Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issue 2, pp. 309–321, DOI 10.12694:/scpe.v21i2.1690

NOVEL METRIC FOR LOAD BALANCE AND CONGESTION REDUCING IN NETWORK ON-CHIP

ABDELKADER AROUI[*], ABOU ELHASSAN BENYAMINA[†], PIERRE BOULET[‡], KAMEL BENHAOUA[§], AND AMIT KUMAR SINGH[¶]

Abstract. The Network-on-Chip (NoC) is an alternative pattern that is considered as an emerging technology for distributed embedded systems. The traditional use of multi-cores in computing increase the calculation performance; but affect the network communication causing congestion on nodes which therefore decrease the global performance of the NoC. To alleviate this problematic phenomenon, several strategies were implemented, to reduce or prevent the occurrence of congestion, such as network status metrics, new routing algorithm, packets injection control, and switching strategies. In this paper, we carried out a study on congestion in a 2D mesh network, through various detailed simulations. Our focus was on the most used congestion metrics in NoC. According to our experiments and performed simulations under different traffic scenarios, we found that these metrics are less representative, less significant and yet they do not give a true overview of reading within the NoC nodes at a given cycle. Our study shows that the use of other complementary information regarding the state of nodes and network traffic flow in the design of a novel metric, can really improve the results. In this paper, we put forward a novel metric that takes into account the overall operating state of a router in the design of adaptive XY routing algorithm, aiming to improve routing decisions and network performance. We compare the throughput, latency, resource utilization, and congestion occurrence of proposed metric to three published metrics on two specific traffic patterns in a varied packets injection rate. Our results indicate that our novel metric-based adaptive XY routing has overcome congestion and significantly improve resource utilization through load balancing; achieving an average improvement rate up to 40 % compared to adaptive XY routing based on the previous congestion metrics.

Key words: Embedded Systems, Network on-Chip, Routing, Load balancing, Congestion reducing.

AMS subject classifications. 68M10, 68M12, 68M14

1. Introduction.

The Network-on-Chip (NoC) has proved to be more reliable, modular and reusable [4] solutions than shared buses. NoC is proposed as an alternative interconnection solution in mutli-processors on chip (MPSoC).As it handles the high throughput and latency communication demands amongst various on-chip tiles, it connects cores, caches and memory controllers using packet switching routers, covering both regular and irregular topologies [17, 10].

Routing algorithm effectively facilitate communication and routing packets between the cores. Although this algorithm decides how to route incoming packets to various destinations, network congestion may occur due to low bisection bandwidth and poor routing. In fact, because of non-uniformity traffic, some network nodes must send more packets than others causing a rise in congestion and affecting network performance significantly i.e. network performance is highly dependent on the routing algorithm.

Routing in netowork on chip can be generally classified into oblivious and adaptive [24]. In oblivious routing, the packets are routed without any information about the traffic levels of the network, whereas in adaptive routing algorithm, the routing decision is made by taking into consideration the current congestion status of the network.

The process of an adaptive routing algorithm is also divided into two phases: the routing function and the selection function. The routing function defines the set of available output channels for a packet dependent on the source and destination positions; whereas the selection function selects an output channel from the set of

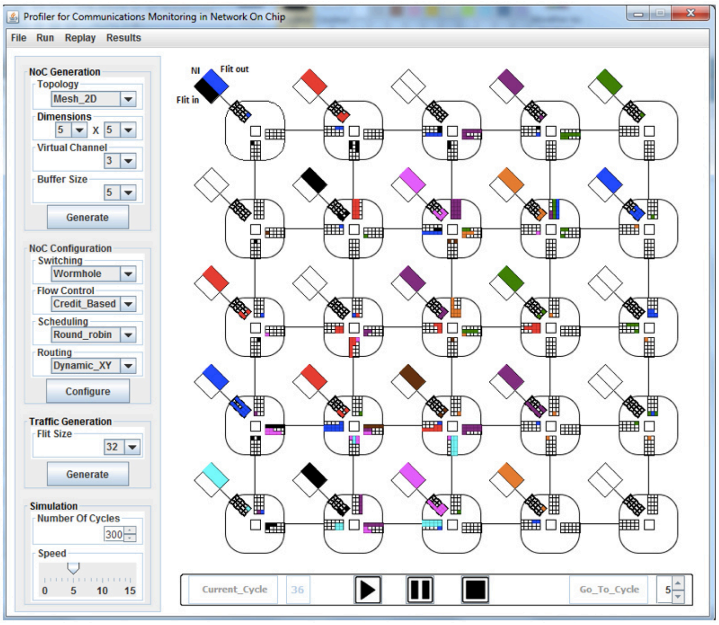
[*] Computer Science Department, University of Oran 1 Ahmed Ben Bella, Algeria (aroui.abdelkader@edu.univ-oran1.dz).

[†] Computer Science Department, University of Oran 1 Ahmed Ben Bella, Algeria. (benyamina.lahssan@univ-oran1.dz).

[‡] CNRS, University Lille, Centrale Lille, IMT Lille Douai, UMR 9189-CRISTAL, Lille, France (pierre.boulet@univ-lille1.fr).

[§] Computer Science Department, University of Mustapha Stambouli, Mascara, Algeria. (kbenhaoua@gmail.com).

[¶] School of Computer Science and Electronic Engineering, University of Essex, UK. (a.k.singh@essex.ac.uk).



NoC Generation
Topology: Mesh_2D
Dimensions: 5 X 5
Virtual Channel: 3
Buffer Size: 5

NoC Configuration
Switching: Wormhole
Flow Control: Credit_Based
Scheduling: Round_robin
Routing: Dynamic_XY

Traffic Generation
Flit Size: 32

Simulation
Number Of Cycles: 300
Speed

Current_Cycle 36
Go_To_Cycle 5

FIG. 1.1. *Our Graphical simulator for Data-Flow Monitoring on Network on-Chip [23]*

available output channels.

Adaptive routing algorithms benefit from reachable metrics of a router to evaluate congestion level of a network. Various metrics have been considered in estimating the congestion level in NoC.

In this paper, we study a subset of the most commonly used metrics in the NoCs congestion domain. After a thorough graphic analysis at each cycle of the execution of some small scenarios on our simulator cf. Fig. 1.1, it appeared that the underestimation of congestion information in some scenarios force routing to congested areas, and random routing decisions are taking place when the candidate output ports have the same congestion value. The indeterminism of some congestion-based routing algorithms and the representativeness of existing congestion metrics are the two issues that motivated our study in this paper.

A novel metric-based approach is proposed and evaluated for 2D mesh topology through simulations using various workloads. The experimental results reveal that the proposed mechanism results in better performances compared to existing techniques.

The rest of this paper is structured as follows: Section 2 describes existing work on congestion treatment. Section 3 gives a general overview about the set-up of our Network-on-Chip architecture. Our proposed technique is presented and illustrated in Sect. 4. Some experiences and analyses are found in Sect. 5 and finally in Sect. 6 we recapitulate and propose some of the future works and perspectives.

2. Related Work. Multi-processor system-on-Chip (MPSoC) has emerged as a positive solution to address the increased computational requirements of modern and future applications [3]. This recent breakthrough of MPSoC which consist multiple processing elements (PEs) in the same chip, thanks to the evolution of semiconductor technology, allow us to integrate several elements in the same chip. MPSoC also provide increased parallelism towards achieving high performance [2]. The NoC has been introduced as a power efficient and scalable communication infrastructure between processors.

Although a large variety of NoC topologies have been presented in the literature, 2D mesh is the most widely studied. Due to spatio-temporal irregularity of network traffic, particularly in 2D mesh topology, some network nodes have to dispatch much more packets than others. Congestion will take place at those nodes and significantly affects network performance [32].

If the congestion arises several times, the concerned resources will be getting additional higher temperature (hotspot formation). This phenomenon reduces the effective rate of the NoC, or in the worst case, can cause blocking of traffic and failure of certain network resources. For this purpose, a huge number of approaches were proposed to surmount the negative effect of congestion and improve the network performance: reconfigurable routers and novel NoC architectures were designed to avoid router and network congestion; congestion metrics were included to complete the design of adaptive routing algorithms intended for congestion reduction [14, 6, 28]; End-to-end flow control technique also was suggested to adjust injecting packets into any emerging network congestion [30, 26, 27]; thus selection strategies were developed to reduce congestion and improve network performance with regards to latency [1, 8, 15].

Routing algorithm [18, 20, 21] is an important part in NoC design, that has a significant impact on the traffic flow and performance improvement of NoC. In adaptive routing, a proper selection strategy is needed to choose the appropriate path for packets to reach their destinations as fast as possible. Several efforts have shown that adaptive routing can reduce network congestion by conducting traffic away from the congested regions. Congestion-aware adaptive routing algorithms were merely designed to select the least congested route to produce the load balance in the network by considering the current congestion status of the network in their routing decisions.

A specific network for congestion data transmission has been added to the proposed NoC architecture [22]. The mechanism that aggregates and transmits metrics of congestion beyond direct neighbors throughout the monitoring network which has been proposed by Gratz and al, it was labelled as Regional Congestion Awareness algorithm (RCA) [9]. It was proved in [29], that destination based adaptive routing algorithms (DAR and DBAR) had provided better performances than regional adaptive routing techniques. The Gratz's work was improved, not long ago, by using a Global Congestion Awareness (GCA) mechanism [28].

Adaptive routing based on Ant Colony Optimization has been presented [13] to exploit traffic historical information to reach load balancing. The inconvenient of this technique was that the pheromone table grows fast with the scaling of NoC and thus the storage cost was too high. This problem was solved and technique was improved by proposing a Regional ACO-Based Cascaded Adaptive Routing [5] with static and dynamic regional table forming technique to decrease the cost of the table storage. In order to reduce NoC area, power and overall packet latency, two routing mechanisms for congestion handling have been presented in [11].The mechanisms use σ bits to capture the congestion information of the links of the network for the node with n-hop visibility at each node. A new methodology of Congestion Aware Adaptive Routing (CAAR) was designed [16], to improve NoC latency and throughput during high congestion by prioritizing packets that suffer higher latency while travelling long distances. A congestion detection mechanism has been proposed in [12], which is capable of locating where the congestion is in the network within several cycles and to change the routing algorithm of congested nodes in the cycle following the congestion detection.

By monitoring and exchanging some network resources status information, we can, therefore, predict the occurrence of congestion in the network. According to [31], congestion may occur at a network node great possibility when more than a quarter of its buffer space has been occupied. The information exchanged in the network and between neighboring nodes is called metric. Various metrics have been used in the literature for estimating congestion level of network resources. The number of free VCs [16], the number of free buffer slots in the downstream router [33], and the active demand that each output port experiences (i.e., crossbar demand) [9, 7] are among such congestion metrics. These metrics were very interested in evaluating the state of the desired downstream router input port and simultaneously they neglected the state of the entire entity (router) that contains this resource and other resources. However, taking into account the downstream router input port state as congestion metric to decide on the next hop is not an effective method, since the router state is the sum of states of resource components that router i.e. all ports and all virtual Channels state. Based on this assumption and through the simulations we have discovered that routing algorithms listed above have made some improvement in overcoming the negative effect of congestion and improving network performance.

Consequently, we managed to propose firstly, a novel congestion metric that estimates the overall state of the router and reflects exactly what is happening within the router. Secondly, a congestion-aware routing algorithm based on this metric was designed to significantly reduce the negative impact of congestion and its occurrences as well as improving NoC's performance.

3. NoC Architecture Oveview. In this work, 2D Mesh topology was chosen. It is the most used in the literature and the easiest to implement. Our platform is a set of processors elements (PEs) interconnected by

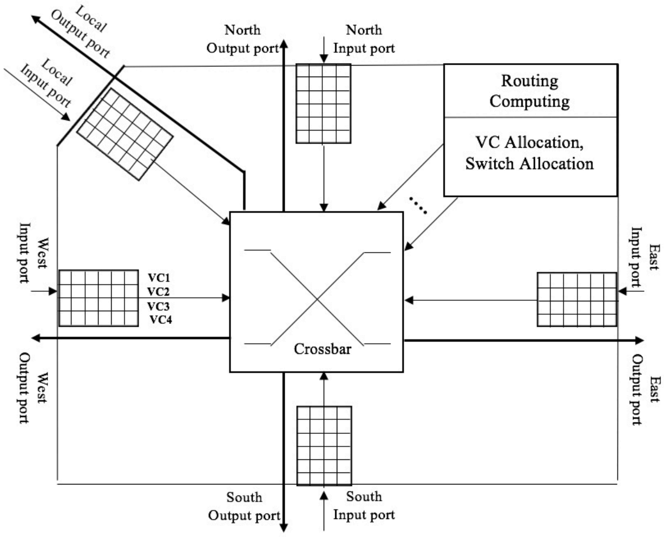


FIG. 3.1. *Architecture of mesh Router*

routers. The data flow exchanged between PEs passes by routers. A typical wormhole NoC router is shown in Fig. 3.1. Every router in our topology has five I/O ports to connect with its adjacent router and IP blocks: four connected to the neighboring routers and one for the local core (PE) through the Network Interface (NI). The input-ports contain the Virtual Channel (VC) buffers, and the output-ports are simple data buses.

The router architecture includes FIFOs (Buffers) for each input port, route computation unit, Virtual Channel (VC) allocation unit, crossbar control logic, and the crossbar. NI transforms data exchanged between cores on many packets then in to smaller data called flits. The flit is the smallest unit on which flow control can perform. A packet in this case is split into a header flit, body flit(s), and a tail flit. A flit enters into the router through one of the ports and got stored in its FIFO. The header flit indicates each start of a new packet, as soon as it is stored in its FIFO, linked to the routing unit which establishes the output port that the packet should follow. Once, this is done, the header flit tries to allocate a virtual channel for the next hop. If that is the case, it enters the switch arbitration stage, where it rivals for the output port with other flits from the other router input ports. Once the crossbar passage is granted, the flit traverses the switch and enters the channel. Following flits belonging to the same packet can proceed directly to the crossbar and lead to the output port [19]. The NoC routers use Round-Robin (RR) scheme for both VC and switch allocation due to its simplicity.

Many routing algorithms dealing with the mesh architecture networks have been proposed, XY routing algorithm is most suitable for networks which are implemented in this architecture. Two variants of this algorithm have been proposed, static and dynamic. In this paper, we opted for the dynamic

For a better distribution of the traffic on the network, the researchers proposed dynamic XY routing methods [18] which provide adaptiveness and ensure deadlock-free and live-lock-free routing at the same time. The adaptiveness lies in making routing decisions by monitoring network status in the proximity, and the deadlock-free and live lock-free features are incorporated by limiting a packet to traverse the network by using the shortest paths between the source and the destination [18].

Different coefficients have been used to complete the design of dynamic XY routing for congestion reduction in NoC, like the number of available VCs, the crossbar demand, and the number of free buffer slots at the port of corresponding entry in the downstream router.

Our approach in this paper is aimed to introduce a novel congestion metric into the design of our dynamic XY routing technique. More details of our contribution will be presented and illustrated in the next section.

4. Proposed technique. Starting with the assumption that load imbalance is one of the main critical issues in NoC, and that good network performance can be achieved by equally distributing traffic between nodes in the network. Since congestion occurs on loaded nodes which affect significantly network performance, it is viable to find the right way to spread the excessive load of a node to its neighbors and reduce the number of congestion in the network. In this context, congestion control scheme which consists of dynamic adaptive routing output port selection is aimed to balance global traffic distribution as well as to alleviate congestion caused by heavy network traffics.

As we argued in the previous section, several proposed works were addressed the problem of network congestion either to reduce or to predict it by proposing routing and selection strategies that were based on most used and approved metrics in literature. Many solutions and methods have been presented as a form of techniques on new architectures that facilitate network monitoring, exchange and exploitation of congestion information in traffic routing decisions.

We have observed that techniques have changed from one article to another however the metrics remained unchanged. A few techniques have been proposed against some metrics to alleviate the congestion phenomenon. This track has motivated us to make an analysis study through simulations on the three most commonly used metrics: Available VC, Demand Crossbar and Buffer occupancy.

By running some small scenarios on our graphical simulator cf. Fig. 1.1 and through step by step debugging option that offer our simulator, two important findings came in sight: 1) the underestimation of node congestion information in certain scenarios forces routing traffic to congested areas, 2) Random routing decisions are made when the candidate exit ports have the same congestion value. The indeterminism of some congestion-based routing algorithms and the non-representativeness of congestion information are the cause of our motivation to our study.

In our bi-objective approach, a novel mechanism for controlling and evaluating node congestion is integrated in the design of the routing algorithm, which allows efficient distribution of heavy packet traffics on the chip and minimizes the congestion occurrence. We introduced a novel metric which is composed of two modules, the first one is original and designed to estimate information and state of node congestion, and to take good decisions of routing; whereas the second one measures the load applied on nodes to better balance the traffic on networks.

We note: OutFlits: the number of Flits leaving router at given cycle correspond to the number of active output port; Candidate VCs: the number of active virtual channels which contains data ready to cross the router by one of its Output ports; the maximum number of output ports in our router architecture is equal to 5. Router congestion status and information evaluation is given by Eq (4.1):

$$Router\ status = \underbrace{\frac{OutFlits}{CandidatVCs}}_{\text{(a)}} \times \underbrace{\frac{OutFlits}{Outputports}}_{\text{(b)}} \tag{4.1}$$

The sub-equation (a) Shows the extent of contention between VCs. The sub-equation (b) gives an idea about what is exactly happening within router's output ports when many are being requested at once, at a given cycle. (b) completes (a) to form router congestion information. In order to make the best use of network resources, we need to properly distribute the traffic and to avoid any uncontrolled routing decision. In the case of the equalization of congestion information between several VCs candidates in contention, the router occupancy rate information is added to the congestion information to form our congestion metric which is given by Eq (4.2):

$$Our\ CM = Router\ status \times Router\ occupancy\ rate \tag{4.2}$$

where *Router occupancy rate* detail is given by Eq. (4.3):

$$Router\ occupancy\ rate = \frac{1}{V}\sum_{i=1}^{V} Buffer\ occupancy\ rate_i \tag{4.3}$$

where V represents the router virtual channels number and $Buffer\ occupancy\ rate_i$ is the occupancy rate of i-th buffer of router.

The division by 0 leads to some complicated situations for our CM. In order to avoid such situation and to keep the efficiency of our metric we have proposed algorithm 1, The CM computation algorithm.

Algorithm 1 Congestion Metric Computation.

```
for (Each RouterIndex) do
    if (Candidat_VCs[RouterIndex]==0) then
        NovelCongestionMetric[RouterIndex]= 1.0
    else
        Router status [RouterIndex]= $\text{OutFlits}^2$ / (Candidat VCs $\times$ Output ports)
        NovelCongestionMetric[routerindex] = Routerstatus[RouterIndex] $\times$ Occupancy_rate[RouterIndex]
    end if
end for
```

As discussed previously, to validate our proposal, we have opted for the XY dynamic algorithm as a routing technique for its simplicity and because it is the most implemented for 2D Mesh topology.

Our congestion control for routing decision making as detailed in Algorithm 2, depends on neighbor nodes congestion condition. it uses Eq. (4.2) for calculating congestion value in each node. To route packets, each node in network is responsible to get and compare the neighbors updated congestion values to select the output port that its congestion value is minimum.

Algorithm 2 Our congestion-based approach

```
for (Each winner Flit of node 's $CandidateVCs$ and each output port) do
    Read the destination of Flit ready to leave
    Compare addresses of the destination and the current router.
    if (the destination is the local core of the current router) then
        Prepare to send the Flit to the local core
    else
        if the destination has the same x or y address as the current router then
            Prepare to send the Flit to the neighboring router on the y-axis or x-axis towards the destination
        else
            Get and Compare the congestion values of neighboring ports leading to destination
            Prepare to send the Flit to the port with the least congestion value according to the destination
        end if
    end if
    Increment the number of leaving flits
    Update $CandidateVCs$ value
    Compute and update node congestion value according to algorithm (1)
    Send the Flit
end for
```

Our approach was designed to select the least congested route to minimize congestion occurrence and to produce the load balancing in the network by taking into account our proposed congestion metric to decide the next hop.

To demonstrate the most factors that adduce our approach in comparison to previous techniques, we put forward the following simulations to clarify and analyze the results and the performances of the approach.

5. Experiment. A NoC java-based simulator [23] is used in order to evaluate our strategy performance. Its graphical presentation offers the possibility to follow the execution details in real time, to know the real state of routers and to locate easily those who suffer from congestion. This simulator is accurate cycle, that means that one cycle is required to transmit one flit to the next router. For each cycle in application lifetime, our simulator starts by identifying candidate flits that request output ports, then running routing technique to determine the path to take, followed by an arbitration technique to obtain winning flits that cross the router continuing their route in network.

In this paper, we simulate 2D-mesh NoC platform that use a wormhole switching mechanism, round-robin

Table 5.1
Simulation setup.

Simulator	[23]
NoC Topology	7×7 2D Mesh
Virtual channel	Yes, 3 VCs per port
Buffers / packet size	5 / 5
Switching	Wormhole
Routing	Congestion-based adaptive routing
Arbitration	Round-robin
Congestion metrics	Available VCs, Buffer Occupancy Crossbar Demand, Our metric
Traffic distribution	Uniform, random (50 xml files)
Simulation Time	300;500;1000 clock cycles per xml file

arbitration and our proposed method as a routing technique. The inter-router transmission is based on the Credit-based flow. Each physical channel in router has a buffer of four flits where packet length is equal to buffer size. The simulation setup is summarized in Table 5.1.

In addition to our novel congestion metric we have chosen the three most common used metrics. We attentively study congestion by focusing on its occurrence in the network for each metric under different types of traffic. We examined uniform and random traffic scenarios. In uniform traffic, node (i, j) only generates packets to node $(2i, 2j)$. In random traffic, each node randomly generates packets to every other node with the same probability where a node (i, j) represents the core/router placed at a row of i and column j of a 2D mesh.

The injection of traffic into the network is regulated by a packet injection mechanism called PIR (Packets Injection Rate), i.e. a node that injects on average a packet into the network every 10 cycles, it has a PIR=0.1.

In our experiment, several runs were performed for 7 × 7 2D-Mesh. Simulations are performed under each traffic and for each congestion metric in varied PIR, to observe how congestion propagates in the network and to quantify the number of its occurrence . For each metric, different values of congestion occurrence are evaluated at different traffic scenarios and different injection rates.

In order to properly conduct the evaluation of our solution, calculated parameters were introduced, and monitored during our simulations i.e. **LU** that is the rate of usage and exploitation of the links in the network. This parameter gives us an indication about the number of resources involved in the distribution of the network traffic. A high rate of LU generates a good distribution load and viable network topology exploitation. To measure and to track the congestion evolution in the network, the number of congested nodes rate **C.N** and the congestion occurrence rate **C.O** were introduced.

To compare the network performance in implemented solutions, we also considered the average packet delay **Lat.** and the average throughput **Th.** as performance metrics, which are defined as follows:

$$Avg\ Packet\ Delay = \frac{1}{N}\sum_{i=1}^{N} latency_i \tag{5.1}$$

where N refers to the total number of received packets and $latency_i$ is the delay of the i-th packet.

$$Throughput = \frac{Total\ received\ flits}{number\ of\ nodes \times Total\ cycles} \tag{5.2}$$

where $Total\ received\ flits$ represent the total number of received flits in network, number of nodes is the number of network nodes, and total cycles is the simulation length in clock cycles.

The average throughput **Th** is usually represented as a measured rate by calculating the number of successfully delivered Flits during the simulation period. This rate is affected by network congestion and packet loss, in other word, reducing congestion and its occurrence in the network nodes is translated by a high throughput rate and a good traffic fluidity.

Table 5.2
Simulation results for simulation time = 300 clock cycles

a) Results for Random and Uniform traffic, for PIR=0.3

	Random Traffic					Uniform Traffic				
Metric	**Th**	**Lat**	**L.U**	**C.N**	**C.O**	**Th**	**Lat**	**L.U**	**C.N**	**C.O**
Available_VCs	0.25	0.94	44%	92%	17%	0.24	0.90	38%	92%	9%
Crossbar Demand	0.32	1.85	50%	92%	15%	0.29	1.47	43%	92%	8%
Buffer_Occupancy	0.21	0.95	39%	92%	17%	0.20	0.85	32%	92%	9%
OurMetric	**0.38**	**0.86**	**59%**	**73%**	**14%**	**0.38**	**0.81**	**59%**	**73%**	**8%**

b) Results for Random and Uniform traffic, for PIR=0.5

	Random Traffic					Uniform Traffic				
Metric	**Th**	**Lat**	**L.U**	**C.N**	**C.O**	**Th**	**Lat**	**L.U**	**C.N**	**C.O**
Availavle_VCs	0.27	1.03	47%	92%	17%	0.26	0.91	43%	92%	10%
Crossbar Demand	0.28	1.25	45%	90%	16%	0.25	1.39	37%	90%	9%
Buffer_Occupancy	0.23	0.89	42%	92%	17%	0.22	0.81	36%	92%	10%
OurMetric	**0.38**	**0.79**	**59%**	**73%**	**14%**	**0.38**	**0.77**	**59%**	**73%**	**8%**

c) Results for Random and Uniform traffic, for PIR=1.0

	Random Traffic					Uniform Traffic				
Metric	**Th**	**Lat**	**L.U**	**C.N**	**C.O**	**Th**	**Lat**	**L.U**	**C.N**	**C.O**
Availavle_VCs	0.21	0.75	39%	92%	17%	0.20	0.64	32%	92%	10%
Crossbar Demand	0.29	1.27	47%	90%	15%	0.24	1.21	38%	90%	9%
Buffer_Occupancy	0.20	0.77	37%	92%	17%	0.20	0.66	31%	92%	10%
OurMetric	**0.38**	**0.90**	**59%**	**73%**	**14%**	**0.38**	**0.86**	**59%**	**73%**	**8%**

Th: Average Throughput, **L.U**: Average Links Usage rate, **Lat:** Average Latency value divided per 10 **C.N:** Average Congested Nodes rate, **C.O:** Average Congestion Occurence rate

In our paper, we aim to maximize **L.U** and reduce **C.O** and **C.N** while reducing the congestion negative effect and improving further the network performance with respect to throughput (**Th**) and latency (**Lat**).

A performance comparison between our congestion metric and the three standard metrics is performed. The simulation results for implemented strategies are summarized in Tables 5.2 and 5.3. Under each metric, PIR and traffic scenario, the five parameters are depicted.

By analyzing the results of the tables, it is quite clear that our metric solution achieved very good results compared to other metrics results. The positive outcome of our studies is due to better traffic distribution and optimal use of the network resources throughout our process. In this framework, our results proved to be more efficient in terms of the overall performance. As a result, we managed to achieve a better traffic fluidity including lower latency and a high rate of successful data delivery towards different network nodes.

In order to confirm this significant difference in the results; As it is shown, Table 5.4 clearly demonstrates the difference in performance between the four solutions. The traffic flow got far better in our solution, simply because of the existing gap in the network traffic load and the number of successfully delivered flits. What is also interesting in our approach is that despite all this doubled data load in the network, the average latency is improved and remained better than those of the other solutions.

The normalized load was introduced as parameter for evaluation. It is defined as the rate of the distribution

Table 5.3
Simulation results for simulation time = 500 clock cycles

a) Results for Random and Uniform traffic, for PIR=0.3

	Random Traffic					Uniform Traffic				
Metric	**Th**	**Lat**	**L.U**	**C.N**	**C.O**	**Th**	**Lat**	**L.U**	**C.N**	**C.O**
Available_VCs	0.22	1.22	42%	98%	16%	0.21	1.12	35%	98%	29%
Crossbar Demand	0.25	1.42	41%	90%	15%	0.23	1.34	35%	98%	26%
Buffer_Occupancy	0.20	1.21	37%	98%	16%	0.18	1.11	30%	98%	29%
OurMetric	**0.31**	**0.96**	**52%**	**82%**	**15%**	**0.31**	**0.95**	**51%**	**82%**	**26%**

b) Results for Random and Uniform traffic, for PIR=0.5

	Random Traffic					Uniform Traffic				
Metric	**Th**	**Lat**	**L.U**	**C.N**	**C.O**	**Th**	**Lat**	**L.U**	**C.N**	**C.O**
Available_VCs	0.22	1.14	42%	98%	16%	0.21	1.04	35%	98%	29%
Crossbar Demand	0.24	1.43	41%	90%	15%	0.23	1.33	35%	90%	26%
Buffer_Occupancy	0.20	1.1	37%	98%	16%	0.18	1.02	31%	98%	30%
OurMetric	**0.31**	**0.93**	**52%**	**82%**	**15%**	**0.31**	**0.92**	**51%**	**82%**	**26%**

c) Results for Random and Uniform traffic, for PIR=1.0

	Random Traffic					Uniform Traffic				
Metric	**Th**	**Lat**	**L.U**	**C.N**	**C.O**	**Th**	**Lat**	**L.U**	**C.N**	**C.O**
Availavle_VCs	0.22	1.04	40%	98%	17%	0.20	0.98	34%	98%	30%
Crossbar Demand	0.24	1.32	41%	90%	15%	0.23	1.23	34%	90%	27%
Buffer_Occupancy	0.19	1.01	36%	98%	17%	0.18	0.91	30%	98%	30%
OurMetric	**0.31**	**1.00**	**52%**	**82%**	**15%**	**0.31**	**0.92**	**51%**	**82%**	**26%**

Table 5.4
Traffic Load and Latency results for Random traffic, PIR= 1.0 and Simulation time = 1000 clock cycles

Metric	**Sent Flits**	**Received Flits**	**Latency**
Available VCs	11820	9905	8,60
Crossbar Demand	11796	10518	10,70
Buffer Ocuppancy	10580	8646	7,90
Our Metric	**16360**	**15424**	**6,90**

of all the packets that have circulated in the network over the number of buffers existing in this network [25].

We compare the normalized load for the four solutions as illustrated in Figure 5.1. The load is almost identical in the case of reduced traffic. The higher the load, the greater the difference between the normalized load. The load results in our solution explain the gap of results and performance and confirm the uniform distribution of traffic across all network resources.

As shown in Figs. (5.2), (5.3) and (5.4), our approach significantly reduces the number of occurrences of congestion. It has better average latency and better performances, more particularly, in terms of network resource utilization and load balancing.

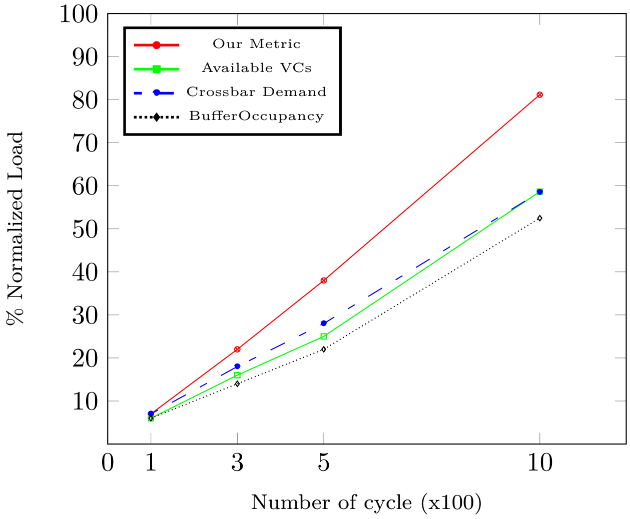


Fig. 5.1. *Network load comparison*

Table 5.5
Our Approach performances improving rates

	Random Traffic					Uniform Traffic				
Simulation Cycles	**Th**	**Lat**	**L.U**	**C.N**	**C.O**	**Th**	**Lat**	**L.U**	**C.N**	**C.O**
Nbr_Cycle = 300	24%-29%	21%	20%-30%	10%	7%	18%-32%	11%	20%-30%	20%	22%
Nbr_Cycle = 500	38%	15%-20%	44%-49%	10%	11%	31%-57%	5%-13	38%-57%	18%	26%
Nbr_Cycle = 1000	**38%-86%**	**12%**	**44%-98%**	**20%**	**29%**	**45%-48%**	**1%-14%**	**62%-67%**	**11%**	**15%**

Our approach gives advantages even in the case of the uniform traffic cf. Figs 5.2.b, 5.3.b and 5.4.b of which the majority of the packets are supposed to cross small distances to reach their destinations and few packets are supposed to send over longer distances in the NoC.

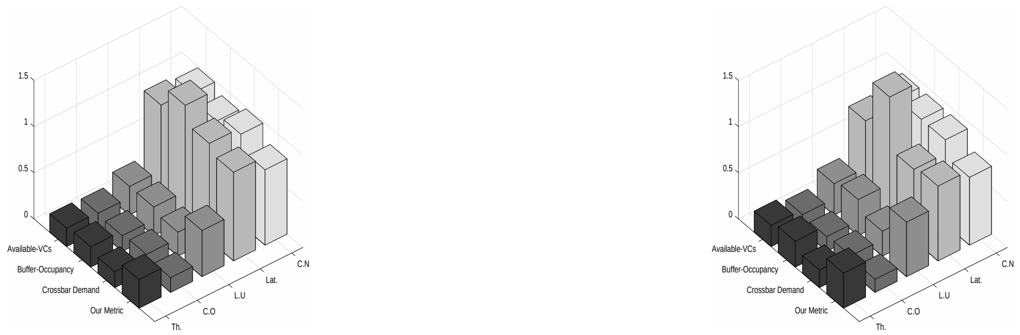
The benefit of our strategy appeared clearly in the random traffic scenarios, as it is illustrated in Figs. 5.2.a, 5.3.a and 5.4.a. that's where the performance gaps between the strategies are huge.

The experimental results summarized in Table 5.5 show that performances are dramatically improved by our congestion control mechanism and this for both types of traffic.

Compared to the three-standard metrics, our metric shows an improvement rate up to 86% in throughput, up to 21% in latency, up to 98% in use of network resources, of up to -20% in number of congested nodes and up to -29% in terms of reduction of occurrence of congestion in the network, those for the uniform and random traffic and for different configurations of simulations. The improvement rate of our congestion-control-based approach has reached its top for random traffic, PIR=1.0 and simulation time = 1000 clock cycles.

Compared to a existing congestion-based strategies and metrics, our proposed congestion control mechanism requires additional arithmetic and negligible extra cost, but offers better performance for both types of traffic: uniform and random, on 2D NoC mesh platforms.

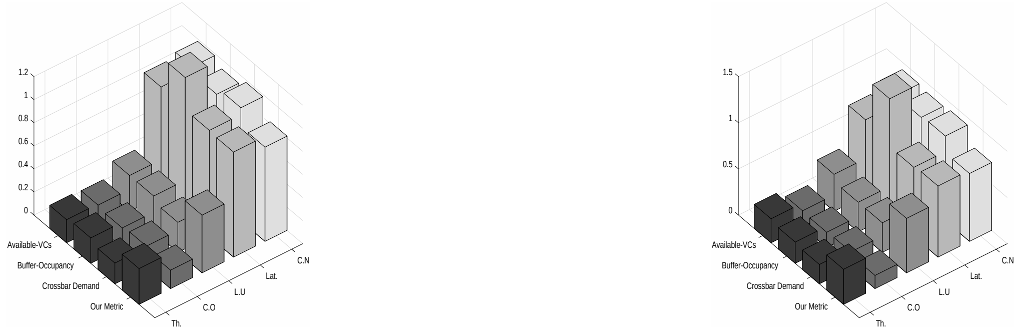
6. Conclusion. In this paper we carried out studies in order to reduce the occurrence and effect of congestion in NoC. The graphical aspect and the step by step debug of the NoC behavior in simulator [23], has motivated us to propose a new control mechanism to overcome congestion and to compensate the performance degradation caused by the traditional indetermination of congestion. An efficient congestion metric is proposed while designing the adaptive XY routing algorithm in so as to overcome congestion and improve resource



a) Under Random traffic

b) Under Uniform traffic

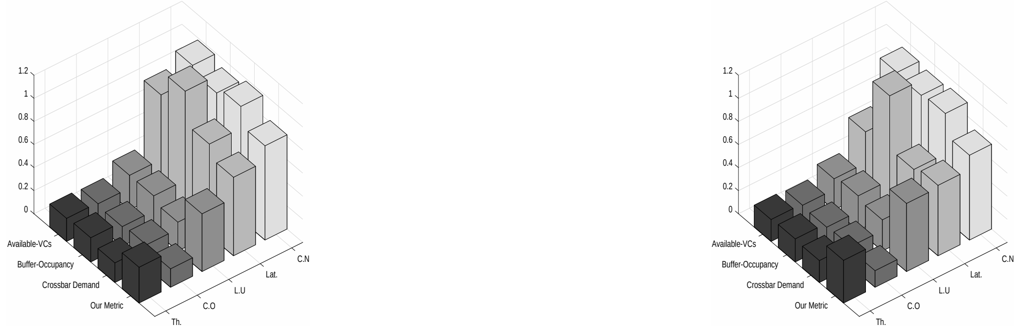
FIG. 5.2. *Average performances under implemented metrics, for simulation time =1000 clock cycles and PIR = 0.3.*



a) Under Random traffic

b) Under Uniform traffic

FIG. 5.3. *Average performances under implemented metrics, for simulation time =1000 clock cycles and PIR = 0.5.*



a) Under Random traffic

b) Under Uniform traffic

FIG. 5.4. *Average performances under implemented metrics, for simulation time =1000 clock cycles and PIR = 1.0.*
Th.: Average Throughput, **C.O:** Average Congestion Occurrence rate, **L.U**: Average Links Usage rate, **Lat.:** Average Latency value divided per 10 and **C.N:** Average Congested Nodes rate

utilization through load balancing.

In addition, experiments show that our newly introduced method has higher performances, besides our results have reached an average rate of improvement ranging up to 40%, compared to past techniques.

In the short term, we intend to study the impact of this novel metric on other routing algorithms and other selection strategies.

Acknowledgments. This work has been supported in part by the PHC TASSILI 14MDU917 project of the University of Oran 1, and by IRCICA,Université de Lille, USR 3380, F-59650 Villeneuve d'Ascq, France.

REFERENCES

[1] G. Ascia, V. Catania, M. Palesi, and D. Patti, *Implementation and Analysis of a New Selection Strategy for Adaptive Routing in Networks-on-Chip*, IEEE Transactions on Computers, 57 (2008), pp. 809–820.

[2] M. K. Benhaoua, A. Singh, A. E. H. Benyamina, and P. Boulet, *DynMapNoCSIM : A Dynamic Mapping SIMULATOR for Network on Chip based MPSoC*, Journal of Digital Information Management, 13 (2015), pp. 45–54.

[3] M. K. Benhaoua and A. K. Singh, *Dynamic Communications Mapping in Multi-tasks NoC-based Heterogeneous MPSoCs Platform*, Int. J. High Perform. Syst. Archit., 5 (2015), pp. 240–251.

[4] L. Benini and G. De Micheli, *Networks on chips: a new SoC paradigm*, Computer, 35 (2002), pp. 70–78.

[5] E. J. Chang, H. K. Hsin, C. H. Chao, S. Y. Lin, and A. Y. . Wu, *Regional ACO-Based Cascaded Adaptive Routing for Traffic Balancing in Mesh-Based Network-on-Chip Systems*, IEEE Transactions on Computers, 64 (2015), pp. 868–875.

[6] E. J. Chang, H. K. Hsin, S. Y. Lin, and A. Y. Wu, *Path-Congestion-Aware Adaptive Routing With a Contention Prediction Scheme for Network-on-Chip Systems*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 33 (2014), pp. 113–126.

[7] A. Dana and N. Salehi, *Congestion Aware Routing Algorithm for Mesh Network-on-chip Platform*, Indian Journal of Science and Technology, 5 (2012), pp. 2822–2830.

[8] M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, and H. Tenhunen, *Agent-based on-chip network using efficient selection method*, in 2011 IEEE/IFIP 19th International Conference on VLSI and System-on-Chip, Oct. 2011, pp. 284–289.

[9] P. Gratz, B. Grot, and S. W. Keckler, *Regional congestion awareness for load balance in networks-on-chip*, in 2008 IEEE 14th International Symposium on High Performance Computer Architecture, Feb. 2008, pp. 203–214.

[10] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu, *Express Cube Topologies for on-Chip Interconnects*, in 2009 IEEE 15th International Symposium on High Performance Computer Architecture, Feb. 2009, pp. 163–174.

[11] N. Gupta, A. Sharma, V. Laxmi, M. S. Gaur, M. Zwolinski, and R. Bishnoi, *$_n$LBDR: generic congestion handling routing implementation for two-dimensional mesh network-on-chip*, IET Computers Digital Techniques, 10 (2016), pp. 226–232.

[12] Z. Han, M. C. Meyer, X. Jiang, and T. Watanabe, *Low-Cost Congestion Detection Mechanism for Networks-on-Chip*, in 2019 IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC), Oct. 2019, pp. 157–163.

[13] H. K. Hsin, E. J. Chang, C. H. Chao, and A. Y. Wu, *Regional ACO-based routing for load-balancing in NoC systems*, in 2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC), Dec. 2010, pp. 370–376.

[14] N. Jiang, D. U. Becker, G. Michelogiannakis, and W. J. Dally, *Network congestion avoidance through Speculative Reservation*, in IEEE International Symposium on High-Performance Comp Architecture, Feb. 2012, pp. 1–12.

[15] K. Jin, C. Li, D. Dong, and B. Fu, *HARE: History-Aware Adaptive Routing Algorithm for Endpoint Congestion in Networks-on-Chip*, International Journal of Parallel Programming, 47 (2019), pp. 433–450.

[16] G. N. Khan and S. Chui, *Congestion Aware Routing for On-Chip Communication in NoC Systems*, in Complex, Intelligent, and Software Intensive Systems, Advances in Intelligent Systems and Computing, Springer, Cham, July 2017, pp. 547–556.

[17] J. Kim, J. Balfour, and W. Dally, *Flattened Butterfly Topology for On-Chip Networks*, in 40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007), Dec. 2007, pp. 172–182.

[18] M. Li, Q.-A. Zeng, and W.-B. Jone, *DyXY - a proximity congestion-aware deadlock-free dynamic routing method for network on chip*, in 2006 43rd ACM/IEEE Design Automation Conference, 2006, pp. 849–852.

[19] P. Lotfi-Kamran, *Per-packet global congestion estimation for fast packet delivery in networks-on-chip*, The Journal of Supercomputing, 71 (2015), pp. 3419–3439.

[20] P. Lotfi-Kamran, M. Daneshtalab, C. Lucas, and Z. Navabi, *BARP-a Dynamic Routing Protocol for Balanced Distribution of Traffic in NoCs*, in Proceedings of the Conference on Design, Automation and Test in Europe, DATE '08, New York, NY, USA, 2008, ACM, pp. 1408–1413.

[21] P. Lotfi-Kamran, A. M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, and Z. Navabi, *EDXY A low cost congestion-aware routing algorithm for network-on-chips*, Journal of Systems Architecture, 56 (2010), pp. 256–264.

[22] S. Ma, N. E. Jerger, and Z. Wang, *DBAR: An efficient routing algorithm to support multiple concurrent applications in networks-on-chip*, in 2011 38th Annual International Symposium on Computer Architecture (ISCA), June 2011, pp. 413–424.

[23] M. A. Meghabber, A. Aroui, L. Loukil, A. E. H. Benyamina, K. Benhaoua, and T. Djeradi, *A flexible network on-chip router for data-flow monitoring*, in 2017 5th International Conference on Electrical Engineering - Boumerdes (ICEE-B), Oct. 2017, pp. 1–6.

[24] L. Ni and P. McKinley, *A survey of wormhole routing techniques in direct networks*, Computer, 26 (1993), pp. 62–76. Conference Name: Computer.

[25] M. Nickray, M. Dehyadgari, and A. Afzali-kusha, *Adaptive routing using context-aware agents for networks on chips*, in 2009 4th International Design and Test Workshop (IDT), Nov. 2009, pp. 1–6.

[26] G. Nychis, C. Fallin, T. Moscibroda, and O. Mutlu, *Next Generation On-chip Networks: What Kind of Congestion Control Do We Need?*, in Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, Hotnets-IX, New York, NY, USA, 2010, ACM, pp. 12:1–12:6.

[27] G. P. Nychis, C. Fallin, T. Moscibroda, O. Mutlu, and S. Seshan, *On-chip Networks from a Networking Perspective: Congestion and Scalability in Many-core Interconnects*, in Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM '12, New York, NY, USA, 2012, ACM, pp. 407–418.

[28] M. Ramakrishna, P. V. Gratz, and A. Sprintson, *GCA: Global congestion awareness for load balance in Networks-on-Chip*, in 2013 Seventh IEEE/ACM International Symposium on Networks-on-Chip (NoCS), Apr. 2013, pp. 1–8.

[29] R. S. Ramanujam and B. Lin, *Destination-based adaptive routing on 2D mesh networks*, in 2010 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), Oct. 2010, pp. 1–12.

[30] M. S. Talebi, F. Jafari, A. Khonsari, and M. H. Yaghmae, *A Novel Congestion Control Scheme for Elastic Flows in Network-on-Chip Based on Sum-Rate Optimization*, in Computational Science and Its Applications – ICCSA 2007, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, Aug. 2007, pp. 398–409.

[31] M. Tang and X. Lin, *Quarter Load Threshold (QLT) flow control for wormhole switching in mesh-based Network-on-Chip*, Journal of Systems Architecture, 56 (2010), pp. 452–462.

[32] M. Tang, X. Lin, and M. Palesi, *Local Congestion Avoidance in Network-on-Chip*, IEEE Transactions on Parallel and Distributed Systems, 27 (2016), pp. 2062–2073.

[33] H. C. Touati and F. Boutekkouk, *FACARS: A novel fully adaptive congestion aware routing scheme for network on chip*, in 2018 7th Mediterranean Conference on Embedded Computing (MECO), June 2018, pp. 1–6.

Edited by: Dana Petcu
Received: Jan 8, 2020
Accepted: May 7, 2020