Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A fuzzy integrated congestion-aware routing algorithm for network on chip

Shahrouz YASREBI[1], Akram REZA[†‡1], Mohammad NIKRAVAN[1], Seena VAZIFEDAN[2]
[1]*Department of Computer Engineering, Shahr-e-Qods Branch, Islamic Azad University, Tehran, Iran*
[2]*Department of Computer Architecture, Science and Research Branch, Islamic Azad University, Tehran, Iran*
[†]E-mail: a.reza@qodsiau.ac.ir; a.ak.reza@gmail.com

Received Feb. 11, 2020; Revision accepted June 16, 2020; Crosschecked Mar. 29, 2021

Abstract: Network on chip (NoC) is an infrastructure providing a communication platform to multiprocessor chips. Furthermore, the wormhole-switching method, which shares resources, was used to increase its efficiency; however, this can lead to congestion. Moreover, dealing with this congestion consumes more energy and correspondingly leads to increase in power consumption. Furthermore, consuming more power results in more heat and increases thermal fluctuations that lessen the life span of the infrastructures and, more importantly, the network's performance. Given these complications, providing a method that controls congestion is a significant design challenge. In this paper, a fuzzy logic congestion control routing algorithm is presented to enhance the NoC's performance when facing congestion. To avoid congestion, the proposed algorithm employs the occupied input buffer and the total occupied buffers of the neighboring nodes along with the maximum possible path diversity with minimal path length from instant neighbors to the destination as the selection parameters. To enhance the path selection function, the uncertainty of the fuzzy logic algorithm is used. As a result, the average delay, power consumption, and maximum delay are reduced by 14.88%, 7.98%, and 19.39%, respectively. Additionally, the proposed method enhances the throughput and the total number of packets received by 14.9% and 11.59%, respectively. To show the significance, the proposed algorithm is examined using transpose traffic patterns, and the average delay is improved by 15.3%. The average delay is reduced by 3.8% in TMPEG-4 (treble MPEG-4), 36.6% in QPIP (quadruplicate PIP), and 20.9% in TVOPD (treble VOPD).

Key words: Network on chip; Routing algorithm; Congestion control; Fuzzy logic
https://doi.org/10.1631/FITEE.2000069 **CLC number:** TN913

1 Introduction

Currently, the semiconductor industry is rapidly progressing, which makes it feasible to integrate multiple processor cores and memories onto a single chip homogeneously or heterogeneously. Therefore, parallel processing of applications on a multi-core chip is favorable. However, in addition to processing cores, it is vital to have a communication infrastructure to exchange data. Network on chip (NoC) is a flexible, scalable, and reusable solution proposed for chip multi-processor (CMP) systems (Benini and de Micheli, 2002; Pande et al., 2005).

Using the wormhole-switching method to share network resources, the NoC achieves high system throughput rates. However, the impacts of packet blocking in the switches result in an unpredictable delay for each packet stream. As the system size is extended, the network traffic load becomes unbalanced while dealing with various applications (Varatkar and Marculescu, 2004; Pande et al., 2005; Marculescu et al., 2008). Since the routers and channels are vulnerable to congestion, these situations can delay the routing queues and lead to extensive power consumption.

1. Router blocking: blocking occurs if multiple packets compete to take control of an output port on a router.

[‡] Corresponding author
ORCID: Akram REZA, https://orcid.org/0000-0003-3853-2734

2. Router congestion: when a packet is transmitted through the output port, packets receiving a failed output request must be blocked and queued for incoming buffers. Then, the routed packet must wait for this channel to be released.

3. Channel congestion: due to the limited size of the input buffer, the router buffer space may get full and the newly incoming packets cannot be stored when arriving at a router, which results in the loss of the packets.

Furthermore, repetitive concurrent requests for an output channel by different packets result in congestion in the switch. This switch congestion and blocked traffic can upset the previous nodes and may even spread to the source node. Thus, path congestion begins and expands from a point and can enter into the entire network. This drawback can drastically reduce the system's overall performance, especially with real-time applications demanding precise scheduling requirements. The routing function is one of the primary solutions to overcome the traffic congestion impacts and improve the performance of the NoC (Chang et al., 2014).

As the NoC performance highly depends on the performance and delay of the routers, routing strategies play a key role in the communication and performance of the networks, and great improvement efforts have been made. Routing algorithms can be subdivided into several categories such as definitive vs. adaptive, minimal vs. non-minimal, and congestion aware vs. congestion unaware. Moreover, these routing algorithms are designed to be fault-tolerant (Rezaei-Ravari and Sattari-Naeini, 2018).

1. Definitive routing

Since definitive routing algorithms always select a predetermined route from the source to the destination node, they are the simplest and cheapest routing methods. This type of method does not consider the diversity of network paths and ignores the network status. For instance, in the definitive XY routing algorithm, packets move only in the X and Y directions (Dally and Towles, 2004).

2. Adaptive routing

Adaptive routing algorithms can be divided into two subgroups: semi-adaptive and fully adaptive routing algorithms. In the first case and to avoid deadlock, the algorithm determines certain turn restriction models, while the packets select only the shortest path. In the latter case, the algorithm can direct packets through all possible paths, and packets are allowed to choose any path available between the source and destination. The fully adaptive routing algorithm provides path variety to demolish congestion. However, this can cause deadlock problems affecting the stability of the system. Employing a fully adaptive routing algorithm by considering virtual channels or deadlock prevention methods by implementing allocation graphs increases the complexity and implementation cost (Chang et al., 2014).

3. Congestion-unaware routing

Adaptive congestion-unaware routing methods do not consider the output link status. For instance, the random selection function selects an output channel from the candidate channels (Dally and Aoki, 1993). A subsequent selection function selects an output channel with the lowest dimensions of the candidate channels (Dally and Aoki, 1993). A zigzag selection function selects the output channel that takes the maximum step away from the destination router; in other words, it attempts to maximize the number of remaining paths to the destination (Badr and Podar, 1989). An external selection function also attempts to keep the packets away from the network's center (Feng and Shin, 1997). Although these functions are easy to implement, they are incapable of effectively balancing the traffic load, mainly when non-uniform traffic patterns are applied (Feng and Shin, 1997; Martinez et al., 2000). These selection functions usually direct a packet to a congested area, which reduces the overall network performance.

4. Congestion-aware routing

Adaptive congestion-aware routing methods choose an output channel based on different types of network congestion information. As a result, these selection functions can adjust the route selection based on timing and congestion status. Congestion-aware routing algorithms accept two types of spatial information: local information and regional information.

To evaluate the traffic status, the former encompasses information such as the downstream buffer count and the buffer space available. For example, the output buffer length selection function selects the output channel buffer with the maximum amount of available space (Ascia et al., 2008; Fazzino et al., 2008). This buffer space information can be exchanged between two neighboring routers. Likewise, the proximity congestion awareness methods use pressure values to avoid possible congestion, which also contains information regarding the neighboring routers' loads (Nilsson et al., 2003).

The latter type uses regional information. For instance, a low-delay router equipped with congestion detection techniques uses a free buffer counter in a downstream router to estimate congestion (Kim et al., 2005). The neighbor on the path selection function selects an output channel with the least buffer occupied in the possible channels of the neighboring routers. The neighbor on the path selection function attempts to obtain additional spatial congestion information congestion (Ascia et al., 2008). Routing algorithms aware of regional congestion use probabilistic information collected and distributed by monitoring the network among adjacent routers to balance the load (Gratz et al., 2008). Destination-based adaptive routings use a distinct monitoring method to obtain non-local congestion information and then employ this information to estimate the delay to the destination (Ramanujam and Lin, 2010). However, to control congestion, these routing methods require additional wires between adjacent routers.

2 Related work

Turn-based models are a vital method for adaptive routing. One of these algorithms is the odd-even (OE) algorithm (Chiu, 2000). The OE algorithm is an adaptive routing algorithm that specifies two-turn restrictions, one for odd columns and the other for even columns. It can solve the problem of deadlock in the network. Compared to previous methods, it is a more compatible process (Chiu, 2000). Dynamic adaptive deterministic routing design is based on a combination of the benefits of both deterministic and adaptive routing algorithms and provides the opportunity to change the routing mechanism between these two routing modes when confronting different statuses of network congestion (Hu and Marculescu, 2004). All possible paths must be selected according to the turn models to increase routing adaptivity and network performance. By employing the proposed arbitration model, the choice of antecedent routing provides more candidate routes for the reduction of congestion, and therefore reduces the delay and improves the network capacity. Besides, fair judgment leads to a better balance of load as well as better performance in the network (Liu et al., 2017).

The adaptive congestion-aware routing algorithm, which is called dynamic XY (DyXY), is based on the static XY algorithm and sends packets on the X or Y dimension according to congestion conditions. Considering only local information in the routing decision is one of the disadvantages of DyXY, where a packet may be led to a non-congested router, while the adjacent resources are over-congested. Destination-based adaptive routing is one of the congestion-aware routing algorithms, but its main drawback is that it is unscalable. Although the enhanced global congestion-aware routing algorithm based on global congestion awareness solves the problems of timing and accuracy regarding congestion information, it can also cause high latency and power loss (Li et al., 2006). The shared resource routing algorithm not only increases network efficiency, but also uses resource sharing mechanisms, which is an effective solution used to balance the network load. The shared resource routing scheme is based on a shared resource mechanism, in which routers are divided into groups, and each group shares a set of specified link resources. A shared resource routing algorithm is a minimal routing algorithm capable of selecting a better route for each packet based on the network status (Shu et al., 2014).

In neighbor on path (NoP), the congestion information of the direct neighbors is ignored, and the status of the two-hop neighbors is considered (only those residing on the shortest path are considered while the rest are eliminated). However, suboptimal decision-making remains, and packet routing through congested areas is sometimes unavoidable. Unlike classical computer networks where congestion information is transmitted mainly through data packets, NoC can use dedicated proprietary control wires to create a congestion network, which facilitates the exchange of useful information, such as the buffer status of specific nodes, without imposing any additional traffic costs (Khan and Chui, 2017; Touati and Boutekkouk, 2017). Regional congestion awareness (RCA) is the first global awareness routing plan using both local and non-local information and with the aim of improving global network equilibrium. Instead of relying on local congestion information, congestion in the network segments via adjacent nodes is computed and aggregated with locally calculated congestion information before transmission to the upstream nodes (Gratz et al., 2008). Three types of RCAs are proposed, one of which is RCA-1D, where the congestion information is independently propagated and aggregated along each dimension. RCA delivers the statuses of all nodes along all acceptable paths, even the inaccessible minimum path, which results in performance degradation by inappropriate data collection. In destination-based adaptive routing (DBAR), congestion information (the number of virtual channels available) along both the X and Y dimensions is considered for route selection and comparison to RCA-1D. This inspects only the nodes in the minimal path, which might lead to an unfair comparison, especially when the number of nodes in one dimension is greater than that in the other one. Both DBAR and RCA consider the comparison of 50% weights for both local and non-local congestion information (Ma et al., 2011).

The delay caused by congestion, errors, and process variations degrades NoC performance. Thus, a congestion-aware, fault-tolerant, and process-variation-aware adaptive routing algorithm (CFPA) is introduced for heterogonous NoC. The proposed routing algorithm involves two routing tables to determine the routing path of packets: one for the routing direction based on the propagation delay (including a process variation delay) and the other to track queue delays at each router port. Moreover, the delay in the queue is considered a sign of congestion. The proposed routing tables store multiple routes to each destination via polar routes, which makes CFPA a fault-tolerant algorithm in the event of a route failure. Unlike other popular NoC routing algorithms, the proposed algorithm is validated by different network topologies and dimensions (Pano et al., 2018).

NoC was introduced to enhance the performance of CMPs and to execute parallel programs. Although NoC is recognized as a scalable infrastructure for communication, there are still some challenges in traditional NoC such as high latency and power consumption due to long-distance communication. In this regard, wireless NoC (WiNoC) is a potential solution that can provide high bandwidth and low latency using the unique features of wireless communication. However, due to the limited number of wireless channels on a chip and the common practice of these channels for use by all processing elements, wireless routers (WRs) still have congestion potential. Muhammad et al. (2019) proposed a load-balanced time-based congestion-aware (LTCA) routing algorithm to eliminate congestion and distribute the traffic load in wired and wireless networks in a balanced manner. LTCA is a deadlock-free routing algorithm that permits only a limited number of packets to use wireless channels. The time required to transmit the selected packets over wireless links is measured by the bandwidth of the wireless channels and the traffic load. Simulation results on synthetic traffic patterns and real-world 3-tuple traffic patterns indicate a considerable improvement in latency, throughput, wired and wireless link utilization, and packet loss probability (Muhammad et al., 2019).

NoC, which is a new scalable connection infrastructure paradigm for multiprocessor chips, necessitates the analysis of degradation and lifetime effects of network traffic and distributed workloads on a chip. Reliability and redundancy methods are employed to bypass minor defects or return to a previous safe state. These reliability techniques are reactive and do not focus on avoiding analysis that may lead to a complete system failure. Current methods are used to reduce cost and improve lifetime in design time and runtime. Mamaghani and Jamali (2019) proposed a workload-aware routing (WAR) algorithm that complements known techniques and optimizes the overall balance of network traffic. The end-product is a method that exploits workload and priority based routing to improve port and router usage across the network (Mamaghani and Jamali, 2019). The proposed WAR algorithm stays within 2% of the latency and the average hop count of NoC using XY routing. The simulation results using the WAR algorithm depict lifespan improvement when facing traffic up to 17% (an average improvement of 8.6%) on NoC (Mamaghani and Jamali, 2019).

Fig. 1 reveals that selection is made based on the free input buffer and total free input buffer of neighboring nodes. The north node is introduced as the candidate path to the destination since it has fewer (20) occupied buffer slots compared to the east node (25). However, path diversity (the variety of the minimum number of paths to the destination) is less in the north

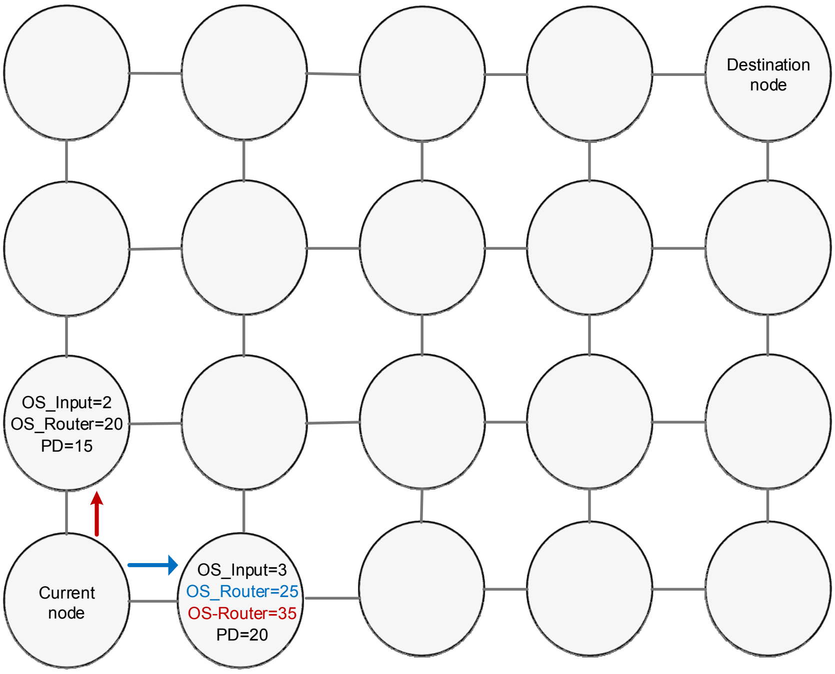

Fig. 1 Path diversity and its effect on path selection

direction (15) compared to the east node (20). In addition, since less path diversity indicates a higher congestion probability, blocking is more probable when selecting the north node.

However, the proposed algorithm selects the path with more path diversity when the congestion condition is low. Since the proposed algorithm benefits from a path diversity parameter and the other parameters previously mentioned, the east node is selected as a candidate node (employing fuzzy logic). Nonetheless, if the number of occupied buffer slots of the east node exceeds 35, this node is no longer a candidate due to its congestion condition.

3 Proposed method

As the NoC desires to be scalable, congestion might be distributed among the network and not centralized over an area, which controls the traffic obligates as a decent visual over the entire network. Although monitoring the whole network might provide useful information regarding path decision making, in addition to the resulting vast computational overload, it can be extremely costly to implement. However, trusting local information is more comfortable and cheaper to implement, but it does not provide a suitable view or information on the network condition. Moreover, regional information has more overhead than local information. Additionally, it does not provide as comprehensive information as monitoring the whole network does; however, it provides better information compared to local information and has less overhead compared to monitoring the whole network. Therefore, considering free buffers in the input buffers of the next routers provides local information about the network, while observing free output buffers of the next router offers regional information of the network. However, it has a local implementation cost since it checks only the buffer of the routers one step away. Thus, it enhances the visual over the network by compounding both local and regional views. Additionally, the path diversity method expunges traffic blocking in the next router. Furthermore, fuzzy logic, which is one of the enhanced uncertainty logical algorithms, is used to heighten system performance and throughput.

The proposed method consists of two steps. In the first step and based on the minimum routing algorithm, one or more channels are determined by considering the current node position and the destination node position. In the second step and by employing fuzzy logic, a path among the candidate ones is selected to transfer the packet.

3.1 Minimal adaptive routing algorithm

Considering the current node location and destination node location, in the first step, one or more output channels are specified as candidate ones. The channels of candidate minimum paths are specified based on the pseudo code illustrated in Fig. 2.

Then, one of the channels specified in the previous step with the lowest congestion is selected after considering the total input buffer slots occupied in the next router, total buffer slots occupied in the next router, and path diversity according to the pseudo code in Fig. 3.

```
Input:
    current node address ($x_c$, $y_c$)
    destination node address ($x_d$, $y_d$)
    free-slot-router
    free-slot-buffer-input
Output: selected direction channel
Procedure:
    xoffset=$x_d-x_c$;
    yoffset=$y_d-y_c$;
    If ($x_c=x_d$ and $y_c=y_d$) then
        out channel=local;
    If ($x_c>x_d$ and $y_c>y_d$) then
        out channel=select function ($X$+, $Y_1$+, $Y_2$+);
    If ($x_c>x_d$ and $y_c<y_d$) then
        out channel=select function ($X$+, $Y_1$−);
    If ($x_c>x_d$ and $y_c=y_d$) then
        out channel=$X$+;
    If ($x_c<x_d$ and $y_c>y_d$) then
        out channel=select function ($X$−, $Y_1$+);
    If ($x_c<x_d$ and $y_c<y_d$) then
        out channel=select function ($X$−, $Y_2$−);
    If ($x_c<x_d$ and $y_c=y_d$) then
        out channel=$X$−;
    If ($x_c=x_d$ and $y_c>y_d$) then
        out channel=$Y_2$+;
    If ($x_c=x_d$ and $y_c<y_d$) then
        out channel=$Y_1$−;
```

Fig. 2 Pseudo code of the minimal adaptive routing algorithm

```
Input:
    Direction[ ], xoffset, yoffset, free-slot-router,
    free-slot-buffer-input
Output: selected channel
Procedure:
    1. Fuzzification inputs
    2. Define the path cost based on the rule table
    3. Compute the cost of path based on
       defuzzification of every path
    4. Select the path with the minimum cost
```

Fig. 3 Pseudo code of the selection function based on the fuzzy algorithm

In the minimal adaptive routing algorithm and to avoid deadlock while benefiting from a high adaptivity rate, two virtual channels (VCs) in the Y dimension along with a turn restriction (Y_1, Y_2) are considered (Fig. 4). Employing two VCs in the Y dimension creates four cycles, clockwise and counterclockwise; to break the cycles, turn restrictions are defined in each cycle.

The path diversity parameter is determined by the distance to the destination in the X and Y dimensions to select the channel with the lowest congestion. Moreover, using the number of free input buffers from the next router and the total free buffers of the next router, the low congestion path is specified by employing fuzzy logic through the four steps of fuzzification of the inputs, determining the rules of route cost, defuzzification of route cost, and low-cost route selection, which is the output.

The congestion parameters embraced in this study are:

1. The number of free buffers in the next router's input port determines the neighbor traffic status.

2. The number of total free buffers in the next router (the total free buffers of the node) provides more accurate traffic information.

3. The route variation in the next router until the destination is reached (the higher path diversity on the next node can reduce the probability of blocking).

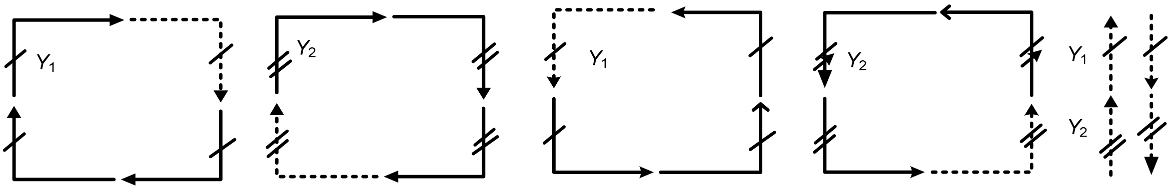

Fig. 4 Turn restriction for deadlock freedom with two virtual channels in the *Y* dimension

3.2 Fuzzy system

The fuzzy inference system (FIS) is employed to select a path for the packet through the routing procedure and consists of four different stages. These stages are known as an input stage or fuzzification, an inference system, a composition unit, and an output stage or defuzzification. Since only the fuzzy inputs are appreciated when practicing the fuzzy system, converting numerical input values into fuzzy values is exercised in the fuzzification step. In the next step, which is the fuzzy inference system, the fuzzy inputs are converted to fuzzy outputs by considering a set of linguistic rules. In the composition stage, the fuzzy outputs of all rules are combined to obtain a single fuzzy output. Finally, in the defuzzification stage, the fuzzy output is converted to a numerical output value. To clarify the system, Fig. 5 illustrates a fuzzy logic system used in a router. A packet can be forwarded to the destination router through a maximum of two directions. The cost of sending the package to the destinations is calculated using fuzzy logic for both directions. The package is then shipped via the lowest cost direction (Ebrahimi et al., 2013).

3.3 Fuzzy logic based routing algorithm

In this subsection, a fuzzy logic based path selection algorithm, called FRA, is introduced. This algorithm provides a new paradigm for designing an NoC router using a fuzzy controller to achieve better performance.

The FRA method has three input variables:

(1) OccupiedSlots_Input;

(2) OccupiedSlots_Router;

(3) PathDiversity(xoffset, yoffset).

In addition, FRA also has a cost output.

The mapping of numerical inputs is placed into their membership functions, known as fuzzy sets, and the fuzzification procedure determines the membership degree of a numerical input (x) and applies it to the appropriate fuzzy set (μ). A membership degree is a number between 0 and 1, where a value of 0 means that x is not a member of the fuzzy set, a value of 1 means that x is entirely a member of the fuzzy set, and values between 0 and 1 denote that the fuzzy member partially belongs to the fuzzy set (Ebrahimi et al., 2013).

3.3.1 OccupiedSlots_Input membership function

Given the number of buffers occupied in the input buffer (OccupiedSlots_Input), the input space is a number between 0 and 8, where a value of 0 indicates that the input buffer is empty, while a value of 8 denotes that the buffer is full. The triangular membership function is employed to elucidate the number of buffers occupied in the input buffer, which can be a number from 0 to 8 (OccupiedSlots_Input), with five fuzzy sets of Z (zero), VS (very small), S (small), M (medium), and L (large). This allocation is shown in Fig. 6. According to this figure, the fuzzy sets are {Z: triangle (0, 0, 2)}, {VS: triangle (0, 2, 4)}, {S: triangle (2, 4, 6)}, {M: triangle (4, 6, 8)}, and {L: triangle (6, 8, 8)}.

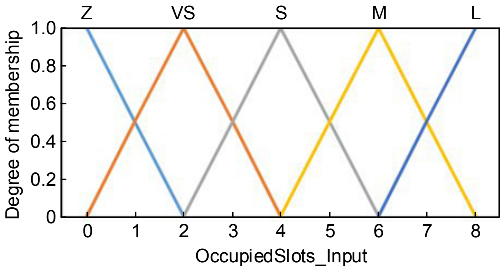

Fig. 6 Allocating the OccupiedSlots_Input membership function to five fuzzy collections

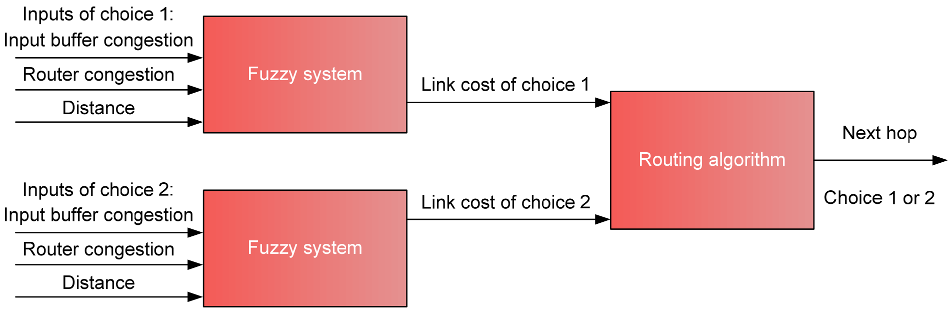

Fig. 5 Fuzzy routing sequence

3.3.2 OccupiedSlots_Router membership function

The OccupiedSlots_Router input can range from 0 to 40, where a value of 0 indicates that all of the buffers in the router are empty, while a value of 40 shows that all buffers are full. This variable can be divided into five fuzzy sets of Z, VS, S, M, and L. As shown in Fig. 7, the fuzzy sets are {Z: triangle (0, 0, 10}, {VS: triangle (0, 10, 20)}, {S: triangle (10, 20, 30)}, {M: triangles (20, 30, 40)}, and {L: triangle (30, 40, 40)}.

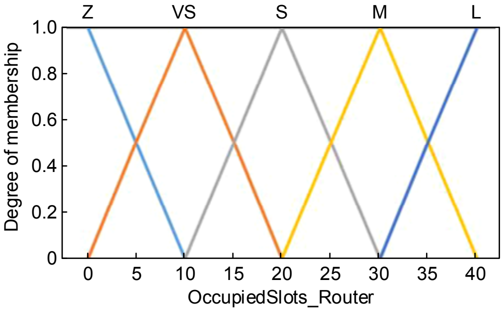

Fig. 7 Dividing the OccupiedSlots_Router membership function into five fuzzy collections

3.3.3 PathDiversity membership function

The distance input consists of the path distance in the *x* dimension (xoffset) and the distance in the *y* dimension (yoffset) and ranges from 0 to 12. The path diversity is defined based on Eq. (1). If xoffset or yoffset=0, there is no path diversity and only one path is available; otherwise, the path diversity equals the total path diversity of the two neighboring nodes on the path.

$$\begin{gathered} W(i,0)=1,\ W(0,j)=1, \\ W(i,j)=W(i-1,j)+W(i,j-1), \\ \text{PathDiversity}(\text{xoffset},\text{yoffset})=\sum_{j=1}^{n}W(\text{xoffset},j). \end{gathered} \tag{1}$$

In a 7×7 grid, 0 indicates that the distance to the destination is the minimum, while 12 indicates the maximum. Accordingly, the range of path diversity is a number between 0 and 462. This path diversity variable is subdivided into three fuzzy sets: Low, Medium, and High. As shown in Fig. 8, the fuzzy sets are {Low: trapezoidal (0, 20, 60)}, {Medium: triangle (40, 80, 120)}, and {High: triangle (100, 140, 140)}.

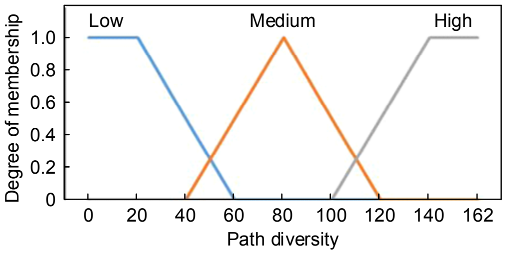

Fig. 8 Allocating the path diversity membership function to three fuzzy sets

3.3.4 Cost membership function (route congestion)

The cost is defined as a value between 0 and 40. Again, five fuzzy sets consisting of Z, VS, S, M, and L are employed. The triangular membership function maps an input element to a specified fuzzy set using a membership degree. As shown in Fig. 9, the fuzzy sets include {Z: triangular (0, 0, 10)}, {VS: triangular (0, 10, 20)}, {S: triangular (10, 20, 30)}, {M: triangle (20, 30, 40)}, and {L: triangle (30, 40, 40)}.

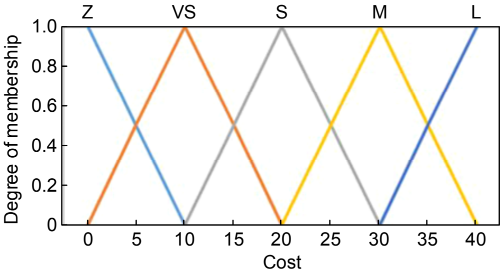

Fig. 9 Allocating the cost function to five fuzzy sets

For instance, in Fig. 5 when OccupiedSlots_Input has a value of 2, the input is entirely owned by the VS membership function. However, a value of 3 indicates that the input is equally part of the two membership functions of VS and S.

3.4 Fuzzy inference system

An inference engine with fuzzy rules is equipped to determine the output channel based on the current state of the network. The inference engine is characterized by a set of linguistic statements to describe the system using a set of "IF-THEN" conditional rules, in which the "IF" section is called "antecedent" and the "THEN" section is called "consequent." Moreover, human knowledge is rapidly used to formulate fuzzy inference system rules (Ebrahimi et al., 2013). Table 1 contains the rules used in the FRA with three fuzzy inputs and one fuzzy output.

There are two techniques to obtain fuzzy laws in the proposed method:

In the first method, five membership functions are considered for the first and second inputs and there are three membership functions for the third inputs. However, since the input space includes the distance to the destination in the *X* and *Y* directions, combining these three inputs includes 225 rules, which is quite numerous.

In the second method, the first and second inputs are combined in the first stage, and then the results are combined with the third input, which reduces the number of rules to 70 (25 in the first step and 45 in the second step). Thus, due to the number of rules, using this method is advisable. These rules are fully presented in Tables 1 and 2 (Ebrahimi et al., 2013).

Table 1 Fuzzy table in the first step

OccupiedSlots_Input	Cost in the first stage				
	OccupiedSlots_Router				
	Z	VS	S	M	L
Z	Z	Z	VS	S	M
VS	Z	VS	VS	S	M
S	VS	VS	S	M	M
M	S	S	M	L	L
L	M	M	L	L	L

Table 2 Fuzzy table in the second step

Cost in the first stage	Final cost		
	PathDiversity		
	Low	Medium	High
Z	Z	Z	VS
VS	Z	VS	VS
S	VS	VS	S
M	S	S	M
L	M	M	L

Tables 1 and 2 stipulate different output values for disparate input ranges. Determining a data table with fuzzy attributes is a subjective procedure. These tables are filled by considering the primary information regarding the impact of each metric on the overall performance of the system. According to these experiments, small changes in the tables result in insignificant effects on the performance (Ebrahimi et al., 2013).

The fuzzy rule sets usually have several antecedent rules combined with fuzzy operators such as fuzzy intersection (AND) and fuzzy unity (OR). If a rule employs the "AND" relationship to map two input variables, the minimum value is considered the output, whereas the maximum value is employed when practicing the "OR" relationship (Ebrahimi et al., 2013).

Some of the fuzzy rules used for path selection described above are mentioned in the following:

Rule 1: IF (OccupiedSlots_Input is S) AND (OccupiedSlots_Router is VS) AND (PathDiversity is H) THEN (cost is VS).

Rule 2: IF (OccupiedSlots_Input is S) AND (OccupiedSlots_Router is S) AND (PathDiversity is H) THEN (cost is S).

Rule 3: IF (OccupiedSlots_Input is M) AND (OccupiedSlots_Router is M) AND (PathDiversity is H) THEN (cost is L).

Rule 4: IF (OccupiedSlots_Input is M) AND (OccupiedSlots_Router is S) AND (PathDiversity is L) THEN (cost is S).

3.5 Composition and defuzzification

To determine the result, the process of generating a quantifiable result in fuzzy logic and converting the fuzzy control action into a numerical value is called defuzzification. The outputs of all rules must be aggregated into a single output. Two widely used methods for defuzzification are used:

1. The center of gravity method (CoG): This method finds the geometric center and considers the law of maximum output area.

2. The mean of maxima method (MoM): This method finds values that have obtained the maximum membership degree regarding the fuzzy membership function.

The MoM method loses useful information, and even though its components are simpler than those of CoG, CoG is more efficient.

In the proposed method, the CoG defuzzification method is employed to generate numerical values. As mentioned earlier, the total area of membership functions is subdivided into several areas. According to the algorithm's uncertainty, each input can be a member of one or more membership functions. This is because based on the fuzzy interface, the combination of different inputs results in various outputs. To compute the output value of the defuzzification steps, we must first compute the outputs based on the membership degree. For the discrete membership function, the value of the defuzzification step using CoG is defined as

$$X^{*}=\sum_{i=1}^{n}x_i\mu(x_i)\Big/\sum_{i=1}^{n}\mu(x_i), \tag{2}$$

where x_i represents the sample element, $\mu(x_i)$ denotes the membership function, and n is the number of elements in the sample. Eq. (2) is employed to obtain the outputs and then to calculate the final result.

For continuous membership, the defuzzified value is computed based on the centroid of area (CoA) method:

$$X^{*}=\frac{\int x\mu_{\mathrm{A}}(x)\mathrm{d}x}{\int \mu_{\mathrm{A}}(x)\mathrm{d}x}. \tag{3}$$

4 Evaluation

In this study, the Noxim simulator is used to evaluate the proposed method. Noxim is a SystemC (a system description library written in C++) based simulator. The Noxim simulator is expandable and scalable (Catania et al., 2016). Noxim can be used to define topologies, routers, and routing algorithms to adjust network size, traffic type, routing type, correlation type and rate, packet injection, and other aspects.

The numbers of gates and variables used to implement the proposed selection function based on fuzzy logic are calculated (Table 3). The power consumption of this function is 3.6 times that of the buffer level selection function, and is assumed in the simulation with the lowest cost (in total, this hardware redundancy cost of the router and an additional 15% power consumption overhead in the simulation). To achieve acceptable simulation results, the power consumption overhead of the proposed routing algorithm is considered 20% higher than that of the basic routing algorithm.

In this section we compare the different injection rates to evaluate the performance of the proposed routing algorithm in hotspot and butterfly traffic patterns. These are examples of congestion and congestion-free traffic patterns. Also, in hotspot traffic, 20% of the traffic is generated at one point, while 80% occurs in other parts of the network. Table 4 clarifies the simulation parameters.

Table 4 Simulation parameters

Topology	Mesh
Network size	8×8
Packet size	8 flits
Buffer depth	4 flits
Simulation time	100 000 cycles
Reset time (warm-up time)	1000 cycles
Traffic patterns	Hotspot, transpose, realistic benchmarks: TVOPD (treble VOPD), TMPEG-4 (treble MPEG-4), and QPIP (quadruplicate PIP)
Packet injection rate	0.001–1

Table 3 Number of gates of the selection function using fuzzy implementation and their signal sizes

Step	Main function	Number of gates	Signal size (bit)
Fuzzification step	OccupiedSlots_Input	8*	5×2=10
	OccupiedSlots_Router	40*	5×4=20
	PathDiversity	8*	5×4=20
	Cost	–	5×4=20
Rule step	Cost-first	25** 12***	5
	Cost-final	15** 12***	5
Defuzzification step	Multiplier/Adder/Divider	604****	6
Total count		724	86
Hardware redundancy in the selection function		$3.2\times\text{gate\_count}_{\text{Buffer level}}$	$1.6\times\text{signal\_count}_{\text{Buffer level}}$

* For the membership function/set defined; ** for the cost set; *** for the cost membership function; **** parallel

For clarification, the routing and selection functions of each method exhibited in Figs. 10–16 are disclosed in Table 5.

Considering the overhead of the proposed method (FA-MPD_CBL), the delay increases with a low traffic injection rate. Nevertheless, since the proposed algorithm benefits from selecting the low congested route as the optimal path to distribute traffic in the network, in the congested traffic patterns, such as the hotspot traffic pattern, as the traffic injection rate increases, the average delay is reduced by 14.88% (Fig. 10). Based on Fig. 10, the DOR algorithm shows the worst results because it determines a path regardless of the network conditions. The DyXY and NOP algorithms lack fuzzy logic in path selection; this method's average delay results are slightly different from those of the FA-BL method. In contrast, other methods have proven to perform better with FA-MPD_CBL being the best. In contrast, this method selects the least congested path to the destination by employing the minimal congestion parameter and the maximum path diversity parameter.

Furthermore, the maximum delay in the hotspot traffic pattern, as shown in Fig. 11, is reduced by 19.39% compared to the improvement in the FA_CBL method, which proves the superior performance of the proposed method in the congested situation.

As shown in Fig. 12, employing the DOR without an aware congestion strategy causes saturation at the 0.075 injection rate, whereas the NOP and DyXY algorithms become saturated at injection rates of 0.10 and 0.15, respectively. Simultaneously, the proposed method becomes saturated at around 0.30. Thus, the total number of received packets in the hotspot pattern is improved by 11.59% in the FA_CBL method using the proposed method.

Regarding energy consumption, the proposed method is equipped with fuzzy logic and low congestion conditions in the network and suffers from additional overhead. As congestion increases, however, better performance starts to appear because there is a decrease in the delay in the network results regarding power consumption reduction. The energy consumption is improved by 7.98%, showing the least power consumption compared to the other methods (Fig. 13).

Table 5 Methods of fuzzy algorithms along with their routing and selection functions

Method	Routing and selection functions
FA_BL	Fuzzy logic + Minimal full adaptive routing buffer (input buffer) level selection
FA_CBL (Ebrahimi et al., 2013)	Fuzzy logic + Minimal full adaptive routing + Buffer level & Router level (total router buffer) selection
FA-MPD_BL	Fuzzy logic + Minimal full adaptive routing + Buffer level & Maximum path diversity selection
FA-MPD_CBL	Fuzzy logic + Minimal full adaptive routing + Buffer level & Router level & Maximum path diversity selection
DyXY (Li et al., 2006)	Minimal full adaptive routing + Buffer level & Router level
NOP (neighbors-on-path) (Ascia et al., 2008)	Minimal full adaptive routing + Buffer level + Channel available
DOR	XY routing + No selection function

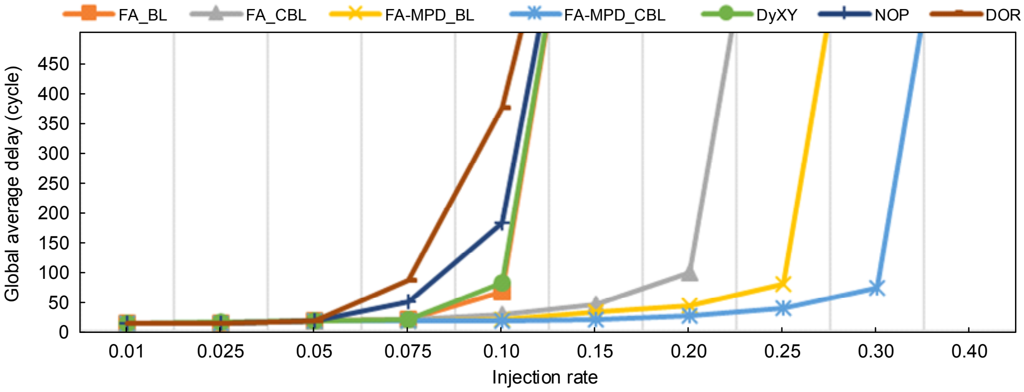

Fig. 10 Average delay using the hotspot traffic pattern

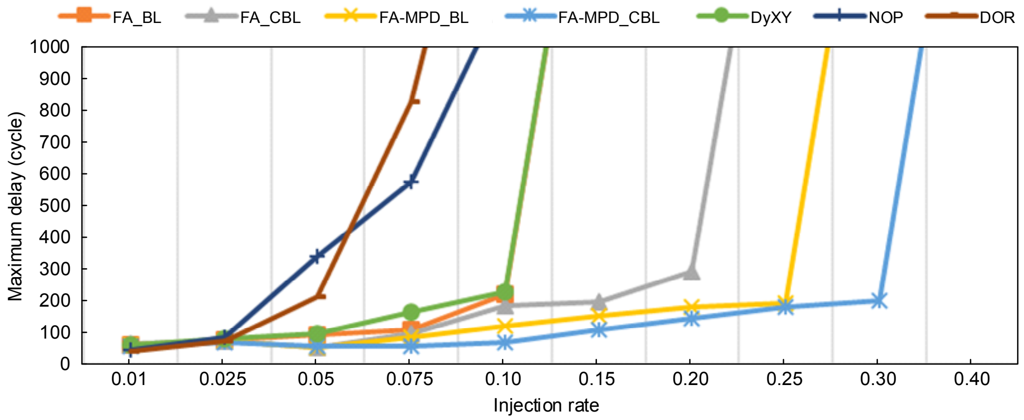

Fig. 11 Maximum delay using the hotspot traffic pattern

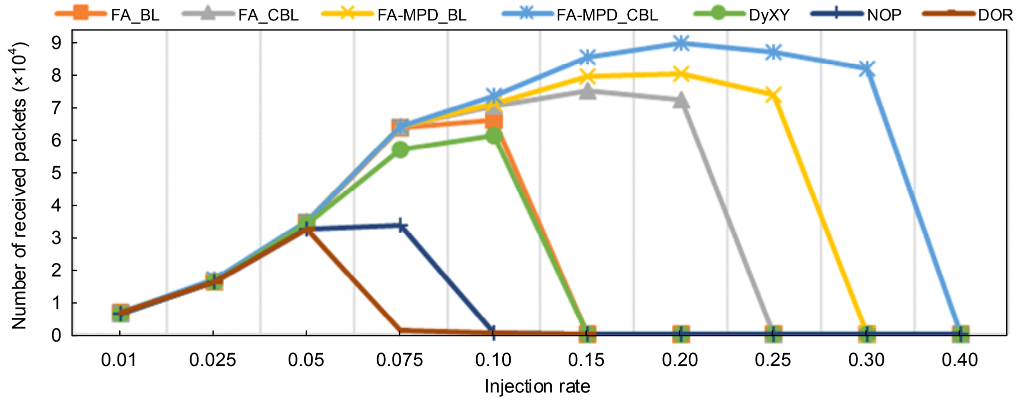

Fig. 12 Total number of received packets using the hotspot traffic pattern

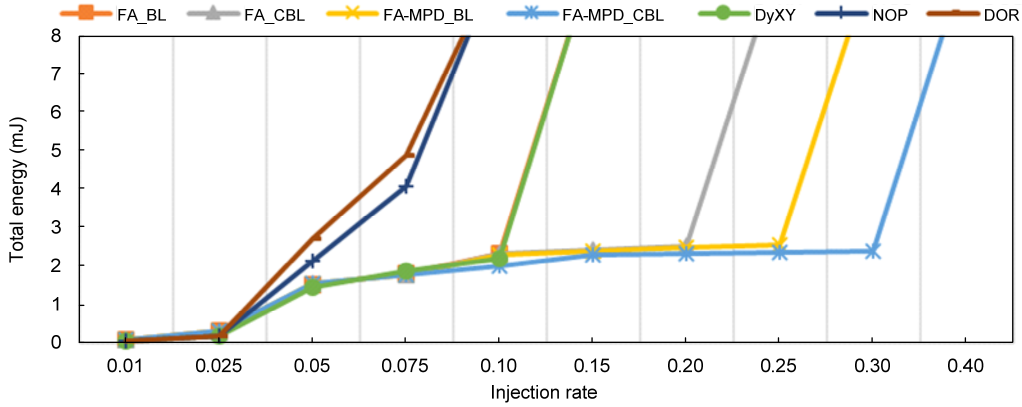

Fig. 13 Energy consumption using the hotspot traffic pattern

Considering throughput, FA-MPD_CBL provides the best results, while the throughput of DOR is minimal because of the lack of a congestion strategy. The throughputs of NOP and DyXY algorithms are less than that of FA_BL. By employing the proposed method, the least congested path with the maximum path diversity is chosen in each step to minimize the utilization of the blocking probability and fuzzy logic, and the quantity of packages sent increases. Therefore, the throughput of the network is improved by 14.9% (Fig. 14).

Furthermore, using the transpose traffic pattern, the average delay (Fig. 15) is decreased as the packet injection rate increases using the proposed algorithm. In contrast, the average delays of the DyXY and NOP algorithms experience steady growth as the injection rate hits 0.04. As elucidated below, FA-MDP_CBL demonstrates better performance compared to the other methods and provides a 15.3% improvement in terms of average delay.

By examining the proposed algorithm with realistic benchmarks, the average delays for the TMPEG-4, QPIP, and TVOPD applications while employing random mapping are exhibited in Fig. 16. The results reveal that the average delay while using these applications is reduced by 3.8% in TMPEG-4, 36.6% in QPIP, and 20.9% in TVOPD compared to that of the FA-CBL method, which is notable.

5 Conclusions

One of the main challenges in NoC is thermal fluctuation, and since one of the leading causes of heat in these networks is congestion, providing methods to control congestion is a leading design challenge. Therefore, to control congestion, this paper presents a fuzzy logic based congestion control routing algorithm, which uses congestion information and then employs this information to select a path that benefits from fuzzy logic uncertainty.

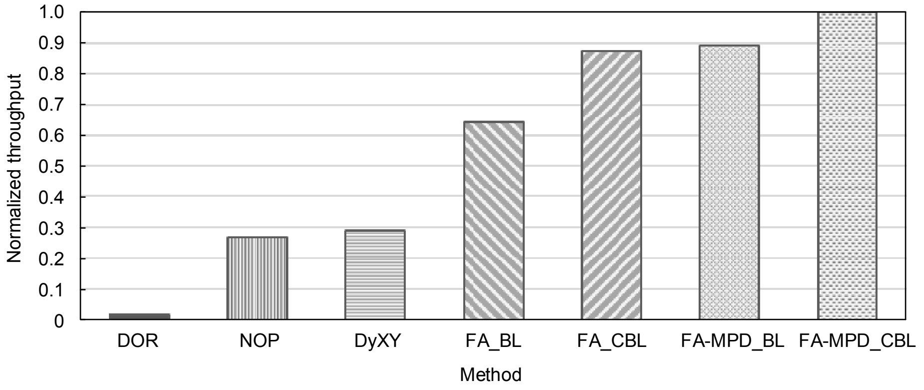

Fig. 14 Average throughput using the hotspot traffic pattern

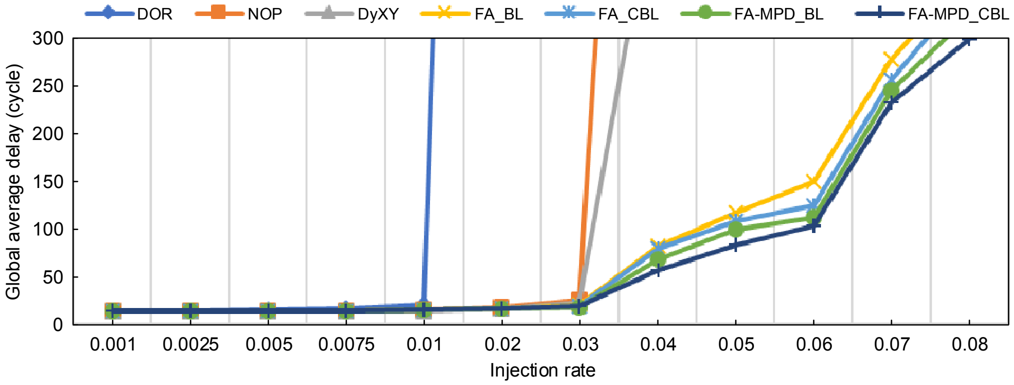

Fig. 15 Average delay using the transpose traffic pattern

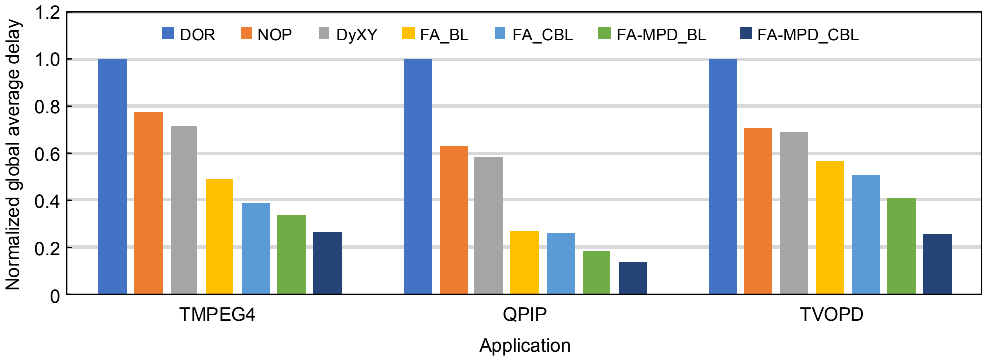

Fig. 16 Average delay using the three applications (References to color refer to the online version of this figure)

Due to the overhead of the proposed method, the delay increases at a low traffic injection rate. However, as it increases, the proposed algorithm chooses an optimal route to reach the ideal traffic distribution, which benefits from the uncertainty of fuzzy logic. As a result, the average delay is improved by 14.88%, the energy consumption is enhanced by 7.98%, the maximum delay is reduced by 19.39%, and the throughput is improved by 14.9%. Moreover, to show the significance, the proposed algorithm is examined using the transpose traffic patterns, and the average delay is improved by 15.3%. Correspondingly, the average delay is reduced by 3.8% in TMPEG-4, 36.6% in QPIP, and 20.9% in TVOPD.

For future research and to control congestion in the network, the use of network information for clustering and time difference implementation is proposed so that the congestion information is transferred from one cluster to another in chronological order.

The separation of the congestion of the network from endpoint congestion is also suggested to manage congestions and provide an individual solution for each.

Contributors

Shahrouz YASREBI and Akram REZA designed the research, implemented the simulations, and drafted the manuscript. Mohammad NIKRAVAN and Seena VAZIFEDAN helped organize the manuscript. Akram REZA and Seena VAZIFEDAN revised and finalized the paper.

Compliance with ethics guidelines

Shahrouz YASREBI, Akram REZA, Mohammad NIKRAVAN, and Seena VAZIFEDAN declare that they have no conflict of interest.

References

Ascia G, Catania V, Palesi M, et al., 2008. Implementation and analysis of a new selection strategy for adaptive routing in networks-on-chip. *IEEE Trans Comput*, 57(6):809-820. https://doi.org/10.1109/TC.2008.38

Badr HG, Podar S, 1989. An optimal shortest-path routing policy for network computers with regular mesh-connected topologies. *IEEE Trans Comput*, 38(10):1362-1371. https://doi.org/10.1109/12.35831

Benini L, de Micheli G, 2002. Networks on chip: a new paradigm for systems on chip design. Design, Automation and Test in Europe Conf and Exhibition, p.418-419. https://doi.org/10.1109/DATE.2002.998307

Catania V, Mineo A, Monteleone S, et al., 2016. Cycle-accurate network on chip simulation with Noxim. *ACM Trans Model Comput Simul*, 27(1):4. https://doi.org/10.1145/2953878

Chang EJ, Hsin HK, Lin SY, et al., 2014. Path-congestion-aware adaptive routing with a contention prediction scheme for network-on-chip systems. *IEEE Trans Comput Aided Des Integr Circ Syst*, 33(1):113-126. https://doi.org/10.1109/TCAD.2013.2282262

Chiu GM, 2000. The odd-even turn model for adaptive routing. *IEEE Trans Parall Distrib Syst*, 11(7):729-738. https://doi.org/10.1109/71.877831

Dally WJ, Aoki H, 1993. Deadlock-free adaptive routing in multicomputer networks using virtual channels. *IEEE Trans Parall Distrib Syst*, 4(4):466-475. https://doi.org/10.1109/71.219761

Dally WJ, Towles BP, 2004. Principles and Practices of Interconnection Networks. Elsevier, Francisco, USA.

Ebrahimi M, Tenhunen H, Dehyadegari M, 2013. Fuzzy-based adaptive routing algorithm for networks-on-chip. *J Syst Arch*, 59(7):516-527. https://doi.org/10.1016/j.sysarc.2013.03.006

Fazzino F, Palesi M, Patti D, 2008. Noxim: Network-on-Chip Simulator. http://sourceforge.net/projects/noxim

Feng WC, Shin KG, 1997. Impact of selection functions on routing algorithm performance in multicomputer networks. Proc 11th Int Conf on Supercomputing, p.132-139. https://doi.org/10.1145/263580.263616

Gratz P, Grot B, Keckler SW, 2008. Regional congestion awareness for load balance in networks-on-chip. IEEE 14th Int Symp on High Performance Computer Architecture, p.203-214. https://doi.org/10.1109/HPCA.2008.4658640

Hu JC, Marculescu R, 2004. DyAD-smart routing for networks-on-chip. Proc 41st Annual Design Automation Conf, p.260-263. https://doi.org/10.1145/996566.996638

Khan GN, Chui S, 2017. Congestion aware routing for on-chip communication in NoC systems. Proc 11th Int Conf on Complex on Complex, Intelligent, and Software Intensive Systems, p.547-556. https://doi.org/10.1007/978-3-319-61566-0_50

Kim J, Park D, Theocharides T, et al., 2005. A low latency router supporting adaptivity for on-chip interconnects. Proc 42nd Design Automation Conf, p.559-564. https://doi.org/10.1109/DAC.2005.193873

Li M, Zeng QA, Jone WB, 2006. DyXY: a proximity congestion-aware deadlock-free dynamic routing method for network on chip. Proc 43rd Annual Design Automation Conf, p.849-852. https://doi.org/10.1145/1146909.1147125

Liu L, Zhu ZM, Zhou D, et al., 2017. A fair arbitration for network-on-chip routing with odd-even turn model. *Microelectron J*, 64:1-8. https://doi.org/10.1016/j.mejo.2017.04.002

Ma S, Jerger NE, Wang ZY, 2011. DBAR: an efficient routing algorithm to support multiple concurrent applications in networks-on-chip. Proc 38th Annual Int Symp on Computer Architecture, p.413-424.
https://doi.org/10.1145/2000064.2000113

Mamaghani SM, Jamali MAJ, 2019. A load-balanced congestion-aware routing algorithm based on time interval in wireless network-on-chip. *J Amb Intell Human Comput*, 10(7):2869-2882.
https://doi.org/10.1007/s12652-018-1020-z

Marculescu R, Ogras UY, Peh LS, et al., 2008. Outstanding research problems in NoC design: system, microarchitecture, and circuit perspectives. *IEEE Trans Comput Aid Des Integr Circ Syst*, 28(1):3-21.
https://doi.org/10.1109/TCAD.2008.2010691

Martinez JC, Silla F, López P, et al., 2000. On the influence of the selection function on the performance of networks of workstations. Proc 3rd Int Symp on High Performance Computing, p.292-299.
https://doi.org/10.1007/3-540-39999-2_27

Muhammad ST, Saad M, El-Moursy AA, et al., 2019. CFPA: congestion aware, fault tolerant and process variation aware adaptive routing algorithm for asynchronous networks-on-chip. *J Parall Distrib Comput*, 128:151-166.
https://doi.org/10.1016/j.jpdc.2019.03.001

Nilsson E, Millberg M, Oberg J, et al., 2003. Load distribution with the proximity congestion awareness in a network on chip. Design, Automation and Test in Europe Conf and Exhibition, p.1126-1127.
https://doi.org/10.1109/DATE.2003.1253765

Pande PP, Grecu C, Jones M, et al., 2005. Performance evaluation and design trade-offs for network-on-chip interconnect architectures. *IEEE Trans Comput*, 54(8):1025-1040. https://doi.org/10.1109/TC.2005.134

Pano V, Lerner S, Yilmaz I, et al., 2018. Workload-aware routing (WAR) for network-on-chip lifetime improvement. IEEE Int Symp on Circuits and Systems, p.1-5.
https://doi.org/10.1109/ISCAS.2018.8351621

Ramanujam RS, Lin B, 2010. Destination-based adaptive routing on 2D mesh networks. Proc 6th ACM/IEEE Symp on Architectures for Networking and Communications Systems, p.1-12.
https://doi.org/10.1145/1872007.1872030

Rezaei-Ravari M, Sattari-Naeini V, 2018. Reliable congestion-aware path prediction mechanism in 2D NoCs based on EFuNN. *J Supercomput*, 74(11):6102-6125.
https://doi.org/10.1007/s11227-018-2515-2

Shu H, Ma PJ, Shi JY, et al., 2014. SRNoC: a novel high performance shared-resource routing scheme for network-on-chip. *Microelectron J*, 45(8):1103-1117.
https://doi.org/10.1016/j.mejo.2014.05.008

Touati HC, Boutekkouk F, 2017. A weighted minimal fully adaptive congestion aware routing algorithm for network on chip. Proc 1st Int Conf on Embedded & Distributed Systems, p.1-5.
https://doi.org/10.1109/EDIS.2017.8284033

Varatkar GV, Marculescu R, 2004. On-chip traffic modeling and synthesis for MPEG-2 video applications. *IEEE Trans Very Large Scale Integr Syst*, 12(1):108-119.
https://doi.org/10.1109/TVLSI.2003.820523