Path-Diversity-Aware Adaptive Routing in Network-on-Chip Systems

Yu-Hsin Kuo[1], Po-An Tsai[1], Hao-Ping Ho[1], En-Jui Chang[2], Hsien-Kai Hsin[2], and An-Yeu (Andy) Wu[1, 2]
[1]Department of Electrical Engineering, National Taiwan University, Taipei 10617, Taiwan
[2]Graduate Institute of Electronics Engineering, National Taiwan University, Taipei 10617, Taiwan
Email: b97901030@ntu.edu.tw

***Abstract*—The partially adaptive routing plays an important role in the performance of Network-on-Chip (NoC). It uses information of the network to select a better path to deliver a packet. However, it may have imbalanced path diversity in different directions, which makes their tolerances of traffic load differ a lot from each other. This characteristic would cause problems in traffic balancing but give us extra information of the network. To achieve load balancing, in this paper, we present an adaptive routing scenario with Path-Diversity-Aware (PDA) and Augmented-PDA (A-PDA) selections, which use the information of path diversity. Moreover, we derive a formula to quantify the characteristic of path diversity. Experiments with different scenarios were conducted. The simulation results show that our proposed selections have an advantage over other selection functions in saturation throughput, with up to 36.84%, and have better scalability in large scale NoC. In addition, a low-cost router architecture is proposed to implement PDA and A-PDA and the synthesized results are also shown in this paper.**

Keywords-Network-on-Chip; adaptive routing; path diversity; scalability

I. INTRODUCTION

With high demands of throughput and bandwidth but within feasible cost, System-on-Chip (SoC) designers find on-chip interconnections a limiting factor in terms of performance and energy consumption [1]. It is believed that the Network-on-Chip (NoC) architecture, which regards on-chip cores as nodes in a network and transmitted information as packets, is a solution to the problems above and provides a reusable model [2]. However, the NoC architecture has not yet become a mainstream because of some challenges, such as power issues, EDA tools issues, and performance issues [3]. Among them, the performance issues have drawn the most attention in recent years.

The performance issues in NoC are greatly influenced by the routing algorithms. Routing algorithms are used to determine how the packets are transmitted from source to destination in NoC. There are two important factors in a routing algorithm [4]. The first one is to be deadlock- and livelock-free, which can guarantee the stability of the system. This factor can prevent packets from staying on the network in an infinite period. Another one is to have high adaptiveness, meaning higher degree of freedom when the router is routing packets. This factor, which aims at achieving traffic balancing and higher performance, decides how much path diversity packets can have. The path diversity here is defined as all the possible paths a packet can be routed given a source-destination pair. Routing algorithms with high adaptiveness are usually robust when faced with hot-spot nodes, faulty nodes and heavy workload. However,

This work is supported in part by the National Science Council, Taiwan R.O.C., under Grant NSC 100-2221-E-002-091-MY3.

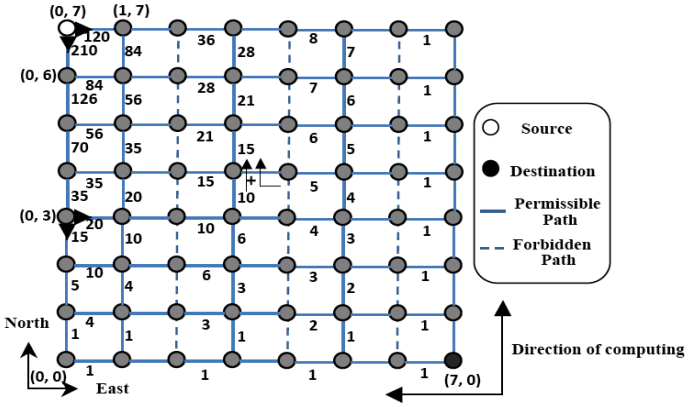


Figure 1. The schematic sketch of path diversity of a source-destination pair (0, 7) and (7, 0). Due to the restriction of the Odd-Even turn model, some paths are forbidden.

there are trade-offs between these two factors. That is, to achieve deadlock- and livelock-free, routing algorithms must at the cost of adaptiveness and vice versa.

Since the trade-offs exist between these factors, routing algorithms with different choices were proposed. We can classify them into two main categories [16] [17] [18]:

- *Deterministic Routing*: It has the lowest adaptiveness since it predetermines a single path for a packet. Nevertheless, the predetermined path also guarantees deadlock- and livelock-free property.
- *Adaptive Routing*: It consists of a turn model and a selection function. The former selects a set of paths and the latter selects a single path among this set according to the condition of the network. Furthermore, we can divide the adaptive routing into two subgroups: *minimal fully adaptive* and *partially adaptive* routing algorithms. The former routes packets toward all shortest paths, which yields the highest adaptiveness yet at the risk of deadlock. The later one, by restricting packets to be delivered toward a subset of shortest paths, routes packets with lower adaptiveness. However, it could achieve deadlock- and livelock-free as deterministic routing algorithms do.

Although the deadlock issue can be solved by adding virtual channels in NoC [5], it needs extra interconnections and implementation cost [7]. Therefore, using adaptive routing algorithms well can solve this issue within reasonable cost yet has a little loss of adaptiveness. Among these routing algorithms, partially adaptive routing algorithms outperform the others for they have both acceptable adaptiveness and desirable deadlock- and livelock-free property. Nonetheless, compared with fully adaptive routing algorithms, the adaptiveness of partially adaptive routing algorithms still has some way to go, which leaves it possible to be improved. In

this paper, we focus on the adaptiveness in partially adaptive routing algorithms and its improvement in overall performance.

Since some possible paths are forbidden, partially adaptive routing algorithms would yield imbalanced path diversity in different directions. Fig. 1 shows the path diversity of a packet with source-destination pair (0, 7) and (7, 0) when we apply a well-known partially adaptive routing algorithm, Odd-Even [4] routing, in an 8×8 mesh topology. From Fig. 1, due to the restriction of the routing algorithm, there is a pair of uneven path diversity, 210 and 120, in the two output channels toward node (1, 7) and (0, 6), respectively. This could be an important defect for performance because imbalanced path diversity would cause different directions having different tolerance of traffic load, which leads to an imbalanced traffic load in NoC. Thus, we need to take this imbalanced path diversity into consideration.

The number of path diversity (PD) is influenced by two factors, routing algorithms and the topology size. There is a non-linear relationship between these two factors and PD. We simply define PD as a function of them, that is,

$$f(R,T) = PD, the\ number\ of\ possible\ paths \quad (1)$$

where R is a factor related to routing algorithms and T is a factor related to the topology size. The positive correlation between PD and network topology size can be shown easily. Also, routing algorithms with higher adaptiveness will lead to higher PD. These two factors affect the performance of the partially adaptive routing functions, which always produce imbalanced workload. To overcome this problem, we are likely to utilize the information of PD to achieve traffic balancing in our works.

The main contribution of this paper is to analyze the effects of the different PD in partially adaptive routing algorithms and make a new criterion by quantifying this characteristic. Then, we adopt this criterion for selection functions to improve performance in NoC. Besides, we propose router architectures based on the idea delivered in the paper and evaluate their area cost.

The remainder of this paper is organized as follows: In section II, turn model, a fundamental property of adaptive routing, is illustrated and other selection functions of adaptive routing in NoC are summarized. In section III, we analyze the property of imbalanced PD in partially adaptive routing and explain how this property can be calculated in a clear way. In section IV, a set of algorithms using the information of PD are proposed. We evaluate the improvement by several experiments in section V. The router architectures are proposed with area summary in section VI. Finally, we conclude this paper in section VII.

II. Related Works and Background

Adaptive routing algorithms consist of two parts. The first one is the turn models, which return several possible outputs. The second one is the selection functions that select the only output among these candidates. Therefore, we will briefly present the two components in the following parts.

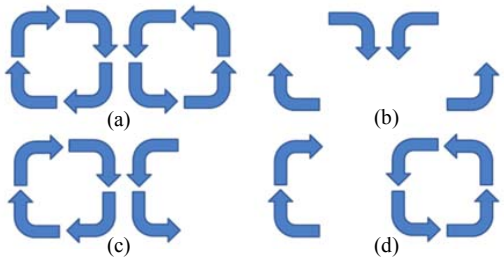


Figure 2. The turn model of (a) fully adaptive, (b) X-Y routing, (c) Odd-Even in even column and (d) Odd-Even in odd colum.

A. Turn Model in Adaptive Routing Algorithms

Turn model is based on the directions in which packets can turn in the network [6]. The most common topology of NoC is a two-dimensional (2D) mesh, which contains K_0 columns in X dimension and K_1 rows in Y dimension. To simplify the description, we define the four sides of a NoC to be *North, East, West*, and *South*, as showed in Fig. 1. In a 2D mesh, turn model contains 8 kinds of 90-degree turns. They are north to east (*N-E*), *N-W*, *E-N*, *E-S*, *W-N*, *W-S*, *S-E* and *S-W*. These eight turns represent how a packet can turn in a 2D mesh. As shown in Fig. 2(a), they perform a clockwise cycle and a counter-clockwise cycle. These cycles represent deadlock situations that four packets in a cycle are waiting for each other. Thus, if one of the turns in a cycle is prohibited, the deadlock will be prevented.

Besides, turn model can be used to describe the properties of a routing algorithm. For example, Fig. 2(a) illustrates the turn model of minimal fully adaptive routing, which contains all possible turn in a mesh topology. Keeping all possible turns in a 2D mesh, it has the most number of routing paths or the highest adaptiveness, yet possibilities of deadlock. Fig. 2(b) shows the turn model of X-Y routing, one kind of deterministic routing. Due to the elimination of some turns, it is impossible for X-Y routing to form a cycle, which guarantees its deadlock-free property. Fig. 2(c) and Fig. 2(d) present the turn model of Odd-Even routing. They show that only a few turns have been forbidden and so a relatively higher adaptiveness is retained in Odd-Even routing. Also, it has deadlock-free property, which is proved in [4].

B. Selection Functions in Adaptive Routing Algorithms

To improve both fully and partially adaptive routing algorithms in performance, a unit called *selection function* is needed. This function helps to select the better output channel among all candidates returned by adaptive routing functions. To judge which path is better, the most straightforward way is to use the local information, such as the remaining buffers or free-slots in the next router, called *output channel buffer-level* (OBL). In [7], the information of buffer size in the *neighbor-on-path* (NoP) is considered to be a score of an output channel. These scores can be used to decide an output channel, making packets more able to avoid congestion in local area. Other congestion-avoided selections based on local information like [8], *Regional Congestion Awareness* (RCA) and [9], *Dynamic Bandwidth Estimation* (DBE) were proposed. Besides local information, the

selection strategy can be based on other information. For example, in [10], historical information and the buffer size are combined to form a more reliable metric to determine which output channel is better.

Nevertheless, among these works, the difference in PD is seldom discussed. To the best of our knowledge, only few researches have focused on the information of PD to prevent congestion. The concept of PD was used in [11] as a probability to select path randomly in order to prevent congestion. In [12], PD is used in the selection function for fault tolerance because packets could have more choices to avoid the faulty nodes when the PD is high. In addition, in [13], the concept of difference in PD is brought into the algorithm in NoP and *Advanced-NoP* was proposed with better performance than the original NoP; this gives us a great incentive of using PD in the routing algorithms. Therefore, our motivation is to avoid congestion by considering the information of PD as a new selection method.

III. PATH DIVERSITY IN PATH DETERMINATION

Although we can easily compute the PD in every direction in Fig. 1, we still need a thorough numeric analysis to develop a general method for every single node in NoC. In this section, we are going to analyze the property of PD and derive a formula to quantify it for our later use in path determination.

A. The Concept of Path Diversity

The PD represents the number of paths the packets can take during routing and is affected by routing algorithms and topology to a certain extent. From the view of topology, if a network has more connections between adjacent nodes, the PD will be higher. Besides, a square mesh will have higher PD than a rectangular one and a long-distance source-destination pair will have higher PD than short-distance one. From the view of routing algorithms, those with less turn model restriction will have higher PD.

However, when it comes to compute the exact number of possible paths, it is incorrect to do with the two factors, routing algorithms and topology size, in (1) separately because they do not have a linear relationship. That is, we can only make PD as a function of the combination of them and there are different expressions of (1) according to different combinations. For example, if the factor R is the partially adaptive routing function, Odd-Even Routing, and factor T is an 8×8 mesh, it will lead to the formula list in [4] to calculate the exact path diversity for packet going to east as,

$$\mathrm{PD} = \frac{[h' + (hc\_y)]!}{h'!\,(hc\_y)!}, h' = \begin{cases} \dfrac{hc\_x - 1}{2}, hc\_x \text{ is odd} \\ \dfrac{hc\_x}{2}, hc\_x \text{ is even} \end{cases}, \quad (2)$$

where *hc_x* and *hc_y* are hop counts between source and destination in *x* and *y* direction, respectively. This formula shows how routing functions and topology influence PD. Both *hc_x* and *hc_y* imply the distance and topology in network. The two possible choices of h' are a result of the special restrictions of Odd-Even Routing.

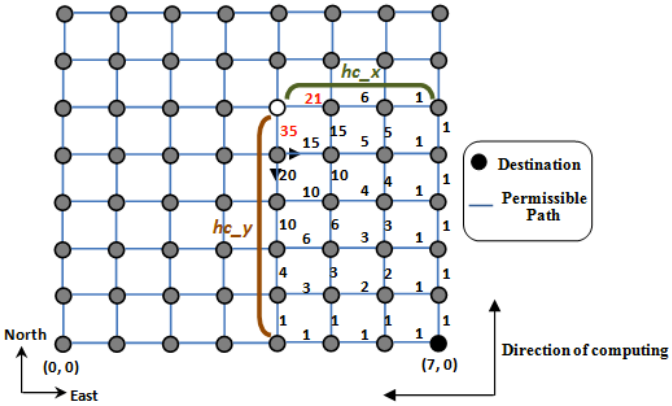


Figure 3. The schematic sketch of path diversity for a fully adaptive routing function with destination (7, 0).

An example of using the formula is the result PD in Fig. 1. Fig. 1 reflects the fact that due to restrictions in turn model, node (0, 6) has higher PD than node (1, 7) even if they have same destination (7, 0). For another example, node (0, 3) has higher PD in east direction. However, the situations in these examples are not totally the same because we have included an additional factor in computing the PD. This additional factor is *distance* which also means the topology factor. To make the PD only affected by the factor R, which is the restriction in turn model, we need to eliminate the effect of distance in computing PD.

B. Path Diversity Normaliztion

To make the problem clear, Fig. 3 shows the PD of fully adaptive routing function for a packet going from white to black node. For fully adaptive routing function, it has no restriction in both *x* and *y* directions in every node; thus, we expect to have equal PD in output directions for a given node. However, Fig. 3 does not show the result as what we expected. The reason is that the distance is involved in our calculation of PD that makes the direction with higher hop counts have higher PD.

In Fig. 3, we can generally assume that there is a packet going from the white node to the black node. The hop counts from white to black node in *x* and *y* direction are *hc_x* and *hc_y,* respectively. In the case of fully adaptive routing, the path diversity in *x* and *y* direction are

$$\frac{[(hc\_x - 1) + (hc\_y)]!}{(hc\_x - 1)!\,(hc\_y)!} \quad (3)$$

and

$$\frac{[(hc\_x) + (hc\_y - 1)]!}{(hc\_x)!\,(hc\_y - 1)!}, \quad (4)$$

which are not the same in the denominator. To make them identical to each other, we can add the terms *hc_x* and *hc_y* in (3) and (4) to get:

$$\frac{[(hc\_x - 1) + (hc\_y)]!}{(hc\_x - 1)!\,(hc\_y)!\,(\boldsymbol{hc\_x})} = \frac{[(hc\_x - 1) + (hc\_y)]!}{(hc\_x)!\,(hc\_y)!} \quad (5)$$

$$\frac{[(hc\_x)+(hc\_y-1)]!}{(hc\_x)!\,(hc\_y-1)!\,(\boldsymbol{hc\_y})} = \frac{[(hc\_x)+(hc\_y-1)]!}{(hc\_x)!\,(hc\_y)!} \quad (6)$$

With a closer look, the added terms are indeed the hop counts in x and y direction. Therefore, we refine the calculation of PD by dividing it by a *distance factor* (DF) as shown below,

$$Normalized\ PD = \frac{PD}{DF} \quad (7)$$

$$where\ DF = \begin{cases} hc\_x, for\ East/West\ Path \\ hc\_y, for\ North/South\ Path \end{cases}.$$

This normalization makes the derived path diversity only affected by the imbalanced property of the routing function itself, and we call the derived one as *normalized path diversity (*NPD*)*. We have to mention that the DF provided above is for regular mesh and may vary in other topology.

With NPD, we can figure out how imbalanced the routing function is in different directions. For instance, a path with higher NPD means that routing function permits more possibilities in that path, which is inherently robust for heavy load. In other words, we can intuitively select a path with higher NPD to deliver packets when there are two candidates.

IV. Proposed Algorithm

We have already discussed the existence of imbalanced property in routing and introduced the concept of NPD. Unlike the information of free slots used in OBL and NoP, which is local and dynamic, NPD is global, static and can be computed previously. Obviously, due to its offline computation, it does not give the system extra burdens during operation while offers extra information. Next, we propose a set of algorithms to apply it in path determination.

A. NPD Information Storage

After calculating the NPD, we need to store this information in every router. However, storing the exact number of NPD in a router seems to be an unrealistic way. From the router's perspective, it chooses "better" direction during selection; "better" here implies us that instead of storing exact number of NPD, we only need to store the relationship of NPD between possible output directions. For example, we can store the direction with higher NPD.

To begin with, we construct an NPD table in a 5×5 network topology as shown in Fig. 4 for each router, and we use router (2, 2) as an example. The number "*0*", "*1*", "*2*", and "*3*" mean higher NPD in *north*, *east*, *south* and *west* direction, respectively and "*X*" means don't care. Every time a packet passes (2, 2) for subsequent transmission, it checks the position of its destination in the table to see which direction has higher NPD. If the packet is going from node (2, 2) to node (4, 0), it can check the position (4, 0) as indicated in Fig. 4 to know "2 (*south*)" direction has higher NPD. In this way, we can efficiently store NPD information.

B. NPD Table Reduction

The NPD table provides another criterion in path determination for routers. However, as the topology becomes bigger, the size of NPD table will also increase, which means this implementation lacks scalability in area. To overcome this problem, we simplify and reduce the size of NPD table to make it unchanged in different sizes of topology.

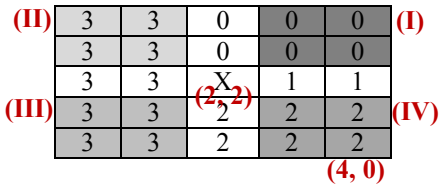


Figure 4. The NPD table stores the direction with higer NPD for router (2, 2) in a 5×5 mesh topology

Quadrant	Direction With Higher NPD
I (North-East)	0 (North)
II (North-West)	3 (West)
III (South-West)	3 (West)
IV (South-East)	2 (South)

Figure 5. The modified NPD table of Fig. 4.

First, we can view the (2, 2) router as the origin of its table in Fig. 4. Besides, the row and column containing (2, 2) are the x- and y-axis. With minimal routing, the packet in (2, 2) can only have a single path when its destination lies in either the same row or column. That is, there is no need for path determination in this region and we can remove the x- and y-axis in the table first.

Second, due to the forbidden turns in the routing function, the numbers of each entry in NPD table are the same within each quadrant. This phenomenon is due to the constraints of routing function discussed previously. For example, the *E-S* turn is forbidden in some columns in Odd-Even Routing. For packets going south-east will always have fewer paths in East because of the forbiddance in E-S turn. Therefore, *south* provides more paths and the numbers in south-east quadrant of the NPD table are the same, "2 (*south*)" in this case.

With the consideration above, the approach of table reduction is obvious. We only need to store one number for each quadrant in the router. The incoming packet can see which direction has higher NPD according to the quadrant of its destination. Moreover, this cost would not be affected by the topology size, which shows its scalability in area. The implementation is shown in Fig. 5.

C. Path-Diversity-Aware Selection

With the elegant design of the NPD table, we can now use it in the path determination. Since higher NPD stands for more paths, directions with higher NPD are likely to avoid congestion and reduce latency under heavy traffic. Therefore, when the routing function returns two directions for a packet, we would select the direction with higher NPD for it. This method depends only on the information of NPD table and we call it as *Path-Diversity-Aware* (PDA) *Selection*. Below, we are going to give a formal description of PDA.

Consider a packet is going to (4, 0) from (2, 2), the same case as in Fig. 4, the routing function will return *East* and *South* direction. Then, we first check the availability of these directions and then decide how to use the NPD table. There are three possible combinations: (i) none of them are

available, (ii) only one of them is available, and (iii) Both of them are available.

In case (i), PDA does nothing because the packet is going to be stalled. In case (ii), PDA selects the available one right away even if its NPD is lower. Since it is difficult to predict the cycles we need for the channel with higher NPD being available, the advantage of choosing it is not clear. Thus, choosing available one can add some variability into PDA selection. Besides, this makes the packets avoid being sent to the same direction every time, while not increase the total latency. In case (iii), since both directions are available, PDA looks up the NPD table and selects the one with higher NPD. The overview of PDA and its pseudo code are shown in Fig. 6(a) and (b), respectively.

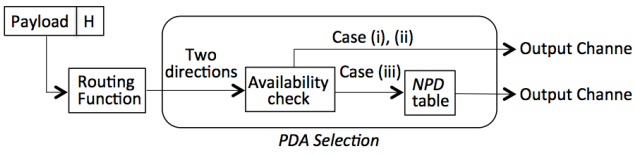
D. Augmented-Path-Diversity-Aware Selection

Due to the low cost of NPD table, another scheme is to use it with the existing selection functions. Though those existing selection functions work well most of the time, they sometimes cannot make decisions and we call this scenario as *congestion indetermination* (C.I.). For example, in OBL and NoP, the same free slots or same scores for two output directions make them unable to decide which one is better. According to Table I, C.I. happens from 30.05% to 67.43% of the time given a reasonable traffic load. Therefore, using the NPD table can provide extra information for further determination.

To use the NPD table, we simply combine PDA and existing selection functions together. When the existing selection functions cannot judge which direction is better, we then use PDA for further decision. Since we add PDA to the existing selection functions, we call this approach as *Augmented-Path-Diversity-Aware (*A-PDA*) Selection*. The overview of A-PDA is shown in Fig. 7.

TABLE I. THE PERCENTAGE OF C.I. IN OBL AND NOP.

Traffic	Transpose1		Random	
Selection Function	OBL	NoP	OBL	NoP
%	67.43	44.12	49.98	30.05



(a)

```
1   PDA Selection ( directions[], dst ) {
2     case = Availability_check (directions[])
3     if case == 1 or 2
4       return directions[RAND()%2]
5     else
6       assign quadrant according to dst's location
7       return NPD_Table [quadrant]
```

(b)

Figure 6. (a) The overview of the adaptive routing function based on PDA selection and (b) PDA's pseudo code.

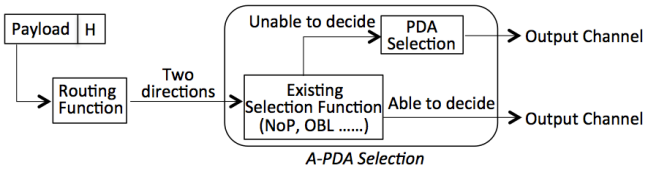


Figure 7. The overview of the adaptive routing function based on A-PDA selection.

V. PERFORMANCE EVALUATION

Firstly, we compare PDA with other selection functions under different traffic patterns in Experiment 1. Then in Experiment 2, the analysis of A-PDA is compared with the original selection function that A-PDA combines with. In Experiment 3, we use PDA and A-PDA selection to deal with the scenario of hot spot traffic. In Experiment 4, we analyze the scalability of A-PDA and other selection functions in different sizes of topology. Finally, in Experiment 5, we simulate our proposed methods in a real traffic.

The experiments are evaluated by the NoC Simulator, Noxim [15], which is an open source SystemC simulator. The topology we use is a 16×16 mesh network. The system runs with wormhole switching mechanism and matrix arbitration. The cycle of handshaking is one. Besides, we use the *Odd-Even* adaptive routing function to do the simulations under different traffic patterns, such as *transpose1*, *random* and *hotspot*. The results of other adaptive routing functions having imbalanced NPD conform to the performance of evaluation.

In *random* traffic, each source is equally likely to send to each destination. In *transpose1* traffic, a source node (*i*, *j)* only sends packets to node (16-*j*, 16-*i*). In the *hotspot* traffic, one or more nodes are designated as the hotspot nodes, which receive hotspot traffic in addition to the regular traffic pattern. Given a hotspot percentage *h*, a newly generated packet is directed to each hotspot node with an additional *h* percent probability. In *hs-br* traffic, we locate hot spot nodes in the bottom-right corner of the network, that are (14, 14), (15, 14), (14, 15) and (15, 15) and *h* equals 0.12. In *hs-c*, we locate hot spot nodes in the center of the network, that are (6, 7), (7, 7), (8, 7) and (9, 7) and *h* equals 0.12.

Besides, each channel has an input buffer with size of 4 flits and each packet has 8 flits. For each run of simulation, the total time is 20,000 cycles and the first 2,000 cycles is warm-up time of NoC system. For the instant at which a packet is injected is chosen on the basis of a Poisson distribution. To guarantee the accuracy of results, the simulation at each point of packet injection rate (*pir*) has been repeated 20 times. The saturation throughput is used as performance metric in our experiments and its definition is where average latency equals to twice of the zero-load latency [19].

Experiment 1: Global Information versus Local Information

To see how global information, NPD table, affects the overall performance, we compare it with different selection functions using local information in *transpose1* and *random* traffic. The experimental results are shown in Fig. 8. They

show that *PDA* selection is superior to all the other selections in saturation throughput. The improvement in *transpose1* traffic ranges from 16.07% to 36.84%; in *random* traffic, it ranges from 1.22% to 13.79%. Since both traffic patterns have different properties, the improvement differs a lot between them. For traffics having regular source-destination pairs, such as *transpose1* traffic, global information dominates the overall performance and the local one can be ignored. On the other hand, for traffics having irregular source-destination pairs, such as *random* traffic, the improvement is not significant.

Experiment 2: Improvement in Saturation Throughput of A-PDA

Using A-PDA, we tend to solve the problem when C.I. happens. To analyze its performance, we use A-PDA combined with OBL and NoP in *transpose1* and *random* traffic. The experimental results are shown in Fig. 9 and they show that A-PDA further improves the saturation throughput of original selection functions. Under *Transpose1* traffic, the improvements are 8.03% and 23.15% compared to NoP and OBL, respectively. However, we find the amounts of improvement decrease a little compared to the result of *Experiment 1*. Since A-PDA uses PDA as the second pass in selection, the reduction supports our inference in *Experiment 1* that global information dominates the performance for traffic having regular source-destination pairs. Even so, A-PDA still has significant improvement for both combinations with NoP and OBL. Under random traffic pattern, the improvements are 3.75% and 8.19% and A-PDA effectively improves both selection functions.

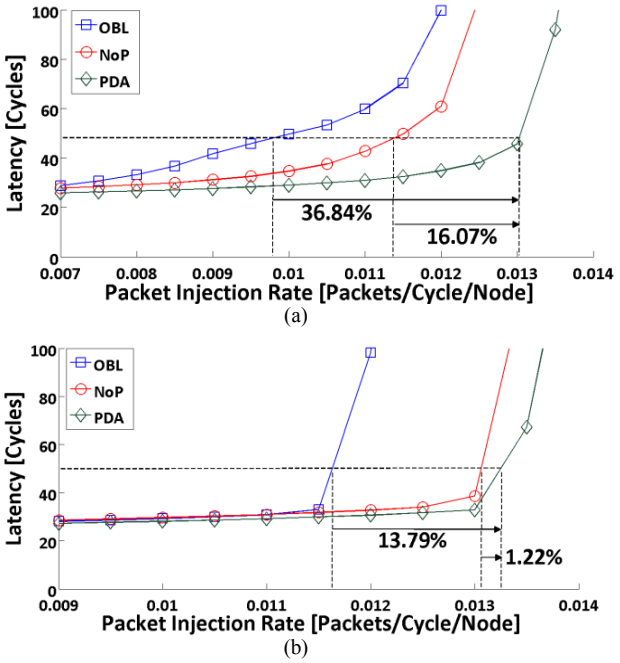


Figure 8. The latency variation of PDA under (a) *transpose1* and (b) *random* traffic.

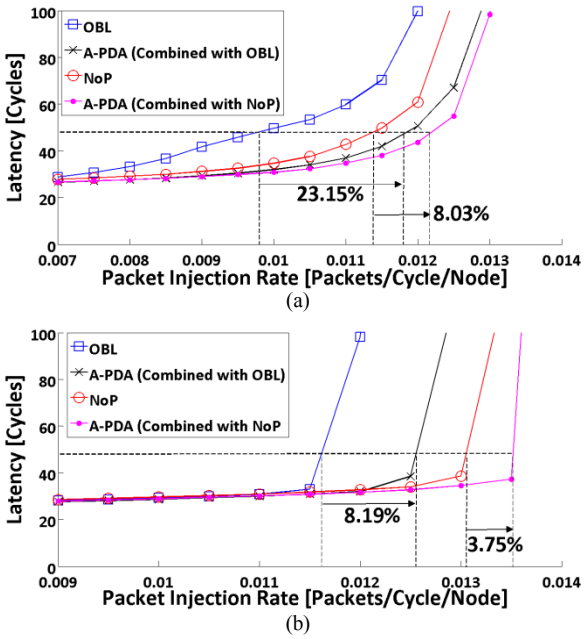
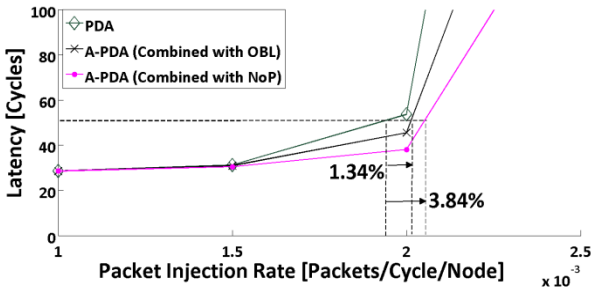
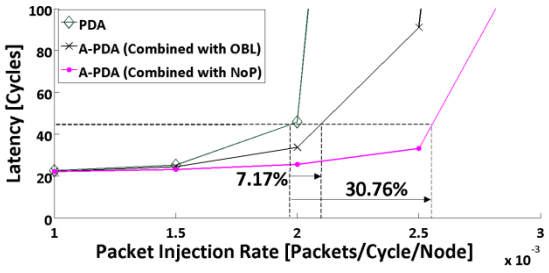


Figure 9. The latency variation of A-PDA and other original selection functions under (a) *transpose1* and (b) *random* traffic.

Experiment 3: Hotspot Traffic Scenario

When designing the NoC system, the designers always plan the whole scheme carefully to avoid hotspots occurring in some nodes. Besides, hotspots rarely happen in a well-designed NoC system. However, if this situation indeed happens, the overall performance will be greatly affected. Hence, the designers should take this scenario into consideration. In the following, we analyze the performance of PDA and A-PDA under *hs-c* and *hs-br* traffic and the experimental results are shown in Fig. 10. Obviously, A-PDA combined with either OBL or NoP outperforms PDA in both hotspot traffics. This shows PDA which lacks local information cannot perform well under such heavy traffic, and this is not seen in previous experiments. On the other hand, A-PDA using local and global information deals well with such heavy traffic. To be conservative in our designs, we can use A-PDA to help us handle the hotspot traffic and still have a considerable improvement.



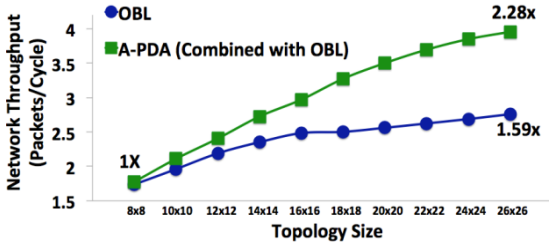


(b)

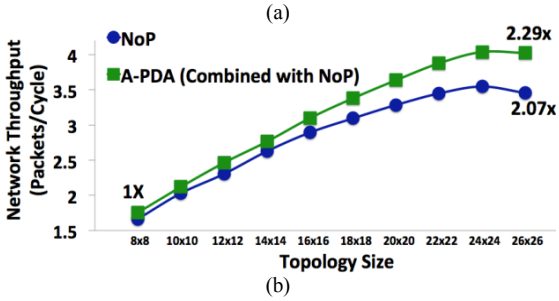
Figure 10. The latency variation of A-PDA and PDA under (a) *hs-br* and (b) *hs-c* traffic.

Experiment 4: Performance Scalability of A-PDA

To see the effects of adding PDA to the original selection functions, we analyze the scalability of A-PDA in performance by examining the network throughput in different topology sizes and traffic patterns. The network throughput (NT) is defined as the accepted traffic of the network at a given latency. Here, we use the latency of *Experiment 2*, which equals to twice of the zero-load latency in a 16×16 mesh topology. Besides, we use the value in 8×8 mesh as our standard NT. For an $N \times N$ mesh, its ideal NT should be $N^2/8^2$ times as big as the standard NT. The experimental results are shown in Fig. 11, and we use transpose1 traffic as our example. In a 26×26 mesh, the ideal NT should increase by 10.56 times. In Fig. 11(a), OBL saturates much sooner than A-PDA and it only increases by 1.59 times while A-PDA increases by 2.28 times. Similar result is shown in Fig. 11(b) with NoP increasing by 2.07 times and A-PDA increasing by 2.29 times. This trend applies to other traffic patterns as well. Obviously, as the topology size grows bigger, A-PDA grows more steadily than the original selection functions in the network throughput, which means A-PDA improves the scalability of them. Moreover, the cost of A-PDA does not change as the topology size grows due to the constant cost of reduced NPD table proposed above.



(a)



(b)

Figure 11. The network throughput of different topoloty sizeds for (a) OBL and (b) NoP under *transpose1* traffic pattern.

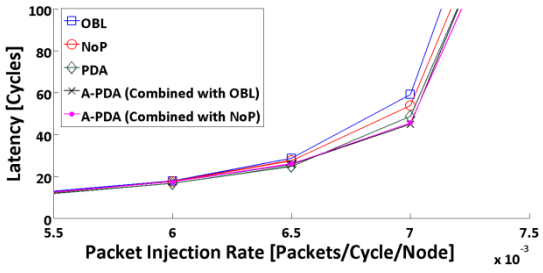


Figure 12. The latency variation under the *mms* traffic scenario.

Experiment 5: Real Traffic Scenario

In a more realistic traffic scenario, we consider a generic *mms*, which includes an h.263 video encoder, an h.263 video decoder, an MP3 audio encoder and an MP3 audio decoder [14]. These tasks of application were assigned and scheduled onto 4×4 selected IPs as in [14]. As we can observe from Fig. 12, both PDA and A-PDA outperform the other selections. For a pir value of 0.007, our proposed algorithms exhibit an average delay from 44 to 48 cycles and others exhibit from 53 to 59 cycles. Due to the small scale of topology, the improvement is not as great as previous experiments. However, with the progress of time, the application will be assigned onto a large-scale topology in the future and the improvement will become much clearer, which is supported by the results of *Experiment 4*.

VI. ARCHITECTURE

The router architecture for a mesh-based network topology implementing PDA is shown in Fig. 13. The top level shows that each router contains a set of first-in first-out (FIFO) input buffer, an adaptive routing algorithm, a matrix arbiter and a crossbar switch circuit.

The routing algorithm is shown in the central part of Fig. 13, and it consists of two components. The first component is adaptive routing function that returns *candidate output channels* (COC). The COC has 4 bits and each bit represents *north, east, south,* and *west* output channel, respectively. The second component is our proposed PDA selection to choose the better output. The detailed architecture of PDA selection is shown in the bottom and there are three main parts.

Part (A) is used for checking the availability of output channels. Part (B) is the construction of our NPD table. In part (C), it selects the source of output channel according to the combination of two availability check blocks in part (A). The detail operations of these parts are discussed in the previous section and would not explain again here. The architecture of A-PDA is easy to derive; we simply modified the central part of Fig. 13 by adding PDA to original selection function as shown in Fig. 14.

Table II reports the summary of the area cost, which is obtained by using Verilog HDL codes synthesized with standard cell library of UMC 90 nm CMOS process. The result shows that with reduced NPD table, the router with PDA has the smallest area among these selections. Moreover, PDA does not require wire connections between routers, and this advantage is not shown in the table. As the scale of NoC grows, the difference in cost will become larger between

PDA and other selections. For A-PDA, it replaces the random selected unit in OBL and NoP with NPD table, which further reduces the cost a little compared to the selections it combines with.

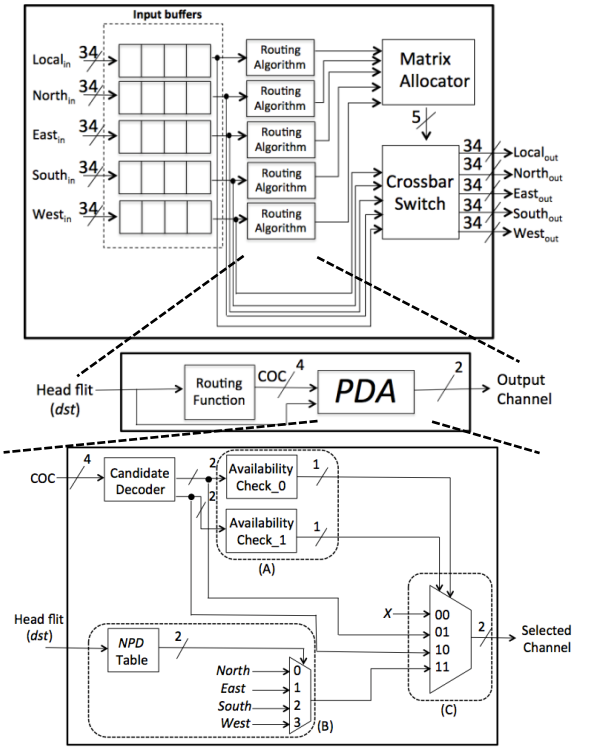


Figure 13. The architecture of the proposed PDA selection function.

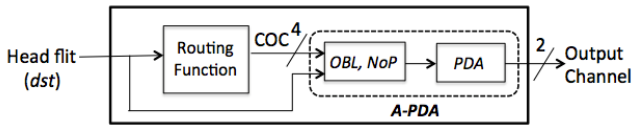


Figure 14. A-PDA architecture derived from the central part of Fig. 13.

TABLE II. AREA PER ROUTER WITH DIFFERENT SELECTION FUNCTIONS

Router with different selection functions	Area (μm^2)
PDA	33,825
OBL	36,556
A-PDA (Combined with OBL)	35,831
NoP	43,671
A-PDA (Combined with NoP)	42,961

VII. CONCLUSION

This paper introduces the concept of PD and NPD and devises an NPD table to store the information. We propose PDA and A-PDA selection using the NPD table. From our experimental results, using global information in selection effectively improves the performance of NoC in every traffic scenario. Besides, PDA outperforms other selections yet has low area cost and A-PDA costs more than PDA but has more stable performance under every traffic scenario.

On the other hand, our proposed selections have great scalability. The experimental results suggest that the improvements of overall performance become larger as the size of topology grows. With its constant cost, PDA and A-PDA provide an enormous potential for large-scale NoC in the future design.

REFERENCES

[1] L. Benini and G. DeMicheli, "Networks on Chip: A New SoC Paradigm," *IEEE Computer*, 35(1): pp.70-78, Jan. 2002.

[2] W.J. Dally and B. Towles, "Route Packets, not Wires: On-Chip Interconnection Networks," *Proc. ACM/IEEE Design Automation Conf.*, pp. 684-689, 2001.

[3] J.D. Owens, W.J. Dally, R. Ho, D.N. Jayasimha, S.W. Keckler, and L.-S. Peh, "Research Challenges for On-Chip Interconnection Networks, " *IEEE Micro*, pp. 96–108, Sep.– Oct. 2007.

[4] G.-M. Chiu, "The Odd-Even Turn Model for Adatpive Routing," *IEEE Trans. on Parallel and Distributed Systems*, vol. 11, no. 7, pp. 729-738, July. 2000.

[5] W.J. Dally, H. Aoki, "Deadlock-free Adaptive Routing in Multicomputer Networks Using Vitual Channels," *IEEE Trans. Parallel Distributed System.,* vol 11, no. 7, pp.466-475, April. 1993.

[6] C.J. Glass and L.M. Ni, "The Turn Model for Adaptive Routing," *19th International Symposium on Computer Architecture*, pp. 278-287, Sep.1992

[7] G. Ascia, V. Catania and M. Palesi, "Implementation and Analysis of a New Selection Strategy for Adaptive Routing in Networks-on-Chip, " *IEEE Trans. Computers*, vol. 57, no.6, pp.809-920, June. 2008

[8] P. Gratz, B. Grot, and S.W. Keckler, "Regional Congestion Awareness for Load Balance in Network-on-chip." *IEEE International Symposium on High-Performance Computer Architecture*, pp. 203-214, Feb. 2008.

[9] Y. Lin, D. Xiang, "An Effective Congestion-Aware Selection Function for Adaptive Routing in Interconnection Networks," *International Conference on Parallel and Distributed Computing, Applications and Technologies*, pp.156-165, Dec. 2010

[10] M. Daneshtalab and A. Sobhani, "NoC Hot Spot Minimization Using Antnet Dynamic Routing Algorithm," *IEEE International Conf. on Application Specific Systems, Architectures and Processors*, pp.33-38, 2006

[11] M.H. Cho, M. Lis, K.S. Shim, M. Kinsy and S. Devadas, "Parameterized Path Based, Randomized, Oblivious, Minimal Path Routing," *NoC Architecture 2009,* pp. 23-28, Dec. 2009

[12] S. Pasricha, Y. Zou, D. Connors and H.J. Siegel, " OE+IOE: A Novel Turn Model Based Fault Tolerant Routing Scheme for Networks-on-Chip," *CODES+ISSS*, pp. 89-94, Oct. 2010

[13] Tang, M.H., Lin, and X.L, "An Advanced NoP Selection Strategy for Odd-Even Routing Algorithm in Network-on-Chip," ICA3PP. LNCS, vol. 5574, pp. 557–568. Springer, 2009.

[14] J. Hu and R. Marculescu, "Energy- and Performance-Aware Mapping for Regular NoC Architectures," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 4, pp. 551-562, Apr. 2005.

[15] "Noxim: Network-on-Chip Simulator," http://sourceforge.net/projects/noxim , 2008.

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

[17] N.E. Jerger and L.-S. Peh, *On-Chip Networks*, Morgan and Claypool, 2009.

[18] F. Gebali and H. Elmiligi, *Networks on Chips: Theory and Practice*, Taylor and Francis Group LLC – CRC Press, 2009.

[19] L. Shang, L. Peh, and N. Jha, "PowerHerd: Dynamic Satisfaction of Peak Power Constraints in Interconnection Networks," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, pp. 92-110, 2006.