Improving Negative First Routing Algorithm with Load Equalization of Virtual Channel in NoC

Tao He*[1], Xingming Zhang[1], Tao Qin[2]
[1]National Digital Switching System Engineering Technological Research Center
Zhengzhou 450002, Henan, China
[2]National Computer Network Emergency Response Technical Team Coordination Center
Beijing 100029, China
*Corresponding author, e-mail: hetaotao1980@gmail.com

Abstract

This article study chip communication mechanisms deeply based on the NoC (Network on Chip) technology and a load equalization of virtual channel (LEVC) routing algorithm for Torus networks is proposed, which splits the physical network into virtual networks in the case of physical link load equalization. According to the predefined rules, packets use different virtual networks on their way to destinations and network achieve no deadlock, no livelock purpose. The algorithm study load equalization on the level of virtual channel by introducing the random factor. Simulation results show that compared with the negative first for Torus networks algorithm, the network performance can be dramatically improved for different traffic pattern by selecting a specific value of the random parameter respectively.

Keywords*: network on chip, virtual channel, load equalization*

Copyright © 2013 Universitas Ahmad Dahlan. All rights reserved.

1. Introduction

SoC (System on Chip) integrates microprocessors, analog IP cores, digital IP cores and storage units on a single chip based on reuse of IP (Intellectual Property) with embedded system and nanotechnology. The main idea of the SoC is design reuse, connecting reusable IP cores on-chip. Researchers do a lot of improvement and standardization of work on chip bus, but, with the sharp increase of the number of IP cores contained in the SoC, SoC technology based existing communication bus structure is facing huge challenges in the aspects of power consumption, latency, reliable and so on [1-5]. Around 2001, some research institutions learn and absorb the thoughts of communications network to presents SoC chip communication methods (NoC, Network on Chip) taking communication as the core, which solves the problems been facing by complex SoC [6-9]. In recent years, with the development of NoC, it has become communication infrastructure of SoC replacing the on-chip bus. NoC is on-chip communication network between computing resources and storage resources [10]. This paper focuses on one of the core issues of NoC network layer, the Routing algorithm.

The routing algorithm is used to establish transmission path between the sending node and the receiving node of each message or packet. Routing algorithms can be classified according to different standard, which are shown in Table 1.

Table 1. Routing Algorithm Classification

basis	content	basis	content
addresses number	Unicast Routing Multicast Routing	path length	Shortest Path Routing Misrouting
decision position	Centralized Routing Distributed Routing	path certainty	Deterministic Routing Adaptive Routing

Under conditions of a given topology, the NoC performance is determined by the switching technology and routing algorithm used in NoC. Some routing algorithms based on mesh are proposed in the literature. Dimension order routing is that the message does not enter into the next dimension until the offset of current dimension is reduced to zero. The XY routing

Received April 1, 2013; Revised June 17, 2013; Accepted July 15, 2013

and eCube routing is typical dimension order routing. Glass and Ni proposed turn model in 1992 [11]. It provided the method of developing the minimum or non-minimum routing algorithm under a given network conditions. The literature [12] proposed three kinds of adaptive routing algorithm, West-First, North-Last and Negative-First. On the basis of the literature [11], Chiu proposed the odd-even turn model [13] in 2000, so that the network performance was more stable and balanced. The appearance of malfunction nodes or links in the NOC communication network is inevitable. Therefore, the scholars try to study the fault-tolerant routing algorithm. That is the routing algorithm is able to ensure the normal communication between other normal nodes when the malfunction nodes or links appeared in communication network. Wu proposed EX-XY routing algorithm based on the odd-even turn model [14] in 2003. Applying the method of constructing fault block proposed by Wu in 2D-mesh, the EX-XY routing algorithm can be tolerant well. However, the fault-tolerant misrouting decreases the network performance. To solve this problem, Lin and Huang proposed OAPR routing algorithm in [15], which makes network performance stable in a fault-tolerant process. In [16], Zou proposed another fault-tolerant routing algorithm NARCO based on turn model, which successfully combines mechanisms of odd-even dimension order routing and inverted odd-even turn dimension order routing. In [17], Li and Gu proposed a fault-tolerant routing algorithm based on innovative artificial potential field (APF) model, applying the basic principles of the electric field in physics to the NoC routing.

Virtual channel often combines with the routing algorithm to achieve efficient data transmission. The virtual channel divides a physical link into multiple virtual channels, so as to solve the packets blocking problem of wormhole switchi ng by allowing other packets to use the idle link bandwidth in the case of packet block. In addition, the virtual channel is also widely used in deadlock avoidance. Boppana and Suresh introduce the concept of fault rings with four virtual channels for deadlock-free in [18]. Introduction of virtual channels simplifies the routing algorithm design, but brings new problem, that is system need to increase the additional buffer space and complex control logic, which greatly improve the cost. Z. Zhang proposes a single fault node tolerant algorithm for the Mesh structure in [19]. The algorithm uses XY algorithm in the fault-free area and avoids deadlock by prohibiting data turning in the northeast corner when data bypasses faulty nodes. Bypassing regularity of this algorithm is strong and the algorithm complexity is low, but the load on the bypass loop is heavy, which affects the performance of NoC. An adaptive load equalization routing algorithm is proposed in [20]. In this algorithm, network load is distributed to the different physical links in balance according to the routing algorithm and current status of the network. However, load equalization on each physical link cannot ensure the load equalization on the corresponding virtual channel of physical link.

In general, each virtual channel has virtual channel buffers of fixed size. Virtual channel load imbalance refers to the state that some of the virtual channel buffers are long-term saturated while other virtual channel buffers are not utilized fully. Therefore, on virtual channel allocation the routing algorithm should try to equalize the use of the virtual channels, thereby enhancing the entire NoC communication performance. This paper is to study load allocation strategy of routing algorithm further based on physical link load equalization further, and proposes a load equalization routing algorithm on virtual channel level in Torus.

2. Load Equalization of Virtual Channel Routing Algorithm in Torus

2.1. Related Definitions and Concepts

For convenience of description below, first some concepts will be defined.

Definition 1. Direct connecting network can be defined as a directed graph $G=<V,C>$. V are the nodes in the graph and C is the set of directed links. A directed link may be defined as an ordered pair $c=<v_a,v_b>\in C$, where $v_a,v_b\in V$.

Definition 2. An n-dimensional Torus network is a direct connection network $G=<V,C>$, which are constituted by $k_0\times k_1\times\cdots\times k_{n-1}$ nodes. k_i represents the number of i^{th} dimensional nodes. Each node in the network can be expressed by an n-dimensional vector $v_x=(x_0,x_1,\cdots x_{n-1}),0\le x_i\le k_i-1$. C represents the links set of n-dimensional Torus network. $C=\{c\,|\,c=<v_x,v_y>,x_i=(y_i\pm 1)\bmod k_i\wedge(\forall j\ne i,x_j=y_j)\}$.

Definition 3. In a 2D-Torus network, positive channel on dimension i is $c = <(x_0, x_1), (y_0, y_1)>, x_i = (y_i - 1) \bmod k_i$, and negative Channel is $c = <(x_0, x_1), (y_0, y_1)>, x_i = (y_i + 1) \bmod k_i$. Corresponding direction is positive (negative) direction and positive (negative) channel set is marked as $C^+ (C^-)$. Left channel is $c = <(x_0, x_1), (y_0, y_1)>, x_i \le \lfloor k_i/2 \rfloor \wedge y_i \le \lfloor k_i/2 \rfloor$ and right channel is $c = <(x_0, x_1), (y_0, y_1)>, x_i \ge \lfloor k_i/2 \rfloor \wedge y_i \ge \lfloor k_i/2 \rfloor$. The sets of left and right channels are denoted respectively as C^L and C^R.

Definition 4. In network $G = <V, C>$, each physical link can be divided into two or more virtual channels. A virtual network $G_v = <V, C_v>$ is constituted by network nodes V and virtual channels C_v. In TNF algorithm [20], each physical link is divided into three virtual channels recorded as 0, 1, and 2. Entire network can be divided into two virtual networks, $G_v^1 = <V, C_v^*>$ and $G_v^2 = <V, C_v^\#>$, where $C_v^* = C_v^0 \cup \{C_v^1 \cap [(C_v^+)^L \cup (C_v^-)^R]\}$, $C_v^\# = C_v^2 \cup \{C_v^1 \cap [(C_v^-)^L \cup (C_v^+)^R]\}$. C_v^i is the set of all i^{th} virtual channels. Figure 1 shows an example of the two virtual networks.

Definition 5. Channel alteration line in the virtual network refers to the boundary used by a virtual channel. When a packet in the network goes across the channel alteration line, the data packet must use another number of virtual channels for data transmission. CAL_{v1}^x is an alteration line in G_v^1, between the nodes of $(k_x - 1, y)$ and $(0, y)$, and is perpendicular to $<(k_x - 1, y), (0, y)>$. CAL_{v2}^x is an alteration line in G_v^2, between the nodes of $(\lfloor k_x/2 \rfloor - 1, y)$ and $(\lfloor k_x/2 \rfloor, y)$, and is perpendicular to $<(\lfloor k_x/2 \rfloor - 1, y), (\lfloor k_x/2 \rfloor, y)>$. Similarly, CAL_{v1}^y and CAL_{v2}^y can also be defined. Channel alteration lines in 4×4 Torus network are shown in Figure 1.

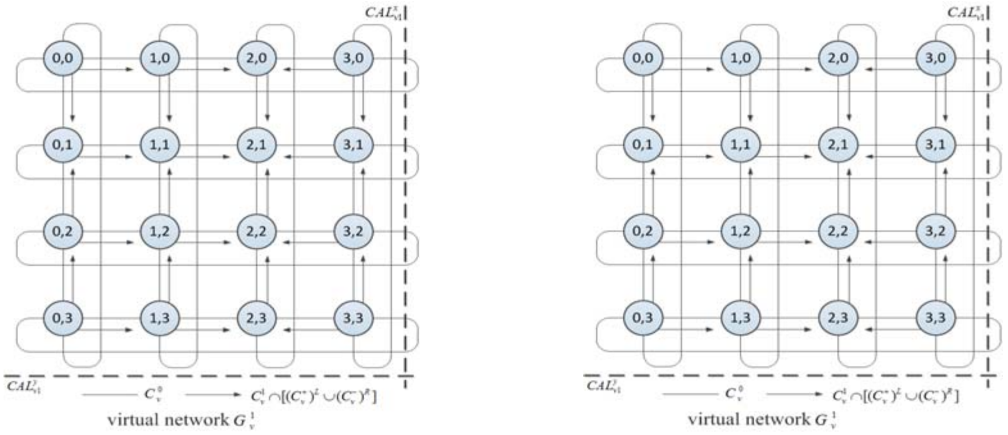


Figure 1. Virtual Network and Channel Alteration Line

The TNF algorithm uses the virtual channel allocation strategy based on turn model in order to avoid deadlock.

(1) In initial condition, when a new packet is injected into the network, the virtual channel in C_v^0 of virtual network G_v^1 is used for data transmission.

(2) In G_v^1, if a packet is going to go across a channel alteration line CAL_{v1} in routing process, it will select a virtual channel in $(C_v^* - C_v^0)$ of virtual network G_v^1 to transmit instead of current virtual channel.

(3) In G_v^1, once the packet routing occurs turn according to routing algorithm, the virtual channel in C_v^2 of virtual network G_v^2 is selected.

(4) Once the packet in G_v^2 is going to cross a channel alteration line CAL_{v2}, it will select a virtual channel in $(C_v^\# - C_v^2)$ of virtual network G_v^2 for routing.

Using the virtual channel allocation strategy, TNF algorithm provides higher network performance compared with dimension order algorithm, but corresponding virtual channel load of each link is extremely unbalanced by analysis and simulation. The experimental results show that most of the traffic is assigned to the virtual channel 0, and the virtual channel 2 has not been fully utilized. The use of virtual channels in uniform traffic pattern, hot traffic pattern and transpose traffic pattern are shown in Figure 2.

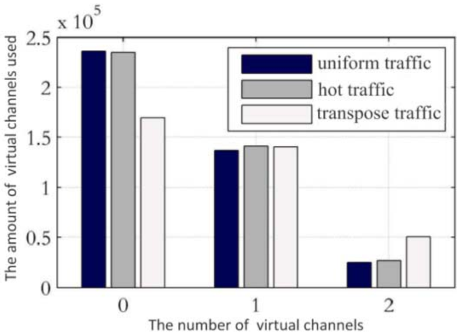


Figure 2. Virtual Channel Load Distribution in TNF Algorithm

It can be seen that load on virtual channel 0 is about 4 to 9 times heavier than that on virtual channel 2. This unbalanced load on the virtual channel affects further enhance of the system performance.

2.2. Improvement of Routing Algorithm

By theoretical analysis and experimental simulation, we find that the main reason for the uneven distribution of the load on the virtual channel lies in virtual channel allocation strategy in TNF algorithm. The algorithm will not use the virtual channel 2 for packet routing until packet is about to turn and cross CAL. Therefore, using C_v^0 for most of the packets routing results in overload on virtual channel 0 and insufficient load on virtual channel 2. The LEVC routing algorithm introduces random factor $p(0 \le p \le 1)$ and modify virtual channel allocation strategy, which makes part of the packets in the routing process to enter virtual channel 2 in advance with probability p. Improved virtual channel allocation strategy is described below.

(1) When a new packet is injected into Torus network, algorithm select C_v^0 of virtual network G_v^1 for packet routing.

(2) In G_v^1, when a right-angle turn occurs, the packet is transmitted directly to C_v^2 of virtual network G_v^2 with probability p.

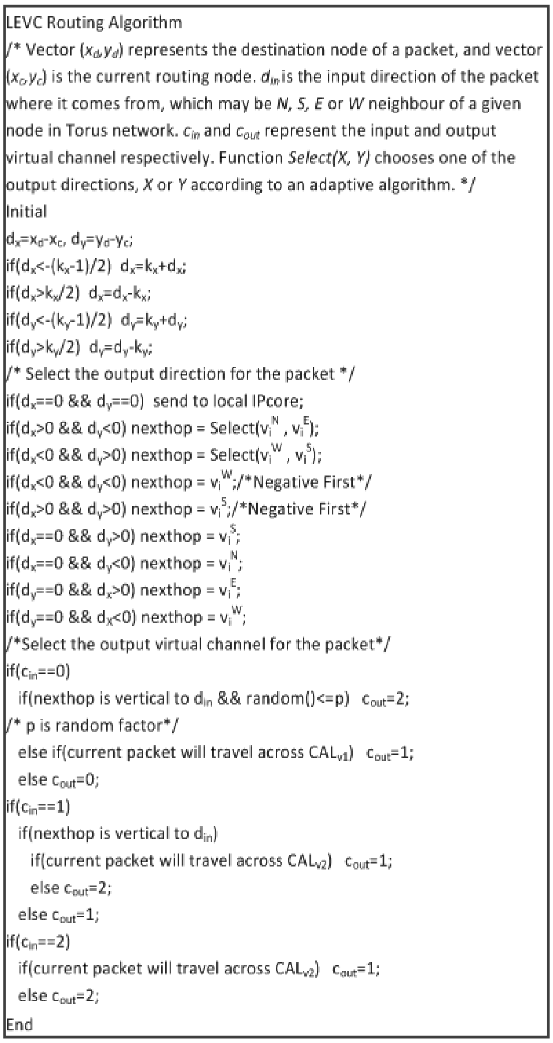
(3) In G_v^1, If a packet is about to cross the CAL_{v1}, then is selected.

(4) Once the packet routed in virtual channel in $(C_v^* - C_v^0)$ of virtual network G_v^1 turns, the virtual channel in C_v^2 of virtual network G_v^2 is selected for packet routing.

(5) Once the packet routed in virtual channel in C_v^2 of virtual network G_v^2 is about to cross the CAL_{v2}, the virtual channel in $(C_v^\# - C_v^2)$ of virtual network G_v^2 is selected for packet routing.

The core idea of LEVC routing algorithm is to forward the packets in C_v^0 of virtual network G_v^1 to the virtual network G_v^2 as soon as possible compared with channel allocation strategy in TNF algorithm, which can reduce the occupancy of C_v^0 of virtual network G_v^1 and improve utilization of C_v^2 of virtual network G_v^2. In this routing algorithm, when a packet turns, the packet will be transmitted to the virtual network G_v^2 with the probability p regardless of whether the packet is about to cross the CAL.

In Torus network, each routing node v is directly connected to its four neighbors denoted as E, S, W, N, and the local node is referred to as IPcore. According to the virtual network division basis and virtual channel allocation strategy above, the following Figure 3 is pseudo code of LEVC routing algorithm. In this algorithm, the virtual channel allocation strategy just forwards part of packets in advance to the virtual network G_v^2 from the virtual network G_v^1, and does not change the order of using virtual channel. All packets routing processes use these virtual channels in accordance with the numbering sequence (ascending or descending). In addition, it can be seen from the routing algorithm pseudo code that once the packet enter G_v^2 from G_v^1, it will not return to G_v^1. So the use of two virtual networks is also ordered. In a word, the routing algorithm is deadlock-free and livelock-free.



```
LEVC Routing Algorithm
/* Vector ($x_d$,$y_d$) represents the destination node of a packet, and vector
($x_c$,$y_c$) is the current routing node. $d_{in}$ is the input direction of the packet
where it comes from, which may be N, S, E or W neighbour of a given
node in Torus network. $c_{in}$ and $c_{out}$ represent the input and output
virtual channel respectively. Function Select(X, Y) chooses one of the
output directions, X or Y according to an adaptive algorithm. */
Initial
$d_x$=$x_d$-$x_c$, $d_y$=$y_d$-$y_c$;
if($d_x$<-($k_x$-1)/2)  $d_x$=$k_x$+$d_x$;
if($d_x$>$k_x$/2)  $d_x$=$d_x$-$k_x$;
if($d_y$<-($k_y$-1)/2)  $d_y$=$k_y$+$d_y$;
if($d_y$>$k_y$/2)  $d_y$=$d_y$-$k_y$;
/* Select the output direction for the packet */
if($d_x$==0 && $d_y$==0)  send to local IPcore;
if($d_x$>0 && $d_y$<0) nexthop = Select($v_i^N$ , $v_i^E$);
if($d_x$<0 && $d_y$>0) nexthop = Select($v_i^W$ , $v_i^S$);
if($d_x$<0 && $d_y$<0) nexthop = $v_i^W$;/*Negative First*/
if($d_x$>0 && $d_y$>0) nexthop = $v_i^S$;/*Negative First*/
if($d_x$==0 && $d_y$>0) nexthop = $v_i^S$;
if($d_x$==0 && $d_y$<0) nexthop = $v_i^N$;
if($d_y$==0 && $d_x$>0) nexthop = $v_i^E$;
if($d_y$==0 && $d_x$<0) nexthop = $v_i^W$;
/*Select the output virtual channel for the packet*/
if($c_{in}$==0)
  if(nexthop is vertical to $d_{in}$ && random()<=p)   $c_{out}$=2;
/* p is random factor*/
  else if(current packet will travel across $CAL_{v1}$)   $c_{out}$=1;
  else $c_{out}$=0;
if($c_{in}$==1)
  if(nexthop is vertical to $d_{in}$)
    if(current packet will travel across $CAL_{v2}$)   $c_{out}$=1;
    else $c_{out}$=2;
  else $c_{out}$=1;
if($c_{in}$==2)
  if(current packet will travel across $CAL_{v2}$)   $c_{out}$=1;
  else $c_{out}$=2;
End
```

Figure 3. Pseudo Code of LEVC Routing Algorithm

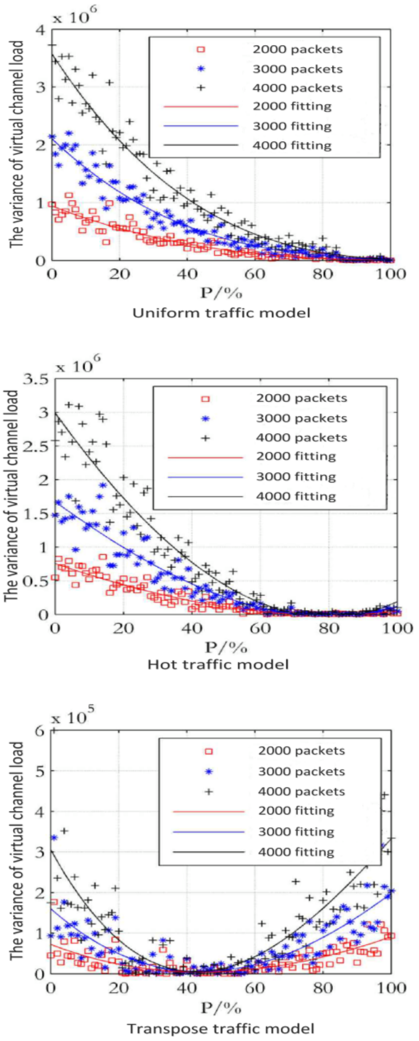


Figure 4. Virtual Channel Load Distribution with Factor P

3. Simulation and Evaluation of the Algorithm

This section evaluates the performance of LEVC routing algorithm taking TNF algorithm as a reference. This paper uses Nirgam simulation platform to model and simulate LEVC and TNF algorithm based the System C. Evaluation carries out in three aspects that is virtual channel load distribution, network latency and normalized throughput. Uniform traffic model, hot traffic model and transpose traffic model are three typical test flow models used in performance simulation.

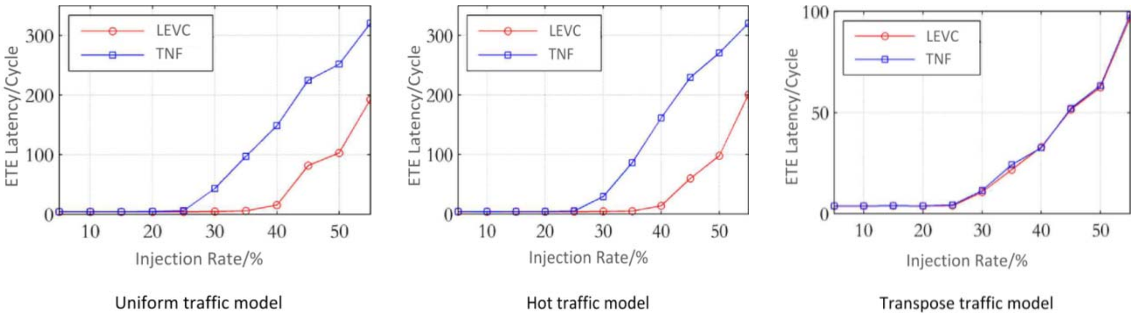
3.1. Virtual Channel Load Distribution

In the algorithm, random factor p decides the number of entering to virtual network G_v^2 from the virtual network G_v^1 in advance. When $p=0$, there is no traffic entering G_v^2 from G_v^1. In this case virtual channel loads of LEVC algorithm and TNF algorithm are the same. When $p=1$, once the packet in C_v^0 occurs turn, it is injected to C_v^2 immediately and this leads to a large number of packets entering G_v^2 from G_v^1 in advance, which relieves pressure on the virtual channels 0 and increases the utilization of virtual channel 2. When $0 \leq p \leq 1$, load distribution is between the two extremes. In order to accurately measure the load distribution of each virtual channel, the variance ($Var(X)=E[(X-E(X))^2]$) of number (X) of utilization virtual channel is introduced to describe the equalization distribution degree of virtual channel load. The smaller variance is, the more balanced load distribution is. Figure 4 shows the three flow model simulations with three traffic and random factor $p\,(0 \sim 1)$.

In order to describe the distribution law more clearly, the flow distribution in the Figure 4 uses a polynomial curve fitting. It can be seen that variance of load distribution is minimum ($Var(X) \approx 0$) for uniform flow model when random factor $p=1$. That is to say, load distribution achieves ideal equalization; for hot traffic model, when the random factor $p=0.75$, the load distribution achieves ideal equalization; for transpose traffic model, when the random factor $p=0.45$, the load distribution achieves ideal equalization. From the simulation results above, in order to achieve an equalization distribution of the load, the selection of the random factor p for corresponding traffic models is different. When $p=0.75$, load distributions for three typical flow models are acceptable. Therefore, in the following discussion, select uniform $p=0.75$.

3.2. Network Average Latency

In this paper, the average end-to-end latency is used to describe the network performance, which is shown in Figure 5.



Uniform traffic model

Hot traffic model

Transpose traffic model

Figure 5. The Average end-to-end Latency

Figure 5 shows the corresponding end-to-end latency characteristic of LEVC algorithm and TNF algorithm in different flow models, as the network load increases and $p=0.75$. With the raise of injection rate, the increasing trend of end-to-end latency emerges. In uniform traffic model and hot traffic model, when the injection rate reaches to 25%, the latency of TNF algorithm starts to increase sharply, but for LEVC algorithm, when the injection rate reaches to 40%, the latency just begins to increase rapidly. It can be seen that LEVC algorithm outperforms TNF algorithm in the network latency besides in transpose traffic model.

3.3. The Normalized Throughput of the Network

The normalized throughput of the network refers to the ratio of the number of packets transmitted successfully to the number of packets sent into network, which is used to measure network throughput performance. Figure 6 shows the corresponding normalized throughput trend of LEVC algorithm and TNF algorithm in different flow models, as the network load increases and $p = 0.75$. In uniform traffic model and hot traffic model, when the injection rate reaches to 25%, the normalized throughput of TNF algorithm starts to reduce sharply(a large number of packets loss), but for LEVC algorithm, when the injection rate reaches to 40%, the latency just begins to reduce rapidly. It can be seen that equalization utilization of virtual channels can greatly improve the communication capability of the network and LEVC algorithm outperforms TNF algorithm in the normalized throughput of network besides in transpose traffic model.

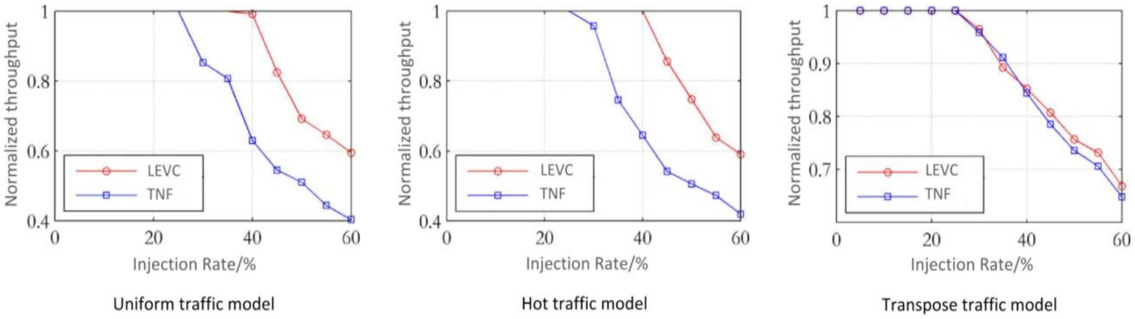


Figure 6. The Normalized Throughput

3.4. Discussion

As can be seen from Figure 5 and Figure 6, uniform traffic model and hot traffic model, network latency and throughput have significant performance improvement with LEVC algorithm for uniform traffic model and hot traffic model. However, for transpose traffic model, the performance has no significant improvement. The reason for this phenomenon is that, in transpose traffic model, the corresponding virtual channel load distribution of LEVC algorithm and TNF algorithm are similar when $p = 0.75$ in LEVC algorithm (LEVC algorithm is equal to TNF algorithm when $p = 0$). Therefore, there is no significant improvement on load equalization of each virtual channel and on the network performance. In transpose traffic model, if adaptive random factor p is chosen, for example $p = 0.45$, the algorithm can improve the load distribution of virtual channel and the network communication performance significantly. In order to achieve the load equalization of virtual channels, algorithm must select an appropriate values of factor p based on the traffic model in actual application to improve communication performance of the network.

4. Conclusion

In order to improve the communication efficiency of Torus networks in further, this paper proposes load equalization of the virtual channel routing algorithm on the level of virtual channel. The algorithm allocates network load to each virtual channel in equalization, which makes packets use different virtual channels in different virtual networks for routing so as to avoid deadlock and livelock. Compared with the negative first algorithm for Torus networks, the simulation results show that the network latency and throughput can be improved by selecting the appropriate random factor for different traffic models, which significantly improves the communication performance of Torus networks.

Acknowledgements

This work was supported by Main project of National 863 Program “2009AA012201” and the key scientific and technological projects of Science and Technology Commission of Shanghai Municipality “08dz501600”.

References

[1] Dally WJ, Towles B. *Route packets, not wires: On-chip interconnection networks.* Proceedings of Design Automation Conference 2001. Las Vegas. 2001; 684-689.
[2] Hemani A, Jantsch A, Kumar S. Network on chip: An architecture for billion transistor era. Proceeding of the IEEE NorChip Conference. Turku. 2000; 166-173.
[3] Rahmati D, Murali S, Benini L, Angiolini F. Computing Accurate Performance Bounds for Best Effort Networks-on-Chip. *IEEE Transactions on Computers*. 2013; 62(3): 452-467.
[4] Marculescu R, Ogras UY, Li-Shiuan P, Jerger NE, Hoskote Y. Outstanding research problems in NoC design: system, microarchitecture, and circuit perspectives. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. 2009; 28(1): 3-21.
[5] Ho R, Mai KW, Horowitz MA. The future of wires. *Proceedings of the IEEE*. 2001; 89(4): 490-504.
[6] Benini L, De Micheli G. *Powering networks on chips: energy-efficient and reliable interconnect design for SoCs.* Proceedings of the 14th international symposium on Systems synthesis. Montreal. 2001; 33-38.
[7] Kumar S, Jantsch A, Millberg M. *A network on chip architecture and design methodology*. Proceedings of IEEE Computer Society Annual Symposium on VLSI. Pittsburgh. 2002; 105-112.
[8] Jantsch A, Tenhunen H. Networks on chip. Netherlands: Springer Press. 2003.
[9] Benini L, De Micheli G. Networks on chips: Technology and Tools. San Francisco: Morgan Kaufmann Press. 2006.
[10] Salminen E, Kulmala A, Hamalainen T D. Survey of Network-on-chip Proposals. *OCP-IP white paper*. 2008; 1-13.
[11] Glass CJ, Ni LM. *The turn model for adaptive routing.* Proceedings of the 19th International Symposium on Computer Architecture. Gold Coast. 1992; 278-287.
[12] Dally W, Towles B. Principles and Practices of Interconnection Networks. San Francisco: Morgan Kaufmann Press. 2004.
[13] Chiu G. The Odd-even Turn Model for Adaptive Routing. *IEEE Transactions on Parallel and Distributed Systems*. 2000; 11(7): 729-738.
[14] Jie W. Fault-tolerant and deadlock-free routing protocol in 2D meshes based on odd-even turn model. *IEEE Transactions on Computers*. 2003; 52(9): 1154-1169.
[15] Lin SY, Huang CH, Chao CH. Traffic-balanced Routing Algorithm for Irregular Mesh-based on-Chip Networks. *IEEE Transactions on Computers*. 2008; 57(9): 1156-1168.
[16] Zou Y, Pascricha S. NARCO: Neighbor Aware Turn Model based Fault Tolerant Routing for NoCs. *IEEE Embedded Systems Letters*. 2010; 2(3): 85-89.
[17] Li YH, Gu HG. Fault tolerant routing algorithm based on the artificial potential field model in Network-on-Chip. *Applied Mathematics and Computation*. 2010; 217: 3226-3235.
[18] Boppana RV, Chalasani S. Fault-tolerant wormhole routing algorithms for mesh networks. *IEEE Transactions on Computers*. 1995; 44(7): 848-864.
[19] Zhang Z, Greiner A, Taktak S. *A Reconfigurable Routing Algorithm for a Fault- Tolerant 2D-Mesh Network-on-chip.* Proceedings of 45th ACM/IEEE Design Automation Conference. Anaheim. 2008; 441-446.
[20] Gu H, Liu Z, Wang K. Distributed adaptive routing algorithms in Torus networks. *Journal of Xidian University*. 2006; 33(3): 352-358.