J Supercomput (2016) 72:567–587
DOI 10.1007/s11227-015-1583-9

Enhanced global congestion awareness (EGCA) for load balance in networks-on-chip

Jili Yan[1]

Published online: 9 December 2015

Abstract As the core count increases in a single chip, traditionally centralized communication architecture has not met the communication demand in new situations, such as system-on-chip (SoC) and chip multi-processor (CMP). Networks-on-chip (NoC), which emerges as an interconnection and communication fabric between components on a single chip, has been regarded as an effective solution. Routing algorithm plays a key role for the performance of NoCs and congestion awareness adaptive routing algorithms take more and more attentions as they have the better latency and throughput performance than the other routing algorithms. However, the existing congestion awareness adaptive routing algorithms often suffer from poor timeliness and accuracy of congestion information, which may lead to high latency and reduced throughput. To address these issues, we propose an enhanced global congestion awareness (EGCA) adaptive routing algorithm which is based on the global congestion awareness (GCA) adaptive technique. EGCA improves the distribution mechanism of congestion information of GCA, which leads to a better timeliness and accuracy of congestion information. Extensive simulations compared the proposed mechanism with typical local, region and global congestion awareness adaptive routing algorithms show that the proposed mechanism can improve system throughput by up to ∼14 and ∼8 % and reduces latency by up to ∼30 and ∼15 % compared with minimal adaptive routing and GCA, respectively.

Keywords Adaptive routing · Network-on-chip · Chip multi-processor · Congestion

✉ Jili Yan
yanjili@mail2.sysu.edu.cn

1 College of Mathematics Physics and Information Engineering, Jiaxing University, Jiaxing 314000, China

1 Introduction

With the continuing advances in semiconductor technology, the increasing complexity and interconnection delay are becoming the limiting factors in SoC and CMP designs [1,2]. The efficiency of interconnection and meeting data transfer requirement are more and more important for on-chip systems, and networks-on-chip (NoC) has been proven to be a flexible, scalable, and reusable solution to these problems [3,4].

In NoCs, routing algorithm, which is used to determine the path traversed by a packet from the source to the destination, plays a vital role in network performance. Routing algorithm can be classified into two categories, deterministic and adaptive routings. The implementation of deterministic routing algorithm is simple and low cost. However, they cannot balance the load across the links in non-uniform or burst traffics [5,6]. Adaptive routing algorithms are proposed to address these limitations. In adaptive routing algorithms, the path traversed by a packet from the source to the destination is determined by the network condition or a predefined rule, which can decrease the probability of passing a packet through a congested link.

Despite the high complexity of implementation, adaptive routing is attractive for the large NoCs especially when these NoCs face with the non-uniform or burst traffics. This approach consists of a routing function and a selection function. The routing function which generates a set of candidate output channels is usually based on deadlock-free models [6–8,27], and the selection function chooses one candidate [8,9]. The selection function markedly affects performance for all adaptive routing algorithms [9,10]. Therefore, to effectively select the optimal output port, the selection function should jointly consider the accurate and timely congestion information. Congestion awareness adaptive routing mechanism is proved to be of a better performance than other oblivious adaptive routing algorithms [10–19,27–32].

Congestion awareness adaptive routing methods can be divided into local, region and global. Local congestion awareness adaptive routing technique makes a routing decision only based on the local congestion information, where the metric of congestion information is a local resource or a combination of local resource, such as, the number of free VCs, the buffer availability at adjacent nodes and the link utilization. [10–14,30,32]. These methods often make sub-optimal routing decisions at a global level as a near-sighted view of network congestion and also react slower to congestion due to the use of network back pressure for information propagation.

Region congestion awareness adaptive technique gathers congestion information from nodes beyond the adjacent. It achieves a better view of the network congestion than local schemes [15,16,29,31]. However, because the accuracy of congestion information may be quite poor due to the weighted aggregation of congestion and interference with the uncorrelated congestion information from different nodes, these techniques often lead to degraded performance.

Global congestion awareness adaptive technique can achieve the best view of network congestion [17–19]. However, updating network congestion information is usually very slow, which may lead to that the congestion information used to select output is staleness, and then the optimal output port cannot be selected.

To summarize existing congestion awareness adaptive routing techniques, the main drawbacks of these techniques are:

1. Insufficient congestion information: without the global network congestion state, the congestion information achieved by local congestion awareness adaptive routing is not sufficient to select the optimal output port [10–14,30,32].
2. Inaccurate congestion information: as the weighted aggregation and interference with uncorrelated congestion information out of routed region, region congestion awareness adaptive routing algorithms may make sub-optimal output port selections [15,16,29,31].
3. Stale congestion information: global congestion awareness adaptive routing techniques are often inefficient in making optimal output port selections due to the staleness of congestion information maintained by them as these information update may be very slow [17–19]. GCA routing algorithm has a higher network performance than other global congestion awareness adaptive routing techniques [19]. However, the staleness of congestion information often leads to sub-optimal output selection [19] (the analysis of performance of GCA seen in Sect. 3).

In this paper, EGCA is proposed to overcome the limitations of GCA. The rest of this paper is organized as follows: Sect. 2 discusses the related works. Section 3 describes the GCA routing algorithm and its limitations. EGCA model is described in Sect. 4, while experimental results are discussed in Sect. 5. Finally, the conclusion of this paper is given in Sect. 6.

2 Related works

2.1 Local congestion awareness adaptive routing

Local congestion awareness adaptive routings consider local resource as the congestion metric and select the output port based on the local congestion information. Work [10] uses the downstream free VC count as a congestion metric and picks the port with the higher number of free VCs. Available buffers at downstream node are used as the congestion metric in work [11]. While Singh et al [23–26] use the output queue lengths for the same purpose. DyXY and DyAD switch between deterministic and minimal adaptive routing (MAR) based on available buffers at downstream neighbor node [12,13]. Work [32] and [30] use the local switch congestion and channel congestion as a congestion metric.

Local congestion awareness adaptive routing techniques have the low implementation complexity and they have no harmful impact on network traffic due to congestion information propagated by separated links. However, these schemes often make sub-optimal routing decisions at a global level due to insufficient network congestion information. Figure 1 highlights these limitations. If we route a packet from S to D using local congestion awareness routing (LCA), the link with a lower congestion value will be chosen at each hop. By this way, the path S $\rightarrow$ B $\rightarrow$ E $\rightarrow$ Z $\rightarrow$ D on which the sum of congestion values is $3 + 3 + 1 + 6 = 13$ would be selected. However, the path S $\rightarrow$ B $\rightarrow$ E $\rightarrow$ C $\rightarrow$ D is more better, as the total sum of the congestion values on this path is $3 + 3 + 2 + 1 = 9$.

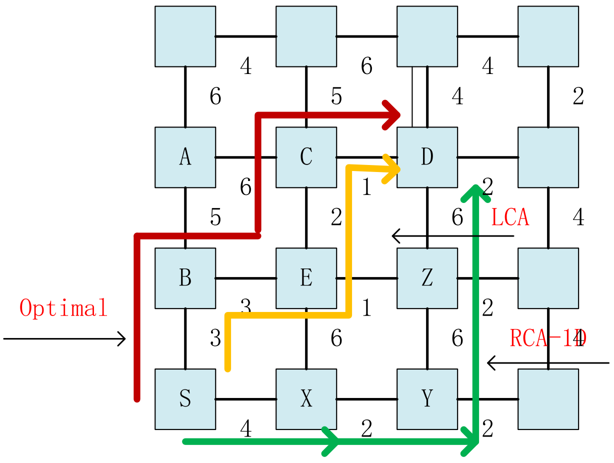

Fig. 1 Sub-optimal rouging decision

2.2 Regional congestion awareness adaptive routing

To the best of our knowledge, regional congestion awareness (RCA) is the first work to explore adaptive routing based on congestion visibility beyond the non-local nodes [15]. RCA utilizes the congestion information from adjacent and non-adjacent nodes by propagation over a light-weight monitoring network. However, the algorithm introduces redundant congestion information in making routing decisions, especially, under workload consolidation. Destination-based adaptive routing (DBAR) is another regional congestion awareness adaptive routing algorithm, which weights congestion information about the destination node while selecting the route. It can reduce the noise aggregation accumulation by integrating the destination into the selection function [16]. DBAR exploits the locality of many traffic patterns. So it is particularly useful for multiple different application regions, where congestion information of different application regions is not used for routing decisions in other application regions. However, DBAR's performance is very similar to RCA in single application region. Furthermore, congestion information from different nodes in same region inferences with each other is just as RCA.

Although these region congestion awareness adaptive techniques provide a better view of the network than local congestion awareness adaptive schemes, the accuracy of congestion information may be quite poor, as the interference of congestion information with each other could degrade the network performance. For example, if we route a packet from S to D using a regional congestion awareness algorithm RCA-1D [15] in Fig. 1, we can see these limitations. At node S, it compares between the aggregated congestion of all links in the EAST direction against in the NORTH direction. This comparison leads it to pick the EAST output port. Similarly, at downstream nodes, it picks EAST at X to reach Y from where it has only one admissible dimension to traverse. It ultimately selects the path, S $\rightarrow$ X $\rightarrow$ Y $\rightarrow$ Z $\rightarrow$ D, and the sum of congestion values is 18, while the path S $\rightarrow$ B $\rightarrow$ E $\rightarrow$ C $\rightarrow$ D is more better.

The sub-optimal route, S $\rightarrow$ X $\rightarrow$ Y $\rightarrow$ Z $\rightarrow$ D, is chosen because RCA considers congestion status of all links in the NORTH and EAST direction, and introduces redundant congestion information.

2.3 Global congestion awareness adaptive routing

Recently, some researchers have proposed some global congestion awareness adaptive routing techniques, such as adaptive toggle dimension ordered routing (ATDOR), destination-based adaptive routing (DAR) and global congestion awareness (GCA) [17–19]. ATDOR introduces a monitoring network to transmit congestion information by each node to a dedicated node. For every source–destination pair, the path experienced by a packet is adaptively switched between XY and YX DOR [17]. In DAR, every node estimates the delay to every other node in the network, and then communicates their local congestion information with the rest of the network [18]. Every node then determines the amount of traffic that it has to be split for a particular destination among the candidate output ports. GCA removes the need for a sideband network by embedding ("piggybacking") congestion information in the packet header, which uses a low-complexity and low-latency route computation unit to provide a lower response time to the changed network congestion [19].

An additional sideband network in ATDOR and DAR add to implementation overheads. Especially, ATDOR and DAR have not solved the inaccuracy of congestion information in making routing decisions. For GCA, one shortcoming is the increase of queuing latency of congestion information in input buffer in destination router, when network becomes congested. Another is the insufficient utilization of congestion information as congestion information has not been shared with neighbors. Above these may lead congestion information to be stale.

3 Global congestion awareness (GCA) model and its limitations

To address inefficiency of other global congestion awareness routing algorithms due to the interference of redundant congestion information with each other, GCA routing algorithm offers the complete view of the congestion status of the whole network and the accurate congestion information of all links. GCA algorithm for 2D mesh consists of the following steps [19]:

1. A new field called the "traffic vector" is added to the header flit. It consists of "back-annotated" congestion information as seen by the packet as it traverses the network, along with bits recording the path taken. At each hop, the traffic vector field is appended with five new bits, where three bits for free buffers and two bits for directions.
2. In routing, every node on the path extracts this traffic vector field from the header flit and updates its own current view of the network congestion status based on this new information.
3. Each node maintains a congestion map of all links in the network. Upon extraction of the traffic vector from an incoming packet, the node performs a route computa-

tion of the affected sub-network to determine the best output port for every node in that sub-network. These output ports are maintained in a pre-route table.
4. Each node also appends congestion information about the incoming link into the header flit before sending out the packet.
5. Packet is routed to the appropriate output port in each node by looking up the pre-route information embedded in the header flit. The current node also adds in output port information for the next hop by looking up its routing table.

GCA algorithm can achieve a better network performance than other typically congestion awareness adaptive routing algorithms [19]. However, update of its congestion information of GCA cannot be viewed by the neighbors of the destination of the routed packets. This may lead to that the neighbors cannot select the best output port while sending packet to the destination in the same sub-network. For example, Fig. 2a shows that a packet from the source node S is sent to the destination node D along with the route path S $\rightarrow$ A $\rightarrow$ B $\rightarrow$ C $\rightarrow$ D. At the destination node D, it will update congestion status of the links on the route path in its own congestion map (as shown in the Fig. 2a in red font). At the later time, the neighbor node S$'$ of the D node in Fig. 2b sends a packet to the destination A node, according to the rule of minimal adaptive routing and its congestion map, it will select the route path S$'$ $\rightarrow$ D$'$ $\rightarrow$ C$'$ $\rightarrow$ B$'$ $\rightarrow$ A. However, there exists a better route path S$'$ $\rightarrow$ D $\rightarrow$ C $\rightarrow$ B $\rightarrow$ A. GCA can provide an accurate congestion status on the specific links of the route path. On the one hand, these new congestion statuses cannot be shared with neighbor nodes until they accepted packet traversing this route path. On the other hand, as injection rate of packet increases, congestion information in head flit will experience longer time in the input buffer of the destination router before being removed from it. It may incur staleness of the network congestion information included in the traffic vector of packets. EGCA improves the distribution mechanism of congestion information of GCA and can achieve a better latency and throughput performance with negligible cost.

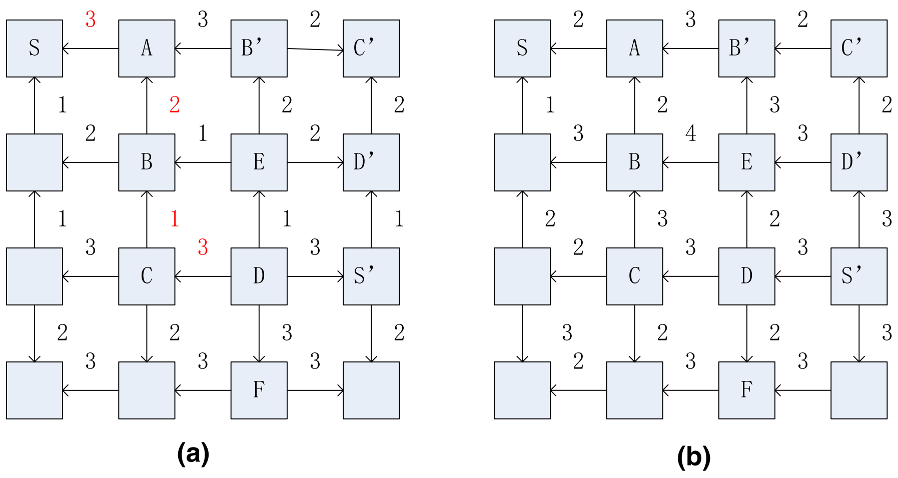

Fig. 2 The limitations of GCA. **a** Node D$'$s update of congestion information. **b** Neighbor nodes without sharing latest congestion information

4 Enhanced global congestion awareness model

The enhanced global congestion awareness (EGCA) adaptive routing algorithm is an improved version of global congestion awareness adaptive routing algorithm for 2D mesh NoC. EGCA provides a timely and complete view of the congestion status of the whole network for each router and its neighbors. It not only provides more timely and accurate congestion status of links of sub-network, but also improves the latency of congestion information. The deadlock freedom of EGCA model is guaranteed by implementing deterministically routed escape channel VCs.

4.1 EGCA model

Accurate and timely network congestion information is a decisive factor for the optimal output port selection of global congestion awareness adaptive algorithm. EGCA algorithm can eliminate the redundant congestion information and interference, and share the latest congestion information among neighbors at same time. The main difference between EGCA and GCA is that the EGCA model improves the distribution mechanism of congestion information of GCA model and reduces the staleness and latency of congestion information. As a result, EGCA can achieve a better performance than GCA algorithm.

4.2 Update of congestion information

The congestion status of links experienced by a packet is “annotated back” in the traffic vector field of the head flit in GCA. The node experienced by a packet extracts the traffic vector from the head flit and updates its own current view of the network based on the new congestion information contained in the traffic vector. However, only the nodes traversed by the packet can view the network congestion, the neighbors of the destination cannot benefit from the latest congestion information. In EGCA, every destination node in network not only updates its current view of the network congestion status according to the accepted traffic vector, but also its neighbors can benefit from the updating of congestion information.

4.2.1 Type of packet

The EGCA model has two types of packets, one is the network data packet that transfers network data and another is the congestion information packet consisting of a congestion information flit, which is only transferred to the neighbors. To mark different type of packets, we introduce a packet type field with one bit in head flit. When a node receives a packet, it extracts the traffic vector and updates its view of network congestion status. At the same time, the packet type field is checked, and if it is a congestion information packet, this packet will be dropped. Otherwise, it is a network data packet, and then it will be put into the corresponding buffer queue according to its VCID.

4.2.2 Update of congestion information

Timely and accurate congestion information can reduce the inefficiency of global congestion awareness selection strategy. In EGCA, when a node receives the head flit of the packet in input port, the packet type field will be checked. If the packet is a congestion information packet, the traffic vector is extracted to update the congestion map with the new congestion information and it is discarded. If not, the traffic vector of a network data packet is extracted to update the congestion map, and the packet is inserted into the corresponding buffer queue according to its VCID. Furthermore, if the current node is also the destination, the traffic vector and the congestion value of the backward link linking the input port will be encapsulated as a congestion information packet and sent to neighbors excluding the incoming neighbor. The congestion information distribution algorithm (Algorithm 1) details the procedure of congestion information distribution. The time

```
Input: Congestion Information Packet $P$, Update Frequency Array $F$
Output: Update Frequency Array $F$
1.  for ($i = 0$; $i < 4$; $i$++)
2.      if ($F[i] == 0$ and $i$ != input port)
3.          Sending $P$ to the neighbor node linked $i$ port;
4.          $F[i] = 1$;
5.      endif
6.  endfor
7.  return $F$;
```

complexity of algorithm 1 is $O(n)$ by analyzing the pseudo-codes of it. Moreover, the checked output ports never exceed four, so the time cost of algorithm 1 is low.

A node receives the congestion information packet from the neighbor and updates only its path congestion view of network. That is, the node does not forward it to its neighbors. For example, Fig. 3 shows congestion information multicast and update of congestion map of the neighbors. Figure 3a shows the destination node D has received a packet from source node S. The destination node D extracts the traffic vector from the packet to update its congestion map and then encapsulates the traffic vector and the congestion value of the backward link into a congestion information packet. This congestion information packet is forwarded to the neighbors of the destination node D, E, F and S′. Here, the neighbor C is not included because the neighbor C has updated its congestion map at prior hop.

However, the congestion information of some links included in the traffic vector may not be on the shortest path from the neighbor to other nodes. The congestion information of these links must be pruned when the neighbor updates its congestion map. For example, when the node E receives a congestion information packet from the node D, it will update its congestion map with this latest congestion information. But the link C $\rightarrow$ B is not on the possible shortest paths from node E to other nodes, so the congestion information of the link must be discarded. The neighbor E only updates the congestion information of the links D $\rightarrow$ C, B $\rightarrow$ A and A $\rightarrow$ S, which is shown in Fig. 3b. The nodes F and S′ will update the congestion information of all

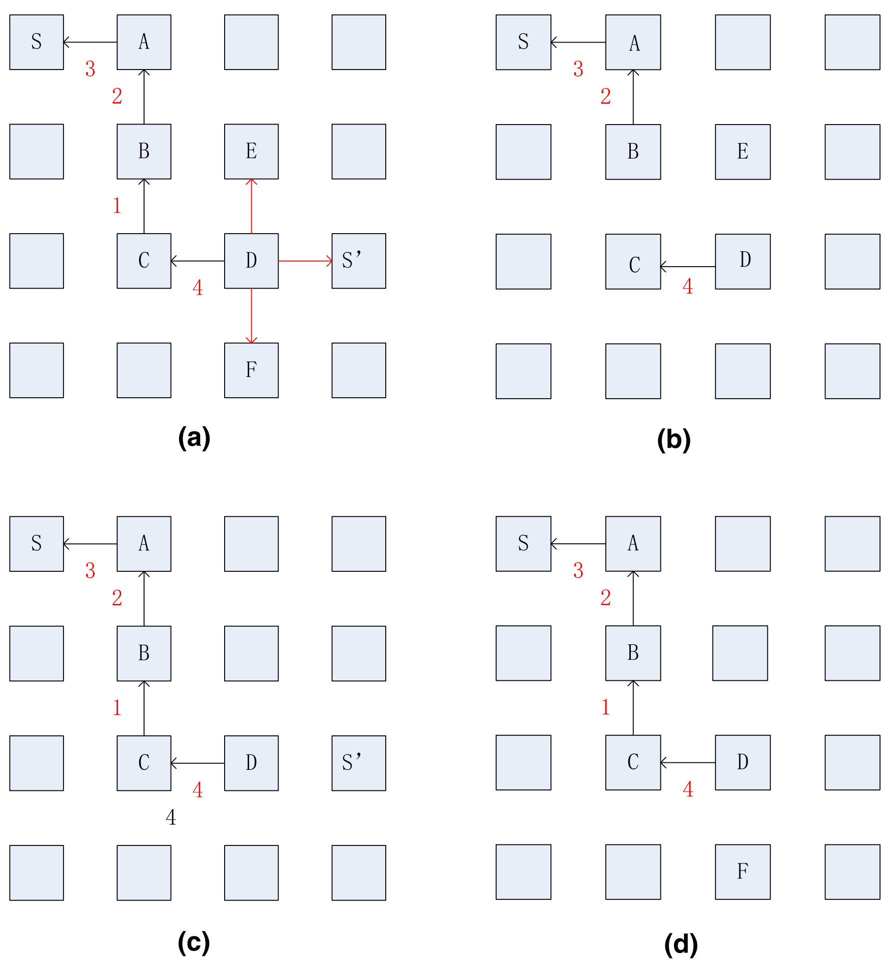

Fig. 3 Update of congestion information mechanism of EGCA. **a** Multicast congestion information. **b** Update of congestion information of neighbor E. **c** Update of congestion information of neighbor s$'$. **d** Update of congestion information of neighbor F

links included in the traffic vector, because these congestion information included in the traffic vector are on the shortest paths from the node F or S$'$ to the other nodes as shown in Fig. 3c, d, respectively.

Algorithm 2 describes the update of congestion information in each node. By analyzing the pseudo-codes, the time complexity of algorithm 2 is $O(n)$, which is low and similar to that of GCA.

4.2.3 Update frequency

With the increase of packet injection rate (it pir), the neighbors will receive more and more the congestion information packets, which will lead to downgrade network performance due to over-consuming network bandwidth. To keep network performance, the update frequency of congestion information from the neighbors must keep smaller. In the EGCA mechanism, the update frequency of congestion information packet from neighbors is a design parameter f, which is set in Sect. 5.

Algorithm 2: Congestion Information Update Algorithm
Input: Congestion Information *traffic vector*, Congestion Map *M*;
Output: Congestion Map *M*;

```
1. C←traffic Vector;
2. for(i = C.length; i<= C.end; i++)
3.     If(i is on the minimal adaptive routing)
4.         Update congestion information of M with C[i];
5.     endif
6. endfor
7. return M[]
```

4.3 Route computation

In a congestion map, every link is assigned a value which represents its current congestion status. A path between a pair of nodes is made up of multiple links and its congestion value is the sum of the congestion values of all those links. EGCA picks the path with the least congestion value within all the minimal paths. We can easily find the shortest path from congestion map using Dijkstra [20], which is widely used in network routing protocols.

We take Fig. 4 as the example to experience the process of routing computation. Figure 4a shows the initial state of routing computation in source node 5, and the number on the arrow is the value of congestion information of the link in a node of congestion map. The number on the top left corner in each box is the serial number of a node. As there is one and only route from the source node to the destination nodes on the same X or Y axes in the minimum adaptive routing algorithm, the output port from the source node to these destination nodes can be obtained from the congestion map and immediately written into pre-routing table. For example, the output port from the source node 5 to the destination node 7 is the east output port of the node 5 and its congestion value is 3, which is expressed as 3/E in the node 7. The routing computation in a node needs only to compute the route from the source node to the nodes that are not on the axes (X or Y axes) following as steps:

1. In the first step, all minimal adaptive routings of the node in two hops away from the source node are computed. For example, the source node 5 can reach node 10 along with the node 6 or 9. The congestion value of the routing from node 5 to 10 passed node 6 is 2 + 2 = 4. However, the congestion value of the routing passed node 9 is 0 + 3 = 3. So the optimal output port from the source node 5 to 10 is the south output port of node 5 and the congestion value of routing is 3, shown as 3/S in node 10 box. By this way, we can compute the optimal output ports from the source node 5 to node 0, 2 and 8 and the corresponding congestion values,

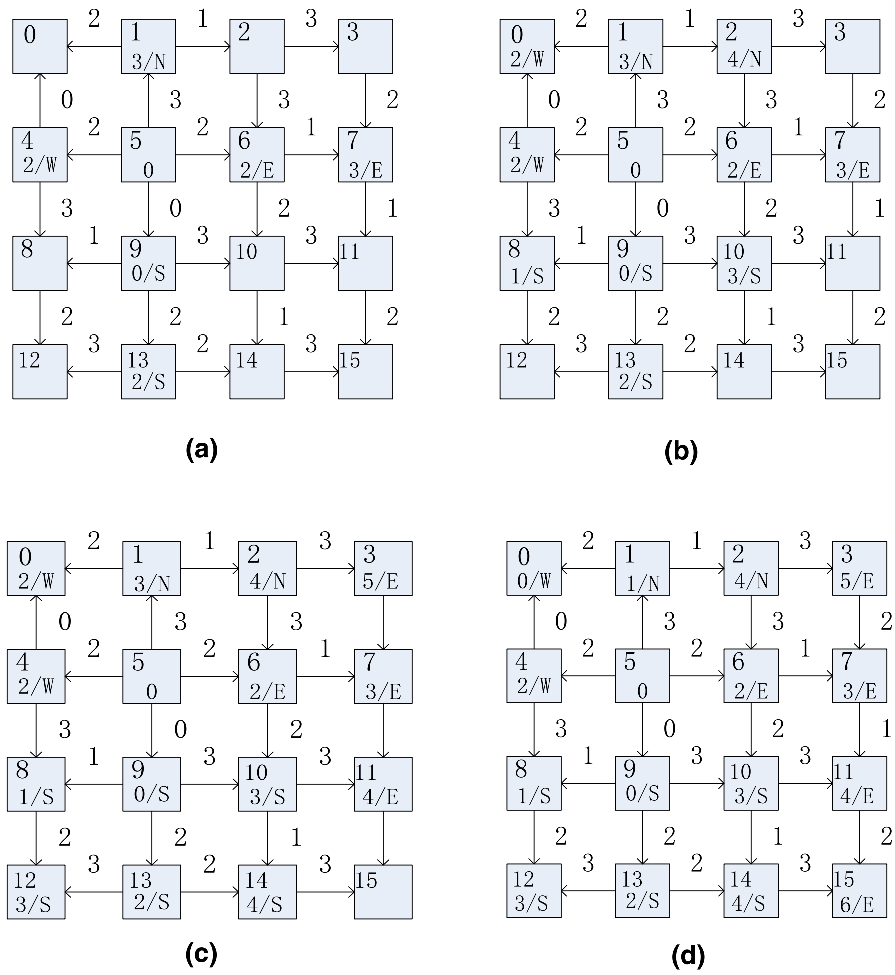

Fig. 4 Routing computing. **a** Initial state. **b** After step 1. **c** After step 2. **d** After step 3

respectively, showed in Fig. 4b. The achieved optimal output ports are written into pre-routing table.

2. In the second step, EGCA computes the optimal output ports through which the source node 5 can reach all nodes that are three hops away from the source node 5. In this step, the optimal output ports reaching nodes 3, 11, 12 and 14 and the corresponding congestion values are shown in Fig 4c. These optimal output ports are written into pre-routing table.
3. According to the above rules, EGCA model can compute the optimal output ports from the source node 5 to all the destination nodes with 4, 5 and …… hops away from the source node 5, until it gets all optimal output ports for all other nodes. For example, routing computation of all optimal output ports from the source node 5 to other nodes will be finished in the third steps as shown in Fig. 3d.

For a $N \times N$ mesh network, the maximal computation steps that a source node computes optimal output ports from itself to all the nodes in the network is $2 \times N - 3$. Whenever a node updates the links in the map, route computation only needs to re-compute a sub-graph of the network because the traffic flow from the source node to the destination node is unidirectional and thus the change can potentially affect only the nodes downstream of that link.

4.4 Other design issues

There are other design issues, including congestion information staleness, congestion information scaling and limited EGCA (LEGCA), in EGCA model. In this paper, we adopt the similar strategy used in GCA to address these issues. To save space, we omit the design of LEGCA and the performance evaluation due to it having similar design, and performance with LGCA.

The congestion values of some links in congestion map may not be updated in a longer time and become stale. If routing algorithm uses stale congestion value to make routing decisions, it usually leads to sub-optimal output port selections. To combat it, EGCA mode employs a fading mechanism. When a congestion value in congestion map is not updated in the last n cycles, the value will be increased or decreased (fading) to average value of all possible congestion values in steps of x, where n and x are set in Sect. 5.

The actual congestion value of a distant link would be changed when congestion map receives the congestion value due to latency experienced by the packet or less update frequency. Moreover, the likelihood that congestion status would be changed before a packet arrives to the link increases considerably for distant nodes. To combat these issues, we assume that the influence on routing decision from the near link congestion is larger than that from distant link. EGCA weights the congestion values from differently distant links by the scaling mechanism with a scaling factor S_i, where i ($i \geq 0$) denotes the distance from the source node to the destination node in hops and w ($w < 1$) is an empirically determined constant, according to Eq. (1).

$$S_i = 1 - (w{*}i) \tag{1}$$

According to Eq. (1), the output links of the source node are not scaled ($i = 0$, $S_i = 1$) and the scaling increases in steps of w as you move away from the source node. For all the links with $i \geq 1/w$, the value of S_i is fixed at w as we do not allow negative congestion metrics.

4.5 Implementation

4.5.1 GCA router microarchitecture

The GCA router microarchitecture is shown in Fig. 5a. It is a three-stage pipeline adaptive virtual channel router, which consists of some additional hardware components: congestion map, route computation module and optimal output port table. Compared

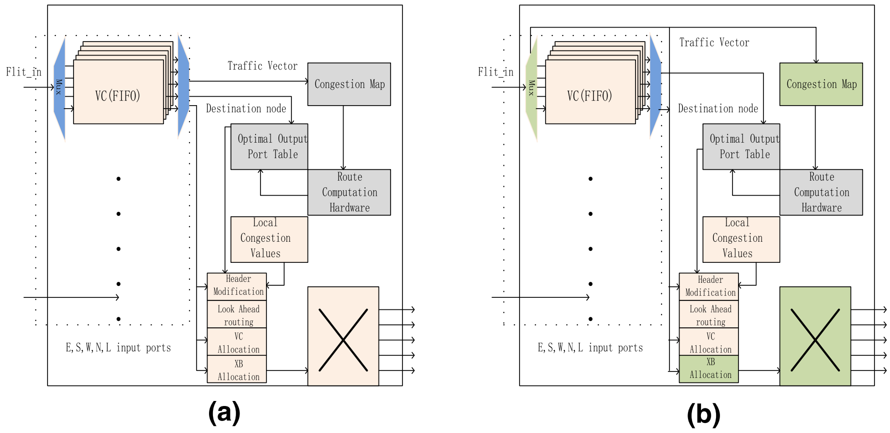

Fig. 5 Microarchitecture of router. **a** GCA. **b** EGCA

Table 1 Summary of number of bits of storage overhead for one node in 64-node network

Storage element	GCA	EGCA
Congestion map	336	336
Optimal output port table	49	49
Flag array	112	112
Update array	0	4
Summary of number of bits	497	501

to the baseline adaptive virtual channel router, additional hardware components need additional storage to store corresponding information. Table 1 presents the storage requirement of GCA for one node of an 8 × 8 size mesh network with five bits for a link, three bits for congestion value per link and two bits for direction (the details of computing storage requirement of GCA seen in Ref. [19]). Especially, GCA router microarchitecture not has the Update Array.

4.5.2 EGCA router microarchitecture

EGCA router provides support to the distribution of congestion information to neighbor nodes and the update of congestion information. The light green modules shown in Fig. 5b are modified modules against the GCA router (shown in Fig. 5a). Mux of input port in EGCA router checks every head flit received by it, extracts traffic vector and sends it to update congestion map. If the packet is a network data packet, the Mux puts the packet into corresponding input buffer according to its VCID. If not, the Mux drops it. Upon the destination node receiving the network data packet, and the multicast signal for the congestion information packet is set. According to the transform rules of congestion information, the transmission of network data packet is temporarily stopped and the traffic vector with the congestion value of the incoming link is forwarded to its neighbor nodes. Especially, the transmission of the congestion information packet does not change the prior holding state of ports. That is, when the transmission of congestion information packet ends, the transmission of the network data packet can go on. As same as the above Mux, the congestion map, XB allocation and Crossbar modules are also modified for supporting congestion information multicast.

For a local port, one bit should be used to indicate whether the port has transferred the congestion information packet by given congestion information window. So that four bits are needed in updating frequency mechanism for a node in EGCA, which is held by the update array structure. The storage requirement for one node of a 8×8 2D mesh network in EGCA is shown in Table 1.

5 Performance evaluation

In this section, we present the results of performance evaluation of the EGCA routing algorithm on different synthetic traffic patterns and are compared against DOR (dimension order routing), MAR and GCA routing techniques in 4×4 and 8×8 network sizes.

5.1 Evaluation methodology

We evaluate the effectiveness of our EGCA model by extending the open source simulator, *noxim* [21], which is a flit-accurate simulator based on *systemC*. The Simulation parameters for evaluation are described in Table 2. Other design parameters are set by our experimental observation, where update frequency (f) is 100 cycles and the values of n, x, and w of EGCA are the same as that of GCA. The design parameters are shown in Table 3.

We evaluate our proposed strategy with four standard synthetic traffic patterns: uniform random (UR), transpose (TP), bit-reverse (BR) and butterfly (BT). These workloads provide insight into the strengths and the weaknesses of the proposed EGCA model for different scenarios. All synthetic traffic patterns employ a uniform random injection process. Within each simulation for a synthetic traffic, there is a warm-up period of 1000 cycles. Thereafter, the volume of 10,000 flits has been injected or 10,000 cycles have been expired.

Table 2 Simulation parameters

Routing algorithm	DOR	MAR	GCA	EGCA
Network size	4×4, 8×8	4×4, 8×8	4×4, 8×8	4×4, 8×8
Latency/hop	3 cycles	3 cycles	3 cycles	3 cycles
Virtual channels/port	8	8	8	8
Flit buffers/VC	4	4	4	4
Link width	128 bits	128 bits	128 bits	128 bits
Packet size	2–8 flits	2–8 flits	2–8 flits	2–8 flits

Table 3 Design parameters

Design parameters	Values of parameter
n	100 cycles
x	0.1 unit
w	0.1
f	100 cycles

In this work, we focus on the throughput and latency performance of EGCA. For easier comparison, we define the global average throughput and the average latency similar to that in Ref. [22] as followings:

$$\text{Global average throughput} = \text{total received flits}/\,(\text{number of nodes}) \times (\text{total cycles}) \tag{2}$$

where the total received flits refers to the number of flits received by all destination nodes, the number of nodes refers to the number of the network nodes and the total cycles refers to the number of clock cycles between the injection of the first flit and the reception of the last flit.

$$\text{Avg}_{\text{lat}} = \frac{1}{k}\sum\nolimits_{i=1}^{k} \text{lat}_i \tag{3}$$

where k refers to the total number of flits received by all destination nodes and lat_i is the latency of the ith flit received by its destination node.

5.2 Performance evaluation results and analysis

The results of performance evaluation of EGCA, DOR, MAR and GCA are presented in Fig. 6, where the traffic model is UR, and the size of mesh network is 4 × 4. Under

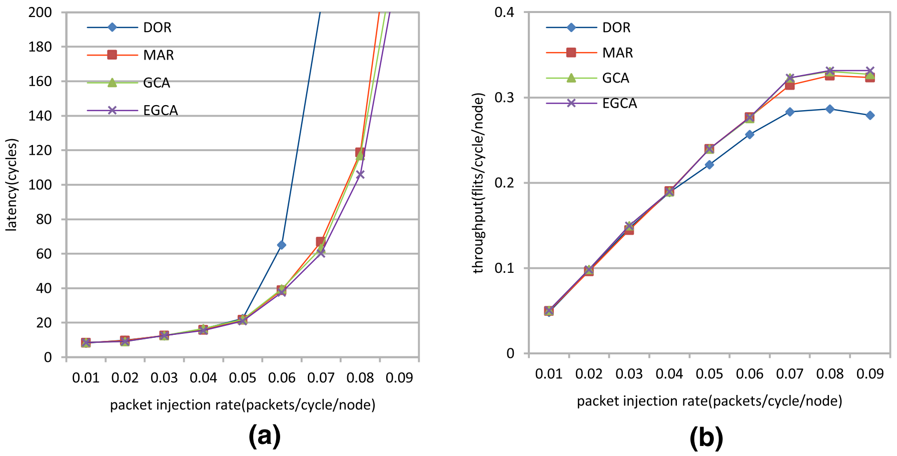

Fig. 6 UR traffic pattern (4×4). **a** Latency. **b** Throughput

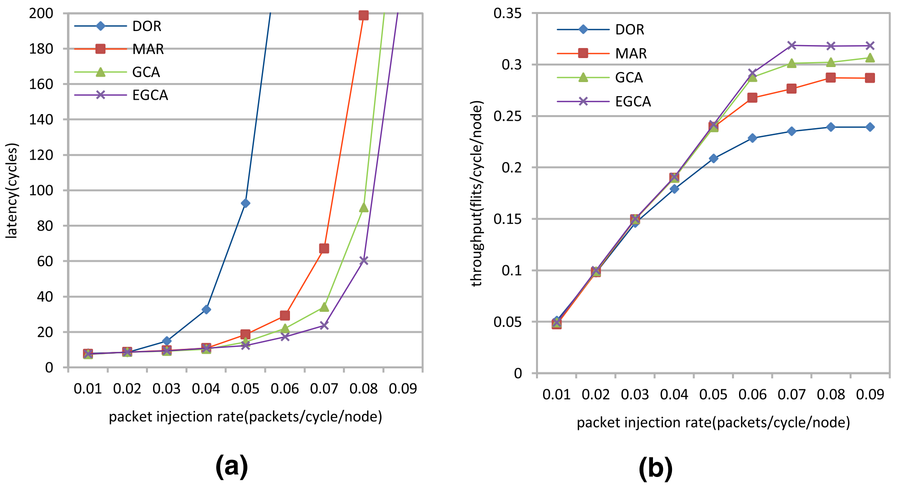

Fig. 7 TP traffic pattern (4 × 4). **a** Latency. **b** Throughput

UR traffic model, DOR can achieve a better latency and throughput performance, because UR traffic model has traffic balance characteristic. Other three adaptive routing algorithms have similar performance in latency and throughput. However, three adaptive routings can use more paths from the source node to the destination node, which can avoid temporary network congestion, and then these three methods can achieve a better performance than DOR with the increase of packet injection rate.

The results of performance evaluation of DOR, MAR, GCA and EGCA under TP traffic model are presented in Fig. 7, where the size of mesh network is 4 × 4. For TP traffic model, the packets generated by a node are sent to the same destination node, so these packets always take the same path to reach the destination node when DOR is employed in network. The coefficient of link utilization of network is imbalanced, and then network will go fast into congestion status with increasing packet injection rate. MAR has ability to select randomly next hop output port and can balance traffic in all minimum path, so it achieves a better performance of latency and throughput than DOR. However, MAR does not consider the current network congestion status when it selects the next hop output port, which usually leads to non-optimal network performance. Compared with MAR, GCA and EGCA can achieve an improved performance as they consider the current network congestion status. Furthermore, EGCA with more efficient distribution of congestion information improves the latency by up to ∼15 % and the throughput by up to ∼8 % than that of GCA.

The latency and the throughput performance of four routing algorithms under BR traffic pattern is shown in Fig. 8. As can be seen from Fig. 8, BR traffic model is of similar average hops to TP traffic model and characteristics in traffic distribution [22], so the latency and the throughput of four routing algorithms under BR are very similar to that of four routing algorithms under TP traffic model in 4 × 4 mesh network. EGCA achieves the best improved performance by up to ∼14 % improvement of throughout and reduced latency ∼30 % on MAR.

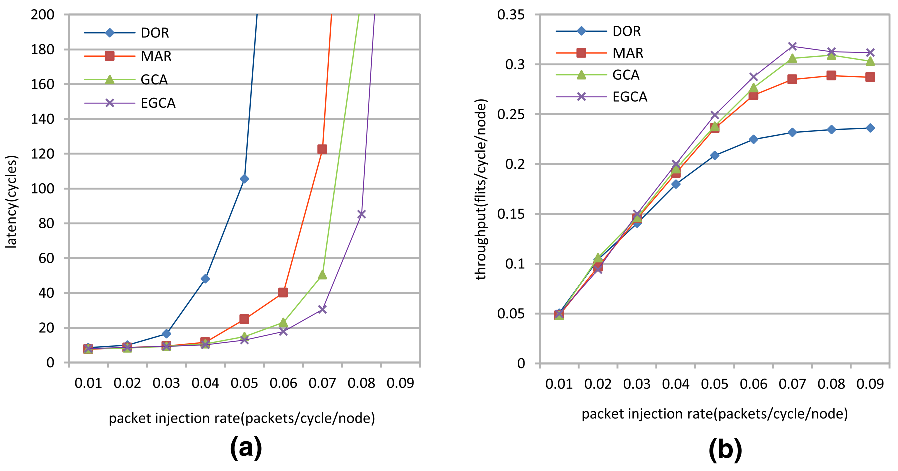

Fig. 8 BR traffic pattern (4 × 4). **a** Latency. **b** Throughput

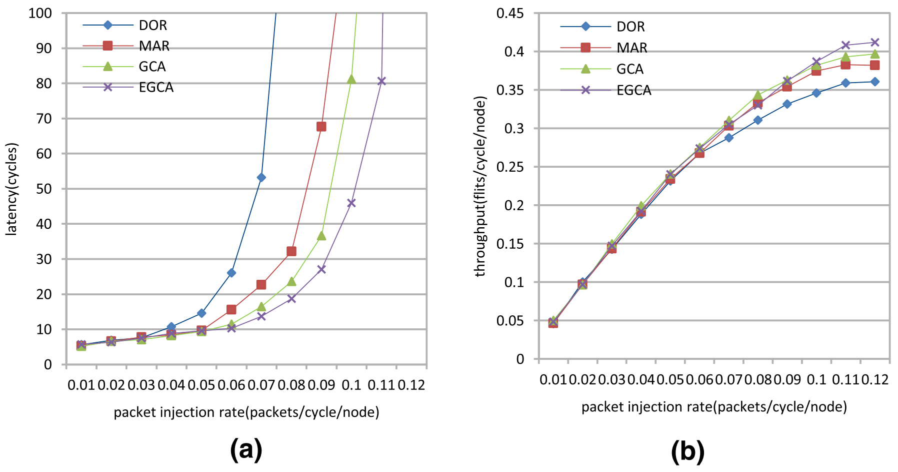

Fig. 9 BT traffic pattern (4 × 4). **a** Latency. **b** Throughput

Figure 9 shows the latency and the throughput performance of four routing algorithms under BT traffic model in a 4 × 4 mesh network, we can observe that four routing algorithms achieve a better latency and throughput performance under BT traffic model than that of other three traffic models. It is the main reason that BT traffic model is of the shorter average hop count than other three traffic models. The shorter hop count not only helps to reduce the network latency and increase the network throughput, but also helps decreases the output port contention, which benefits to improve further the network latency and throughput.

To sum up Figs. 6, 7, 8 and 9, GAC and EGCA models have a stable advantage in comparing with other two routing models. It is also showed that considering network congestion status benefits in routing decision. With more accurate and timely network congestion information, EGCA model performs well than GCA model.

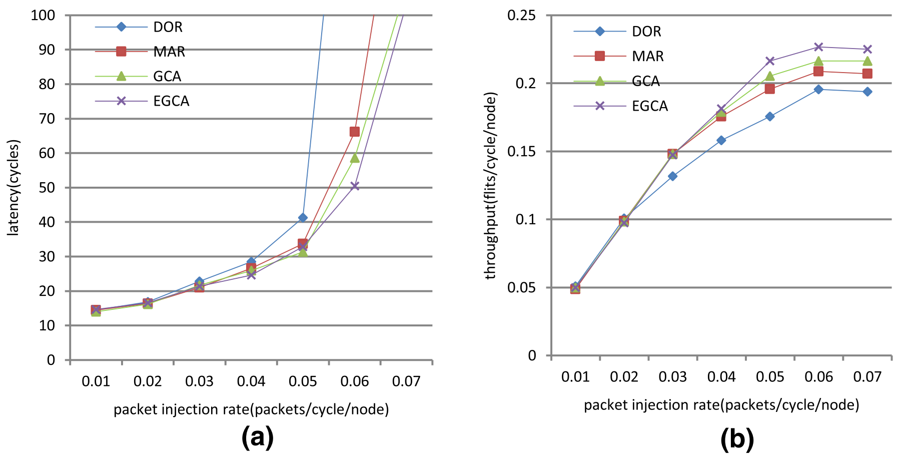

Fig. 10 UR traffic pattern (8 × 8). **a** Latency. **b** Throughput

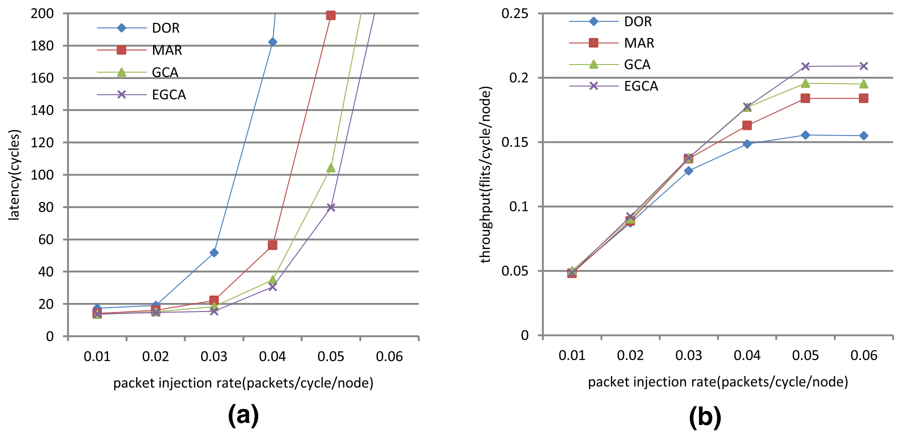

Fig. 11 TP traffic pattern (8 × 8). **a** Latency. **b** Throughput

The performance of four routing algorithms under four traffic models is shown in Figs. 10, 11, 12 and 13 in a 8 × 8 mesh network. We can observe from Figs. 10, 11, 12 and 13 that, with the increase of network size, more hops and more output contentions experienced by a network data packet lead to the increase of network latency and the drop of throughput performance. However, in same environments, we can observe the following results in large network size from these Figures:

1. Adaptive routing algorithm compared with DOR has a stable performance advantage in latency and throughput, which indicates that the adaptive routing algorithm should be adopted in NoCs design.
2. In the adaptive routing algorithm, the random selection strategy of output port cannot balance network traffic well, which easily leads to local congestion and reduces network performance. However, global congestion awareness adaptive routing can obtain a better network performance due to considering current net-

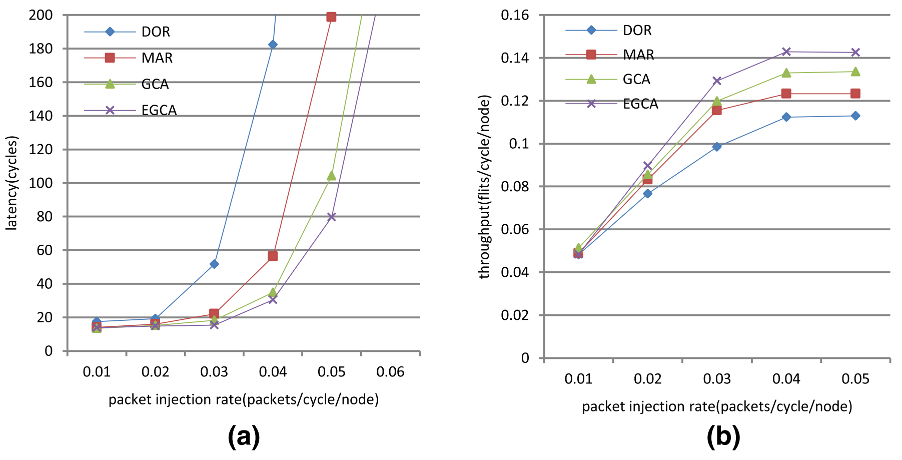

Fig. 12 BR traffic pattern (8 × 8). **a** Latency. **b** Throughput

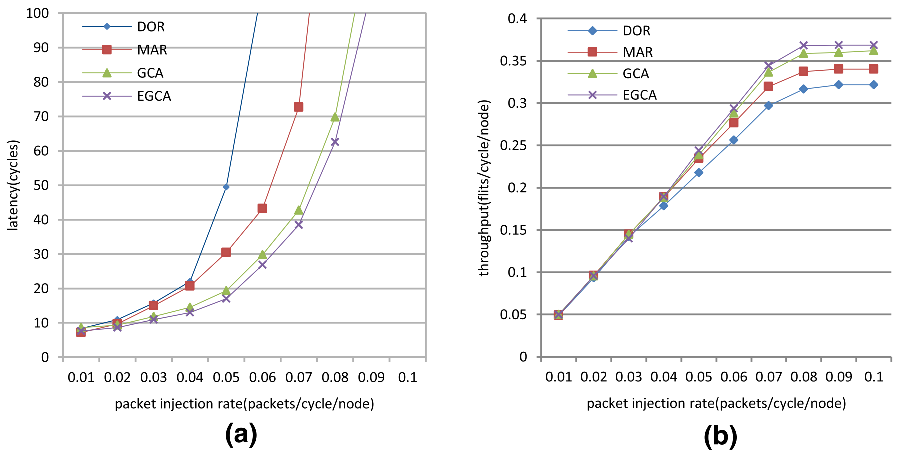

Fig. 13 BT traffic pattern (8 × 8). **a** Latency. **b** Throughput

work congestion status when it makes routing decision. This selection strategy can potentially balance network traffic, relieve network congestion and improve network performance.

3. The advantage of EGCA over GCA shows that how to enhance the timeliness and accuracy of congestion information is the key factor of designing the selection strategy of congestion awareness adaptive routing algorithm.

6 Conclusions

In this paper, we have proposed a new global congestion awareness named EGCA by improving the distribution mechanism of congestion information of GCA. The new mechanism can increase the timeliness and accuracy of congestion information and reduce the latency of congestion information in router, especially when it is congested. EGCA has a better utilization of congestion information, because the congestion information can be shared with neighbors in EGCA. To support these new mechanisms, the router in EGCA has been redesigned. We have conducted extensive simulations to compare the proposed mechanism with typical Local, Region and Global congestion awareness (GCA) adaptive routing algorithms under UR, TP, BR and BT traffic models. The simulation results show that the proposed mechanism can improve the network throughput up to ∼14 and ∼8%, and reduces the network latency up to ∼30 and ∼15 % compared with MRA and GCA, respectively.

Acknowledgments This paper is supported by Zhejiang Provincial Natural Science Foundation of China under Grant No. LY15F020040 and No.LY16F020028, the Doctoral Scientific Research Foundation of JiaXing University under Grant No. N70514009. Humanity and Social Science Youth foundation of Ministry of Education of China under Grant no. 15YJCZH088.

References

1. Benini L, Micheli GD (2002) Network-on-chip: a new paradigm for systems-onchip design. In: Proceedings of the design, automation and test in Europe conference and exhibition, pp 418–419
2. Marculescu R, Ogras UY, Peh L-S, Jerger NE, Hoskote Y (2009) Outstanding research problems in NoC design: system, microarchitecture, and circuit perspectives. IEEE Trans Comput Aided Des Integr Circuits Syst 28:3–21
3. Pande PP, Grecu C, Jones M, Ivanov A, Saleh R (2005) Performance evaluation and design trade-offs for network-on-chip interconnect architectures. IEEE Trans Comput 54(8):1025–1040
4. Sankaralingam K, Nagarajan R, Gratz P, Desikan R, Gulati D, Hanson H, Kim C, Liu H, Ranganathan N, Sethumadhavan S, Sharif S, Shivakumar P, Yoder W, McDonald R, Keckler S, Burger D (2006) The distributed microarchitecture of the TRIPS prototype processor. In: International symposium on microarchitectures, pp 480–491
5. Vangal S, Howard J, Ruhl G, Dighe S, Wilson H, Tschanz J, Finan D, Iyer P, Singh A, Jacob T, Jain S, Venkataraman S, Hoskote Y, Borkar N (2007) An 80-tile 1.28 TFLOPS network-on-chip in 65 nm CMOS. In: IEEE international solid state circuits conference, pp 98–99
6. Dally WJ, Towles B (2004) Principles and practices of Interconnection Networks. Morgan Kaufmann, New York
7. Duato J, Yalamanchili S, Ni L (2002) Interconnection networks: an engineering approach. Morgan Kaufmann, New York
8. Glass CJ, Ni LM (1994) The turn model for adaptive routing. J ACM 41(5):874–902
9. Chiu GM (2000) The odd–even turn model for adaptive routing. IEEE Trans Parallel Distrib Syst:729–738
10. Dally WJ, Aoki H (1993) Deadlock-free adaptive routing in multicomputer networks using virtual channels. IEEE Trans Parallel Distrib Syst 4(4):466–475
11. Kim J, Park D, Theocharides T et al (2005) A low latency router supporting adaptivity for on-chip interconnects. In: Proceedings of the 42nd annual design automation conference. ACM, pp 559–564
12. Li M, Zeng Q-A, Jone W-B (2006) DyXY-A proximity congestion-aware deadlock-free dynamic routing method for network on chip. In: Proceedings of 43rd design automation conference, pp 849–852
13. Hu J, Marculescu R (2004) DyAD: smart routing for networks-on-chip. In: Proceedings of 41st design automation conference, pp 260–263
14. van den Brand JW, Ciordas C, Goossens K, Basten T (2007) Congestion-controlled best-effort communication for networks-on-chip. In: Proceedings of the design, automation and test in Europe conference and exhibition, pp 1–6
15. Gratz P, Grot B, Keckler SW (2008) Regional congestion awareness for load balance in networks-on-chip. In: High performance computer architecture, IEEE 14th international symposium on IEEE, pp 203–214

16. Ma S, Enright Jerger N, Wang Z (2011) DBAR: an efficient routing algorithm to support multiple concurrent applications in networks-on-chip. In: ISCA
17. Manevich R, Cidon I, Kolodny A et al (2011) A cost effective centralized adaptive routing for networks-on-chip. In: Digital system design (DSD), 14th Euromicro conference on IEEE, pp 39–46
18. Ramanujam RS, Lin B (2010) Destination-based adaptive routing on 2D mesh networks. In: Proceedings of the 6th ACM/IEEE symposium on architectures for networking and communications systems, p 19
19. Ramakrishna M, Gratz PV, Sprintson A (2013) GCA: Global congestion awareness for load balance in networks-on-chip. In: Networks on chip (NoCS), 2013 seventh IEEE/ACM international symposium on, pp 1–8
20. Dijkstra EW (1959) A note on two problems in connexion with graphs. In: Numerische mathematik, pp 269–271
21. Palesi M, Patti D, Fazzino F (2010) Noxim: the noc simulator. http://noxim.Sourceforge.net
22. Feng W, Shin KG (1997) Impact of selection functions on routing algorithm performance in multicomputer networks. In: Proceedings of the 11th international conference on supercomputing, pp 132–139
23. Singh A, Dally WJ, Gupta AK, Towles B (2003) GOAL: a load-balanced adaptive routing algorithm for torus networks. ACM SIGARCH Comp Archit News 3(12):194–205
24. Singh A, Dally WJ, Towles B, Gupta AK (2004) Globally adaptive load-balanced routing on tori. Comput Archit Lett 31(1):2
25. Ascia G, Catania V, Palesi M, Patti D (2008) Implementation and analysis of a new selection strategy for adaptive routing in networks-on-chip. Comput IEEE Trans 57(6):809–820
26. Hu W-H, Lee SE, Bagherzadeh N (2008) DMesh: a diagonally-linked mesh network-on-chip architecture. In: First international workshop on network on chip architectures workshop
27. Kumar M, Laxmi V, Gaur MS et al (2014) CARM: congestion adaptive routing method for on chip networks[C]//VLSI design and 2014 13th international conference on embedded systems, 2014 27th international conference on IEEE, pp 240–245
28. Gupta N, Kumar M, Laxmi V et al (2015) σLBDR: Congestion-aware logic based distributed routing for 2D NoC[C]. VLSI design and test (VDAT), 2015 19th International Symposium on IEEE, pp 1–6
29. Zong W, Agyemen MO, Wang X et al (2015) Unbiased regional congestion aware selection function for NoCs[C]. In: Proceedings of the 9th international symposium on networks-on-chip. ACM, p 19
30. Dana A, Salehi N (2012) Congestion aware routing algorithm for mesh network-on-chip platform. Indian J Sci Technol 5(6):2822–2830
31. Chang EJ, Hsin HK, Chao CH et al (2015) Regional ACO-based cascaded adaptive routing for traffic balancing in mesh-based network-on-chip systems. Comput IEEE Trans 64(3):868–875
32. Chang EJ, Hsin HK, Lin SY et al (2014) Path-congestion-aware adaptive routing with a contention prediction scheme for network-on-chip systems. Comput Aided Des Integr Circuits Syst IEEE Trans 33(1):113–126