Efficient and Balanced Adaptive Routing in Two-Dimensional Meshes*

Jatin H. Upadhyay, Vara Varavithya, and Prasant Mohapatra
Department of Electrical and Computer Engineering
Iowa State University
Ames, IA 50011
E.mail: *prasant@iastate.edu*

Abstract

In this paper, we present a new concept of region of adaptivity *with respect to various routing algorithms in wormhole networks. Using this concept, we demonstrate that the previously proposed routing algorithms, though more adaptive, cause an uneven workload in the network, which limits the performance improvement. It is observed that balanced distribution of traffic has greater impact on system performance than the adaptivity or efficiency of the algorithm. Based on these motivating factors, We have presented a new fully adaptive routing algorithm for 2-dimensional meshes using one extra virtual channel. The algorithm is more efficient in terms of the number of paths it offers between the source and the destination and also distributes the network load more evenly and symmetrically. The simulation results are presented and are compared with the results of previously proposed algorithms. It is shown that the proposed algorithm results in much better performance in terms of the average network latency and the throughput.*

Key words: *Adaptive routing, Positive-First – Negative-First algorithm, Region of adaptivity, System performance, Traffic distribution.*

1 Introduction

In distributed parallel computers, tasks are executed by a set of intercommunicating nodes or processors. The communication is usually carried out by means of passing messages from one node to another over the interconnection network. Since direct networks utilize the locality of the message references more efficiently, most of the existing systems use direct networks such as k-ary n-cube or n-dimensional meshes. Examples of such systems include T3D [1], Tera Computer systems [2], Touchstone Delta [3], DASH [4], ALEWIFE [5], nCUBE [6] etc. In [7], Dally has shown that lower dimensional networks offer lower latency and higher throughput than higher dimensional networks.

The performance of the interprocessor communication scheme and therefore the multicomputer systems depends greatly on the the switching technique and the routing algorithm selected. Store and forward, virtual cut-though and wormhole routing are the main switching techniques used for interprocessor communication. Amongst the three, virtual cut-through and wormhole routing result in lower latency than store and forward routing. Due to its lower buffer requirements, wormhole routing is preferred over virtual cut through.

Message passing in multicomputer systems is implemented based on certain routing algorithm that determines the path a message follows to reach its destination. Guaranteed delivery of the message requires that the algorithm be deadlock and livelock free. If the path between every pair of source and destination is fixed, the algorithm is called a *deterministic algorithm.* For better system performance, it is however preferable that the algorithm adapt itself to the network faults and traffic congestion and allow alternate paths. This leads to the development of adaptive algorithms. Depending upon whether the algorithm can use all the possible physical paths between the source and the destination, adaptive algorithms are classified as *partially adaptive* or *fully adaptive.* Partially adaptive routing algorithms use only a subset of the available physical paths between the source and destination. Turn model [8, 9], direction restriction model [10] and planar routing [11] are examples of partially adaptive algorithms. Fully adaptive algorithms are de-

*This research was supported in part by the Carver Trust Grant provided by the Iowa State University.

veloped using the concept of virtual channels[12]. Virtual channels are logical abstractions of the same physical channel. Each virtual channel has its own separate queue and is time-multiplexed with other virtual channels associated with the same physical channel. Linder and Harden presented a fully adaptive algorithm for n-dimensional meshes using 2^{n-1} virtual channels [13]. Duato has proposed a fully adaptive routing algorithm for binary hypercubes using only one additional virtual channel [14]. The algorithm is based on dividing all the virtual channels into two sets - waiting channels and non–waiting channels. At each step, a message can route using any of the non–waiting channels and if it gets blocked, it is routed using waiting channels in strict dimensional order. In [15], Su and Shin have also proposed a similar algorithm, called 3P protocol. Boura and Das presented an algorithm called mesh_route, which utilizes more number of virtual paths than the 3P protocol and thereby demonstrates relatively higher efficiency [16]. Efficiency is a measure of the number of the virtual paths the algorithm utilizes.

The adaptive algorithms presented in [14, 15, 16] try to achieve more adaptivity and more number of alternate paths for message routing. In order to achieve high adaptivity and efficiency, these algorithms often favor some messages or some paths over the others. This causes an uneven network traffic distribution in the network. As a result, a part of the network is heavily loaded whereas other regions may be sparsely utilized. This uneven network utilization often results in an early saturation of the network and limits the system performance.

The system performance, thus not only depends on adaptivity of the algorithm, but also on how evenly it routes the network traffic. In fact, our simulation results clearly indicate that the traffic distribution created by the algorithm has a greater impact on the system performance as compared to the efficiency or adaptivity of the algorithm. In [17] Boppana and Chalasani have also shown that more adaptivity does not necessarily mean better performance. Thus the motivation behind our work was to develop a new routing algorithm that is not only more efficient but also achieves an even and balanced traffic distribution.

In this paper, we have first introduced a new concept of *region of adaptivity*, the region where messages can be routed using all the available paths. Using this concept, we theoretically show how various algorithms cause an uneven traffic load in the network and how it affects the system performance. Next, we present a new minimal fully adaptive routing algorithm called Positive-First– Negative-First(PFNF) algorithm, based on the turn model [8]. Using an analysis similar to [16] we show that our algorithm outperforms all the previously proposed algorithms in terms of the number of paths it allows between the source and the destination. Using *region of adaptivity* and simulation results we also show that it creates a very balanced and symmetric traffic load in the network.

The simulation results are presented and compared with previously proposed algorithms. While the efficiency of 3P protocol asymptotically decreases to zero with increasing network radix and mesh_route stabilizes at around 0.25, PFNF algorithm gives efficiency near 0.5. In terms of the average latency and throughput, PFNF algorithm gives significant improvement over 3P protocol and mesh_route algorithms. The traffic at which the PFNF algorithm saturates is also considerably higher than that of both 3P protocol and mesh_route algorithms.

The rest of the paper is organized as follows. Section 2 summarizes the required definitions and introduces the concept of region of adaptivity. It also shows how region of adaptivity is related to the traffic distribution the algorithm creates in the network. Section 3 discusses the motivation behind our work. A new routing algorithm is proposed in Section 4 along with its efficiency and adaptivity analysis. Section 5 discusses the simulation results and compares the performances of several algorithms. Section 6 concludes the paper.

2 Definitions and Concepts

In this section, we first define the terminologies associated with the adaptive routing scheme. Some of these definitions are reiterated from previous works [7, 14, 16] for the sake of completeness. We also introduce a new term, region of adaptivity, that is useful in designing an efficient adaptive routing algorithm.

Definition 1: A *Physical Interconnection Network*, PN, is a strongly connected graph, $PN(PV, PC)$, where PV represents the set of processing nodes and PC represents the set of physical channels connecting the nodes.

Definition 2: A *Virtual Interconnection Network*, VN, is a strongly connected graph, $VN(PV, VC)$, where PV represents the set of processing nodes and VC represents the set of virtual channels, that are mapped to the set of physical channels PV.

Definition 3: An *n-dimensional mesh* is defined as an interconnection structure that has $K_0 \times K_1 \times \ldots \times K_{n-1}$ nodes with n as the number of dimensions and K_i as

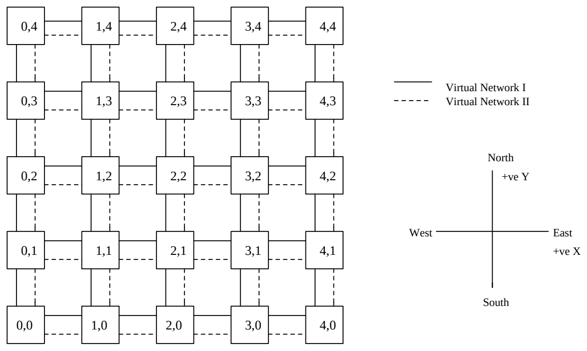

Figure 1: A 5×5 two-dimensional mesh

the number of nodes in ith dimension. Each node in the mesh is identified by an n-coordinate vector $(x_0, x_1, \ldots, x_{n-1})$, where $0 \le x_i \le K_i - 1$. Two nodes, $(x_0, x_1, \ldots, x_{n-1})$ and $(y_0, y_1, \ldots, y_{n-1})$, are connected if and only if there exists an i such that $x_i = y_i \pm 1$, and $xi = y_i$ for all $j \ne i$.

Figure 1 shows a 5×5 two-dimensional mesh with one additional virtual channel per physical channel. Two virtual networks are shown as VN_1 and VN_2. The figure also shows the directional notations we have used throughout this paper.

Definition 4: A *routing algorithm* $R : N \times N \to \rho(C)$, where $\rho(C)$ is the power set of C, supplies a set of alternative output channels to send a message from current node n_c to the destination node n_d. $R(n_c, n_d) = (c_1, c_2, \ldots, c_p)$.

Definition 5: *Efficiency* of a *fully adaptive* routing algorithm is defined as the ratio of the number of the shortest virtual paths that the algorithm can use for the delivery of messages between any two nodes in the network to the total number of all such shortest paths between these two nodes.

Definition 6: *Region of Adaptivity* is the average region (set of nodes) of the network to where all the messages can be routed fully adaptively (i.e. using all the physical paths or all the virtual paths, if virtual channels are used) using the chosen algorithm.

Consider a 2-dimensional mesh network as shown in the Figure 2(a). With node $< 3, 1 >$ as the source node, under the East-First algorithm [8], all the messages directed towards any of the nodes in the shaded region can be routed fully adaptively, while messages to any node outside this region would be routed deterministically. We call the shaded region as the region of adaptivity of the node $< 3, 1 >$ under the East-First algorithm. Figure 2(b) represents the region of adaptivity for the node $< 1, 2 >$. As a whole, the *region of adaptivity* of the the East-First algorithm can be represented as the shaded region shown in Fig-

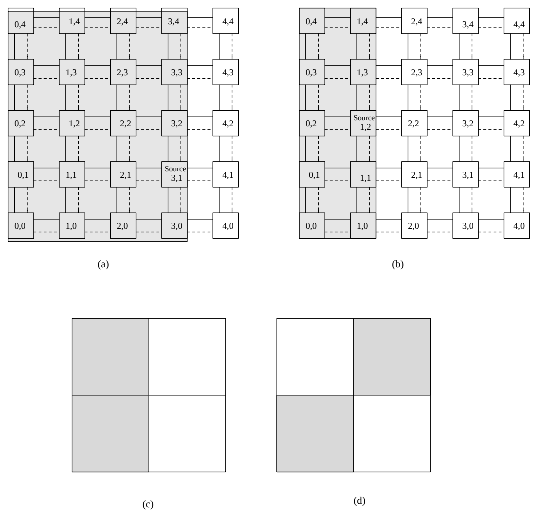

Figure 2: *Region of adaptivity* of (a) node $< 3, 1 >$ (b) node $< 1, 2 >$ (c) the East-First algorithm (d) the Positive-First algorithm

ure 2(c). Figure 2(d) shows the region of adaptivity for the Positive-First routing algorithm[8]. If virtual channels are used, the region of adaptivity can be obtained by considering adaptive regions of all the virtual networks.

The concept of *region of adaptivity* serves well to understand the behavior of the algorithms – particularly on how evenly the algorithm balances the network load. As can be seen from the Figure 2(c), since the East-First algorithm has one half of the mesh as the fully adaptive region, all the messages have more number of alternate paths to route to this region than to the other parts of the mesh. Since this region is not symmetric in the mesh, assuming uniform traffic generation at all nodes, it causes more traffic congestion in one part of the network and hence an early saturation. Saturation in only one part of the network saturates the rest of the network and hence degrades the system performance.

3 Motivation

The motivation for development of our algorithm is driven from our observation that most of the fully adaptive algorithms presented in the literature either have poor efficiency or their efficiency is improved at the expense of uneven traffic distribution in the network. We illustrate this point by comparing two recently proposed algorithms – the 3P protocol [15] and mesh_route [16].

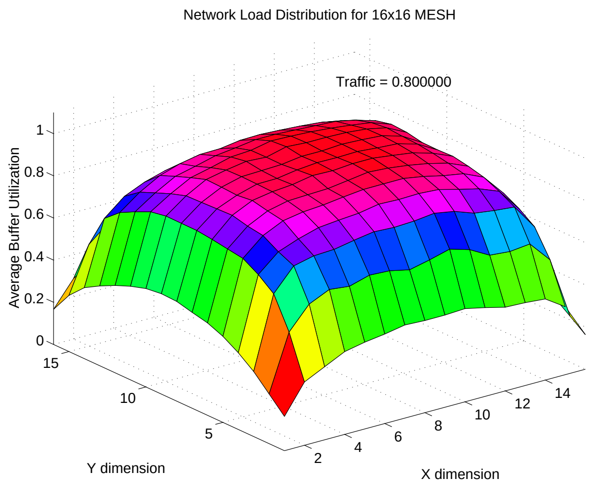

Figure 3: Traffic distribution for 3P algorithms

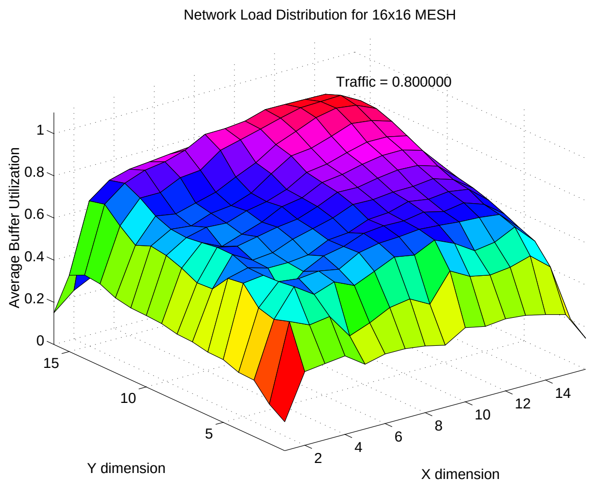

Figure 4: Traffic distribution for mesh_route algorithms

Both the 3P protocol and mesh_route algorithms divide the virtual channels into two separate sets – waiting channels and non–waiting channels. In 3P protocol, a message can travel using any non–waiting virtual channel. If it gets blocked, it travels through the waiting channels in strict dimensional order routing. Similarly in mesh_route algorithm, a message can travel using any available non–waiting channel. If blocked, all the messages are restricted to the dimensional order routing in waiting channels except those who have to go negative both in x and y directions. These messages can use any of the waiting channels without any restrictions.

The traffic distribution in terms of buffer utilization is shown in Figure 3 and 4 for the 3P protocol and mesh_route algorithms, respectively. While 3P protocol distributes the traffic very evenly, the traffic is concentrated in one quadrant of the mesh for the mesh_route algorithm. This uneven load distribution creates a bottleneck as the high traffic areas saturate early and in turn saturate the whole network.

While the 3P protocol gives uniform traffic distribution, it has a very low efficiency due to the dimensional order routing in the waiting–channels[16]. Mesh_route on the other hand is relatively more efficient but favors messages going in the negative directions and hence creates an uneven distribution of traffic in the network. The motivation behind our work was to obtain an algorithm which is not only more efficient but also produces a very balanced and symmetric network traffic load and thereby improves the system performance.

4 Proposed Algorithm

In this section, we propose a fully adaptive routing algorithm, called Positive-First–Negative-First(PFNF) algorithm, for 2-dimensional mesh networks. The algorithm uses one additional virtual channel per physical channel in the network.

The basic concept behind our algorithm is as follows. The physical interconnection network PN is logically divided into two virtual networks, VN_1 and VN_2 such that the two virtual channels associated with the same physical channel are in different virtual networks. A separate routing algorithm is used in both the virtual networks VN_1 and VN_2. At each step, a set of virtual channels are chosen from each of the virtual networks, depending upon the routing tag and the routing algorithm for that particular virtual network. The selection policy then determines which of these channels should the message be routed along.

The overall structure of the routing algorithm is as shown below. Here $routing\_tag$ refers to the vector of length n, the network dimension, where $routing\_tag[i] = destination\_address[i] - current\_node\_address[i]$.

```
Route(routing_tag){
    virtual_channel_set $vc_1$;
    virtual_channel_set $vc_2$;
    virtual_channel $vc$;
    $vc_1$ = Routing_algorithm_for_$VN_1$(routing_tag);
    $vc_2$ = Routing_algorithm_for_$VN_2$(routing_tag);
    $vc$ = Selection policy($vc_1$, $vc_2$)
    Route along channel $vc$.
}
```

Suitable selection of $Routing\_algorithm\_for\_VN_1$ and $Routing\_algorithm\_for\_VN_2$ functions would give different routing algorithms. In order to distribute the traffic load symmetrically within each virtual network and also between the two virtual networks, we use the Positive-First algorithm in one virtual layer and Negative-First algorithm in another. The choice of the selection policy is independent of the routing algorithm. The algorithm is outlined in Figure 5.

Positive-First–Negative-First(PFNF) algorithm:

```
Routing_algorithm_for_$VN_1$:
    /* Positive First Algorithm */
    For all dimension $i${
        If ($\exists$ $routing\_tag[i] > 0$) {
            select channels along direction $i$, if free
            return;
        }
        else{
            For all dimension $i$ {
            If ($\exists$ $routing\_tag[i] \neq 0$)
                select channels along direction $i$,
                if free.
            }
        }
    }

Routing_algorithm_for_$VN_2$
    /* Negative First Algorithm */
    For all dimension $i${
        If ($\exists$ $routing\_tag[i] < 0$) {
            select channels along direction $i$, if free
            return;
        }
        else{
            For all dimension $i$ {
            If ($\exists$ $routing\_tag[i] \neq 0$)
                select channels along direction $i$,
                if free.
            }
        }
    }
```

Figure 5: Positive-First–Negative-First Algorithm for two-dimensional meshes

Lemma 1: Under the PFNF algorithm, a message can be routed to its destination using channels of only one virtual network.

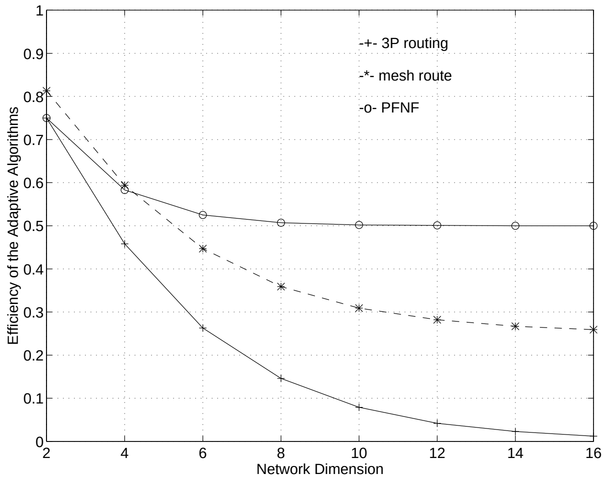

Figure 6: Efficiency of the 3P protocol, the mesh_route and the PFNF algorithms with respect to the network radix for a $K \times K$ mesh

Proof: The PFNF algorithm divides the physical interconnection network PN into two virtual networks VN_1 and VN_2. The two virtual channels associated with each physical channel belong to a different virtual network. Since physical network PN is connected, both VN_1 and VN_2 will also be connected.

Both Positive-First and Negative-First algorithms are connected i.e., a message can be routed from a node to another node using either of the Positive-First or the Negative-First algorithms [8].

The PFNF algorithm uses Positive-First routing in VN_1 and Negative-First routing in VN_2. As both VN_1 and VN_2 are individually connected and so also are the Positive-First and the Negative-First algorithms, a message can be routed to its destination using channels of only one virtual network. Q.E.D.

Lemma 2: In the PFNF algorithm, routing in each virtual network is individually deadlock free.

Proof: The PFNF algorithm has separate routing functions for virtual network VN_1 and VN_2. Routing in VN_1 is using the Positive-First algorithm and routing in VN_2 is using the Negative-First algorithm. Both Positive-First and Negative-First algorithms are deadlock free[8]. Hence under the PFNF algorithm, routing in each of the virtual networks is individually deadlock free. Q.E.D.

Theorem 1: The Positive-First – Negative-First(PFNF) algorithm is deadlock free.

Proof: The proof of this theorem is a direct consequence of Lemma 1 and Lemma 2. Assume the head flit of a particular message is in virtual network VN_i, $i = 1$ or 2. The message can reach its destination using the channels of only VN_i (Lemma 1). Secondly, since virtual network VN_i is deadlock free, it guarantees the delivery of the message to its destination. Thus under PFNF algorithm a message can always reach its destination. Hence the PFNF algorithm is deadlock free. Q.E.D.

4.1 Efficiency Analysis

The efficiency of an algorithm is the percentage of the total number of paths the algorithm allows to be used to route the messages [16]. When virtual channels are used, the total number of paths refer to the total number of virtual paths. If P is the total number of paths between the source and destination and P_R is the number of paths allowed by the algorithm R then the efficiency ϵ_R of the algorithm R can be defined as:

$$\epsilon_R = \frac{P_R}{P} \tag{1}$$

To determine the efficiency of PFNF algorithm, consider a 2-dimensional mesh with source $S(x_0, x_1)$ and the destination $D(y_0, y_1)$.

The total number of all the shortest virtual paths between the nodes $S(x_0, x_1)$ and $D(y_0, y_1)$ will be

$$P = \frac{(\|x_0 - y_0\| + \|x_1 - y_1\|)!}{(\|x_0 - y_0\|)!\,(\|x_1 - y_1\|)!} \times 2^{\|x_0 - y_0\| + \|x_1 - y_1\|}. \tag{2}$$

Let P_i be the number of shortest paths allowed by the algorithm under the condition i. Each of these conditions is considered in the following cases.

Positive-First – Negative-First(PFNF) Algorithm

case 1: $\|x_0 - y_0\| = 0$ and $\|x_1 - y_1\| \neq 0$

$$P_1 = 2^{\|x_1 - y_1\|} \tag{3}$$

case 2: $\|x_0 - y_0\| \neq 0$ and $\|x_1 - y_1\| = 0$

$$P_2 = 2^{\|x_0 - y_0\|} \tag{4}$$

case 3: $x_0 < y_0$ and $x_1 < y_1$

$$P_3 = \frac{(\|x_0 - y_0\| + \|x_1 - y_1\|)!}{(\|x_0 - y_0\|)!\,(\|x_1 - y_1\|)!} \times 2^{\|x_0 - y_0\| + \|x_1 - y_1\|} \tag{5}$$

case 4: $x_0 < y_0$ and $x_1 > y_1$

$$P_4 = \begin{cases} \sum_{k=1}^{\|x_0-y_0\|} \frac{((\|x_0-y_0\|-k)+(\|x_1-y_1\|-1))!}{(\|x_0-y_0\|-k)!\,(\|x_1-y_1\|-1)!} \times 2^k \\ \sum_{k=1}^{\|x_1-y_1\|} \frac{((\|x_0-y_0\|-1)+(\|x_1-y_1\|-k))!}{(\|x_0-y_0\|-1)!\,(\|x_1-y_1\|-k)!} \times 2^k \end{cases} \tag{6}$$

case:5 $x_0 > y_0$ and $x_1 > y_1$

$$P_5 = \frac{(\|x_0 - y_0\| + \|x_1 - y_1\|)!}{(\|x_0 - y_0\|)!\,(\|x_1 - y_1\|)!} \times 2^{\|x_0 - y_0\| + \|x_1 - y_1\|} \tag{7}$$

case 6: $x_0 > y_0$ and $x_1 > y_1$

$$P_6 = \begin{cases} \sum_{k=1}^{\|x_0-y_0\|} \frac{((\|x_0-y_0\|-k)+(\|x_1-y_1\|-1))!}{(\|x_0-y_0\|-k)!\,(\|x_1-y_1\|-1)!} \times 2^k \\ \sum_{k=1}^{\|x_1-y_1\|} \frac{((\|x_0-y_0\|-1)+(\|x_1-y_1\|-k))!}{(\|x_0-y_0\|-1)!\,(\|x_1-y_1\|-k)!} \times 2^k \end{cases} \tag{8}$$

The efficiency of the PFNF algorithm would be:

$$\epsilon_{PFNF} = \frac{P_1 + P_2 + P_3 + P_4 + P_5 + P_6}{P} \tag{9}$$

The efficiency analysis of 3P protocol and mesh_route is presented in [16]. Figure 6 plots the efficiency of the PFNF algorithm. For the purpose of comparison, we have also plotted the efficiency of the 3P Protocol and the mesh_route algorithms. As can be seen from the graph, PFNF exhibits almost twice as much efficiency as mesh_route algorithm and hence twice as much number of alternate paths between the source and the destination. Unlike 3P protocol whose efficiency asymptotically decreases to zero as the network radix is increased, The efficiency of PFNF algorithm stabilizes at a value near 0.5.

4.2 Region of Adaptivity

This subsection compares the region of adaptivity for all the three algorithms discussed earlier. As all these algorithms use one additional virtual channel per physical channel, we shall consider the whole network as consisting of two separate virtual networks. *Region of adaptivity* for each virtual network is(are) the quadrant(s) in which full adaptivity is offered by the algorithm in that particular network. The *region of adaptivity* of the actual network is then obtained by superimposing the adaptive regions of the two virtual network.

The *adaptive regions* for each virtual network and the total *region of adaptivity* for each of these algorithms are shown in Figure 7. Here the shaded areas represent the adaptive regions of the network as explained in Section 2.

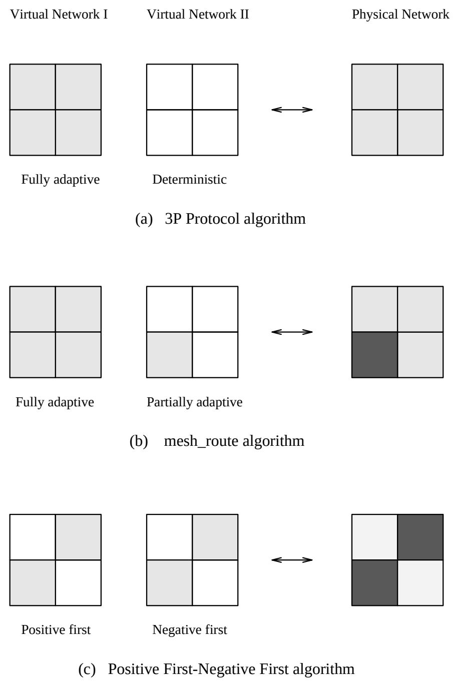

Figure 7: *Region of Adaptivity* for (a) 3P protocol (b)mesh_route (c) PFNF algorithms

The region of adaptivity for the 3P protocol and mesh_route algorithms shown in Figures 2(a) and (b) closely relate to the traffic distribution caused by these algorithms (Figures 3(a) and (b)). Thus the concept of *region of adaptivity* indeed illustrates how various algorithms distribute traffic in the network. It can be inferred from Figure 7 that since the *region of adaptivity* for the PFNF algorithm is symmetric and since the PFNF algorithm is also more efficient, it would perform better than 3P protocol and mesh_route algorithms. The results presented in Section 5 confirm this inference.

5 Performance Evaluation

In this section we present the performance evaluation of PFNF algorithm. The results for various traffic patterns and network loads are presented and compared with those of the 3P protocol and mesh_route schemes.

5.1 Simulation Environment

We have developed a time driven simulator to evaluate the performance of the above mentioned routing algorithms. At each time step all the packets that have been given access to the channels are moved synchronously and the time step is advanced. The results were reproduced several times and were observed to be consistent with maximum deviation of only 1%.

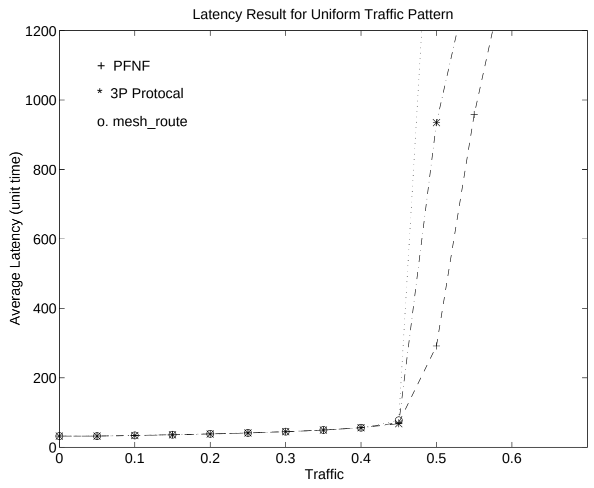

Figure 8: Latency results for the uniform traffic pattern

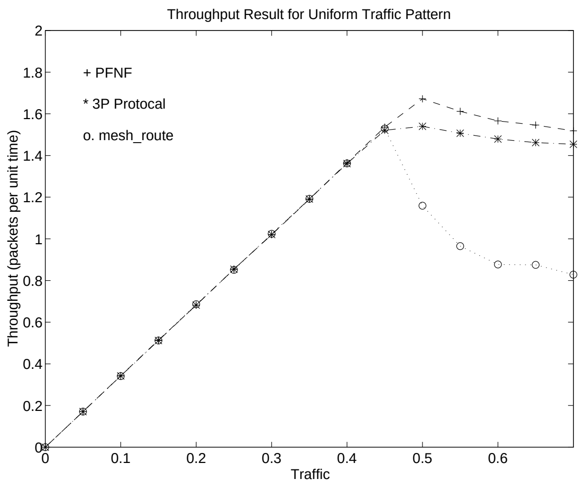

Figure 9: Throughput results for the uniform traffic pattern

The simulations were conducted on a 16×16 mesh. We have assumed 20 flits per packet. Each virtual channel is assumed to have only one flit buffer associated with it. Packet generation rate is Poisson in nature with an exponential distribution of inter-arrival time. We have used *multiplex biased* selection policy for all the algorithms, i.e., from the set of channels defined by the routing algorithm, the one which would avoid multiplexing with other messages is selected, if available. The selection is otherwise random.

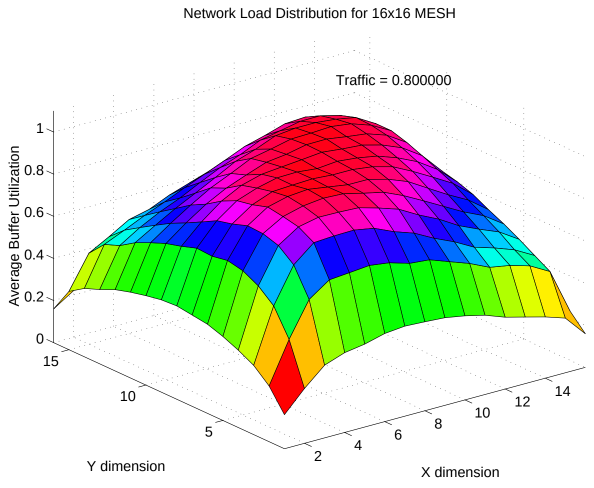

Figure 10: Traffic load distribution produced by the PFNF algorithm

The simulation was carried out for 150,000 packets. The first 40,000 to 60,000 packets were not included in the results to reduce the transient effects in the simulations.

5.1.1 Traffic patterns

Uniform, hotspot, and transpose traffic patterns were considered in our study. Under the uniform traffic pattern, a node sends messages to every other node with equal probability. In hotspot traffic, one particular node receives additional hotspot traffic besides its normal traffic. Using the parameters from [17], we have considered only one hotspot node with the hotspot percentage of four, i.e., in a 16×16 mesh a message is directed to the hotspot node with the probability of 0.0438 and to any other nodes with a probability of 0.0038. We have chosen node $< 5, 5 >$ as the hotspot node. In transpose traffic, a message from node $< i, j >$ is directed to node $< j, i >$ if $i \neq j$. If $i = j$ node $< i, i >$ sends messages to node $< K - i, K - i >$, where K is the network radix.

5.1.2 Parameters of interest

We have studied the average communication latency, the average throughput of the network, and the network load distribution in the network. The communication latency is defined as the average time from the message generation to the time when the tail reaches the destination. The throughput is the average number of messages routed per unit time. The network traffic load is measured by finding the average flit buffer utilization at each node.

All the above parameters are studied against the offered network traffic. The network traffic is defined as the ratio of the average traffic generated by a node to the average bandwidth available per node.

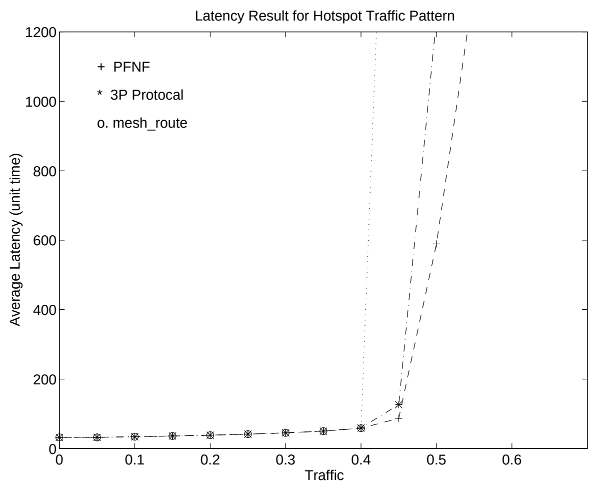

Figure 11: Latency results for the hotspot traffic pattern

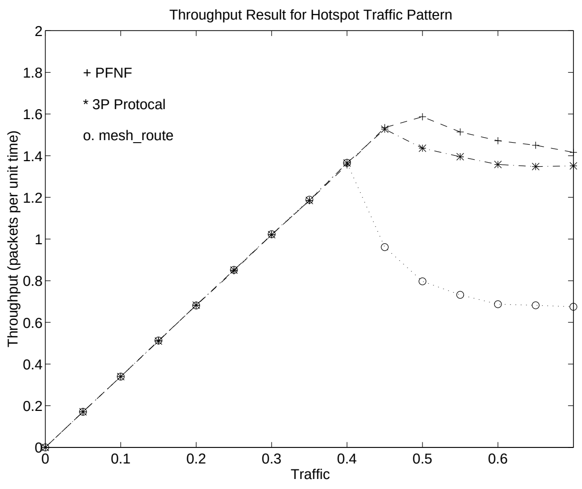

Figure 12: Throughput results for the hotspot traffic pattern

5.2 Results and Discussions

Figures 8 and 9 plots the average latency and average throughput of the network against the offered network traffic for the uniform traffic pattern. In the low traffic region, all the three algorithms result in the same average latency. However, as the traffic is increased, mesh_route saturates first and its latency increases rapidly. The PFNF algorithm performs better

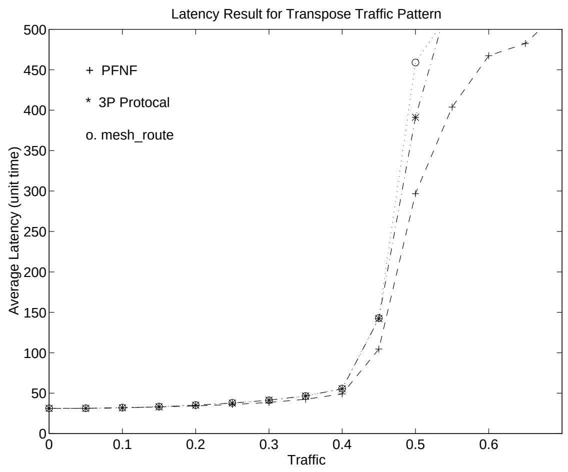

Figure 13: Latency results for the transpose traffic pattern

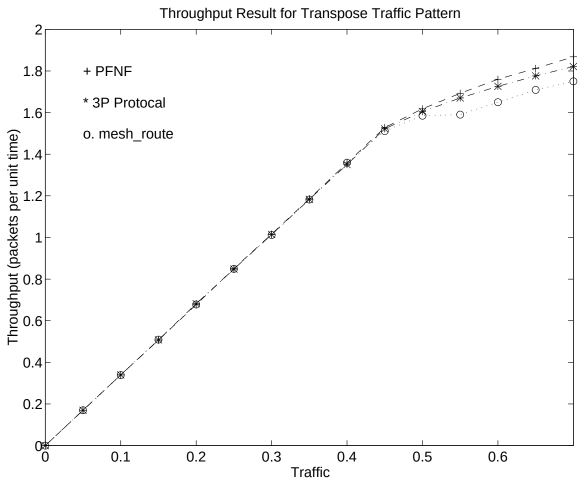

Figure 14: Throughput results for the transpose traffic pattern

than both the mesh_route and 3P protocol schemes. The same pattern is also observed in the throughput results. All the algorithms give the same throughput for lower traffic rates (< 0.4), however at higher traffic, the throughput of the mesh_route algorithm drops very abruptly. 3P protocol and PFNF do not drop abruptly but instead saturate at their maximum values.

As shown in the efficiency analysis, the mesh_route algorithm is more efficient than the 3P protocol, however it creates uneven traffic distribution in the network. The fact that 3P protocol demonstrates higher performance compared to mesh_route confirms our claim that the system performance depends more upon how evenly the traffic is distributed than the efficiency or adaptivity of the algorithm. The PFNF algorithm has the highest efficiency and it also distributes the network load very symmetrically and hence produces better results.

To demonstrate that the PFNF algorithm indeed distributes the network load symmetrically we have plotted the traffic distribution for the PFNF algorithm. Figure 10 shows this graph. As mentioned before, these results are obtained by measuring average flit buffer utilization at each node. The results are in accordance with what was expected from the *region of adaptivity* analysis and clearly illustrate that the traffic distribution is more balanced and symmetric in the PFNF algorithm than in the mesh_route (refer to Figure 3(b)).

Figure 11 and 12 shows the latency and throughput results under the hotspot traffic pattern. As with the uniform traffic pattern, the mesh_route performs very poorly and saturates early. Here also, the PFNF algorithm outperforms 3P protocol scheme in both the latency and throughput results. The same trend is observed in the results under the transpose traffic pattern which are shown in Figures 13 and 14.

6 Conclusions

In this paper we have analyzed adaptive routing in 2-dimensional meshes in light of two new concepts – *region of adaptivity* and balanced traffic distribution. Previously proposed fully adaptive algorithms have aimed at improving the adaptivity and efficiency of the routing schemes. We have demonstrated through simulations that symmetric and balanced network traffic distribution has more impact on the system performance than higher adaptiveness and better efficiency. Using these motivating factors we have proposed a new adaptive scheme, Positive-First–Negative-First(PFNF) algorithm for two-dimensional meshes. The algorithm uses two different routing layers Positive-First and Negative-First to balance the traffic distribution in the network. It is also more efficient than the previously proposed algorithms in terms of the number of alternate paths it allows. The simulation results show that our scheme performs better than all the previous schemes in terms of the average network latency and throughput.

Since the proposed algorithm is based on balancing the network traffic by using two separate routing algorithms in the two virtual networks, it can be easily extended to higher dimensional meshes by proper choice of algorithms for each virtual network. The concept of *region of adaptivity* and how it relates to the network traffic distribution can be used in choosing these balanced algorithms.

References

[1] R. E. Kessler and J. L. Schwarzmeier, "CRAY T3D: A New Dimension for Cray Research," *Compcon*, pp. 176-182, Spring 1993.

[2] R. Alverson *et al.*, "The tera Computer System.",*Proc. of the 1990 Intl. Conference on Supercomputing*, pp.1-6, June 1990.

[3] Intel Corporation, *A Touchstone DELTA System Description*, 1990.

[4] D. Lenoski, J. Laudon, K. Gharachorloo, W. Weber, A. Gupta, J. Hennesy, M. Horowitz, and M. Lam, "The Stanford DASH Multiprocessor", *IEEE Computer*, pp. 63-79, March 1992.

[5] A. Agrawal, B. Lim, D. Kranz,, and J. Kubiatowicz, "APRIL: A Processor Architecture for Multiprocessing", *Proc. of the 17th Annual Intl. Symposium on Computer Architecture*, vol. 18, no. 2, pp. 104-114, June 1990.

[6] NCUBE Company, *NCUBE 6400 Processor Manual*, 1990.

[7] W. J. Dally, "Performance Analysis of k-ary n-cube Interconnection Networks", *IEEE Trans. on Computers*, vol. 39, no. 6, pp. 775-785, June 1990.

[8] C. J. Glass and L. M. Ni, "Turn model for adaptive routing", *Proc. 19th Annual Intl. Symposium on Computer Architecture*, pp. 278-287, May 1992.

[9] C. J. Glass and L. M. Ni, "Adaptive routing in mesh-connected networks", *Proceedings of the 1992 Intl Conference on Distributed Computing Systems*, pp. 12-19, 1992.

[10] Y. M. Boura and C. R. Das, "A class of partially adaptive routing algorithms for n-dimensional meshes", *Proc. of the 23rd Intl. Conference on Parallel Processing*, vol. 3, pp. 175-182, August, 1993.

[11] A. A. Chien and J. H. Kim, "Planar Adaptive routing: Low-cost adaptive networks for multiprocessors", *Proc. of the Intl. symposium on Computer Architecture*, pp. 268-277, may 1992.

[12] W. J. Dally., "Virtual channel flow control", *IEEE Trans. on Parallel and Distributed Systems*, vol 3, pp. 194-205, March 1992.

[13] D. H. Linder and J. C. Harden, "An Adaptive and fault tolerant wormhole routing strategy for k-ary n cubes", *IEEE Trans. on Communications*, vol. 40, pp. 2-12, Jan. 1991.

[14] J. Duato, "A New Theory of Deadlock-Free Adaptive Routing in Wormhole Network", *IEEE Trans. on Parallel and Distributed systems*, vol. 4, No.12, Dec. 1993.

[15] C. Su and K. G. Shin, "Adaptive deadlock-free routing in multicomputers using only one extra channel", *Proc. of the 22nd Intl Conference on Parallel Processing*, vol. 3, pp. 175-182, Aug. 1993.

[16] Y. M. Boura and C. R. Das, "Efficient fully adaptive wormhole routing in n-dimensional Meshes", *Proc. of the 14th Intl. Conference on Distributed Computing Systems*, 1994.

[17] R. V. Boppana and S. Chalasani, "A Comparison of Adaptive Wormhole Routing Algorithms", *Proc. of the Intl. Symposium on Computer Architecture*, pp. 351-360, May 1993.