Journal of Systems Architecture 59 (2013) 516–527

Contents lists available at SciVerse ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Fuzzy-based Adaptive Routing Algorithm for Networks-on-Chip

Masoumeh Ebrahimi [a,*], Hannu Tenhunen [a], Masoud Dehyadegari [b]

[a] Department of Information Technology, University of Turku, Finland
[b] School of Electrical and Computer Engineering, University of Tehran, Iran

ARTICLE INFO

Article history:
Available online 6 May 2013

Keywords:
Networks-on-Chip
Fuzzy logic
Adaptive routing algorithm

ABSTRACT

In this paper, we propose two adaptive routing algorithms to alleviate congestion in the network. In the first algorithm, the routing decision is assisted by the number of occupied buffer slots at the corresponding input buffer of the next router and the congestion level of that router. Although this algorithm performs better than the conventional method, DyXY, in some cases the proposed algorithm leads to non-optimal decisions. Fuzzy controllers compensate for ambiguities in the data by giving a level of confidence rather than declaring the data simply true or false. To make a better routing decision, we propose an adaptive routing algorithm based on fuzzy logic for Networks-on-chip where the routing path is determined based on the current condition of the network. The proposed algorithm avoids congestion by distributing traffic over the routers that are less congested or have a spare capacity. The output of the fuzzy controller is the congestion level, so that at each router, the neighboring router with the lowest congestion value is chosen for routing a packet. To evaluate the proposed routing method, we use two multimedia applications and two synthetic traffic profiles. The experimental results show that the fuzzy-based routing scheme improves the performance over the DyXY routing algorithm by up to 25% with a negligible hardware overhead.

1. Introduction

Nowadays, System-on-Chip (SoC) designers employ traditional buses or hierarchical bus structures to connect Processing Elements (PEs) together. Buses cannot transfer more than one data stream simultaneously so that they act as a bottleneck in future many-core architectures. In addition, the main challenges of SoC designers are to come up with the structured, scalable, reusable, and high performance communication platform for integrating a large number of cores on a single chip. To meet these requirements, many research groups have concurrently proposed the idea of using a packet-switched network for on-chip communication in the realm of many-core architectures. This communication platform is called Network-on-Chip (NoC) [1–3].

NoC provides scalability and reusability of communication infrastructure, reducing time to design and time to market of new products. The performance and efficiency of NoC highly depend on the underlying communication infrastructure which relies on the performance of the on-chip routers. Thus, the design of high performance routers represents the success of the NoC approach. The router is the main component of the interconnection architecture that is responsible for delivering packets from input ports to output ports. A network consists of an interconnection of routers together in order to enable the cores to communicate with each other. A mesh-based NoC architecture has been proposed to interconnect the routers as a solution for the complex on-chip communication problems [4,5]. This architecture consists of typical routers in grid mesh, where each router is connected to a PE (i.e. a general-purpose processor, a DSP, a memory module, etc.) [6].

A routing algorithm defines a path taken by a packet between a source and a destination. Routing algorithms are classified as deterministic and adaptive algorithms. Implementations of deterministic routing algorithms are simple but they are not able to balance the load across the links in a non-uniform or bursty traffic [5,7]. The simplest deterministic routing method is dimension-order routing which is known as XY or YX algorithm. In the dimension-order routing algorithms, packets are routed by crossing dimensions in strictly increasing order, reducing to zero the offset in one dimension before routing in the next one. Adaptive routing has been used in interconnection networks to improve network performance and to tolerate link or router failures [8–10]. In adaptive routing algorithms, the path a packet travels from a source to a destination is determined by the network condition. So, they can decrease the probability of routing packets through congested or faulty regions.

In minimal routing algorithms, the shortest paths can be used for transmitting packets between the source and destination routers. Minimal adaptive routing algorithms that do not allow

* Corresponding author.
E-mail addresses: masebr@utu.fi (M. Ebrahimi), hanten@utu.fi (H. Tenhunen), mdehyadegari@ut.ac.ir (M. Dehyadegari).

1383-7621/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sysarc.2013.03.006

packets to use all of the shortest paths are called partially adaptive, while in fully adaptive methods, packets are able to choose among all the shortest paths available between the source and destination [5,8,11–14].

Two main tasks of a routing algorithm are determining a path and transferring packets from a source to a destination. The packet transformation refers to the packet switching communication which is relatively straightforward while the path determination can be complex. Routing algorithms use a variety of metrics to estimate the optimal path for a packet. Metrics such as path length (hop count), reliability, delay, bandwidth, and load are commonly used to determine an optimal route. In our proposed routing schemes, the number of occupied buffer slots at the corresponding input buffer of the next router and the congestion level of that router are exploited as routing metrics. The proposed routing schemes result in increasing the throughput for the same value of average latency per packet under high offered load conditions or decreasing the average latency per packet under low and moderate load conditions.

Fuzzy controllers are classic knowledge-based controllers that have been successfully implemented in various applications where the human expertise and dealing with uncertainty play a vital role in the decision making process. Fuzzy systems avoid arbitrary rigid boundaries by giving a level of confidence to the data. They are commonly used to improve the performance or to resolve ambiguities in complex problems that are difficult to tackle mathematically. Since control problems in communication systems become increasingly complex (due to their characteristics of having multiple performance criteria), the use of fuzzy and adaptive algorithms is indeed well suited to increase the performance. Most applications using fuzzy logic can be regarded as systems with numerical inputs and outputs [15]. The linguistic descriptions are used to define the relationship between input(s) and output(s).

In this paper, at first we propose a Non-Fuzzy Routing Algorithm (NFRA). NFRA can improve the performance over the well-known adaptive method, DyXY, by using more efficient congestion metrics. Then, we present a Fuzzy-based Routing Algorithm (FRA), utilizing the fuzzy logic controller. The fuzzy system is employed to generate the fuzzy cost of candidate directions for delivering a packet. In both fuzzy and non-fuzzy approaches, the congestion metrics are the number of occupied buffer slots at the corresponding input buffer of the next router and the congestion level of that router. At each router, the output direction with the lowest cost is chosen as the one to deliver the packet. The use of fuzzy logic algorithms in path decision making leads to distribute traffic over the less-congested routers. Hence, the average latency can be improved.

The rest of the paper is organized as follows. Section 2 reviews the related works. In Section 3, the preliminaries about the NoC platform and fuzzy system are given. The proposed non-fuzzy and fuzzy methods are described in Section 4 while experimental results are discussed in Section 5. Finally Section 6 concludes the paper.

2. Related work

There have been significant published works on efficient routing schemes in off-chip and on-chip networks. Nilson et al. [16] proposed a Proximity Congestion Awareness (PCA) technique to avoid congestion in the network. In this method, traffic in the hotspot area can be alleviated by spreading it over a larger region based on employing different routing rules. PCA can be utilized to distribute the traffic load more evenly. The congestion metric in PCA is called stress value that determines the load level of a router. This value is sent from one router to its neighbors in all directions. In this way, each router receives at most four values from their neighbors, obtaining a global view of the congestion condition in surrounding routers. Hu and Marculescu in [17] proposed the DyAD routing technique for on-chip networks which combines the advantages of both deterministic and adaptive routing schemes by switching between them based on the congestion condition of the network. When traffic is light, the deterministic routing algorithm is used, otherwise the adaptive routing algorithm is employed. An adaptive deadlock-free routing algorithm, called Dynamic XY (DyXY), has been proposed in [18]. In this scheme, a packet is sent either to the *X* or *Y* direction depending on the networks' congestion condition. The decision on the next hop is made based on the congestion condition of the neighboring routers. DyXY suffers from making non-optimal routing decisions which may result in forwarding packets to congested regions. Lotfi-Kamran et al. [19] introduced the Enhanced DyXY routing algorithm (EDXY) where the congestion information of a router is propagated to all routers in its row and column via separate wires. Non-local information provided by these wires is used only when the packet is one row or column away from the destination router. Balanced Adaptive Routing Protocol (BARP) [20] is another approach proposed in on-chip networks. This method is able to evenly distribute packets over the output ports.

The fuzzy controllers are widely used in many different fields nowadays, ranging from control applications, robotics, image and speech processing to biological and medical systems [15]. Fuzzy controllers have already received much attention in Ad Hoc, wireless and interconnection networks [21,22]. For example in [21], a fuzzy controller is used to instruct cache decisions and to optimize routing selection, so that only high quality links are used between source and destination routers. In [23], the authors present a fuzzy controller-based QoS routing algorithm in mobile Ad Hoc networks in order to dynamically evaluate the route expiry time. In [22], hop-count, bandwidth, and mobile speed are considered for routing decision based on fuzzy logic system to satisfy the required QoS. In [24], using fuzzy rules, the link cost is dynamically determined depending on the link delay and the number of packets waiting in the queue.

The analysis of control problems in communication systems can be perfectly done using fuzzy logic, due to their characteristics of having multiple performance criteria [25]. However, the concept of fuzzy logic has not been distinctly stated in the realm of NoCs. This motivated us to present an efficient adaptive routing algorithm based on fuzzy sets for NoCs. This algorithm takes advantage of the fuzzy-based system to deliver packets through less congested region. The cost is dynamically determined based on the present condition of the network.

3. Preliminaries

3.1. NoC platform

A 2D-mesh NoC based system is shown in Fig. 1. It consists of routers (R), processing elements (PE), links between routers (L), and network interfaces (NI) where PEs can be intellectual property (IP) blocks or memory modules (M). Each core is connected to the corresponding router port using the network interface [6,26]. Routers have layered structure and contain an address decoder, a switch controller, a routing unit, an arbitration unit, and communication ports. Communication ports connect the routers together. They include input and output channels. Each router has at most five bi-directional ports: East, West, North, South, and Local. Each input port has a buffer to store the incoming packets temporarily. The local port connects the router to the local core. The other ports of the router are connected to the neighboring routers, as presented in Fig. 1.

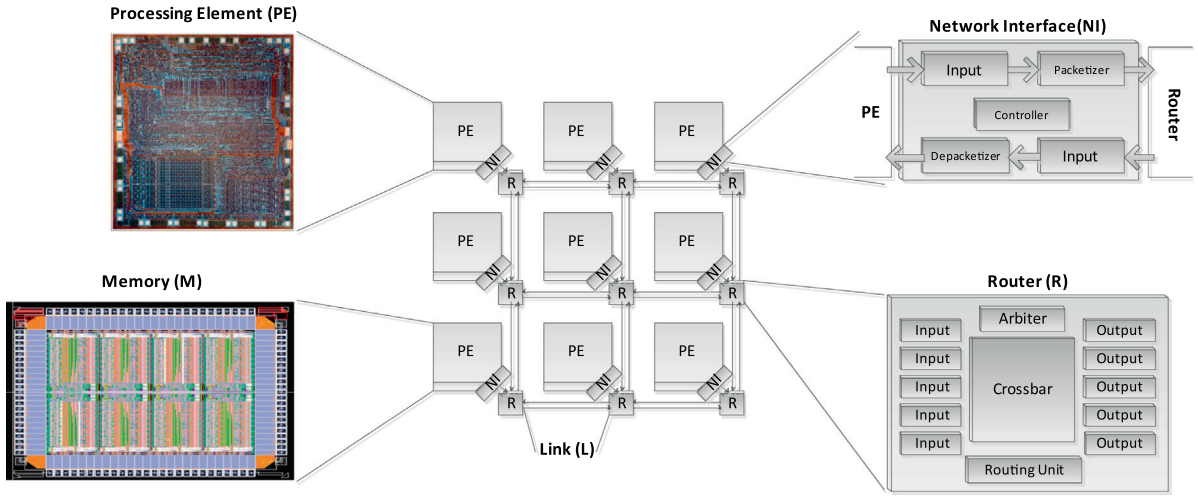

Fig. 1. Tile-based 2D-Mesh topology.

3.2. Fuzzy system

As illustrated in Fig. 2(a), a fuzzy inference system (FIS) consists of an input stage (fuzzification), an inference system, a composition unit, and an output stage (defuzzification). Fuzzification is a process of converting crisp input values to fuzzy values. The fuzzy inference system uses the collection of linguistic rules to convert the fuzzy inputs into fuzzy outputs. In the composition stage, the fuzzy outputs of all rules are combined together to obtain a single fuzzy output. Defuzzification converts the fuzzy output into crisp output value. Fig. 2(b) demonstrates employing a fuzzy logic system in a router. A packet can be sent through at most two directions toward the destination router. The cost is calculated over two candidate minimal directions using fuzzy logic. The packet is sent to a direction with the lowest cost.

4. The proposed non-fuzzy and fuzzy methods

4.1. Non-fuzzy method

Deterministic routing schemes such as XY are more susceptible to hotspot formation as they provide no alternative choice for routing packets when the network becomes congested. In adaptive methods, depending on the traffic condition, the route computation unit may select different paths at different times for the same source and destination pair. Using adaptive routing, the deadlock becomes an issue when packets are waiting for each other in a cycle.

One way to achieve deadlock-free adaptive routing is to use virtual channels [27]. However, adding virtual channels is not free as it requires extra buffering space and complex control logic. Some partial adaptive routing algorithms such as Odd–Even [28] and DyAD [17] do not use virtual channels for avoiding deadlock. These routing algorithms prohibit at least one turn in each of the possible routing cycles [29], thus reducing the degree of adaptiveness. Our Non-Fuzzy Routing Algorithm (NFRA) is based on the fully adaptive routing algorithm, DyXY, using one and two virtual channels along the X and Y dimensions, respectively.

Deadlock freeness should be guaranteed since packets are able to select between X and Y directions at each router. Usually, the following method is used to guarantee deadlock freeness in fully adaptive routing algorithms in 2D mesh networks. Virtual channels in the Y dimension are divided into two parts. Therefore, the network is partitioned into two sub-networks called $+X$ sub-network and $-X$ sub-network, each having half of the channels in the Y dimension. If the destination router is in the east of the source,

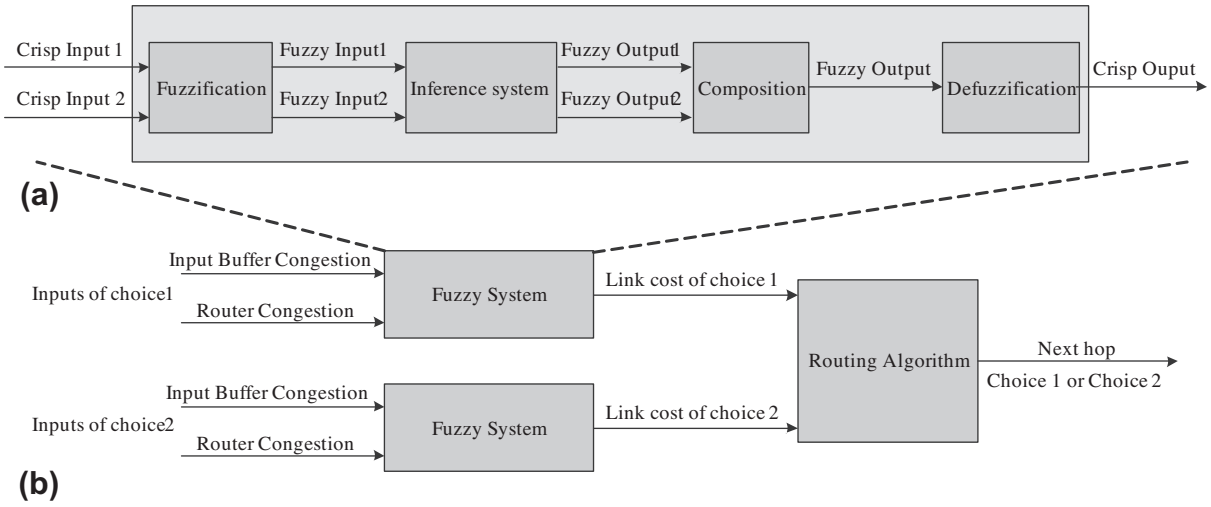

Fig. 2. (a) General fuzzy system (b) fuzzy routing algorithm.

the packet will be routed through the +*X* sub-network. If the destination router is in the west of the source, the packet will be routed through the −*X* sub-network; otherwise the packet can be routed using either sub-networks [30].

In the DyXY method, a pre-port selection unit is added to the router. Based on the number of occupied buffer slots in an instant input buffer of the neighboring routers, the pre-port selection unit selects the best candidate between two minimal directions (i.e. North vs. East for northeast packets; North vs. West for northwest packets; South vs. East for southeast packets; and South vs. West for southwest packets) and makes a routing decision based on this information. Although DyXY is simple, in many cases it leads to non-optimal routing decisions. Let us consider the example of Fig. 3 when the router 5 has to decide whether to send a packet to the router 6 or 9. Since the number of occupied buffer slots in the south input buffer of the router 9 (i.e. five occupied buffer slots) is more than the west input buffer of the router 6 (i.e. four occupied buffer slots), the packet is sent to the router 6 based on the DyXY routing algorithm. The decision is made because of one more occupied buffer slot at the router 9. Now, by looking at the overall congestion level at routers 6 and 9, we notice that the total number of occupied buffer slots at the router 6 (i.e. 28 occupied buffer slots) is considerably larger than that of the router 9 (i.e. 15 occupied buffer slots). In other words, the contention at the router 6 is high and thus packets entering this router from the west input port will be in competition with other packets to receive the desired output channel while this contention situation is mild at the router 9. In this example, obviously it was better to deliver a packet to the router 9 rather than the router 6. It is worth mentioning that DyXY selects a direction by random in the case of the same congestion values in two directions. This example shows that DyXY may lead to a non-optimal decision when the number of occupied buffer slots is comparable in two directions.

In order to improve the performance of DyXY, we proposed a method, named Non-Fuzzy Routing Algorithm (NFRA), which utilizes two parameters for choosing between output directions. These parameters are the number of occupied buffer slots in an instant input buffer of the neighboring router (*OccupiedSlots_Input*) and the total number of occupied buffer slots in the neighboring router (*OccupiedSlots_Router*). To exchange the congestion information of *OccupiedSlots_Input* and *OccupiedSlots_Router* between adjacent routers, 4-bit and 6-bit wires are required. In NFRA, when the differences between *OccupiedSlots_Input* values in two directions are less than or equal to two, *OccupiedSlots_Routers* are checked, so that the packet is sent to a direction which has the lowest *OccupiedSlots_Router*.

OccupiedSlots_Input has been considered as the main factor in NFRA since it has the information about the available slots in an instant input buffer to accommodate a packet rather than the overall router congestion condition (*OccupiedSlots_Router*). Considering the same example as in Fig. 3, since the values of *OccupiedSlots_Inputs* are comparable, the values of *OccupiedSlots_Routers* are compared together and thus the packet is sent to the router 9, resulting in a right decision. The NFRA routing algorithm is shown in Fig. 4.

Although NFRA can alleviate the shortcoming of the DyXY routing algorithm, it is still suffering from making non-optimal decisions due to using rigid boundaries in input parameters. Regardless of the metrics used, this is a common drawback of traditional methods. In the following example, we explain this problem in the case of the NFRA routing algorithm. In Fig. 5(a) suppose that the router 5 has two options (i.e. the router 6 or 9) to forward a packet toward the desired destination. At the router 5, the congestion statuses of the west buffer of the router 6 is compared with the south buffer of the router 9. Since the router 9 has five occupied slots in the input buffer more than that of the router 6, the packet is delivered to the router 6. However, it might not be a right decision as well, as the packet at the switch 9 may get access to its desired output channel earlier than at the switch 6. As another example in Fig. 5(b), when the values of *OccupiedSlots_Input* are three and five in two directions, the decision will depend on the values of *OccupiedSlots_Router*. This may not lead to an optimal decision as the values of *OccupiedSlots_Router* are nearly similar. On the other hand, the router 6 is able to accommodate more flits of the arriving packet and thus it is a better choice. In sum, the inability to find a proper output direction is the weakness of almost all traditional methods. On top of it, solutions vary for different buffer sizes and different metrics. These problems are due to the fact that the decision making is based on rigid boundaries on input variables. A possibility to solve this problem is to qualify the routing algorithms by the means of the fuzzy logic system that allows a flexible and controllable routing process. As the number of control metrics becomes high and controlling the system becomes complex, the use of fuzzy and adaptive control algorithms will be more attractive.

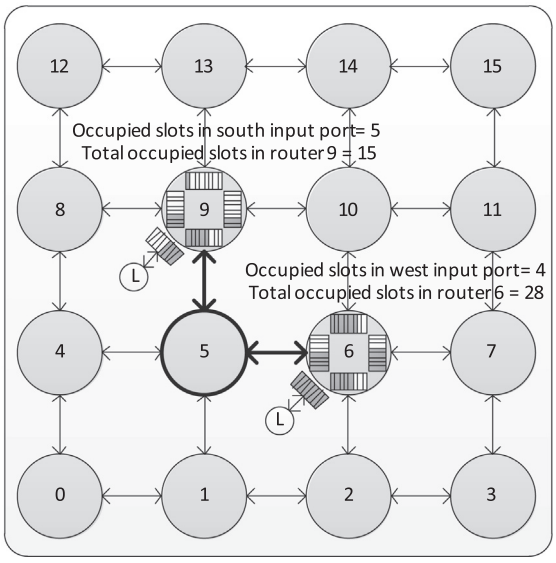

Fig. 3. Non-optimal decision in the DyXY routing algorithm.

4.2. Fuzzy method

In this section, we propose a Fuzzy-based Routing Algorithm named FRA. It provides a new paradigm for NoC router design using the fuzzy controller to reach a better performance gain compared with other router designs. As explained earlier, the fuzzy controller has four parts as fuzzification, inference system, composition, and defuzzification. Fuzzification of a real-valued variable is done with the intuition, experience and analysis of the set of rules and conditions associated with the input data variables. There is no fixed set of procedures for the fuzzification.

4.2.1. Fuzzification

Like NFRA, FRA has two input variables (i.e. *OccupiedSlots_input* and *OccupiedSlots_Router*) and one output (i.e. *Cost*). In the fuzzification stage, the fuzzy controller accepts the crisp inputs and maps them into their membership functions, known as fuzzy set. Fuzzification determines the degree of membership for a crisp input χ being applied to appropriate fuzzy set μ. The degree of membership is a number between 0 and 1.

$$\mu : \chi \rightarrow [0,1]$$

```
ALGORITHM NFRA IS
-- (Cx,Cy): Current  router, (Dx,Dy): Destination router
-- X_dir: candidate port in X direction.
-- Y_dir: candidate port in Y direction.

BEGIN
   IF (Dx = Cx) AND (Dy = Cy) THEN
        Output <= Local_dir;
   ELSIF (Dx = Cx) THEN
        Output <= Y_dir;
   ELSIF (Dy = Cy) THEN
        Output <= X_dir;
   ELSE
        IF (ABS(OccupiedSlots_Input(X_dir)- OccupiedSlots_Input(Y_dir)) <=2) THEN
             IF OccupiedSlots_Router(X_dir) >= OccupiedSlots_Router(Y_dir) THEN
                  Output <= Y_dir;
             ELSE
                  Output <= X_dir;
             END IF;
        ELSIF (OccupiedSlots_Input(X_dir)> OccupiedSlots_Input(Y_dir)) THEN
             Output <= Y_dir;
        ELSE
             Output <= X_dir;
        END IF;
   END IF;
END NFRA;
```

Fig. 4. The pseudo code of NFRA.

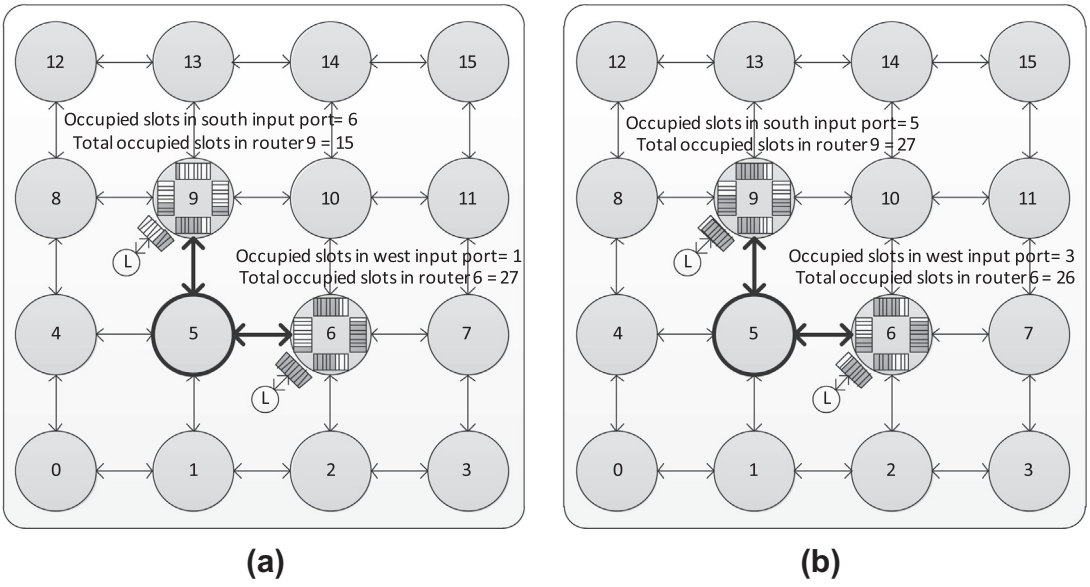

Fig. 5. Two examples of non-optimal decisions in traditional methods.

The value 0 means that χ is not a member of the fuzzy set; the value 1 means that χ is fully a member of the fuzzy set. The values between 0 and 1 characterize fuzzy members, which partially belong to the fuzzy set.

A *membership function* (MF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. The input space is sometimes referred to as the *universe of discourse* [31]. The most commonly used shapes for membership functions are triangular, trapezoidal, and Gaussian. Among them, the *triangular* membership function is the simplest and the most frequently used [31,32]. In the proposed FRA, the assigned membership functions to input and output variables are chosen as triangular. The triangular edges can be identified by the triple (a,b,c) (with $a < b < c$). The parameters (a,b,c) determine the x coordinates of the three corners of the underlying triangular function. Fig. 6(a) illustrates a triangular membership function defined by the triangle (2,4,6). The point 4 has the largest value in the membership function.

In the following, we have defined a fuzzy membership function for *OccupiedSlots_Input* and *OccupiedSlots_Router* as the input metrics and *Cost* as the output parameter.

From three to seven curves are generally appropriate to cover the required range of an input value, or the *universe of discourse* in a fuzzy region. We have examined the fuzzy system using three, five, and seven curves. The performance gain of considering five and seven curves was considerable in comparison with three curves while the difference between five and seven curves was negligible. Thereby, in this work, the fuzzy set includes five states as "zero (Z)", "very small (VS)", "small (S)", "medium (M)", and "large (L)".

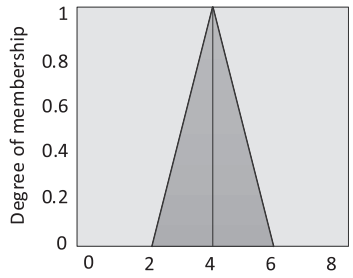

Fig. 6. An example of triangular membership function.

4.2.1.1. Membership function of OccupiedSlots_Input. Considering the number of occupied buffer slots in the input buffer (*OccupiedSlots_Input*), the universe of discourse includes the numbers between 0 and 8. The value of zero indicates that the input buffer is empty while the value of eight indicates that the buffer is full. The triangular membership function maps the number of occupied buffer slots in the input buffer ranged from 0 to 8 (*OccupiedSlots_Input*) to five fuzzy sets (Z, VS, S, M, and L) by a degree of membership. The assignment is illustrated in Fig. 7(a). 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)}.

4.2.1.2. Membership function of OccupiedSlots_Router. OccupiedSlots_Router can be a number between 0 and 40, where the value of zero means that all the input buffers in the router are empty while the value of 40 indicates that all input buffers are full. This variable can be divided into five fuzzy sets as (Z, VS, S, M, and L). As shown in Fig. 7(b), the crisp values are mapped into the sets associated with the degree of membership by defining fuzzy sets as {Z: triangle (0,0,10)}, {VS: triangle (0,10,20)}, {S: triangle (10,20,30)}, {M: triangle (20,30,40)}, and {L: triangle (30,40,40)}.

4.2.1.3. Membership function of cost. We have defined *Cost* as a value between 0 and 40. The fuzzy set includes the states as "zero (Z)", "very small (VS)", "small (S)", "medium (M)", and "large (L)". The triangular membership function maps the input element to a certain fuzzy set by a degree of membership. As illustrated in Fig. 7(c), the fuzzy sets are {Z: triangle (0,0,10)}, {VS: triangle (0,10,20)}, {S: triangle (10,20,30)}, {M: triangle (20,30,40)}, and {L: triangle (30,40,40)}.

With these schemes, the states of input variable are no longer changed abruptly from one state to the next. Instead, as the input changes, it loses a value in one membership function while gaining a value in the next. In other words, an input variable with some degree is part of two membership functions. In Fig. 7(a), for example, when *OccupiedSlots_Input* is 2, the input fully belongs to the membership function VS. However, when *OccupiedSlots_Input* is 3, the input is partially (0.5 each) part of two membership functions VS and S. In general, a fuzzy system is constructed based on human expertise and expert knowledge. The boundaries of the states are also defined in the same manner. The knowledge can be obtained by experiments.

4.2.2. Fuzzy inference system

An inference engine is equipped with fuzzy rules to make a decision for an output channel based on the current condition of the network. The inference engine is characterized by a set of linguistic statements to describe the system by using a number of conditional "IF–THEN" rules where the IF part is called the "antecedent" and the THEN part is called the "consequent". Expert knowledge is usually used to form the rules of a fuzzy inference system. Table 1 contains the rules used in FRA with two fuzzy inputs and one fuzzy output. The table provides various ranges of output for different ranges of inputs. Filling a data table with fuzzy attributes (scaling) is subjective. The table is filled based on the basic knowledge on the impact of each metric in the overall performance of the network. Based on our experiments, small changes in the table have negligible impact on the performance.

Fuzzy rule sets usually have several antecedents that are combined using fuzzy operators, such as fuzzy intersection (AND) and fuzzy union (OR). If the rule uses an AND relationship for the mapping of two input variables, the minimum of those values is used as the output while for the OR relationship, the maximum is used. In FRA, the AND operator is utilized to combine the fuzzy inputs.

Let us consider an example in Fig. 8 where the *OccupiedSlots_Input* and *OccupiedSlots_Router* have the values of 5 and 18, respectively. As shown in Fig. 8(a), *OccupiedSlots_Input* is a part of membership functions S and M while the portion of each membership function is 0.5. The input *OccupiedSlots_Router* is a part of membership functions VS and S as illustrated in Fig. 8(b). In this case, the degree of membership for membership functions VS and S is 0.2 and 0.8, respectively.

As shown in the example of Fig. 9, there are four combinations between *OccupiedSlots_Input* and *OccupiedSlots_Router* as:

- Fig. 9(a): *OccupiedSlots_Input*: S and *OccupiedSlots_Router*: VS.
- Fig. 9(b): *OccupiedSlots_Input*: S and *OccupiedSlots_Router*: S.
- Fig. 9(c): *OccupiedSlots_Input*: M and *OccupiedSlots_Router*: VS.
- Fig. 9(d): *OccupiedSlots_Input*: M and *OccupiedSlots_Router*: S.

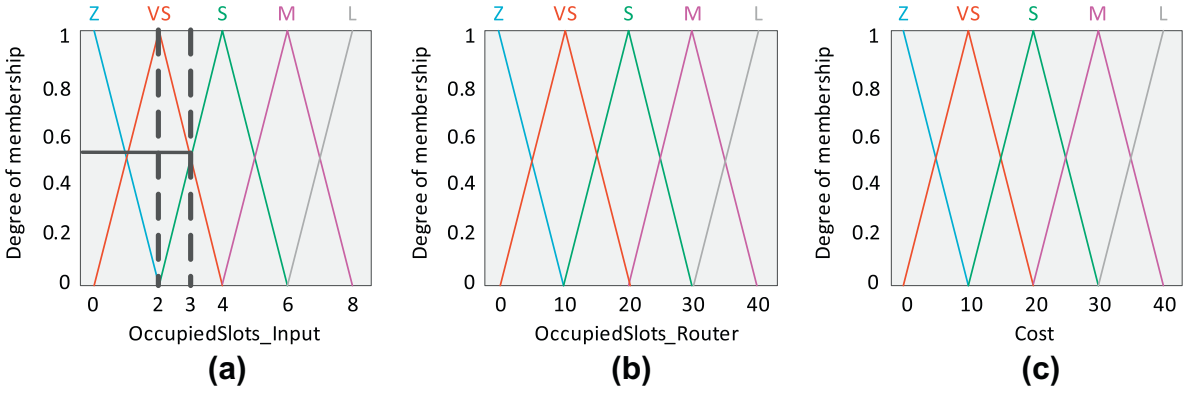

Fig. 7. (a) *OccupiedSlots_Input* (b) *OccupiedSlots_Router* (c) *Cost* membership functions.

Based on these combinations, four following rules are fired according to Table 1:

Table 1
FRA inference rules.

Rules		OccupiedSlots_Router				
		Z	VS	S	M	L
OccupiedSlots_Input	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

- Rule1 (Fig. 9(a)): IF (*OccupiedSlots_Input* is S) AND (*OccupiedSlots_Router* is VS) THEN (*Cost* is VS).
- Rule2 (Fig. 9(b)): IF (*OccupiedSlots_Input* is S) AND (*OccupiedSlots_Router* is S) THEN (*Cost* is S).
- Rule3 (Fig. 9(c)): IF (*OccupiedSlots_Input* is M) AND (*OccupiedSlots_Router* is VS) THEN (*Cost* is S).
- Rule4 (Fig. 9(d)): IF (*OccupiedSlots_Input* is M) AND (*OccupiedSlots_Router* is S) THEN (*Cost* is M).

4.2.3. Composition and defuzzification

Defuzzification is the process of producing a quantifiable result in fuzzy logic, and converting the fuzzy control action into a crisp value. The outputs of all rules should be aggregated and converted into a single output. Two methods for defuzzification are widely used:

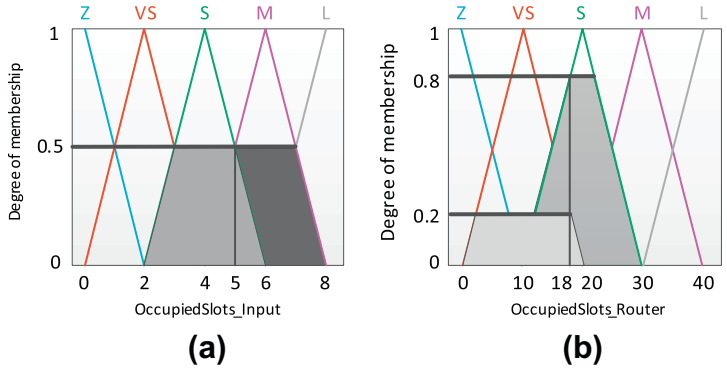

Fig. 8. (a) *OccupiedSlots_Input* (b) *OccupiedSlots_Router* as a part of two membership functions.

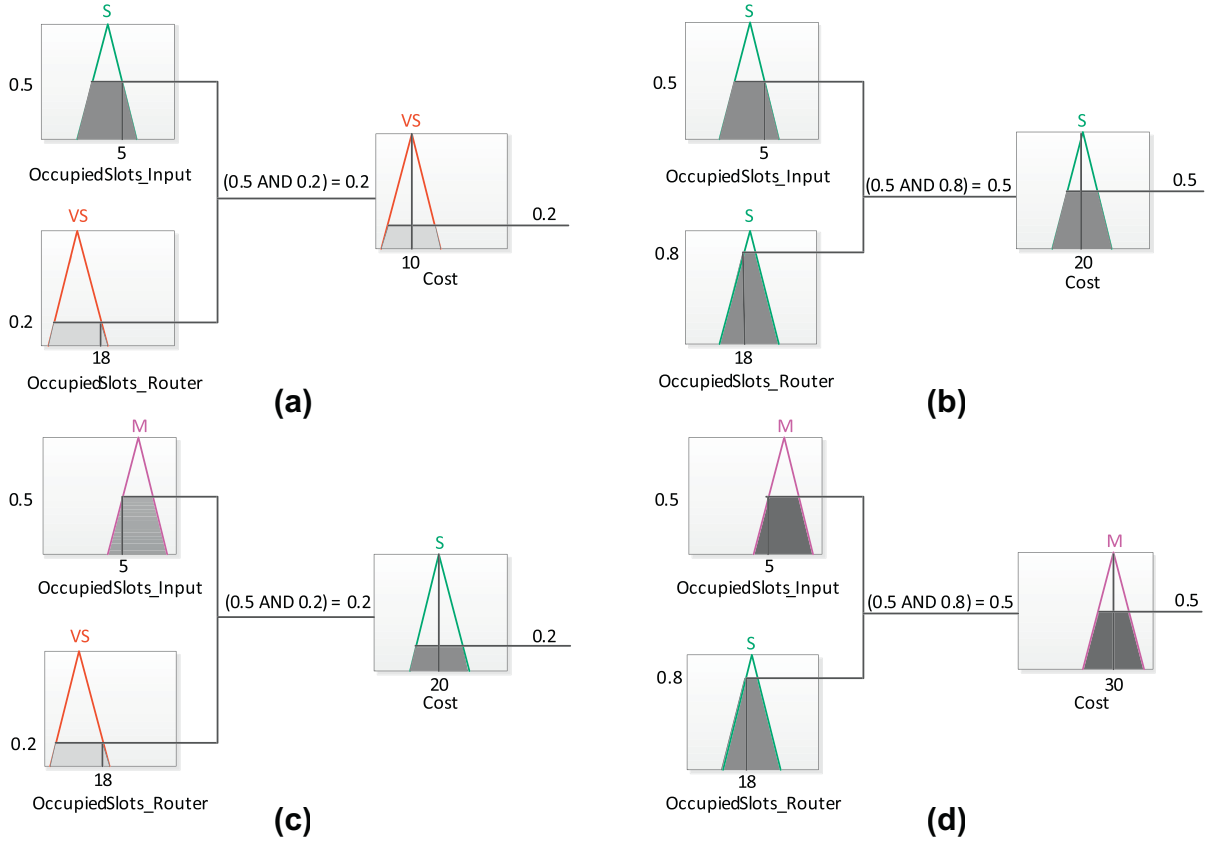

Fig. 9. *Cost* for (a) rule1 (b) rule2 (c) rule3 (d) rule4.

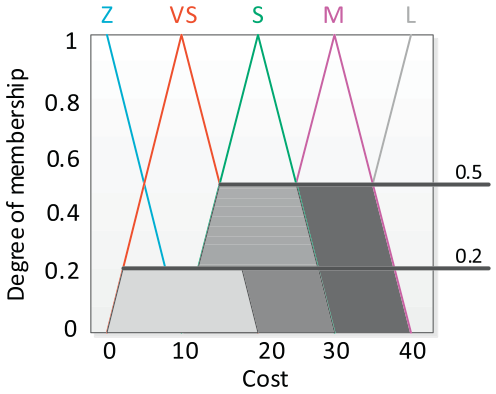

Fig. 10. Composition of the *Cost* membership function of all rules and producing a crisp value using CoG method.

1. The Center-of-Gravity method (CoG). This method finds the geometrical center. It favors the rule with the output of the greatest area.
2. The Mean-of-Maxima method (MoM). This method finds the value which has the maximum membership degree according to the fuzzy membership function.

MoM is simpler but it loses useful information while CoG is the commonly used method as it is more efficient. In this paper, the CoG defuzzification method is used to produce a crisp value.

In the defuzzification stage, the four obtained cost values (Fig. 9) are combined together and by using the Center-of-Gravity method, a single cost value is extracted. As shown in Fig. 10, the fuzzy outputs of the same cost membership function are summed together while the values in different membership functions are united (i.e. the maximum value is considered).

In this case, the cost value can be calculated from the following formula:

$$\begin{aligned} Obtained\ Cost_{node9} &= \frac{ObtainedCosts}{degree\ of\ membership\ functions} \\ &= \frac{(10*0.2)+(20*0.5)+(20*0.2)+(30*0.5)}{0.2+0.5+0.2+0.5} \\ &\approx 22 \end{aligned}$$

According to this formula, the degree of membership function of each rule is multiplied to the cost value associated with the maximum value in the membership function and then divided by the sum of all of the degrees of membership functions.

This procedure is done for both candidate directions and a packet is delivered in a direction with a smaller *Cost* value. Although in this work, the congestion metric parameters of the number of occupied slots (*OccupiedSlots_Input*) and congestion level of a router (*OccupiedSlots_Router*) are used, the proposed approach is generic and can be easily extended to different routing metrics.

Now, let us employ the proposed fuzzy logic system in the example of Fig. 5(b) where a packet should be delivered either through the router 9 or 6. The conditions of these routers are as follows:

$$node9 = \left\{ \begin{array}{l} FreeSlots_Input = 5 \\ FreeSlots_Router = 27 \end{array} \right\}$$

$$node6 = \left\{ \begin{array}{l} FreeSlots_Input = 3 \\ FreeSlots_Router = 26 \end{array} \right\}$$

We first measure the cost of selecting the router 9 and then the router 6. Considering the router 9, the degree of membership function for the input parameter *OccupiedSlots_Input* is shown in Fig. 11(a) and the input parameter *OccupiedSlots_Router* is shown in Fig. 11(b). According to these figures, the degree of membership in four triangles is non-zero: S and M for the input parameter *OccupiedSlots_Input* and S and M for the input parameter *OccupiedSlots_Router*. The combinations of these membership functions result in a new membership function called *Cost*. The type of the membership function (*Cost*) is extracted from Table 1 while the degree of membership function is achieved by using an AND operator. The obtained *Cost* and the degree of membership are shown in Table 2. Finally, the cost of the router 9 is calculated by:

$$\begin{aligned} ObtainedCost_{node9} &= \frac{ObtainedCosts\ Values}{degree\ of\ membershipfunctions} \\ &= \frac{(20*0.3)+(30*0.5)+(30*0.3)+(40*0.5)}{0.3+0.5+0.3+0.5} \\ &\approx 31 \end{aligned}$$

Similarly, the degree of membership function for each input parameter at router 6 is shown in Fig. 12.

The information on the cost membership function is listed in Table 3. Finally, the cost of the router 6 is measured as follow:

$$\begin{aligned} ObtainedCost_{node6} &= \frac{ObtainedCosts}{degree\ of\ membership\ functions} \\ &= \frac{(10*0.4)+(20*0.5)+(20*0.4)+(30*0.5)}{0.4+0.5+0.4+0.5} \\ &\approx 20 \end{aligned}$$

Now, the cost of sending a packet to the router 9 or 6 is 31 and 20, respectively, so the packet is sent toward the destination via the router 6 which is less congested. This choice is reasonable as the router 9 has more occupied buffer slots in the input buffer than the router 6 while the overall congestion conditions of both routers

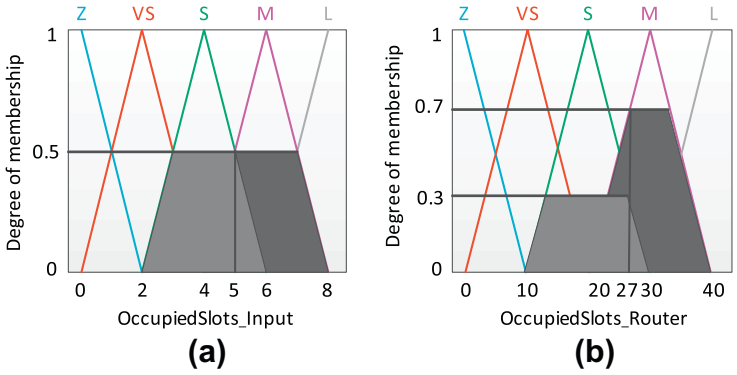

Fig. 11. The degree of membership function for the input parameters at router 9.

Table 2
The cost membership function of the router 9; MF stands for the membership function.

OccupiedSlots_Input	Degree of MF	OccupiedSlots_Router	Degree of MF	Cost	Maxvalue of MF	Degree of MF
S	0.5	S	0.3	Rule(S,S) = S	20	(0.5 AND 0.3) = 0.3
S	0.5	M	0.7	Rule(S,M) = M	30	(0.5 AND 0.7) = 0.5
M	0.5	S	0.3	Rule(M,S) = M	30	(0.5 AND 0.3) = 0.3
M	0.5	M	0.7	Rule(M,M) = L	40	(0.5 AND 0.7) = 0.5

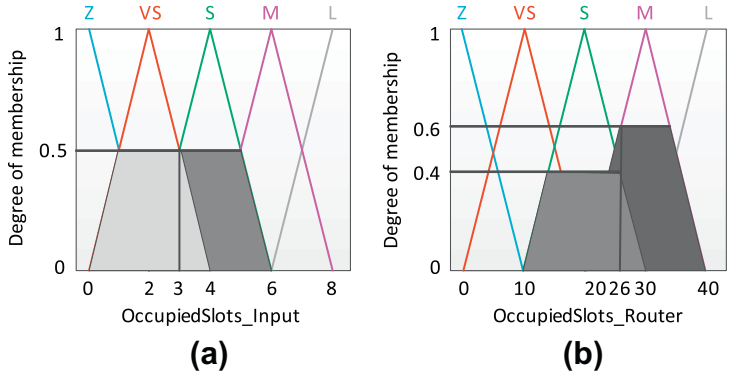

Fig. 12. The degree of membership function for the input parameters at router 6.

Table 3
The cost membership function of the router 6.

OccupiedSlots_Input	Degree of MF	OccupiedSlots_Router	Degree of MF	Cost	Maxvalue of MF	Degree of MF
VS	0.5	S	0.4	Rule(VS,S) = VS	10	(0.5 AND 0.4) = 0.4
VS	0.5	M	0.6	Rule(VS,M) = S	20	(0.5 AND 0.6) = 0.5
S	0.5	S	0.4	Rule(S,S) = S	20	(0.5 AND 0.4) = 0.4
S	0.5	M	0.6	Rule(S,M) = M	30	(0.5 AND 0.6) = 0.5

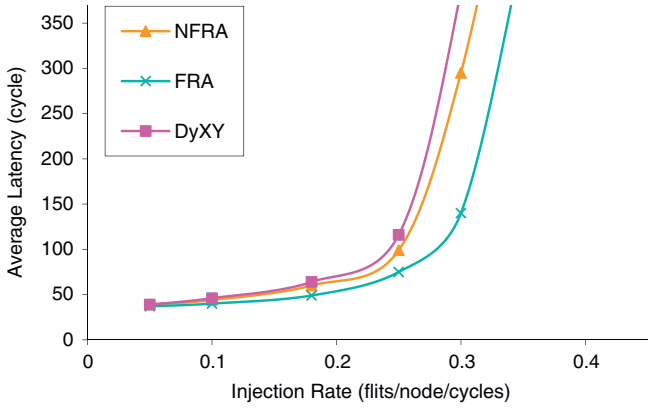

Fig. 13. Performance in an 8 × 8 mesh network under the uniform traffic profile.

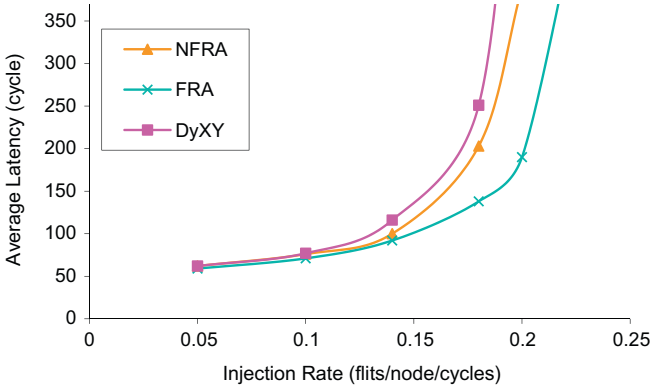

Fig. 14. Performance in an 8 × 8 mesh network under hotspot traffic profile with $H = 10$.

are similar. If the final result is not satisfactory (e.g. the router 6 has a higher cost value than the router 9), it means that the fuzzy rules are not well defined and should be modified.

5. Experimental results

To assess the efficiency of the proposed adaptive routing algorithms, NFRA and FRA, we compare them with DyXY under two types of traffic: synthetic and multimedia. We have developed a synthesizable NoC simulator implemented in VHDL to evaluate the efficiency of NFRA and FRA. This simulator is based on wormhole switching in a two-dimensional mesh configuration. The simulator inputs include the array size, the routing algorithm, the link width, the buffer size, and the traffic type. For all routers, the data width was set to 32 bits, and each input port has a buffer (FIFO) size of 8 flits. For the performance metric, we use the latency defined as the number of cycles between the initiation of the message and the time when the tail of the message reaches the destination.

The delay of computing resources in a router, including the fuzzy system, has been considered in the results. In simulation, it is

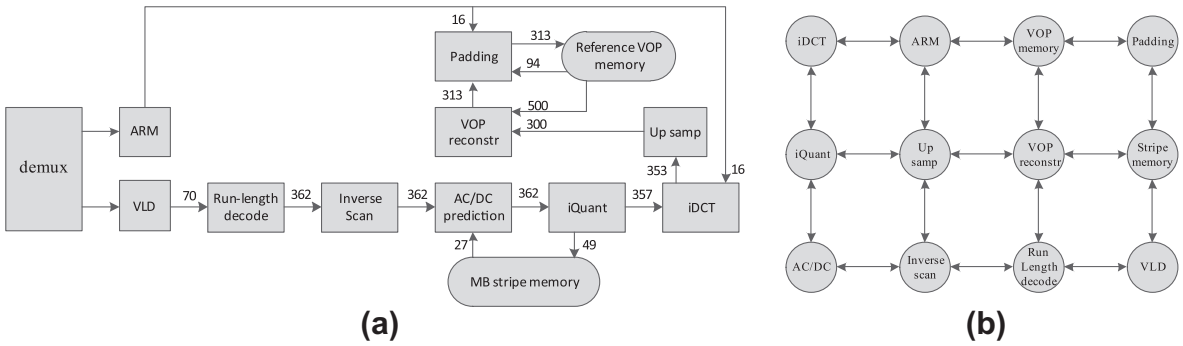

Fig. 15. VOPD block diagram, with communication BW annotated (in MB/s) and (b) its mapping onto a mesh topology [33].

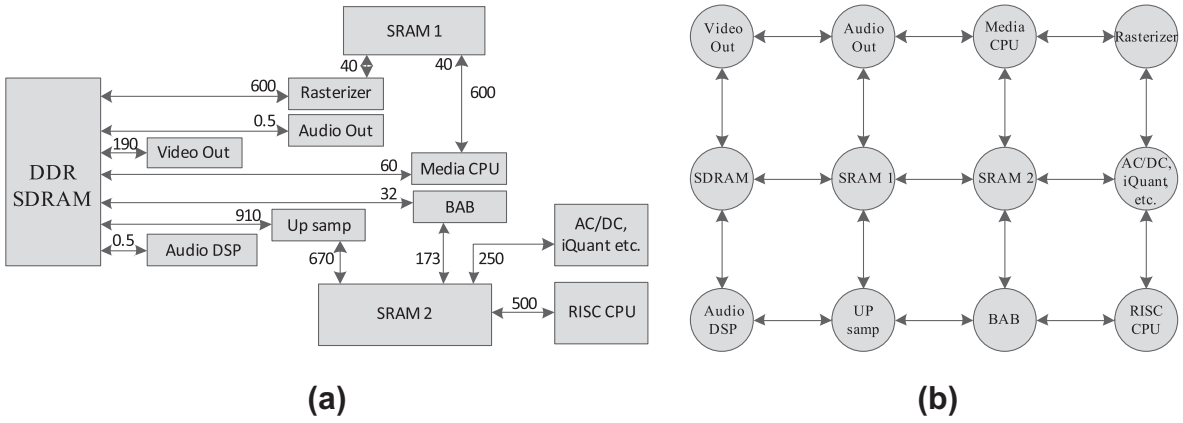

Fig. 16. (a) MPEG4 decoder block diagram, with communication BW annotated (in MB/s) and (b) its mapping onto a mesh topology [33].

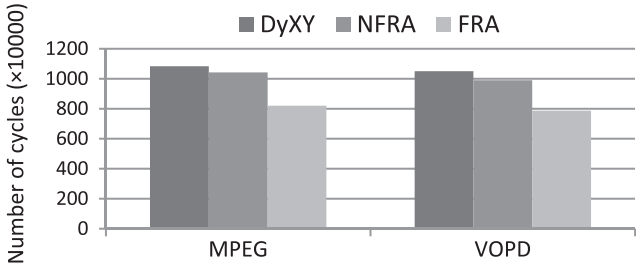

Fig. 17. Simulation results under two multimedia traffic profiles: MPEG and VOPD.

assumed that the links between routers have the same transmission bandwidth and length. This assumption is logical because the propagation delay of a traffic flow in the high performance communication is normally very small in comparison with its queuing delay in the routing nodes. The proposed schemes are evaluated for various traffic loads in an 8×8 mesh network. The packet size is uniformly distributed between 1 and 10 flits.

5.1. Performance evaluation under uniform traffic profile

In the uniform traffic model, each PE sends a packet to any other core with equal probability. As illustrated in Fig. 13, NFRA has better performance than DyXY while FRA performs the best under the uniform traffic profile. This performance improvement of NFRA is due to the fact that it makes a better local decision than DyXY. On other hand, FRA performs the best as the routing decision based on FRA leads of a better distribution of packets over the network.

Table 4
Hardware implementation details.

Network platforms	Area (mm^2)	Power (mW)
DyXY	2.563	1.355
NFRA	2.565	1.274
FRA	2.594	1.303

5.2. Performance evaluation under hotspot traffic profile

In the hotspot traffic model, a PE receives an extra portion (H) of traffic more than the other switches (here we assume the switch (4.4) receives $H = 10\%$ more traffic). As illustrated in Fig. 14, FRA performs the best and then NFRA and DyXY, respectively. This performance improvement of FRA over NFRA and DyXY shows that employing fuzzy logic mechanism results in a better routing decision which in turn reduces the average packets latency.

5.3. Multimedia traffic

NFRA and FRA are also evaluated under two realistic case studies mapped onto a 3×4 mesh topology. We selected two different video processing applications: Video Object Plane Decoder (VOPD) and MPEG4 decoder [33]. Figs. 15 and 16 depict the VOPD and MPEG4 Decoder block diagrams mapped onto 3×4 mesh topologies, respectively. As shown in Fig. 17, FRA decreases the latency considerably where the performance gain is up to 24% and 25% under the MPEG and VOPD traffic profiles, respectively, compared

with the DyXY routing algorithm. The performance gain of NFRA over the DyXY method is around 4% and 6% under the MPEG and VOPD traffic profiles, respectively.

5.4. Hardware overhead

For appraising the area overhead of the switch utilizing the proposed fuzzy logic, each scheme was synthesized by Synopsys Design Compiler using the TSMC 65 nm technology with an operating point of 500 MHz and supply voltage of 1 V. We perform place-and-route, using Cadence Encounter, to have precise power and area estimations. The power dissipation of each scheme is calculated under the uniform traffic profile near the saturation point using Synopsys PrimePower in an 8×8 mesh network. The layout area and power consumption of each platform are shown in Table 4. Comparing the area cost of the switches using DyXY and NFRA and the one employing FRA indicates that the hardware overhead of implementing a switch using the fuzzy logic is larger. However, the FRA consumes less power than two other approaches due to a better traffic distribution over the network.

6. Conclusion

In this paper, at first we proposed a non-fuzzy routing algorithm (NFRA) which overcomes DyXY, the conventional routing algorithm. In the DyXY routing algorithm, based on the number of occupied buffer slots along two minimal directions, a less congested direction is selected for delivering a packet. In NFRA, not only the instant queue length of the corresponding input buffer is considered but also the congestion level of the router is taken into account. Therefore, when buffer statuses are nearly the same in two directions, the congestion levels of the routers are compared together and a packet is sent to a router with a lower congestion level. Experimental results show that NFRA leads to a better performance compared with DyXY. Although NFRA can alleviate the congestion over the network, it still suffers from making non-optimal decisions in some conditions because of considering rigid boundaries on input parameters. To address this issue, we proposed a fuzzy-based routing algorithm (FRA) for on-chip networks employing fuzzy logic mechanism. The number of occupied buffer slots in the input buffer of the neighboring router and the congestion level of the neighboring router are chosen as two input parameters of the fuzzy controller while the cost is the output of the fuzzy controller. We have evaluated the proposed routing algorithm under both synthetic and real application traffic profiles. The results reveal that FRA improves the performance significantly compared with non-fuzzy methods.

Acknowledgment

The first author wishes to acknowledge Nokia Foundation for the partial financial support during the course of this research.

References

[1] A. Jantsch, H. Tenhunen, Networks on Chip, Springer, 2003.
[2] G.D.M. Luca Benini, Networks on chips: a new SoC paradigm, IEEE Computer 35 (1) (2002) 70–78.
[3] M. Daneshtalab, M. Kamali, M. Ebrahimi, S. Mohammadi, A. Afzali-Kusha, J. Plosila, Adaptive input–output selection based on-chip router architecture, Journal of Low Power Electronics 8 (1) (2012) 11–29.
[4] J. Xu, W. Wolf, J. Henkel, S. Chakradhar, A methodology for design, modeling, and analysis of networks-on-chip, in: Proceedings of IEEE International Symposium on Circuits and Systems, 2005, pp. 1778–1781.
[5] W. Dally, B. Towles, Principles and Practices of Interconnection Networks, Morgan Kaufmann Publishers, 2003.
[6] M. Daneshtalab, M. Ebrahimi, P. Liljeberg, J. Plosila, H. Tenhunen, Memory-efficient on-chip network with adaptive interfaces, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 31 (1) (2012) 146–159.
[7] M. Daneshtalab, M. Ebrahimi, J. Plosila, GLB – efficient global load balancing method for moderating congestion in on-chip networks, in: Proceeding of 7th International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2012, pp. 1–5.
[8] P. Gratz, B. Grot, S. W. Keckler, Regional congestion awareness for load balance in networks-on-chip, in: Proceedings of IEEE 14th International Symposium on High Performance Computer Architecture, HPCA, 2008, pp. 203–214.
[9] M. Ebrahimi, M. Daneshtalab, J. Plosila, F. Mehdipour, MD: minimal path-based fault-tolerant routing in on-chip networks, in: Proceeding of 18th Asia and South Pacific Design Automation Conference (ASP-DAC), Japan, 2013.
[10] M. Ebrahimi, M. Daneshtalab, J. Plosila, H. Tenhunen, MAFA: adaptive fault-tolerant routing algorithm for networks-on-chip, in: Proceeding of 15th Euromicro Conference on Digital System Design (DSD), 2012, pp. 201–207.
[11] J. Duato, L. Ni, Interconnection Networks, Morgan Kaufmann, 2003.
[12] M. Ebrahimi, M. Daneshtalab, P. Liljeberg, H. Tenhunen, HAMUM – a novel routing protocol for unicast and multicast traffic in MPSoCs, in: Proceeding of 18th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2010, pp. 525–532.
[13] M. Ebrahimi, M. Daneshtalab, F. Farahnakian, J. Plosila, P. Liljeberg, M. Palesi, H. Tenhunen, HARAQ: congestion-aware learning model for highly adaptive routing algorithm in on-chip networks, in: Proceeding of 6th IEEE/ACM International Symposium on Networks on Chip (NoCS), 2012, pp. 19–26.
[14] M. Dehyadegari, M. Daneshtalab, M. Ebrahimi, J. Plosila, S. Mohammadi, An adaptive fuzzy logic-based routing algorithm for networks-on-chip, in: Proceeding of NASA/ESA Conference on Adaptive Hardware and Systems (AHS), 2011, pp. 208–214.
[15] A. Chandramohan, M.V.C. Rao, M.S. Arumugam, Two new and useful defuzzification methods based on root mean square value, Journal of Soft Computing 10 (11) (2006) 1047–1059.
[16] E. Nilsson, M. Millberg, J. Oberg, A. Jantsch, Load distribution with the proximity congestion awareness in a network on chip, in: Proceedings of Design, Automation and Test in Europe Conference and Exhibition, 2003, pp. 1126–1127.
[17] J. Hu, R. Marculescu, DyAD – smart routing for networks-on-chip, in: Proceedings of ACM/IEEE Design Automation Conference, 2004, pp. 260–263.
[18] M. Li, Q.-A. Zeng, W.-B. Jone, DyXY – a proximity congestion-aware deadlock-free dynamic routing method for network on chip, in: Proceedings of 43rd ACM/IEEE Design Automation Conference, 2006, pp. 849–852.
[19] P. Lotfi-Kamran, A.M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, Z. Navabi, EDXY – a low cost congestion-aware routing algorithm for network-on-chips, Journal of Systems Architecture 56 (7) (2010) 256–264.
[20] P. Lotfi-Kamran, M. Daneshtalab, C. Lucas, Z. Navabi, BARP-a dynamic routing protocol for balanced distribution of traffic in NoCs, in: Proceedings of Design, Automation and Test in Europe (DATE), 2008, pp. 1408–1413.
[21] S. Rea, D. Pesch, Multi-metric routing decisions for ad hoc networks using fuzzy logic, in: Proceedings of 1st International Symposium on Wireless Communication Systems, 2004, pp. 403–407.
[22] M.M. Thaw, Fuzzy-based multi-constrained quality of service distance vector routing protocol in mobile ad-hoc networks, in: Proceedings of the 2nd International Conference on Computer and Automation Engineering (ICCAE), vol. 3, 2010, pp. 429–433.
[23] C.G.B. Sun, Fuzzy controller based QoS routing algorithm with a multiclass scheme for MANET, International Journal of Computers, Communications & Control 4 (2009) 427–438.
[24] A. Pasupuleti, A.V. Mathew, N. Shenoy, S.A. Dianat, Fuzzy system for adaptive network routing, Digital Wireless Communications 4740 (2002) 189–196.
[25] E. Aboelela, C. Douligeris, Fuzzy inference system for QoS routing in B-ISDN, in: Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering, vol. 1, 1998, pp. 141–144.
[26] M. Fattah, M. Daneshtalab, P. Liljeberg, J. Plosila, Transport layer aware design of network interface in many-core systems, in: Proceeding of 7th International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2012, pp. 1–7.
[27] J. Duato, A new theory of deadlock-free adaptive multicast routing in wormhole networks, IEEE Transactions on Parallel and Distributed Systems 6 (1993) 976–987.
[28] G.-M. Chiu, The odd-even turn model for adaptive routing, IEEE Transactions on Parallel and Distributed Systems 11 (7) (2000) 729–738.
[29] C.J. Glass, L.M. Ni, The turn model for adaptive routing, in: Proceedings of the 19th, Annual International Symposium on Computer Architecture, 1992, pp. 278–287.
[30] L.M. Ni, P.K. McKinley, A survey of wormhole routing techniques in direct networks, Computer 26 (2) (1993) 62–76.
[31] J.S.R. Jang, N. Gully, MATLAB Fuzzy Logic Toolbox: User's Guide. MathWorks, 1997.
[32] W. Pedrycz, Why triangular membership functions?, Fuzzy Sets and Systems 64 (1) (1994) 21–30
[33] E.B.V.D. Tol, E.G.T. Jaspers, Mapping of MPEG-4 decoding on a flexible architecture platform, in: Proceedings of Media Processors, 2002, pp. 1–13.

Masoumeh Ebrahimi received her B.S. degree in computer engineering from School of Electrical and Computer Engineering, University of Tehran in 2005, and M.S. degree in computer architecture from Azad University, Science and research branch, in 2009. Since spring 2009 she has been working in the Embedded Computer Systems laboratory, University of Turku. She has expertise in interconnection networks, networks-on-chip, 3D integrated systems, and systems-on-chip. Her PhD thesis is focused on routing protocols in 2D and 3D NoCs. Masoumeh is a member of IEEE and has published more than 30 international refereed journals and conference papers.

Hannu Tenhunen received his PhD from Cornell University, Ithaca, USA in 1985 and since that he has held professor, invited professor, or honorary professor positions in Tampere, Stockholm, Ithaca, Grenoble, Shanghai, Beijing and Hong Kong. During the recent years he has been director of Turku Centre of Computer Science and invited professor at University of Turku where he has established Computer Systems Laboratory, the leading computer architecture and systems research centre in Finland. Prof. Tenhunen's research interest is in new computational architectures, dependability issues, on-chip and off-chip communication and mixed signal and interference issues in complex electronic systems including 3-dimensional integration. He has done over 600 publications or invited key note talks internationally.

Masoud Dehyadegari received his M.Sc. degrees in computer engineering from University of Tehran in 2006. He is currently studying towards a Ph.D. degree in the same university, as a member of the Dependable System Design Lab. His research interests include Low-Power Design, Network-on-Chips, Multi-Processor System-on-Chip.