

# Optimal Many-to-Many Personalized Concurrent Communication in RapidIO-based Fat-trees

Lin Shu, Jie Hao, Yafang Song, Chengcheng Li, Donglin Wang

Institute of Automation, Chinese Academy of Sciences

Beijing, China

{lin.shu, jie.hao, yafang.song, chengcheng.li, donglin.wang}@ia.ac.cn

**Abstract**—Many-to-many ( $M$ -to- $N$ ) personalized concurrent communication is one of the most dense traffic patterns in high performance computing system. Each node in group  $M$  sends different messages to every node in group  $N$ . Hot-spot congestions are the bottleneck of this communication pattern. In this paper, a proactive congestion avoidance mechanism is proposed to optimize many-to-many personalized concurrent traffic in RapidIO-based fat-trees. And a heuristic scheme of virtual destination is proposed to deal with the case that messages from each node in  $M$  group to every other node in  $N$  group are not of the same size. Our proactive congestion avoidance mechanism has been used in a practical network, which is a part of digital signal processing platform for “*the TianLai Pathfinder*”. The network delay is effectively improved, reducing from about 34.784 us to 1.484 us. And the communication overhead is decreased by twice.

**Keywords**—many-to-many; personalized; RapidIO; Fat-Tree; node-level; congestion-avoidance

## I. INTRODUCTION

In the many-to-many ( $M$ -to- $N$ ) personalized concurrent communication, each node in group  $M$  sends different messages to every node in group  $N$ . It is one of the most dense traffic patterns. It occurs in many high performance computing applications, like very large scale array signal processing[1]. Real-time performance is a kind of significant metrics, including computation overhead and communication overhead. As the scaling of computing systems, communication overhead is becoming the bottleneck. During many-to-many personalized concurrent traffic, congestion is the most important factor resulting in degradation of network performance, such as the increase of latency and the loss of throughput.

Many previously proposed approaches have focused on congestion control techniques[2-6], which are reactive schemes when detecting congestions. In this paper, we propose a proactive congestion avoidance mechanism to optimize many-to-many personalized concurrent traffic in fat-tree network[7] based on RapidIO[8]. Fat-trees have full bisection bandwidth, which is an advantage in data-intensive applications. As the network scales, the bisection bandwidth increases as well. RapidIO is a lossless protocol, supporting efficient transmitting of short messages. And the associated switch is power-efficient.

The main contributions in this paper are summarized as follows:

- As the knowledge we know, it is the first time that RapidIO is introduced to a large network application (extended 8-ary 3-tree).
- A modified deterministic routing algorithm for fat-tree is proposed so that potential congestions are predictable.
- A proactive congestion avoidance mechanism is proposed to optimize many-to-many ( $M$ -to- $N$ ) personalized concurrent communication. When  $M < N$  or  $M = N$ , a node-level communication schedule based on contiguous task mapping scheme can avoid hot-spot congestion totally without any extra overhead. When  $M > N$ , a hierarchical scheme is proposed.
- A heuristic scheme of virtual destination is proposed so that messages from each node in  $M$  group to every other node in  $N$  group (including the virtual destinations) are of the same size. And the efficient

implementation is related to Rapid's high efficiency in short message communication.

The rest of the paper is organized as follows. Related works are discussed in section II. We introduce RapidIO-based fat-tree networks and analyze the routing algorithms of fat-trees in section III. In section IV, specifications of many-to-many personalized concurrent communication are discussed, and we analyze how the hot-spot congestion generates and how it impacts the network performance. Then a proactive congestion avoidance mechanism is proposed in section V and experiments and performance analysis are presented in section VI. Section VII concludes the paper.

## II. RELATED WORK

Many reactive congestion control techniques have been proposed. Explicit Congestion Notification (ECN) of InfiniBand Architecture (IBA) is the most popular mechanism which has been adopted by many network systems[2, 3]. It is effective in combating congestion. But studies[2, 3] have also pointed out limitations such as reduced system stability, the need for parameter adjustment, and slow congestion response time. Another kind of schemes use the queues at each switch port to prevent congestion spreading, like OBQA[4, 5] and BBQ[6].

Besides, proactive mechanisms to avoid congestion have been discussed. Jiang et al.[9] introduced the Speculative Reservation Protocol to avoid network hot-spot congestion. It uses a lightweight reservation handshake to avoid endpoint congestion and allows the transmission of lossy speculative messages to make reservation overhead. Similar congestion control schemes are proposed in[10]. The approaches in [9, 10] are effective when traffic occurs in bursts, but not in concurrent dense traffic.

A large body of researches has proposed high-level congestion-control solutions. Luo et al.[11] presented a core stateless optimization approach based on open loop end-point throttling, improving all-to-all collective performance in Cray and Berkeley UPC. Zahavi et al.[12] proposed a fat-tree routing algorithm that provides a congestion-free all-to-all shift pattern for the InfiniBand static routing. Kumar and Kale[13] focused on optimizing all-to-all multicast on fat-tree networks. Yang and Wang[14]

discussed how to optimize all-to-all broadcast on meshes and tori.

In summary, the low-level congestion control mechanisms based on protocol or switch are mostly based on InfiniBand Architecture. And they are effective in burst traffic, but not in concurrent dense communication. High-level algorithmic solutions are associated with network topology and traffic pattern. Many of them focused on collective communication, like all-to-all multicast or broadcast. Each node sends the same message to every node in the network. Although some researches[15, 16] have also been worked on personalized communication, only all-to-all exchange pattern have been discussed. In this paper, we focus on the optimization of many-to-many personalized concurrent communication in RapidIO-based fat-trees.

## III. RAPIDIO-BASED FAT-TREE NETWORKS

The network topology of fat tree is quite common in high performance computing systems. It enables high throughput in data-intensive applications because of full bisection bandwidth. As is known, many supercomputer systems' fat-trees are based on ethernet, InfiniBand or other similar customized high speed protocols. In this paper, we introduce a RapidIO-based fat-tree network. RapidIO is usually used in embedded applications, supporting processor-to-processor and board-to-board interconnection. It is also a good choice in high performance computing. While retaining high bandwidth-cost rate and low latency, RapidIO is especially efficient in short message communication which can achieve small time granularity in on-line applications so as to improve real-time performance. In this section we will introduce RapidIO's advantages compared to ethernet in high real-time systems. And then, fat-tree networks' routing algorithms are discussed in order to match the requirements of many-to-many personalized concurrent communication.

### A. RapidIO

RapidIO supports packet-switched point-to-point interconnection. It defines a three-layer architectural hierarchy including the logical layer, the transport layer and the physical layer. RapidIO packet format is showed in Fig.1. Target/Source address (device ID) can be set to be 8 bits or 16 bits or 32 bits. It allows systems with up to

4,294,967,296 endpoints, which is large enough for a high performance computing system.

RapidIO supports I/O-oriented programming model. It is a non-coherent DMA transfer that enables offload processor. The maximum transaction size is 256 bytes. Fig.2 shows comparisons of protocol efficiency[17]. Although the maximum payload efficiency of RapidIO is only about 93%, it can achieve quite better performance than ethernet in short message traffic. When the message size is 26 bytes, the improvement in terms of payload is 2.63 times. The latency per hop of RapidIO switch (4x mode, 5Gbaud) is about 110.6 ns referred to IDT 1848, while the ethernet's latency of one hop is tens of microseconds. Therefore, a system based on RapidIO can reduce communication overhead to achieve high real-time performance.

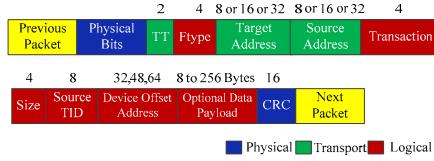


Fig. 1. RapidIO Packet Format: only the size of data payload is counted by byte and the other are counted by bit.

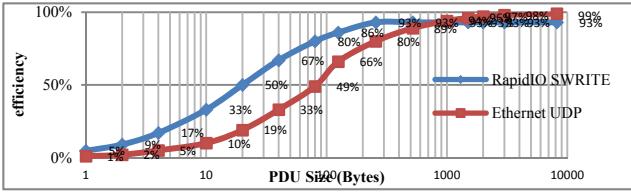


Fig. 2. Comparisons of Protocol Efficiency.

### B. Fat Tree Topology

The fat-tree topology is commonly implemented by building a tree with multiple roots, often following the  $k$ -ary  $n$ -tree definition[18]. It is easy to scale to be a network with  $2k^n$  nodes and  $(2n-1) k^{n-1}$  switches. A 2-ary 3-tree is showed in Fig.3 and Fig.4 shows the scalability.

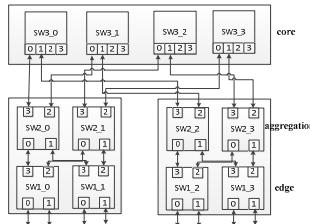


Fig. 3. 2-ary 3-tree: 8 nodes, 12 switches.

The fat-tree pattern eases the implementation of different efficient routing algorithms[19]. Several paths are possible when advancing upward, making possible adaptive routing.

Adaptive routings can balance traffic, especially in burst communication. But, we focus on traffic-aware dense pattern in this paper. If an adaptive routing algorithm is implemented in this system, the congestions caused by hot spots will be much complicated. Therefore, deterministic routing algorithms are preferred in this application.

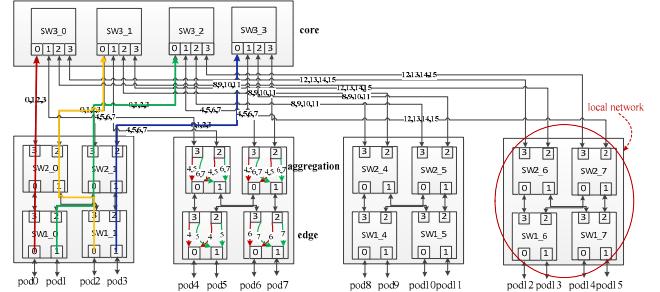


Fig. 4. Extended 2-ary 3-tree: 16 nodes, 20 switches. Deterministic Source mod  $k$  Routing Algorithm is showed using colored links.

Existing deterministic routing algorithms mainly include destination mod  $k$  ( $D\text{-mod-}k$ ) routings and source mod  $k$  ( $S\text{-mod-}k$ ) routings (Fig.4)[20]. The main difference of  $D\text{-mod-}k$  routings and  $S\text{-mod-}k$  routings is how to choose the uplinks. To establish a path from node  $s$  to node  $d$ ,

$D\text{-mod-}k$  routings choose parent  $\left\lfloor \frac{d}{k^{l-1}} \right\rfloor \bmod k$  at hop  $l$ , and

$S\text{-mod-}k$  routings choose  $\left\lfloor \frac{s}{k^{l-1}} \right\rfloor \bmod k$ . A deterministic  $S\text{-mod-}k$  routing is showed in Fig.4. Packets from the same source to different destinations are distributed through the downlinks. Concurrent communication to the same destination may cause contention in the downlinks. We analyze the contentions in section IV.

## IV. MANY-to-MANY PERSONALIZED CONCURRENT COMMUNICATION

We define many-to-many personalized concurrent traffic as  $M\text{-to-}N$  communication. In this section, we describe the specifications. And hot-spot contention is discussed in details.

### A. Specifications

For  $M\text{-to-}N$  personalized concurrent communication, we define  $M$  as the set of transmitting nodes and  $N$  is the set of receiving nodes. When  $M$  is equal to one, it is called one-to-all traffic pattern. When  $N$  is equal to one, it is called all-to-one traffic pattern.

When neither  $M$  nor  $N$  is equal to one, many hot-spots happen in worst case, which result in a catastrophic loss of throughput and the increase of delay. In order to achieve efficient network in high real-time applications, our contribution is to avoid hot-spot contentions since it is traffic-aware pattern.

### B. Hot-Spot Contention

When many sources transfer messages to the same destination concurrently, hot-spot contention happens. For a 2-ary 3-tree network in Fig.4, if sources from *pod0* to *pod3* transfer messages to *pod4* concurrently, packets from *port2* and *port3* concurrently require access to *port0* (in both switch *SW2\_2* and switch *SW2\_3*). But only one packet can cross while the others have to wait until the required output port becomes free. If contention is persistent, a packet waiting to cross will prevent other packets stored behind in the same input buffer from advancing, resulting in increasing of latency and loss of throughput. There is also contention in the lowest stage (switch *SW1\_2*). If each source works independently without congestion control, persistent hot-spot congestions are the bottleneck. When using D-mod-k routing, the generation of hot-spot contention is actually similar.

## V.NODE-LEVEL COMMUNICATION SCHEDULE TO AVOID CONGESTIONS

In last section, we have analyzed that persistent hot-spot congestions are the main bottleneck during many-to-many personalized concurrent communication. We propose a proactive congestion avoidance mechanism to optimize the traffic. It is a node-level communication-scheduling strategy, which is a high-level congestion-control scheme.

This proactive congestion avoidance mechanism needs to work with deterministic routing algorithms we discussed in section III. The uplink routing is modified: packets advancing uplinks must be forwarded to the top switch even if the top switch is not the nearest common ancestor of source and destination. It makes sure that the hops from each node to another are a constant. Then, potential congestions are aware, when many nodes transfer messages concurrently. We can use communication schedules on each node to avoid this kind of congestion.

In this paper, many-to-many personalized concurrent communication we discussed is somewhat regular, which can be divided into two categories: equal size of messages to each node (case 1) and non-equal size of messages to each node (case 2). When messages requested from one node in  $M$  group to another node in  $N$  group is not of the same size (case 2), each size is integral times as big as the minimum size. Even if it is not match with the two cases, some invalid data can be added to make it regular as case 2 since the minimum size of messages can be very small (e.g. 480 bytes in experiment 2 in section VI). Next, we will describe the node-level communication-scheduling strategies to deal with these two different cases.

### A. Equal Size of Messages to Each Node

#### 1) $M < N$ or $M = N$

Here 6-to-10 personalized concurrent communication using *S-mod-k* routing (Fig.4) is an example to describe our proposed node-level communication-scheduling strategy. It is also available when using *D-mod-k* routing. Firstly, we propose a contiguous task mapping scheme, following constraints listed below:

- The minimum  $k$ -ary- $n$ -tree (excluding the case  $n=1$ ) has  $k^2$  nodes. We define the  $k^2$  nodes as a local network (Fig.4). The transmitting nodes in  $M$  group should be mapped to the same local network. If  $M > k^2$ , the other nodes in  $M$  group are mapped to the near local networks. So do the mapping of nodes in  $N$  group.
- It is better not to map nodes in  $M$  group and  $N$  group to the same local network, if there are non-overlapping sets of nodes among group  $M$  and group  $N$ , and the total number of nodes working is smaller than the size of network.

So the nodes in  $M$  group are mapped to set (*pod0*, *pod1*, *pod2*, *pod3*, *pod4*, *pod5*). And the nodes in  $N$  group are mapped to set (*pod6*, *pod7*, *pod8*, *pod9*, *pod10*, *pod11*, *pod12*, *pod13*, *pod14*, *pod15*). Even though many nodes transmit packets concurrently, it is possible that there is no contention in the network. Node-level communication-scheduling strategy is aimed at providing a series of source-destination pairs that have no overlapping link in the network. An easy destination shuffling scheme can meet the

requirement, and the premise is following the contiguous task mapping approach previously proposed. Fig.5 shows the destination shuffling scheme. One destination table is generated according to the nodes' contiguous logical address (in sequence). The first  $M$  destination addresses in the destination table are given to the  $M$  nodes in  $M$  group sequentially as start destinations. After one cycle's transmitting, destination address for packets is shuffled to next one in the table. This operation is repeated until all the destination addresses in the table have been fetched once. It means one processing of many-to-many personalized communication is finished.

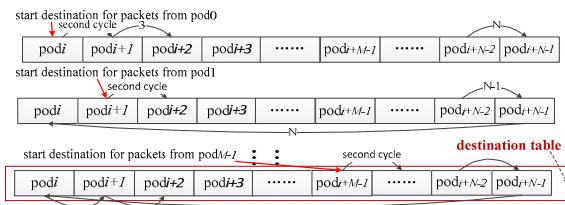


Fig. 5. An Easy destination shuffling Scheme

Table 1 Concurrent Source-Destination Pairs for 6-to-10 Communication

time (cycles)	1	2	3	4	5	6	7	8	9	10
SD Pairs	(0,6) (1,7) (2,8) (3,9) (4,10) (5,11)	(0,7) (1,8) (2,9) (3,10) (4,11) (5,12)	(0,8) (1,9) (2,10) (3,11) (4,12) (5,13)	(0,9) (1,10) (2,11) (3,12) (4,13) (5,14)	(0,10) (1,11) (2,12) (3,13) (4,14) (5,15)	(0,11) (1,12) (2,13) (3,14) (4,15) (5,16)	(0,12) (1,13) (2,14) (3,15) (4,16) (5,17)	(0,13) (1,14) (2,15) (3,16) (4,17) (5,18)	(0,14) (1,15) (2,16) (3,17) (4,18) (5,19)	(0,15) (1,16) (2,17) (3,18) (4,19) (5,20)

( $m, n$ ),  $m$  represents source  $podm$  and  $n$  represents destination  $podn$ .

Table 1 shows the concurrent source-destination pairs during successive cycles according to the destination shuffling scheme we proposed. There is no link/switch/endpoint contention during each cycle. Although there are overlapping time when packets from the same source to different destinations are being forwarded in the network, there is still no any contention. Because the hops from each node to another is the same according to the deterministic routing algorithm we proposed. So our proactive congestion avoidance mechanism is effective in many-to-many personalized concurrent communication.

## 2) $M > N$

It is obvious that in this case if the nodes in  $M$  group transfer messages concurrently, congestions happen no matter how to schedule communication. We propose a hierarchical scheme: dividing the  $M$ -to- $N$  communication into several  $M'$ -to- $N$  communications, where  $M' < N$  or  $M' = N$ . Assuming that  $i = \lceil \frac{M}{N} \rceil$ , then the first  $i-1$

communications are  $N$ -to- $N$  pattern and the last one is  $(M-mod-N)$ -to- $N$  pattern. They can be implemented efficiently using methods proposed in section V-A-1.

## B. Non-equal Size of Messages to Each Node

Each size from one node in  $M$  group to another node in  $N$  group is integral times as big as the minimum size in this case. If it is not, please adding some invalid payload. We introduce the ideal of virtual destination to avoid congestions. For example, if the minimum size of messages among source-destination pairs is  $s$  and the message size from source  $podi$  to destination  $podj$  is  $l \cdot s$ , we can add  $l-1$  virtual nodes and each virtual destination will receive messages of size  $s$  from source  $podi$ . These  $l$  destinations (including  $podj$ ) will be distributed to  $l$  different destination tables. If the maximum size is  $L \cdot s$ , it means the original communication can be divided into  $L$   $M$ -to- $N'$  personalized concurrent communications like case 1 (equal size of messages to each node), where  $N' < N$  or  $N' = N$ .

## VI. EXPERIMENTAL RESULT

### A. Experimental Setup

#### 1) RapidIO Switch

The IDT CPS-1848 is a RapidIO specification compliant central packet switch, supporting 12 ports (each port has 4 lanes). There are four CPS-1848 chips on each switch board, which can be seen as a higher-radix switch. In the following experiments, networks are built upon this switch board and each switch port works at 4x mode, 5.0 Gbaud. The theoretical latency of one hop is 111.6 ns (it is actually about 170 ns referred to our tests).

#### 2) Experiment 1

It is a 2-kay 3-tree network showed in Fig.3, which is consisted of 12 switch chips. This prototype system is used to test the basic specifications of RapidIO-based network and show how node-level communication schedule works. In this experiment, messages from each node to another node are of the same size, once many-to-many personalized concurrent communication is requested.

#### 3) Experiment 2

This is a practical network, belonging to part of digital signal processing platform for “*the TianLai Pathfinder*”, which is a project for exploration of dark energy in the cosmic space. Fig.6 shows the network topology. The

interconnection inside each local network is the same. Each switch chip named  $A$  in the aggregation layer has links to switch chips named  $A$  in the core layer. So do switch chips named  $B$ ,  $C$  or  $D$ . The deterministic  $S$ -mod- $k$  routing algorithm is implemented in this topology. Only 102 leaf nodes are available in this application. There are 48 nodes in the M group, which is mapped to the first three local networks sequentially. And the other 54 ports are in the N group, mapping to the following four local networks. Each two ports in the N group are connected to a compute node, which is consisted of 8 processors, each with 8 cores. The all 216 processors are divided into three groups, each processing one third of data during a processing cycle. In order to balance traffic in the network, the 8 processors in a compute node is divided three groups as well. Two of them include 3 processors each, and the other has 2 processors. So the traffic pattern in this application is actually 48-to-54 personalized concurrent communication, where some leaf nodes' throughput is twice as the others'. We implement three different communication schemes and will analyze the performance.

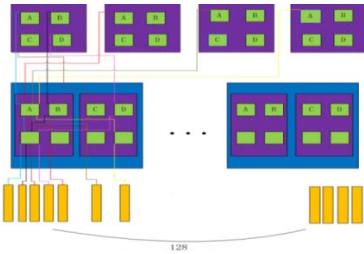


Fig. 6. Extended 8-ary 3-tree Network: the green blocks represent switch chips, the purple blocks represent switch boards, each blue block means a local network, and each yellow block represents a leaf node.

### B. Performance Analysis

Completion time ( $T$ ) of messages is used to evaluate the communication overhead, which follows the time mode below:

$$T = T_1 + mT_2 \quad (1)$$

where  $T_1$  represents the latency from the transmitter to the receiver, including three parts: the cost of processor (transmitter and receiver), the delay of copper cable, and the delay of switch.  $T_2$  represents the transfer time per byte and  $m$  is the size of messages. For a given traffic pattern, the latency  $T_1$  reflects the communication overhead. The latency in this section is counted in cycles and each cycle is 4 ns.

#### 1) Experiment 1

##### a) Single Source-Destination Pair

In this case, there is only single source-destination pair working, making sure that no congestion occurs. Fig.7 shows that the communication overhead of processors is about 608 ns (152 cycles). The delay of each hop in switch is about 170 ns. Although there is no contention in the network, the latency of successive packets forwarding to switch are not a constant as that of the endpoint to endpoint transmitting. The throughput per port can reach to 14.222 Gbps. The theoretical bandwidth is 16 Gbps. The loss of throughput is owed to the bottleneck of processors.

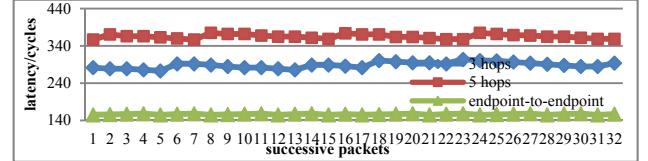


Fig. 7. The Latency of Successive Packets When Having No Congestions.

##### b) Hot-Spot Contention Injection

In this case, two different nodes transfer messages to one node concurrently, which leads to one hot-spot contention. Fig.8 shows that packets from the two source nodes are absolutely concurrent. Since hot-spot contention is persistent, it becomes congestion. But the output port in the switch won't be occupied by one input port all the time until there are no packets from this port (Fig.8-receiver). This is one advantage of RapidIO flow control mechanism.

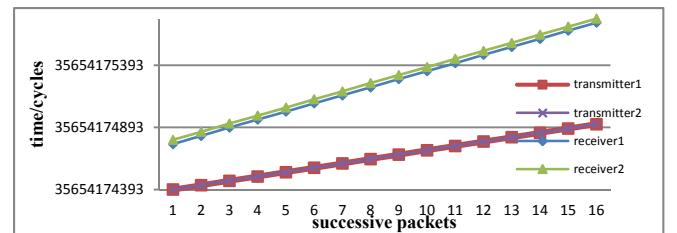


Fig. 8. The Time When Each Packet is Transmitted and Received.

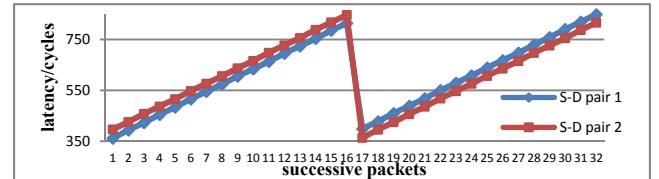


Fig. 9. The Latency of Successive Packets When Having One Hot-Spot Contention.

It is obviously that hot-spot congestion results in the increasing of latency (Fig.9, Fig. 10). Fig.9 shows two 2-to-1 communication transactions, where the average throughput is only 2.4967 Gbps, while the average throughput in the case showed in Fig.10 is nearly full and

the latency is increased continuously, resulting in head-of-line blocking at last. So reducing or avoiding hot-spot is quite significant.

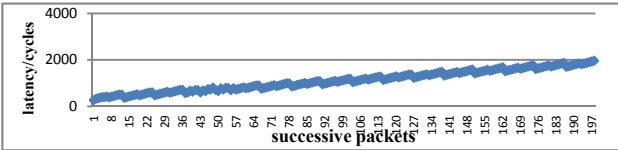


Fig. 10. The Latency of Successive Packets When Having One Hot-Spot Contention and Packets Injection Rate is High.

### c) Node-Level Communication Schedule

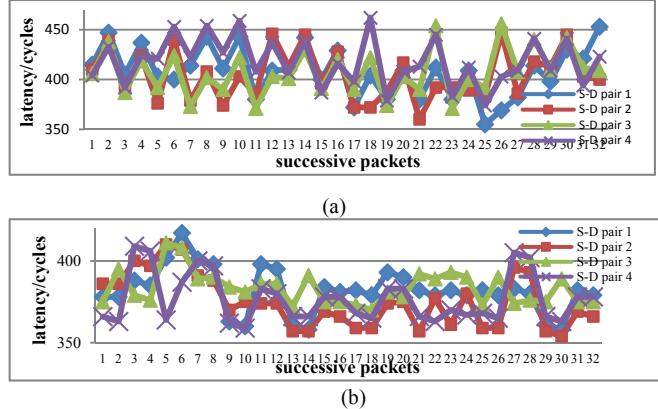


Fig. 11. 2-to-2 Personalized Concurrent Communication: (a) One Hot-Spot Exists; (b) Node-Level Communication Schedule is Implemented.

In this case, we implement two kinds of traffic patterns: 2-to-2 personalized concurrent communication (Fig.11), 2-to-4 personalized concurrent communication (the latency situation is similar to Fig.11). The two nodes in M group are mapped to one local network (*pod0, pod1* in Fig.4). And the nodes in N group are mapped to the other local network (*pod4, pod5, pod6, pod7*). So there is at most one hot-spot. It is clear showed in Fig.11 that if removing the hot-spot, the latency can be decreased about 200 ns. The efficiency of avoiding hot-spots will be much excellent in a bigger network.

### 2) Experiment 2

In this case, messages requested from one node in M group to another node in N group are not of the same size. If we implement it as many concurrent independent one-to-many traffic, the degradation of network performance is serious, especially for high real-time applications. The latency can be as long as 34.784 us (Fig.12), while the theoretical delay of transmitting through five hops is only about 1.484 us. So the communication overhead is too large. And the packets' latencies through

different routing links are quite different, which will result in additional system synchronization overhead.

If we introduce the ideal of virtual destination, the nodes in N group can be increased to 72. Then the communication can be regarded as 48-to-72 personalized concurrent traffic. If it uses a node-level communication schedule, the network congestions can be reduced. Fig.13 shows packets' latency through some classical source-destination pairs. The communication overhead has been reduced a lot. But there are still hot-spot contentions because the virtual destinations share routing links with some of the 54 nodes.

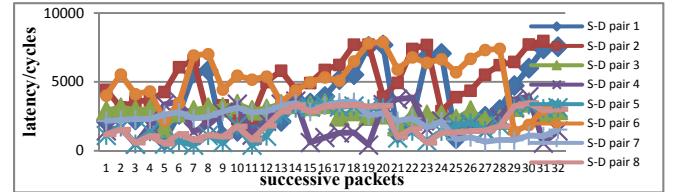


Fig. 12. Original 48-to-54 Personalized Concurrent Communication.

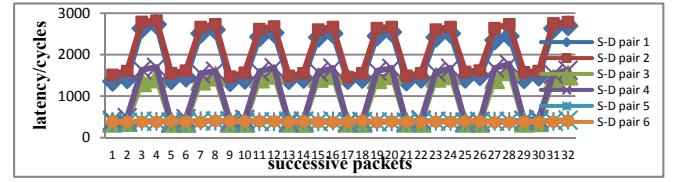


Fig. 13. 48-to-54 Personalized Concurrent Communication: Virtual Destination Based Node-Level Communication Schedule is Implemented.

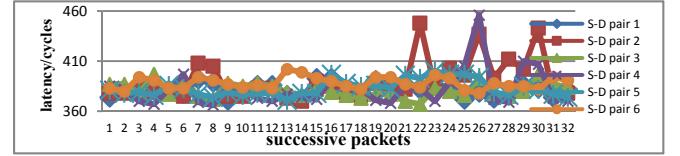


Fig. 14. 48-to-54 Personalized Concurrent Communication: Virtual Destination Based Hierarchical Scheme is Implemented.

In order to avoid hot-spots, communications to the 18 virtual destinations should be seen as a single transaction. It means 48-to-54 personalized concurrent communication is divided into two independent transactions: 48-to-54 personalized concurrent traffic, messages of which are of the same size; 48-to-18 personalized concurrent traffic. And the 48-to-18 personalized concurrent communication is divided to two 18-to-18 transactions and one 12-to-18 transaction. Fig.14 shows the situation of the latency, which is similar to the case with only single source-destination pair. The total four transactions are implemented in sequence. So the processing time of source nodes is increased about 9.216 us, compared to the original case if having no head-of-line

blocking. But the total communication overhead is still improved, which is a decrease of twice.

## VII. CONCLUSION

Many-to-many ( $M$ -to- $N$ ) personalized concurrent communication, one of the most dense traffic patterns, is common in high performance computing system. Hot-spot congestions are the bottleneck of this communication pattern. Once hot-spot contention occurs, the latency of massages increases. If the contentions are persistent, it ultimately causes congestion spreading and even leads to head-of-line blocking, which results in a catastrophic loss of throughput and the increasing of delay.

In this paper we introduce a fat-tree network based on RapidIO to these application systems, which has full bisection bandwidth and low latency. And a proactive congestion avoidance mechanism is proposed to optimize many-to-many personalized concurrent traffic in RapidIO-based fat-trees. It is a node-level communication-scheduling strategy working along with modified deterministic routing algorithms and a contiguous task mapping scheme. When  $M > N$ , we propose a hierarchical scheme. And we propose a heuristic scheme of virtual destination to deal with the case that messages from each node in  $M$  group to every other node in  $N$  group are not of the same size. Our proactive congestion avoidance mechanism has been used in a practical network, belonging to part of digital signal processing platform for “*the TianLai Pathfinder*”, which is a project for exploration of dark energy in the cosmic space. The network delay is effectively improved, reducing from about 34.784 us to 1.484 us. And the communication overhead is decreased by twice.

## ACKNOWLEDGEMENT

This work is supported by the Joint Research Fund in Astronomy (U1531139) under cooperative agreement between the National Natural Science Foundation of China (NSFC) and Chinese Academy of Sciences (CAS), and is supported by Foundation of Chinese Academy of Sciences(YCZB201302).

## REFERENCES

- [1] E. Vermij, L. Fiorin, C. Hagleitner, and K. Bertels, "Exascale Radio Astronomy: Can We Ride the Technology Wave?", in *Supercomputing*, 2014, pp. 35-52.
- [2] G. Pfister, M. Gusat, W. Denzel, D. Craddock, N. Ni, W. Rooney, *et al.*, "Solving hot spot contention using infiniband architecture congestion control," *Proceedings HP-IPC 2005*, 2005.
- [3] E. G. Gran, M. Eimot, S.-A. Reinemo, T. Skeie, O. Lysne, L. P. Huse, *et al.*, "First experiences with congestion control in InfiniBand hardware," in *Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on*, 2010, pp. 1-12.
- [4] J. Escudero-Sahuquillo, P. J. Garcia, F. J. Quiles, and J. Duato, "An efficient strategy for reducing head-of-line blocking in fat-trees," in *Euro-Par 2010-Parallel Processing*, ed: Springer, 2010, pp. 413-427.
- [5] J. Escudero-Sahuquillo, P. J. Garcia, F. J. Quiles, J. Fliech, and J. Duato, "OBQA: Smart and cost-efficient queue scheme for Head-of-Line blocking elimination in fat-trees," *Journal of Parallel and Distributed Computing*, vol. 71, pp. 1460-1472, 2011.
- [6] P. Y. Segura, J. Escudero-Sahuquillo, C. G. Requena, P. J. Garcia, F. J. Quiles, and J. Duato, "BBQ: a straightforward queuing scheme to reduce hol-blocking in high-performance hybrid networks," in *Euro-Par 2013 Parallel Processing*, ed: Springer, 2013, pp. 699-712.
- [7] C. E. Leiserson, "Fat-trees: universal networks for hardware-efficient supercomputing," *Computers, IEEE Transactions on*, vol. 100, pp. 892-901, 1985.
- [8] RapidIO Specification.
- [9] N. Jiang, D. U. Becker, G. Michelogiannakis, and W. J. Dally, "Network congestion avoidance through speculative reservation," in *High Performance Computer Architecture (HPCA), 2012 IEEE 18th International Symposium on*, 2012, pp. 1-12.
- [10] N. Jiang, L. Dennison, and W. J. Dally, "Network endpoint congestion control for fine-grained communication," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2015, p. 35.
- [11] M. Luo, D. K. Panda, K. Z. Ibrahim, and C. Iancu, "Congestion avoidance on manycore high performance computing systems," in *Proceedings of the 26th ACM international conference on Supercomputing*, 2012, pp. 121-132.
- [12] E. Zahavi, G. Johnson, D. J. Kerbyson, and M. Lang, "Optimized InfiniBandTM fat - tree routing for shift all-to-all communication patterns," *Concurrency and Computation: Practice and Experience*, vol. 22, pp. 217-231, 2010.
- [13] S. Kumar and L. V. Kale, "Scaling all-to-all multicast on fat-tree networks," in *Parallel and Distributed Systems, 2004. ICPADS 2004. Proceedings. Tenth International Conference on*, 2004, pp. 205-214.
- [14] Y. Yang and J. Wang, "Near-optimal all-to-all broadcast in multidimensional all-port meshes and tori," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 13, pp. 128-141, 2002.
- [15] W. Y. Chou and C. Chen, "All-to-all personalized exchange in generalized shuffle-exchange networks," *Theoretical Computer Science*, vol. 411, pp. 1669-1684, 2010.
- [16] R. Petagon and J. Werapun, "Embedding the optimal all-to-all personalized exchange on multistage interconnection networks+," *Journal of Parallel and Distributed Computing*, vol. 88, pp. 16-30, 2016.
- [17] The Opportunity for Sub Microsecond Interconnects for Processor Connectivity.[www.rapidio.org](http://www.rapidio.org).
- [18] F. Petrini and M. Vanneschi, "k-ary n-trees: High performance networks for massively parallel architectures," in *Parallel Processing Symposium, 1997. Proceedings., 11th International*, 1997, pp. 87-93.
- [19] C. Gomez, F. Gilabert, M. E. Gómez, P. López, and J. Duato, "Deterministic versus adaptive routing in fat-trees," in *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*, 2007, pp. 1-8.
- [20] G. Rodriguez, C. Minkenberg, R. Beivide, R. P. Luijten, J. Labarta, and M. Valero, "Oblivious routing schemes in extended generalized Fat Tree networks," in *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on*, 2009, pp. 1-8.