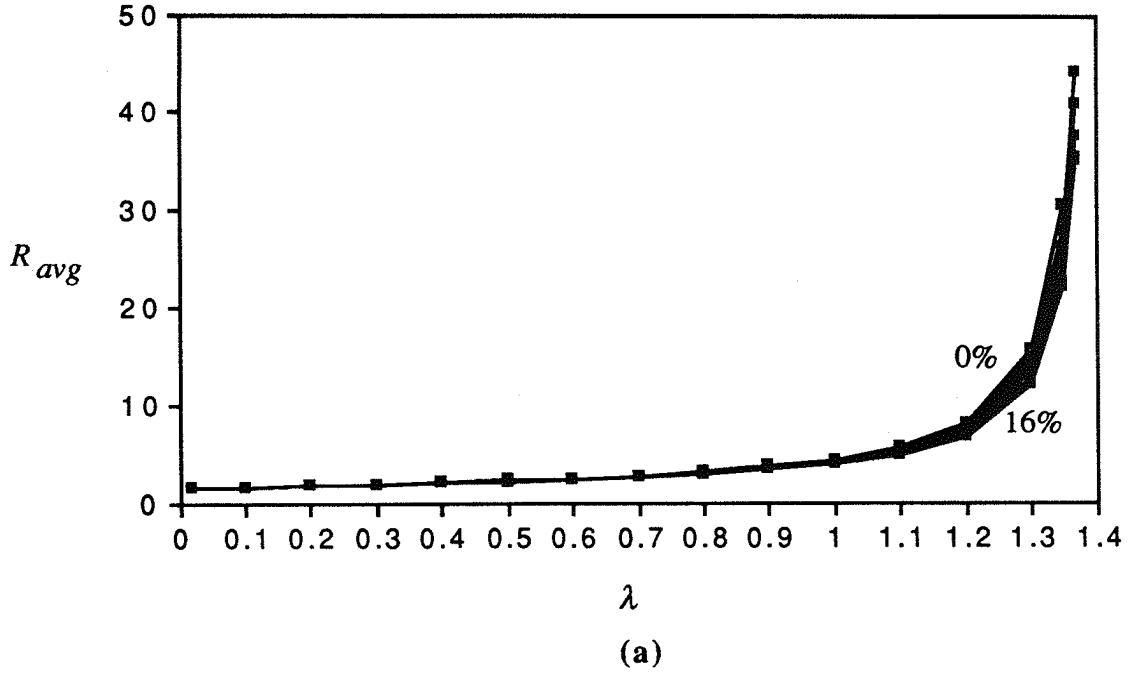
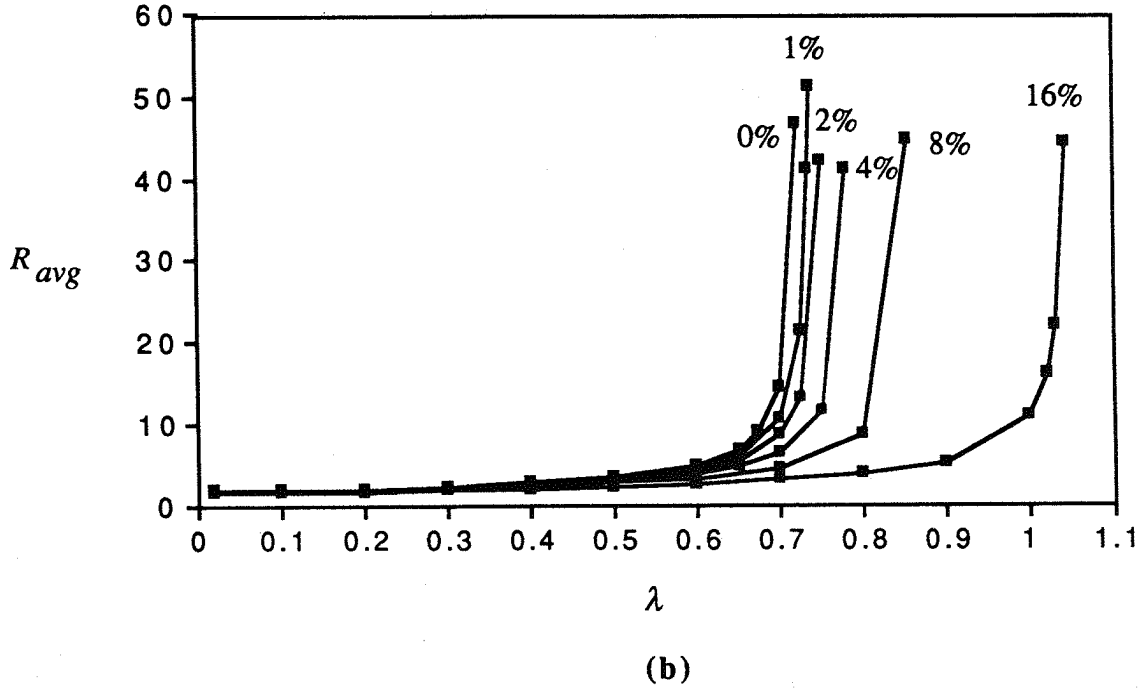


Figure 7 Software combining tree for the BH and BH/BH networks with $D = 5$ and $d = 3$



(a)

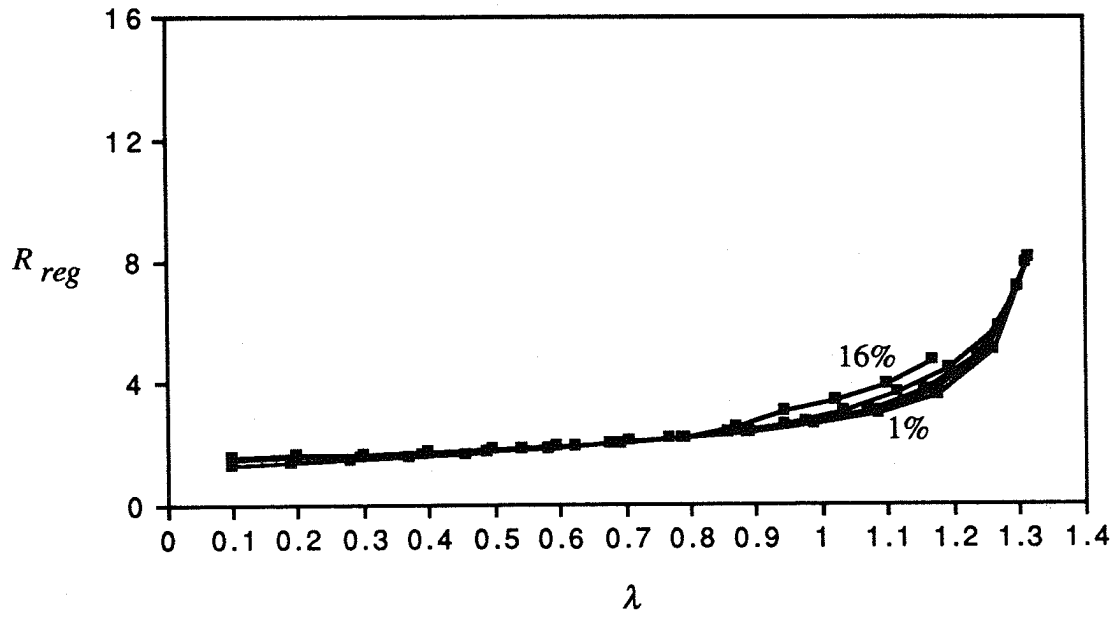


(b)

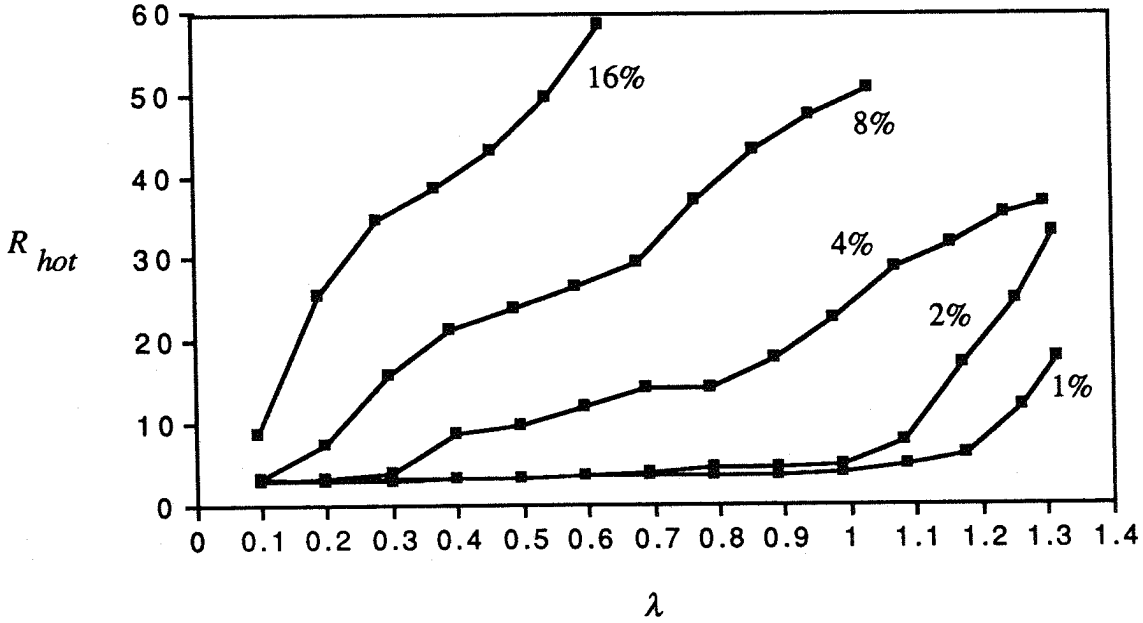
Figure 8 Impact of software combining on hot-spot contention in the BH and BH/BH networks (h is varied from 0% to 16%)

(a) BH network ($d = 3, D = 8, \ \mu_{CL} = \mu_{NCL} = 1.4, \ \alpha = 0.75$)

(b) BH/BH network ($d = 3, D = 8, \ \mu_{CL} = 1.4, \ \mu_{NCL} = 2.8, \ \alpha = 0.75$)



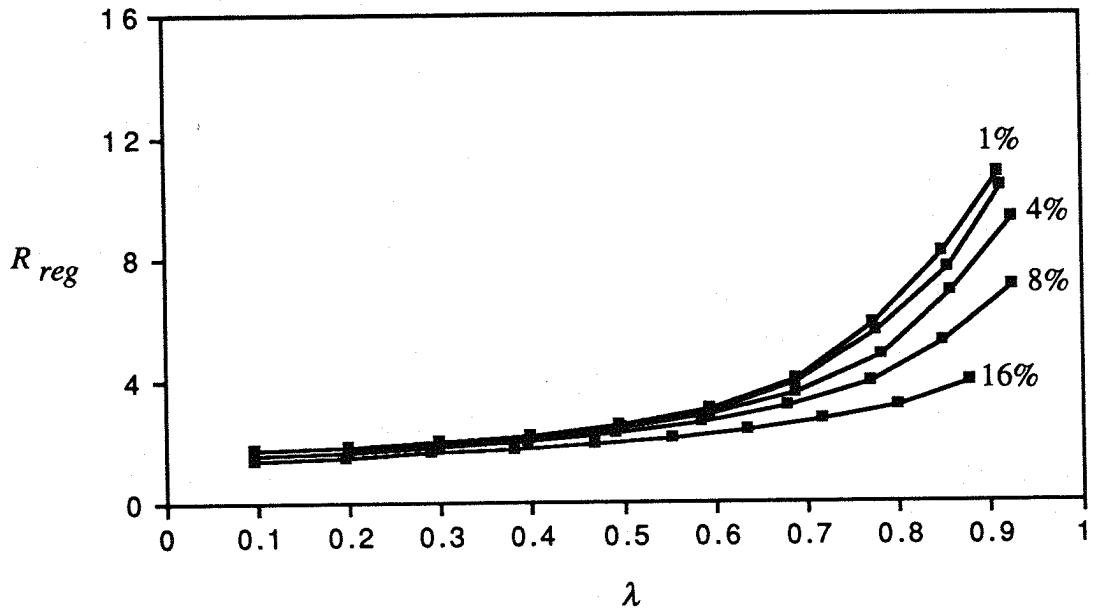
(a)



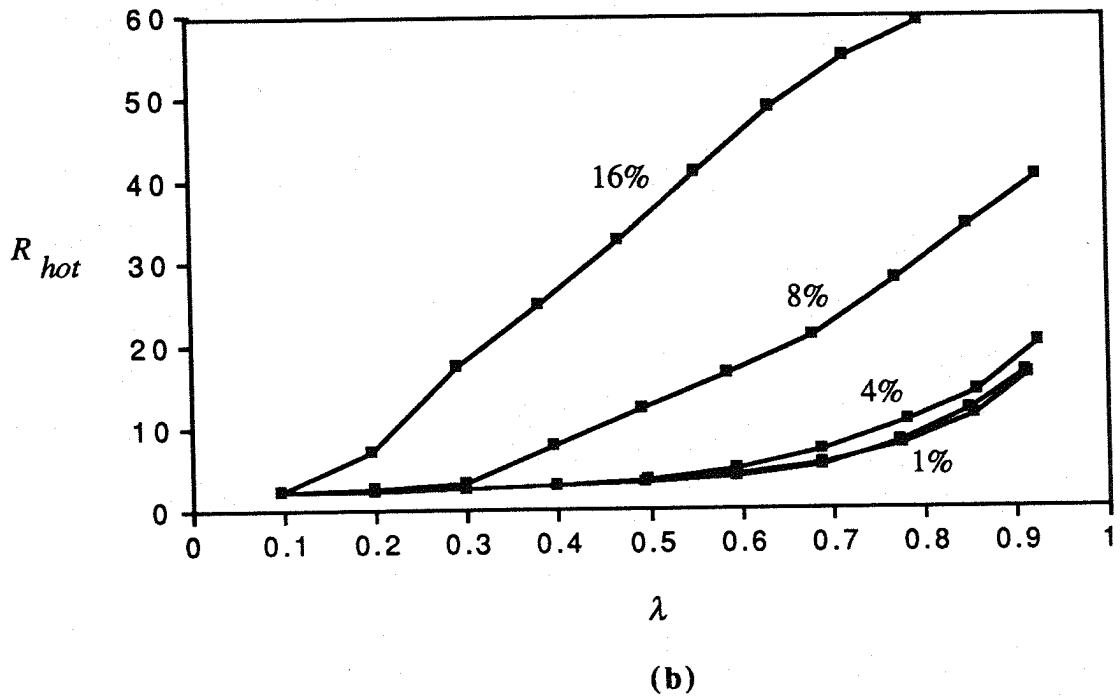
(b)

Figure 9 Impact of an improved routing algorithm on hot-spot contention in the BH network

(h is varied from 1% to 16%, $d = 3, D = 8$, $\mu_{CL} = \mu_{NCL} = 1.4$, $\alpha = 0.75$)

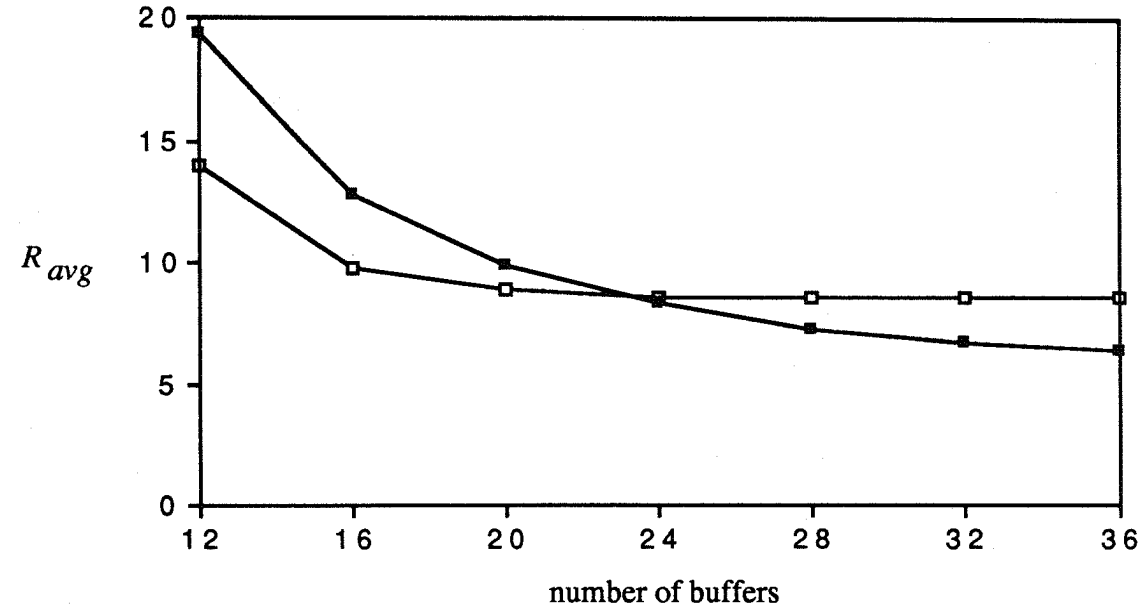


(a)

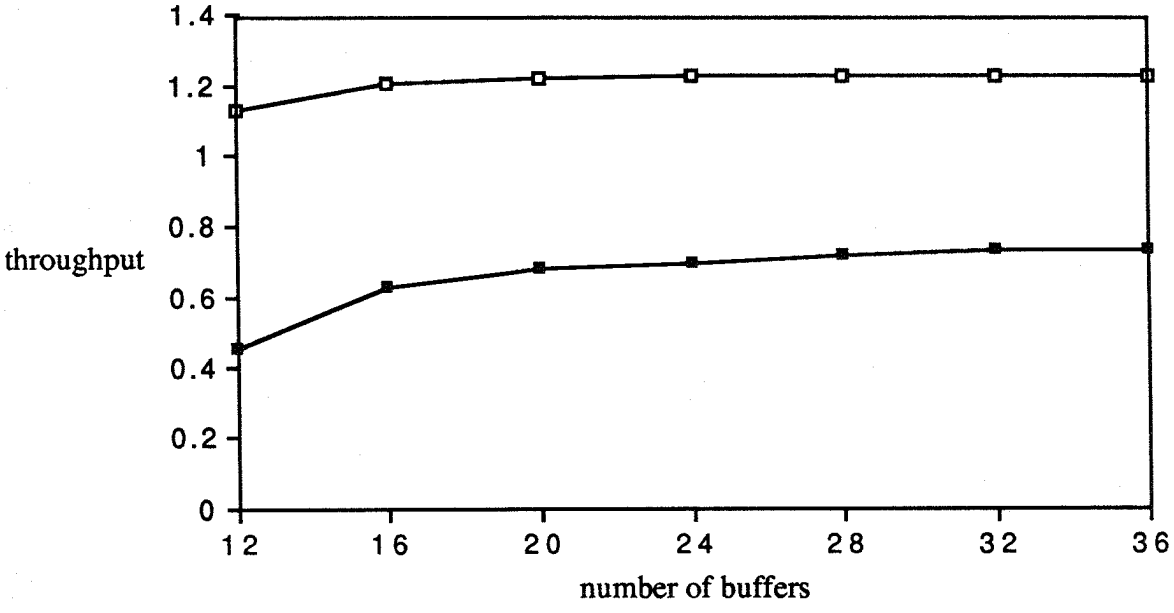


(b)

Figure 10 Impact of an improved routing algorithm on hot-spot contention in the BH/BH network (h is varied from 1% to 16%)
($d = 3, D = 8, \ \mu_{CL} = 1.4, \ \mu_{NCL} = 2.8, \ \alpha = 0.75$)



(a)

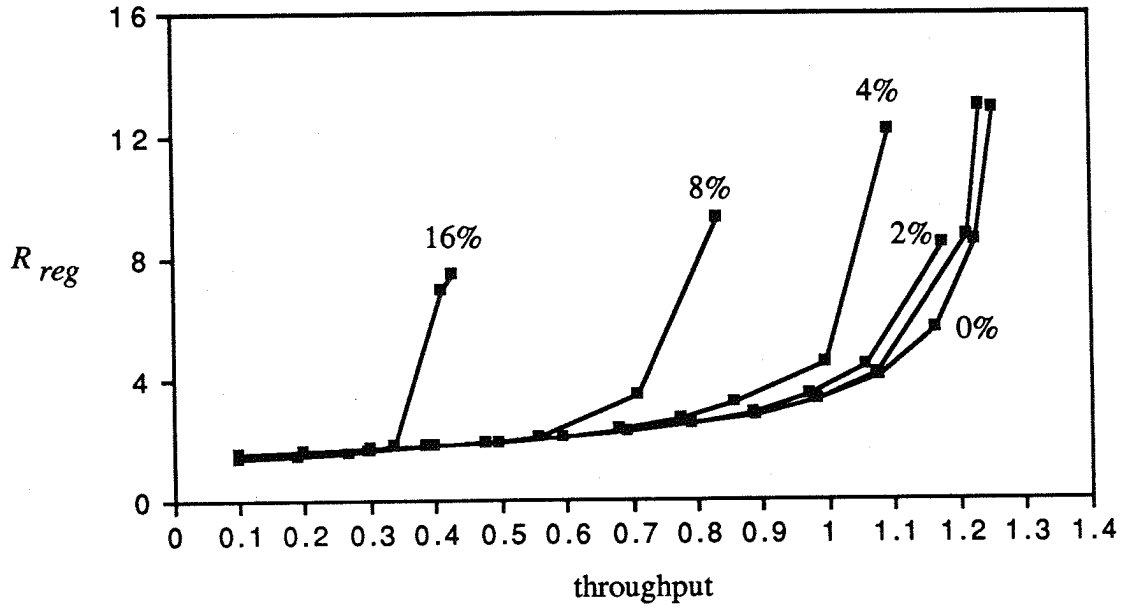


(b)

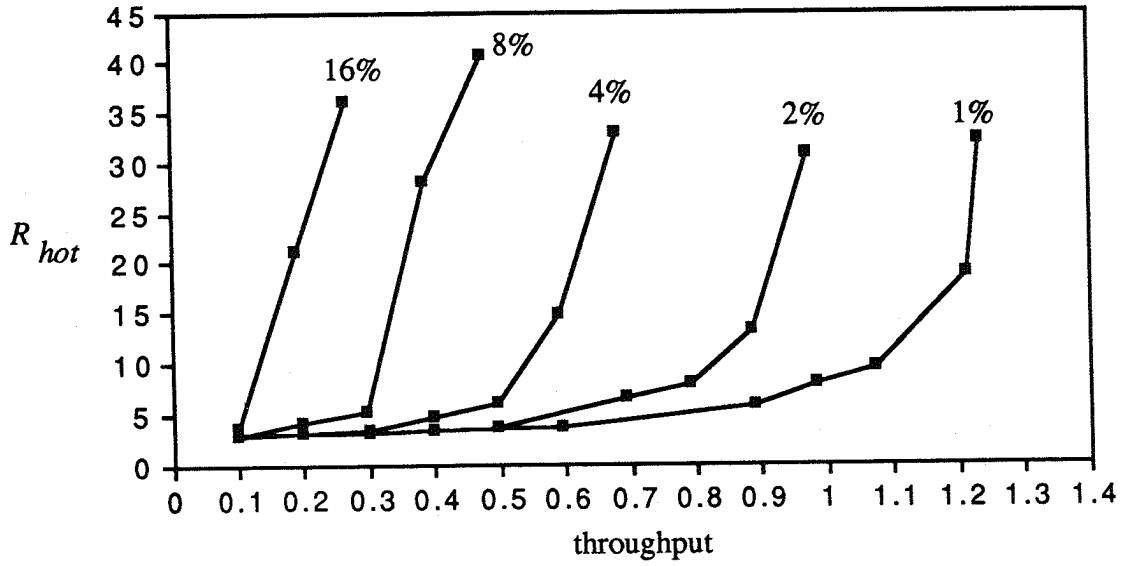
Figure 11 Impact of finite buffering capacity on performance of the BH and BH/BH networks

□ : BH network ($d = 3, D = 8, \mu_{CL} = \mu_{NCL} = 1.4, \lambda = 1.3, \alpha = 0.75, h = 0\%$)

■ : BH/BH network ($d = 3, D = 8, \mu_{CL} = 1.4, \mu_{NCL} = 2.8, \lambda = 0.8, \alpha = 0.75, h = 0\%$)

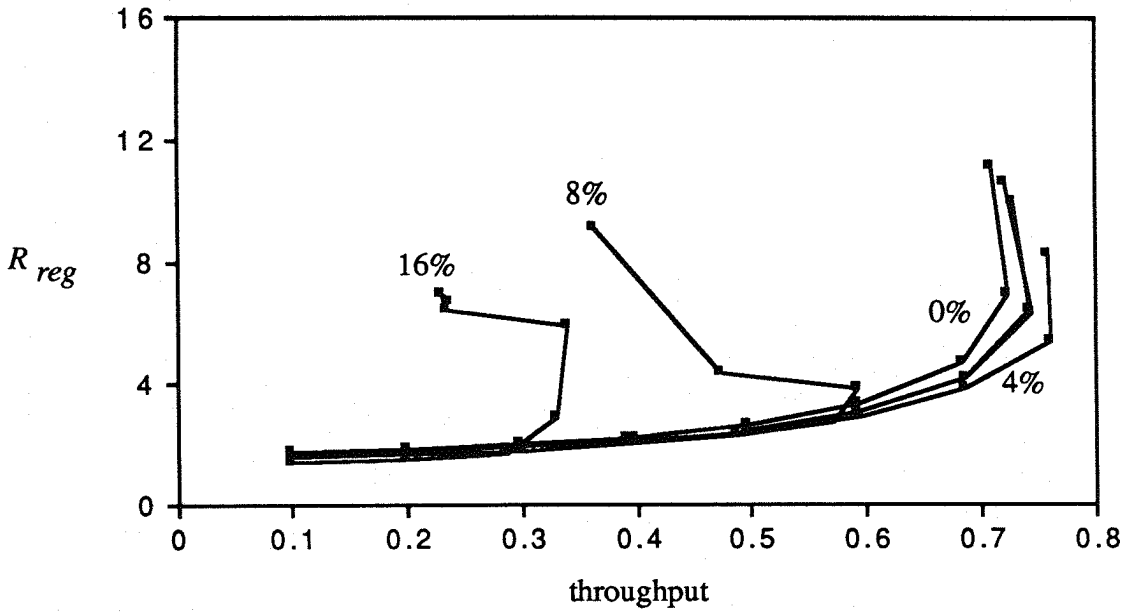


(a)

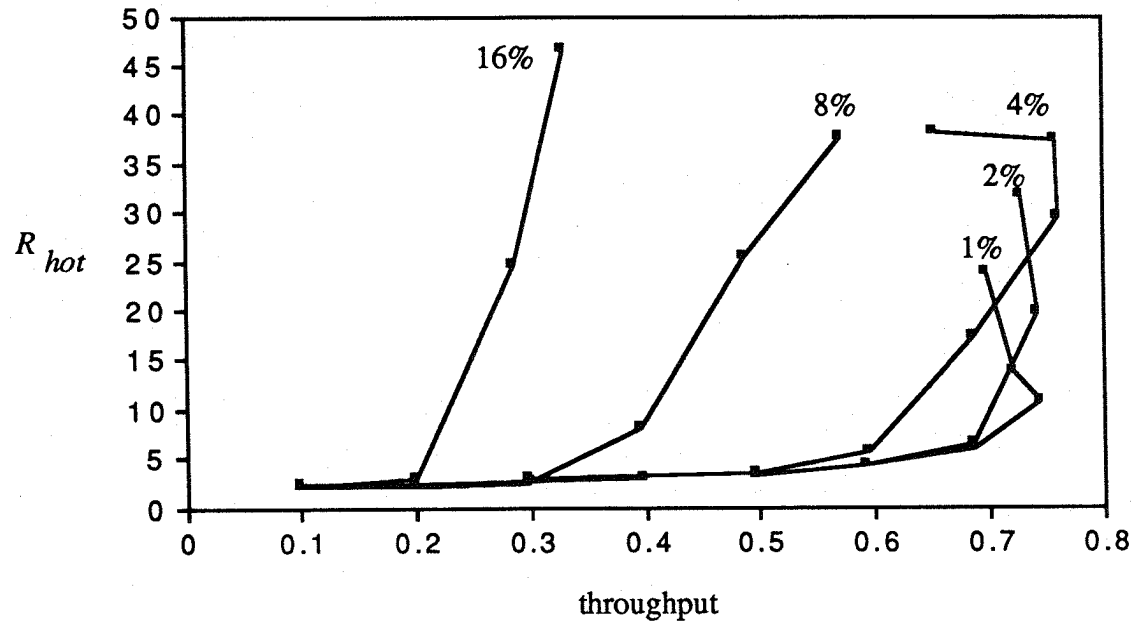


(b)

Figure 12 Impact of hot-spot contention in the BH network -- the case of finite buffering capacity (h is varied from 0% to 16%)
($d = 3, D = 8$, $\mu_{CL} = \mu_{NCL} = 1.4$, $\alpha = 0.75$, number of buffers = 32)

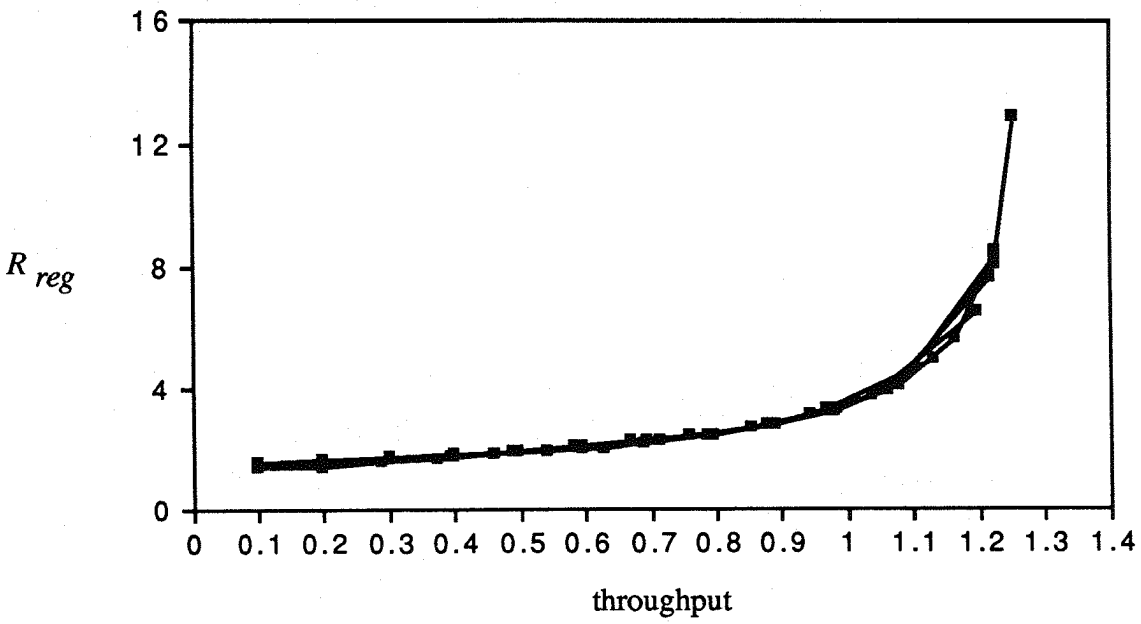


(a)

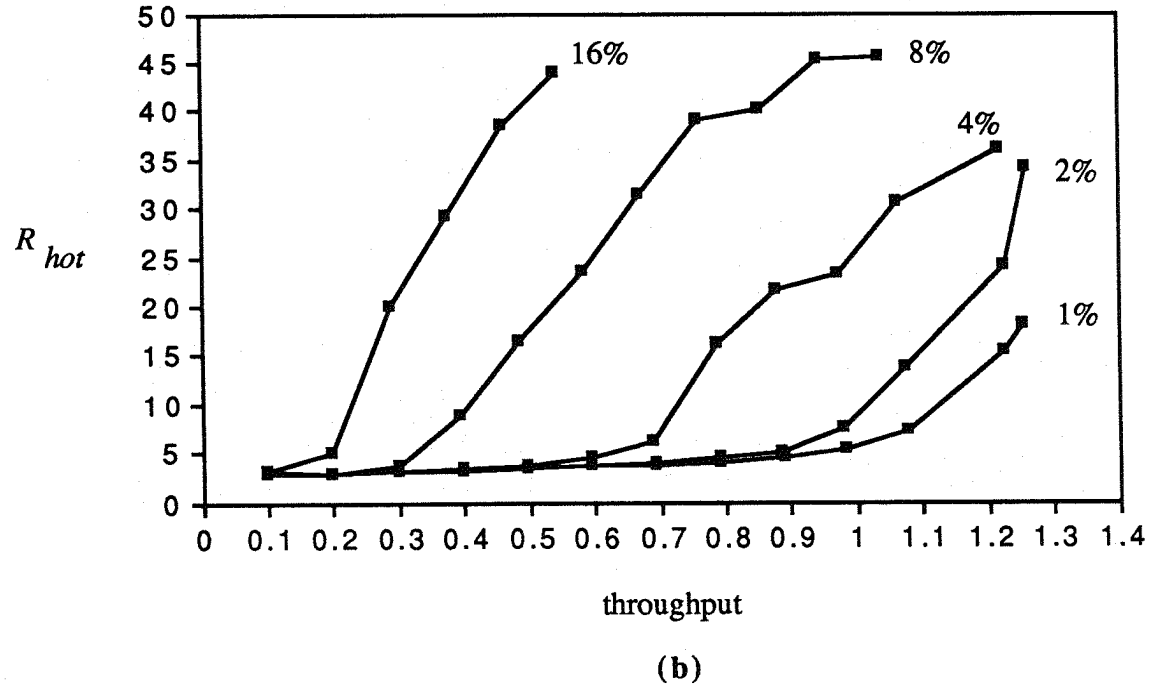


(b)

Figure 13 Impact of hot-spot contention in the BH/BH network -- the case of finite buffering capacity (h is varied from 0% to 16%)
($d = 3, D = 8, \quad \mu_{CL} = 1.4, \quad \mu_{NCL} = 2.8, \quad \alpha = 0.75$, number of buffers = 32)

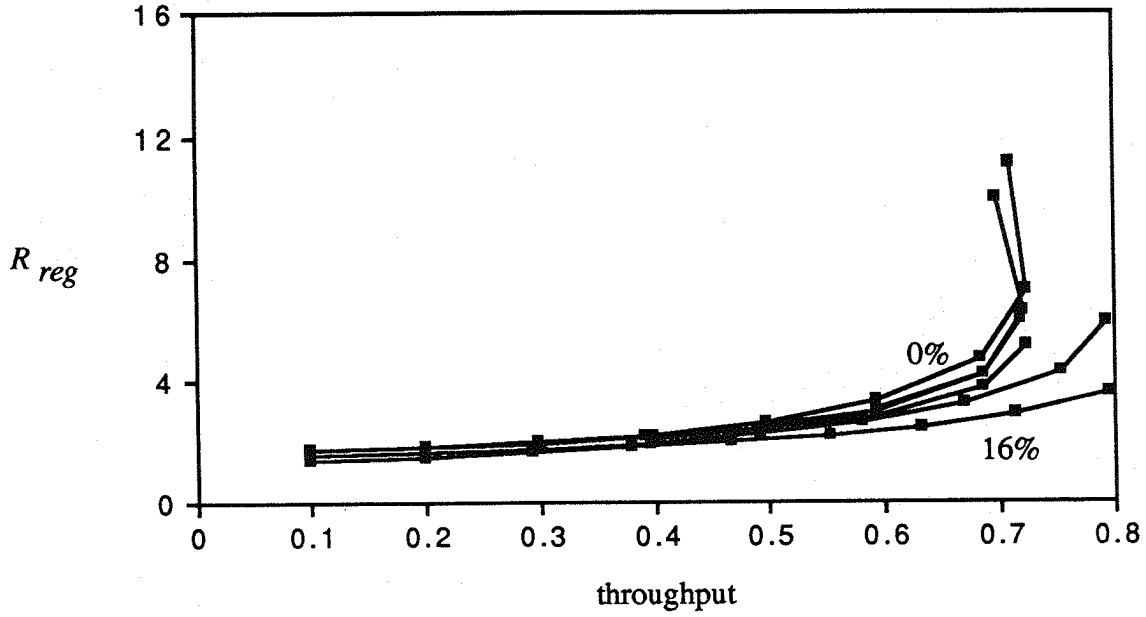


(a)

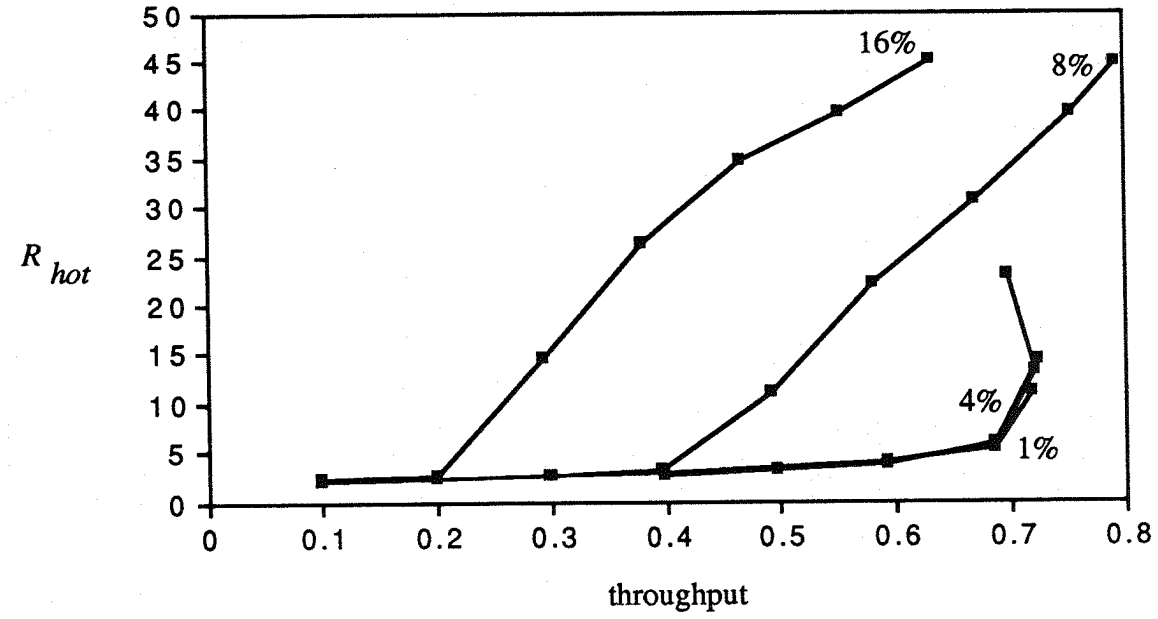


(b)

Figure 14 Impact of hardware combining on hot-spot contention in the BH network -- the case of finite buffering capacity (h is varied from 0% to 16%)
($d = 3, D = 8, \mu_{CL} = \mu_{NCL} = 1.4, \alpha = 0.75$, number of buffers = 32)

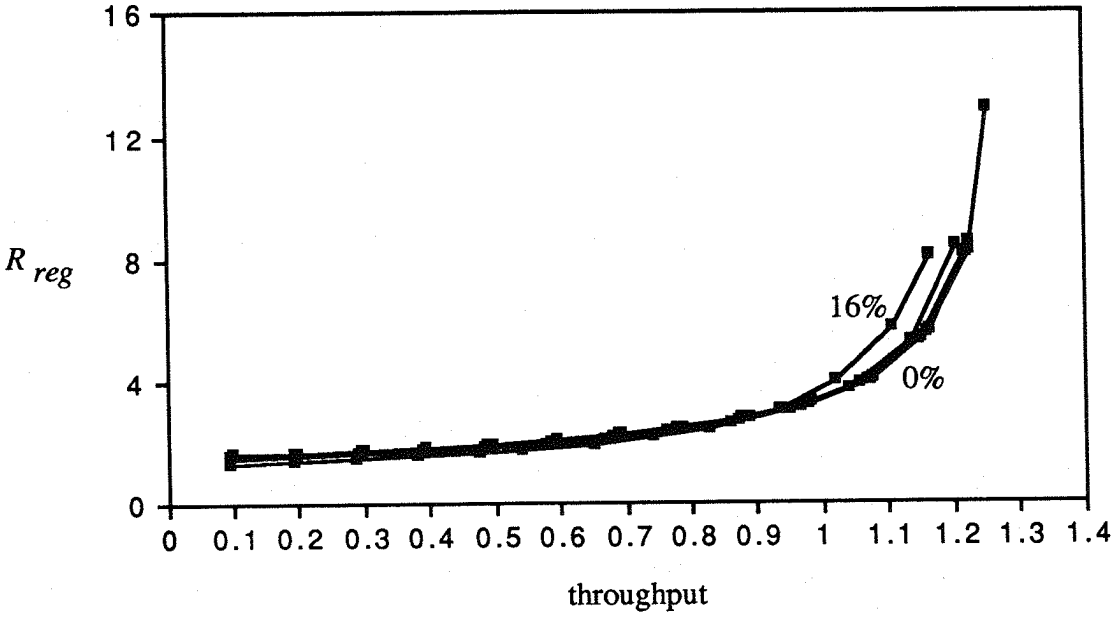


(a)

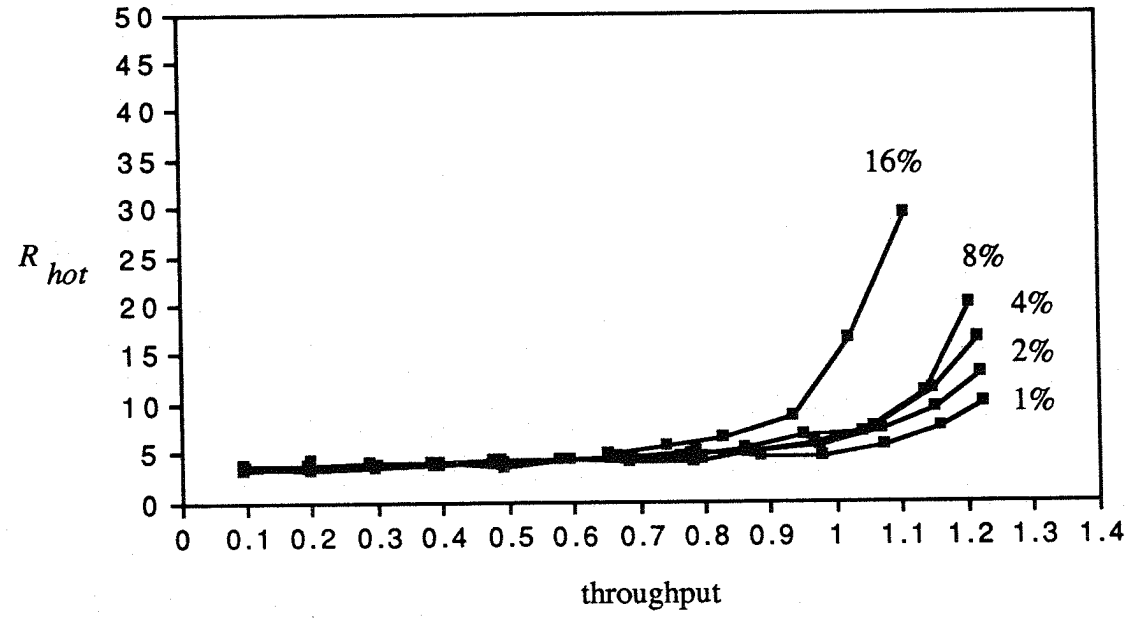


(b)

Figure 15 Impact of hardware combining on hot-spot contention in the BH/BH network -- the case of finite buffering capacity (h is varied from 0% to 16%)
($d = 3, D = 8, \mu_{CL} = 1.4, \mu_{NCL} = 2.8, \alpha = 0.75$, number of buffers = 32)

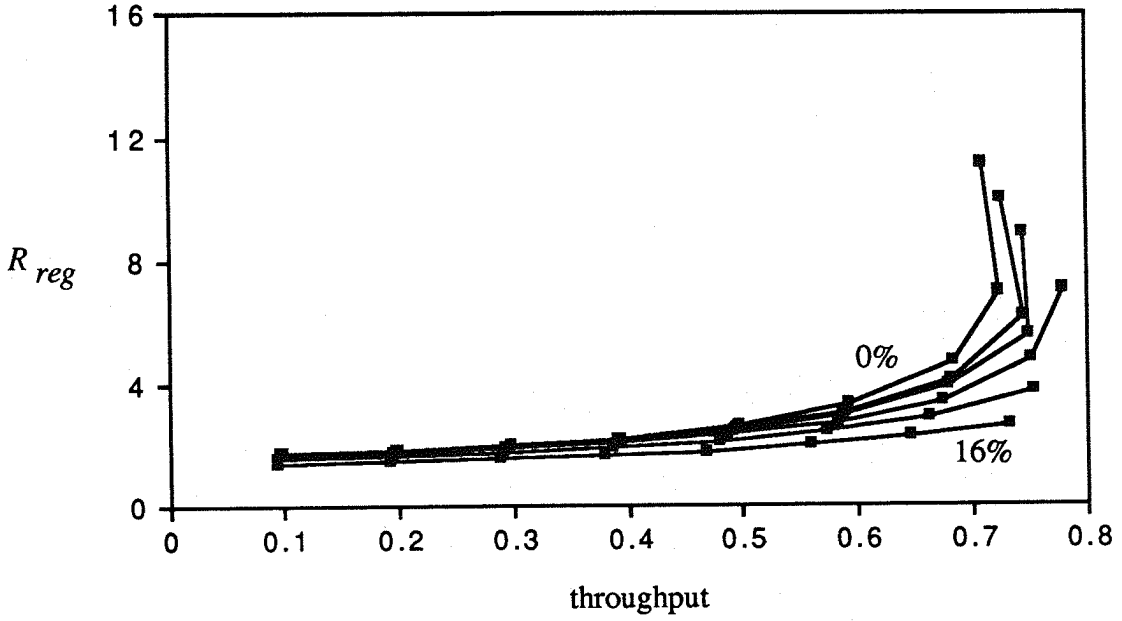


(a)

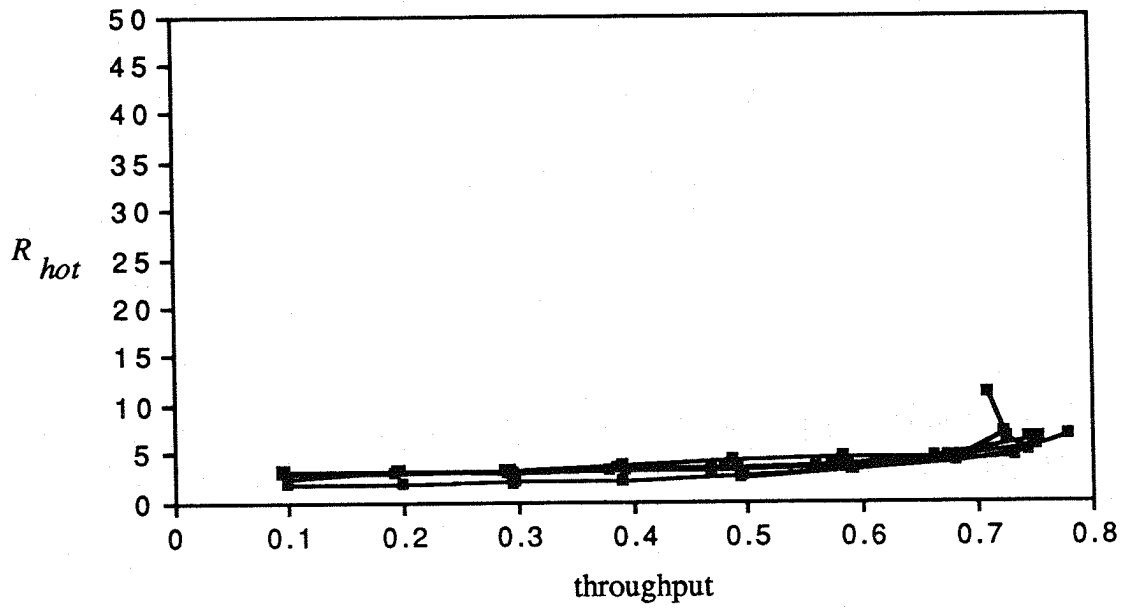


(b)

Figure 16 Impact of software combining on hot-spot contention in the BH network -- the case of finite buffering capacity (h is varied from 0% to 16%)
($d = 3, D = 8, \ \mu_{CL} = \ \mu_{NCL} = 1.4, \ \alpha = 0.75$, number of buffers = 32)



(a)



(b)

Figure 17 Impact of software combining on hot-spot contention in the BH/BH network -- the case of finite buffering capacity (h is varied from 0% to 16%)
($d = 3, D = 8$, $\mu_{CL} = 1.4$, $\mu_{NCL} = 2.8$, $\alpha = 0.75$, number of buffers = 32)