BARP-A Dynamic Routing Protocol for Balanced Distribution of Traffic in NoCs

Pejman Lotfi-Kamran[†], Masoud Daneshtalab[†*], Caro Lucas[†], and Zainalabedin Navabi[†]
[†]School of Electrical and Computer Engineering, The University of Tehran
[*]Islamic Azad University, Parand Branch
plotfi@computer.org, m.daneshtalab@ece.ut.ac.ir, lucas@ipm.ir, and navabi@ece.neu.edu

Abstract

A novel routing algorithm, named Balanced Adaptive Routing Protocol (BARP), is proposed for NoCs to provide adaptive routing and ensure deadlock-free and livelock-free routing at the same time. By evenly distributing input packets of a router among all its shortest path output ports, a novel adaptive routing protocol for avoiding congestion condition emerges. It is observed that BARP can achieve better performance compared to static XY routing, odd-even routing and dynamic XY routing.

1. Introduction

Network on chips (NoCs) [1][2] are proposed to be used in complex SoCs for communicating among cores. Routing algorithms are used in NoCs in order to determine the path that a packet travels from the source to the destination. Routing algorithms are classified as deterministic and adaptive algorithms. Implementations of deterministic routing algorithms are simple but if the packet injection rate of cores in a NoC is greater than a threshold value, the packet service time increases exponentially [3][4]. In adaptive routing algorithms, the path that a packet travels from the source to the destination is determined by the network congestion condition. An adaptive routing algorithm decreases the probability of passing a packet from a congested link. Therefore, an important feature of a routing algorithm is the ability to adapt to congestion condition. Another important feature of a routing algorithm is the lack of deadlock.

Recently, many adaptive deadlock free routing algorithms are proposed for networks with mesh structures. In [5][6][7], the concept of virtual channel is introduced. These references use virtual channels to assist designing adaptive routing algorithms for different network topologies. For a network with mesh structure, a number of routing algorithms are proposed that route packets without using virtual channels [8][9][10][17]. Static routing algorithm XY for two-dimensional meshes is introduced in [8]. In this routing algorithm, each packet first travels along X and then Y direction to reach the destination. In this algorithm deadlock never occurs but no adaptivity exists in this algorithm. In [9], an adaptive routing algorithm named turn-model is introduced. In [10], an adaptive routing algorithm named odd-even turn is proposed based on the turn-model. This method restricts the position that turns are allowed in a network with the mesh topology to avoid deadlock. Another algorithm called DyAD is introduced in [11]. This algorithm is a combination of a static routing algorithm named oe-fix, and an adaptive routing algorithm based on odd-even. The routing algorithms use one of these mechanisms based on the congestion condition of a network. Finally, an adaptive routing algorithm that is called DyXY is proposed in [12]. This algorithm is based on the static XY algorithm but in each router, a packet can be sent to either X or Y direction.

Almost in all adaptive routing algorithms, no mechanism is incorporated to avoid congestion. Congestions happen and then the routing algorithm starts its effort to cop with the situation by changing the path that packets travel. Because of the congestion, service time of a number of packets increases dramatically. In this paper, an adaptive routing algorithm for distributing traffic in a NoC is presented that avoids congestion.

The rest of this paper is organized as follows. In Section 2, our adaptive routing algorithm (BARP) is presented. Section 3 deals with hardware implementation of BARP. Experimental results are discussed in Section 4 and conclusion comes in the last section.

2. The BARP Routing Protocol

In the networks with mesh topology, there may be many paths from the source to the destination, but only some of these paths have a minimal length. The proposed routing algorithm avoids congestion by distributing traffic among minimal paths. In this way, livelock-free feature of our algorithm is guaranteed. BARP has two operating mechanisms to avoid congestion. In the first mechanism, BARP tries to evenly distribute traffic to all network resources just by local information. In the case of unbalanced distribution of traffic because of the lack of

978-3-9810801-3-1/DATE08 © 2008 EDAA

global information, the second mechanism of BARP tries to cop the situation by exchanging as little information as needed.

In the first mechanism, each router uniformly distributes its incoming traffic to all of its ports that are located in the shortest path to the destination. If for an incoming packet, there is just one outgoing port, the packet will be routed to the destination through that port. On the other hand, if a packet can be routed to the destination through two outgoing ports, half the time it is routed from the first port and in the rest, it is routed from the second port. Consider network structure of Figure 1. Suppose Core (0, 3) sent a packet to Core (3, 3) and this packet is already in Router (1, 3). Because of the fact that packets are routed to the destination only through minimal path, just one option is available. Router (1, 3) sends the packet to the Router (2, 3). Now suppose that Core (0, 0) has sent a packet to Core (3, 1) and this packet is already in Router (1, 0). Router (1, 0) has two options [i.e., Router (1, 1) or Router (2, 0)] to route the packet to the desired destination. Therefore, half of these packets are sent to Router (1, 1) and the rest of them are sent to Router (2, 0).

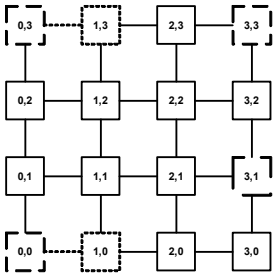


Figure 1. sending packet from Core (0, 3) to Core (3, 3) and from Core (0, 0) to Core (3, 1)

Note that for the deadlock-freeness of BARP routing protocol, two virtual channels are used in each Y-dimension [13]. The network is partitioned into two sub-networks called the +X sub-network and –X sub-network each having a pair of channels in the Y dimension. If the destination node is to the right of the source, the packet will be routed through the +X sub-network. If the destination node is to the left of the source, the packet will be routed through the -X sub-network. Otherwise that packet can be routed using either sub-network [13].

By uniform distribution of incoming packets among different paths, the probability of occurrences of congestions is decreased, but there is still a chance for congestion to occur. The aim of the second mechanism of the BARP routing protocol is to detect the near congestion routers and avoid the congestion by changing the routing mechanism of influencing routers. The main idea is that each router monitors its input buffers and whenever the number of packets in one of these buffers is increased to a specific threshold value, the router becomes a near congestion router. Each near congestion router sends some routing packets to pre-determined routers (based on the position of the near congestion link of the router) to request changing in their routing mechanism in order to decrease the number of packets that are delivered to near congestion router.

The other fact is how to change the routing mechanism in order to shape network traffic to avoid congestion in the near congestion routers. To suggest a solution, the network structure of Figure 2 is used. Suppose the link from Core (2, 2) to Core (3, 2) is getting to become congested. If routers of Group G4 have packets for the members of Group G8, they can choose either east port or south port. If the south port is chosen, the packet certainly passes the near congestion link but it is not the case if the east port is chosen. Therefore, it is desirable to increase the probability of selecting east port with respect to south port in routers of Group G4 as a result of running second mechanism of BARP routing protocol. The same argument can be applied to members of Group G2 when they are sending packets to members of Group G8. In this case, it is desirable to increase the probability of selecting east port with respect to north port as a result of running the second mechanism of BARP routing protocol.

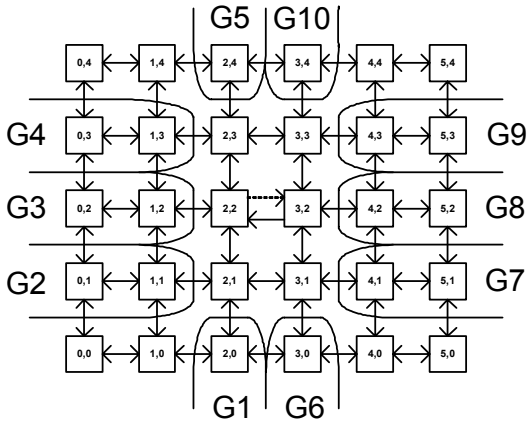


Figure 2. a sample NoC with mesh topology

Now suppose a member of Group G1 has a packet to be sent to a member of Group G10. This packet can be sent to either east port or north port. If the east port is chosen, the packet certainly will *not* pass the near congestion link, but it is not the case if north port is chosen. Therefore, it is better to increase the probability of selecting east port with respect to north port. The same argument applies for the members of Group G5 when they are sending packets to members of Group G6. In this case, it is better to increase the probability of selecting east port with respect to south port. Finally, if members of Group G3 have packets for sending to Groups G7/G9, either east port or south/north port can be chosen. If the east port is chosen, there is a probability that the packets are routed to the near

congestion link, but there is no chance that a packet is routed to this link if the south/north link is chosen. Therefore, it is better to increase the probability of selecting south/north port with respect to east port in members of Group G3.

For packets that are routed to the members of Group G8, if in the routers of Group G2/G4, north/south port is chosen; the packet passes the near congestion link certainly. Therefore, changing the routing mechanism in these group members is called critical. The change that is made to the routing mechanism in routers of Group G1, G3, and G5 makes us assured that these packets do not pass near congestion link but if this change does not happen, there is still a chance that the packet does not pass this link. Therefore, the change that is made in the routing mechanism of these group members is called non-critical. The summary of our discussions has come in Table 1.

If instead of west port of Router (3, 2), one of the north, east, or south port of this router is getting to become congested, a number of influencing group is obtained similarly in which the probability of selecting some port with respect to others should be changed. It is clear that the routing mechanism of other routers that do not belong to these groups should not change.

Table 1. Changing port selection probability for the case of getting congested the link from (2, 2) to (3, 2) in Figure 2

Group	West Port	North Port	East Port	South Port
G1		DEC	INC	
G2		DEC	INC	
G3		INC	DEC	INC
G4			INC	DEC
G5			INC	DEC

As stated, each router monitors its input buffers to detect when these buffers are going to become full. In this case, this router generates a number of routing packets and sends them in appropriate directions to adjust routing mechanism according to Table 1. As an example, if east link of the Router (2, 2) in Figure 2 is getting to become congested, the router will eventually generate 5 routing packets corresponding to the five groups in Table 1. Routing packets corresponding to Groups G1, and G2 are sent to south port, routing packet corresponding to Group G3 is sent to west port, and routing packets corresponding to Groups G4, and G5 are sent to north port. Each routing packet changes the routing mechanism of reaching routers according to Table 1.

The congestion avoidance algorithm in the BARP routing protocol is as following. Two threshold values are defined based on the number of packets in the input buffer of each router. If the input buffer size of a router reaches the first threshold value, two routing packets are generated to change the routing mechanism of critical group (i.e., Groups G2, and G4 of Figure 2). If this cannot avoid congestion to occur, the input buffer size reaches the second threshold value. In this case, five routing packets are generated to change the routing mechanism in all five groups (critical and non-critical groups simultaneously).

3. Hardware Implementation

The implementation of this approach takes advantages of a priority table. For each neighboring ports (i.e., two ports of a router that are adjacent [e.g., north-east, north-west, south-east, and south-west]), there is a location in this priority table. The priority table indicates, for the packets that can be sent to more than one port, how the incoming packets are distributed between these ports. Each location in the priority table has 2-bit width. These 2-bits indicate that how many packets, out of 5 packets, are to be sent to each port. Table 2 shows the number of packets that are sent to each port corresponding to each value of the priority table.

Table 2. No. of transmitted packets to different ports corresponding to different values of the priority table

		Number of Transmitted packet out of 5			
		West	North	East	South
North West	00	4	1		
	01	3	2		
	10	2	3		
	11	1	4		
North East	00		1	4	
	01		2	3	
	10		3	2	
	11		4	1	
South West	00	4			1
	01	3			2
	10	2			3
	11	1			4
South East	00			4	1
	01			3	2
	10			2	3
	11			1	4

In addition, each port has two 3-bit saturating (saturated to 5) counters. When a packet is sent to a port; both counters of that port are incremented. Each counter of a port has a correspondence with one of the adjacent ports of that port. A counter is identified by two letters: the first letter indicates the owning port and the second letter indicates a neighbor of the owning port that the counter is corresponding to it (e.g., NE, NW, EN, ES, SE, SW, WS, and WN). These counters and the priority table together indicate the destination port of a packet. If a packet can be transmitted to more than one direction, only counters with letters corresponding to those directions are used to route

the packet. When the counter of north or south port is less than the number of packets that Table 2 indicates should be sent to north/south port, the incoming packet is sent to north or south port. Otherwise the packet is sent to west or east port. When sum of the counters of two adjacent ports (i.e., counters that their names have the same letters with different order) gets equal to or becomes greater than 5, these counters are cleared.

Initially, all eight counters are zero. The content of priority table is initially set {01, 10, 10, and 01} for {North-East, South-East, North-West, and South-West} to evenly distribute incoming traffic across the network. The contents of the priority table remain constant until a routing packet is delivered. In this case, the contents of the priority table are changed according to Table1.

4. Results and Discussion

For appraising the efficiency of the proposed routing algorithm, three other routing algorithms were also implemented. These algorithms include XY, Odd-Even, and DyXY. We have developed a flit level NoC simulator written in C++ capable of calculating the average delay and the power consumption for the message transmission. This simulator can be used for wormhole switching in two dimensional mesh configurations for the NoC. The simulator inputs include the array size, the router operation frequency, the router algorithm, the link width length, and the traffic type. The simulator can generate different traffic profiles. To calculate the power consumption, we have used Orion library functions [14]. Since the simulator is event driven, the simulation speed is high. For all switches, the data width is set to 16-bits, and each input channel has a buffer (FIFO) size of 12 flits with the congestion thresholds are set at 25% and 75% of the total buffer capacity. The message size was assumed to be 10 flits. The time needed to generate the messages is not considered, because we assumed the messages are generated in the PEs.

4.1. Transpose Traffic Profile

The first sets of simulations were performed for a transpose traffic profile. In this traffic profile, for a $n \times n$ mesh network, a PE at position (i, j) $(i, j \in [0, n))$ only sends a data packet to another PE at position $(n-1-i, n-j-1)$. This traffic pattern is similar to the concept of transposing a matrix [15].

In these simulations, the processing elements (PE) generates ten flit data messages and inject them into the network using the time intervals which are obtained based on the exponential distribution. Two array sizes have been considered 8×8 and 14×14. This traffic profile leads to a non-uniform traffic distribution with heavy traffics for the central nodes of the mesh. Therefore, hotspots close to the center of the network may be created. As shown in Figure 3 if the data packet injection rate is very low, the hotspots are not created. But as the injection rate increases, the proposed algorithm leads to smaller average delays. Note that in our algorithm packets in the buffers waiting to be routed undergo less latency.

4.2. Random Traffic Profile

In these set of simulation, two array sizes have been considered 8×8 and 14×14. In this traffic profile, each PE sends several messages to a set of destination. A uniform distribution is used to construct the destination set of each message [16]. The number of destinations has been set to 10. In Figure 4, the average communication delay as a function of the average message injection rate has been plotted. As observed from the results, the proposed routing algorithm outperforms other algorithms.

4.3. Power Dissipation

The power dissipation of XY, Odd-Even, DyXY and the proposed routing algorithms were calculated and compared under the Transpose and Random traffic models. The results for the average and maximum power under these traffic models in 14x14 2D-mesh are shown in Figure 5 and Figure 6, respectively. As the results presented in Table 3 reveal, the average power dissipation of the network with the proposed algorithm is 22% more than that of the XY algorithm, 12% more than that of the Odd-Even algorithm, and 6.5% less than that of the DyXY algorithm under the transpose traffic profile.

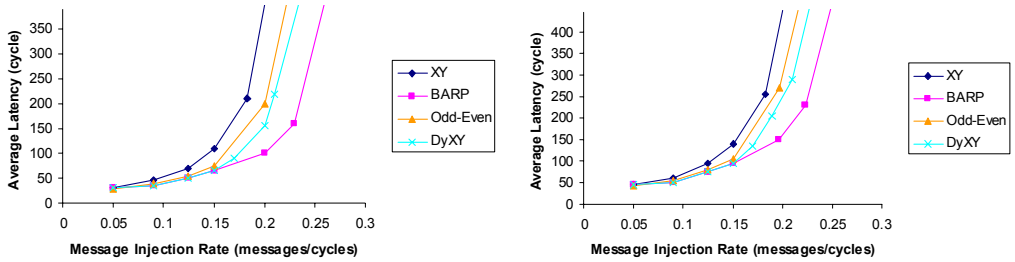
Table 4 indicates that the peak power of the proposed algorithm is 35%, 24%, and 10% less than that of the XY, Odd-Even, and DyXY algorithms, respectively under the transpose traffic model. We can notice that the average power and the peak power compared to other algorithms, is considerably lowered in our proposed algorithm. This is achieved by smoothly distributing the maximum power consumption over the network using the adaptive routing scheme which reduces the number of the hotspots and, hence, lowering the peak power.

Table 3. Comparative Average power dissipation of the proposed algorithm with other algorithms in 16×16 2D-mesh.

Average Power Dissipation	XY	Odd-Even	DyXY
Transpose	22%	12%	-6.5%
Random	24%	11%	-5%

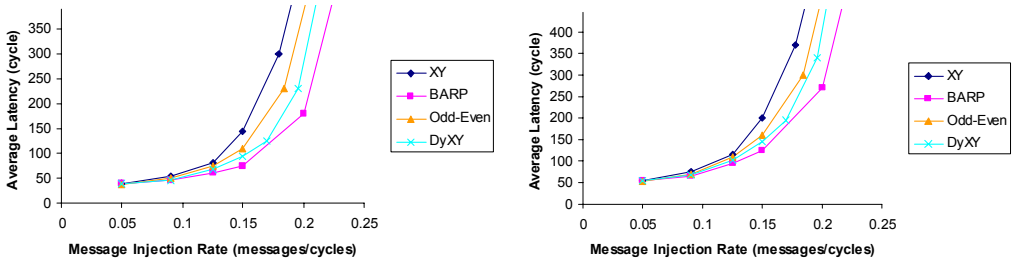
Table 4. Comparative Maximum power dissipation of the proposed algorithm with other algorithms in 16×16 2D-mesh

Peak Power Dissipation	XY	Odd-Even	DyXY
Transpose	-35%	-24%	-10%
Random	-31%	-20%	-6%



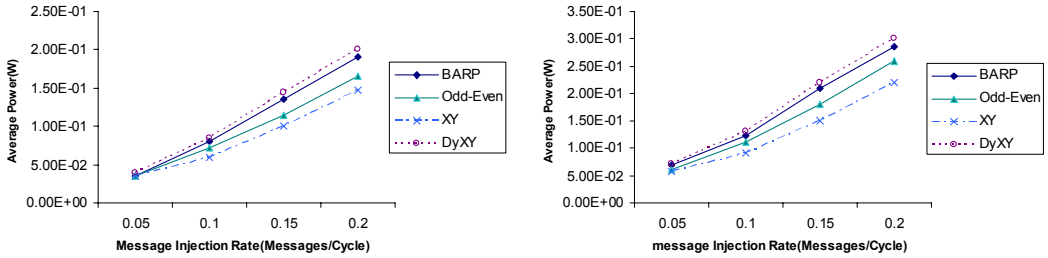
(a) (b)

Figure 3. Performance under different loads in (a) 8×8 2D-mesh and (b) 14×14 2D-mesh under transpose traffic model.



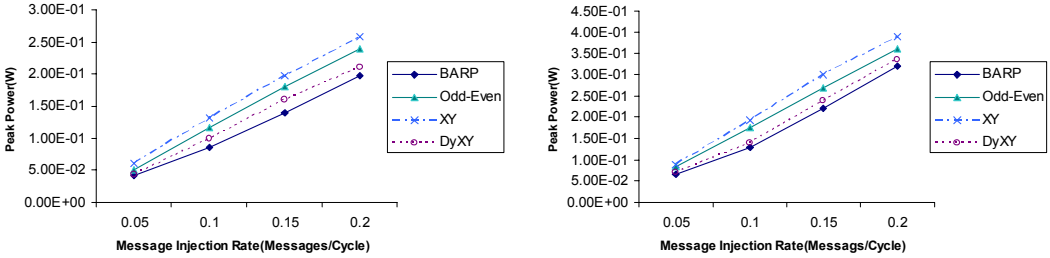
(a) (b)

Figure 4. Performance under different loads in (a) 8×8 2D-mesh and (b) 14×14 2D-mesh under Random traffic model with 10 destinations.



(a) (b)

Figure 5. Average power dissipation of the BARP, the XY, the Odd-Even and the DyXY algorithms in 14×14 2D-mesh under (a) Transpose and (b) Random traffic models.



(a) (b)

Figure 6. Maximum power dissipation of the BARP, the XY, the Odd-Even and the DyXY algorithms in 14×14 2D-mesh under (a) Transpose and (b) Random traffic models**.**

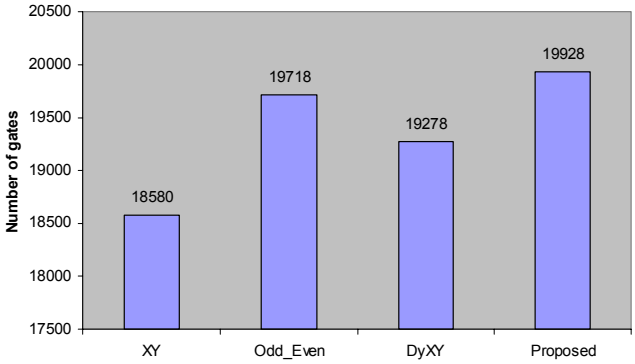


Figure 7. Area cost of the switches

4.4. Hardware Overhead

To evaluate the area overhead of the proposed algorithm, we designed the switches with VHDL and synthesized with Leonardo-Spectrum ASIC using the SCL 0.25µm standard cell library. For all switches, the data width was set to 16 bits (flit size), and each input channel had a buffer size of 12 flits. In order to achieve better performance/power efficiency, the FIFOs were implemented using registers. Figure 7 shows the area cost of the switches. Comparing the area cost of the proposed switch with XY, Odd-Even, and DyXY indicates 6.5%, 1%, and 3% additional overhead, respectively.

5. Conclusions

In this paper a balanced adaptive routing algorithm, BARP, is proposed. In contrast to other adaptive routing protocols, BARP tries to avoid congestion. BARP evenly distributes the incoming packets to different path from the source to the destination. If a port of a router becomes near to the congestion condition, a few number of routing packet is generated to adjust routing mechanism of the network to avoid congestion. Experimental results show effectiveness of BARP compared to static XY routing, odd-even routing, and adaptive XY routing.

References

[1] L. Benini, and G. D. Micheli, "Networks on chips: a new SOC paradigm," IEEE computer, 35:70–78, Jan 2002.

[2] W. J. Dally, and B. Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," in Proc. Of Design Automation Conference, pp. 684–689, 2001.

[3] D. Bertsekas and R. Gallager, *Data Networks*, Prentice Hall, 1992.

[4] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*, Morgan Kaufmann, 2004.

[5] W. J. Dally, "Virtual-channel flow control," IEEE Trans. Parallel and Distributed Systems, 3:194–205, Mar 1992.

[6] Y. M. Boura, and C. R. Das, "Efficient fully adaptive wormhole routing in n-dimensional meshes," in Proc. of Int'l Conf. Distributed Computing Systems, pp. 589–596, 1994.

[7] C. J. Glass, and L. M. Ni, "Maximally fully adaptive routing in 2d meshes," in Proc. of Int'l Conf. Parallel Processing, pages 101–104, 1992.

[8] I. Corporation, "A touchstone delta system description," in Intel Advanced Information, 1991.

[9] C. J. Glass, and L. M. Ni, "The turn model for adaptive routing," Journal of the ACM, 41:874–902, Sept 1994.

[10] G. M. Chiu, "The odd-even turn model for adaptive routing," IEEE Trans. on Parallel and Distributed Systems, 11:729 – 738, July 2000.

[11] J. C. Hu, and R. Marculescu, "DyAD - smart routing for networks-on-chip," in Proc. of Design Automation Conference, pp. 260–263, 2004.

[12] M. Li, Q.-A. Zeng, and W.-B. Jone, "DyXY - a proximity congestion-aware deadlock-free dynamic routing method for network on chip," in Proc. of Design Automation Conference, pp. 849–852, 2006.

[13] L. M. Ni, and P. K. McKinley, "A survey of wormhole routing techniques in direct networks," in IEEE computer, pp. 62–76, 1993.

[14] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A power-performance simulator for interconnection network," in Proc. of international symposium of micro-architectures, pp. 294–305, Nov 2002.

[15] C. J. Glass, and L. M. Ni, "The Turn Model for Adaptive Routing," in Proc of Symp. on Computer Architecture, pp. 278-287, May 1992.

[16] X. Lin, and L. M. Ni, "Multicast Communication in Multi-computer Networks," in IEEE Transactions on Parallel and Distributed Systems, pp. 1105-1117, 1993.

[17] M. Daneshtalab, A. Pedram, M. H. Neishaburi, M. Riazati, A. Afzali-Kusha, and S. Mohammadi, "Distributing Congestions in NoCs through a Dynamic Routing Algorithm based on Input and Output Selections", in Proc. of 20th VLSID, IEEE Press, pp. 546-550, Jan 2007, India.