Available online at www.sciencedirect.com
ScienceDirect
ELSEVIER
April 2013, 20(2): 66–72
www.sciencedirect.com/science/journal/10058885
The Journal of China Universities of Posts and Telecommunications
http://jcupt.xsw.bupt.cn

# Frame-level traffic splitting for link aggregation in data center networks

ZHANG Peng (✉), WANG Hong-bo, CHENG Shi-duan, LI Yang-yang, DONG Jian-kang

State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China

**Abstract**

Ethernet link aggregation, which provides an easy and cost-effective way to increase both bandwidth and link availability between a pair of devices, is well suited for data center networks. However, all the traffic splitting algorithms used in existing Ethernet link aggregation are flow-level which do not work well owing to the traffic characteristics of data centers. Though frame-level traffic splitting can achieve optimal load balance and the maximum benefits from aggregated capacity, it is generally deprecated in most cases because of frame disordering which can disrupt the operation of many Internet protocols, most notably transmission control protocol (TCP). To address this issue, we first investigate the causes of frame disordering in link aggregation and find that all of them either are no longer true or can be prevented in data centers. Then we present a byte-counter frame-level traffic splitting algorithm which achieves optimal performance while causes no frame disordering. The only requirement is that frames in a flow are the same size which can be easily met in data centers. Simulation results show that the proposed frame-level traffic splitting method could achieve higher throughput and optimal load balance. The average completion time of different sized flows is reduced by 24% on average and by up to 46%.

**Keywords** data center networks, Ethernet, link aggregation, frame-level traffic splitting

## 1 Introduction

In recent years, many data centers, including high performance computing, cloud computing and Internet data centers, are built using commodity Ethernet networks because of low cost and ease-of-use. The ratio of Ethernet as cluster inter-connects has grown rapidly from 2% to 42% on the Top500 list of most powerful computers in the past decade (http://www.top500.org/). Moreover, Ethernet is widely deployed in cloud computing and Internet data centers owing to the trend of constructing data centers with commodity components [1].

Data center networks need higher bandwidth and availability to assure application performance. Ethernet link aggregation (LAG), which combines multiple parallel links between a pair of devices to a single higher performance logical link, is well suited for data center networks. Traditionally, the approach increasing bandwidth from the server to the network edge is to add links and use LAG. Moreover, LAG is also commonly used between two switches to increase bandwidth. The bandwidth of the logical link is the sum of multiple physical links and the number of physical links can be adjusted according to the specific need. Additionally, LAG can prevent the failure of any single component link from disrupting communications between the interconnected devices. Another advantage of LAG is that these improvements can be obtained using existing hardware. Because there is always a window in time when aggregated links are less expensive than a speed upgrade and will achieve equivalent performance. Above all, it can be seen that LAG provides an easy and cost-effective way to increase both bandwidth and availability.

In order to make full use of the bandwidth of multiple aggregated physical links between a pair of devices, LAG applies a traffic splitting algorithm to distribute traffic among multiple links. The current traffic splitting

Received date: 05-07-2012
Corresponding author: ZHANG Peng, E-mail: zhp@bupt.edu.cn
DOI: 10.1016/S1005-8885(13)60030-8

algorithms used in Ethernet link aggregation standard IEEE 802.3ad [2] and some improved works [3–4] are all based on flows (also referred as 'sessions'), that is, all frames belonging to one flow are forwarded to the same link. Flow-level splitting methods work well where a large number of network flows is given and no flow dominates the link. However, the second prerequisite is not met in data centers since the measurement results show that the flows vary widely in size [5–6]. This can lead to persistent congestion on some links, while other links remain underutilized.

Another option for traffic distributing is frame-level method which allows frames belonging to one flow to be forwarded to different links. Though frame-level traffic splitting can achieve optimal load balance and the maximum benefits from aggregated capacity, it is generally deprecated in most cases because of frame disordering which can disrupt the operation of many Internet protocol.

Frame disordering is caused by several primary reasons. One of them is that characteristics of the links such as latency and capacity are different. Frames forwarded to different links would experience unequal delay. However, this cause is no longer true in data centers since the links are identical both in latency and capacity.

Another reason is that frames are of variable length. A long frame could be received after a short frame (for example, a 64 B minimum Ethernet frame) started at a later time. As shown in Fig. 1, three Ethernet frames of different size are forwarded to two links, which are labeled to frame 1, frame 2 and frame 3 respectively. Obviously, frame 1 is received after other two frames. In fact, more than 18 minimum-length frames (84 B in total, including the frame length and the 96 bit of inter-frame gap) can be sent and received at the other end before the single maximum-length Ethernet frame (1 538 B in total) is completely received.

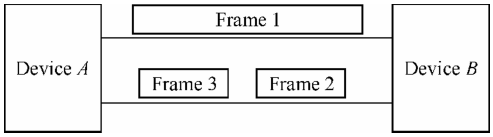



**Fig. 1** Frame disordering caused by variable frame length

However, the causes of variable frame length can also be prevented in data centers. Firstly, setting frame size is feasible in data centers since data centers are usually owned and operated by one organization. Secondly, setting frame size has no effect on the external traffic as connectivity to the external Internet is typically managed through load balancers and application proxies that effectively separate internal traffic from external [7]. Thirdly, setting frame size would cause little overhead to link utilization. The frame size of one of the data centers measured in Refs. [5–6] is in a bimodal distribution with peak at around 40 B and 1 500 B. It should be noted that frames of different flows can be set different size since the frames in one flow is completely independent of the traffic of other flows and it is not strictly necessary to maintain the order of frames from one flow to another.

Moreover, frame disordering may be also caused by the algorithms which decide how the frames are distributed, such as round-robin and random. In this paper, we present a byte-counter frame-level splitting algorithm for link aggregation between a pair of devices, which could achieve optimal performance while cause no frame disordering. The only requirement is that frames in one flow are set to the same size which can be easily met in data centers. Simulation results show that the proposed frame-level traffic splitting method could achieve higher throughput and optimal load balance. The flow completion time of different sized flows is reduced by 24% on average and by up to 46%.

The remainder of this paper is organized as follows. In Sect. 2 we introduce background and related work. The frame-level traffic splitting algorithm is presented in Sect. 3. Sect. 4 provides the evaluation results. Finally, Sect. 5 concludes the paper.

## 2 Background and related works

### 2.1 Data center networks

Typical data center networks usually are layered multi-root two- or three-level trees. Fig. 2 shows a three-tiered data center network which has a core tier in the root of the tree, an aggregation tier in the middle and an edge tier at the leaves of the tree. Most data center network topologies introduce oversubscription as a mean to lower the total cost (TCO) of the design [8]. The oversubscription is the ratio of the worst-case achievable aggregate bandwidth among the end servers to the total bisection bandwidth of a particular communication topology. An oversubscription of 5:1 means that only the total bisection bandwidth is only 200 Mbit/s. Typical designs are oversubscribed by a factor of 2.5:1 (400 Mbit/s)

to 8:1 (125 Mbit/s) [8].

In traditional internet data centers, most of the traffic is between servers and end users so there is less traffic inside data centers. However, cloud computing and online services require much traffic among servers which makes network become the performance bottleneck. This is due to the infra-structure services which are hosted in cloud computing and online services, such as distributed file systems, distributed execution engine. Moreover, virtualization has been widely deployed currently as one of the core technologies in cloud computing where multiple virtual machines are allocated in a physical server and shares the scarce access bandwidth.

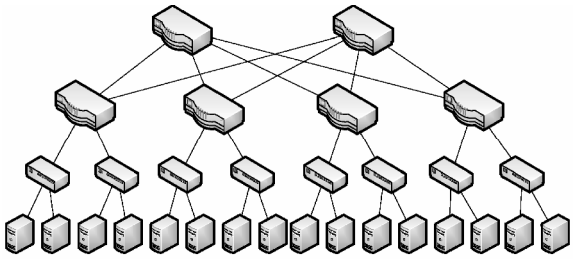


**Fig. 2** Typical data center network topology

### 2.2 Related works

Flow-level splitting method works well where a large number of network flows are given and no flow dominates the link. However, the prerequisite is not met in data centers. Several studies [5–6] have been performed on measuring the traffic characteristics of data centers. The measurement results show that there are usually only dozens of concurrent flows at any point in time and flows vary widely in size. This cannot assure statistical multiplexing and it is difficult to ensure that traffic is distributed evenly over multiple links in data center networks. Another limitation of flow-level splitting is that single flow throughput limited to the speed of a single physical link. This directly affects the performance of some services, such as bulk data transfer.

Packet-level traffic splitting has been studied in multipath routing [9–10]. It can achieve optimal load balance and the maximum benefits from the aggregated capacity. But packet-level traffic splitting is generally deprecated in most cases because of packet disordering which can disrupt the operation of many Internet protocols, most notably TCP, which makes up more than 95% of the data center traffic [5,7]. When TCP receiver gets packets out of order, it sends duplicate acknowledgements (ACKs) to trigger fast retransmission algorithm at the sender. The TCP sender infers a packet has been lost and retransmits it. More serious thing is that the TCP sender assumes it is an indication of network congestion. It reduces its congestion window to limit the transmission speed. If disordering happens frequently, the congestion window is at a small size and can hardly grow larger. As a result, the TCP connection has to transmit packets at a limited speed and cannot efficiently utilize the bandwidth.

By combining with the advantages of both flow-level and packet-level, in Refs. [11–12], Kandula et al. propose new traffic splitting algorithms that operate on bursts of packets, i.e., flow slice or flowlet, to avoid reordering. The method cuts off each flow into flow slices at every intra flow interval larger than a slicing threshold and balances the load on a finer granularity. The main origin of flow slices or flowlets is the burstiness of TCP at round-trip time (RTT) and sub-RTT scales; i.e., a window of frames is transmitted at the very beginning of each RTT time, followed by a long silent period. This behavior is caused by ACK compression, slow-start, and other factors [13]. However, the slicing threshold which is typically set to several milliseconds is larger than typical RTT of data centers (usually hundreds of microsecond). Therefore the burstiness of TCP is not obvious in data center and the traffic splitting algorithms that operate on flow slice or flowlet is not suited for data center networks.

## 3 Frame-level traffic splitting

### 3.1 Byte-counter frame splitting algorithm

To avoid frame disordering, we present the byte-counter frame splitting algorithm which explicitly stores a byte-counter corresponding to each aggregated link. The system model of link aggregation is show in Fig. 3.

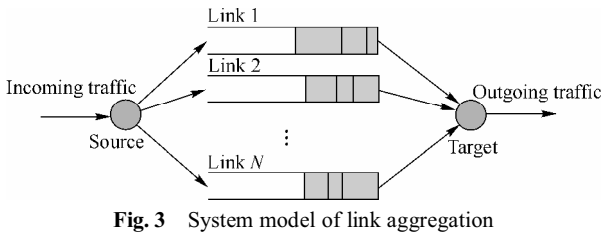



**Fig. 3** System model of link aggregation

We first set frames in a flow to the same size while different flows can have different sized frames. Frame length of a flow is set according to application demand.

For example, flows used to transfer data are set to maximum transmission unit (MTU) sized frame while flows used to send small control messages are set small-sized frame. While forwarding a frame, our design selects the aggregated link with the smallest counter value to put the frame into its output queue and increases the counter by the corresponding frame size (in byte) and the inter-frame gap, as described in Algorithm 1.

Moreover, in order to maintain frame order when multiple counters are equal with each other, the source aggregated device numbers all the aggregated links and always chooses the link with the minimum number when byte-counters are equal. Similarly, at the target aggregated device, the frame which is received from the link with minimum number is processed first when multiple frames are received simultaneously. This can assure no frame disordering even when frames are received at the same time.

Clearly, byte-counter frame splitting algorithm results in keeping all the links as balanced as possible since the maximum difference among byte-counters is no more than the Ethernet MTU. For implementing this scheme, the only extra states required are the byte-counters corresponding to every aggregated links and do not need maintain any other states.

**Algorithm 1** Byte-counter frame splitting algorithm

Variable definition:
  $N$: the number of aggregated links.
  $L_i$: the $i$th link.
  $S_i$: the byte-counter of link $i$ (in byte).
  $Q_i$: the output queue of link $i$.
Initialization:
  a) Numbering the physical links to 1, 2, …, $N$.
  b) Setting a frame size to each flow.
Algorithm:
  At the source aggregated device:
    when receiving a frame $p$
      find $i' \leftarrow \min\{ \text{argmin}(S_i) \}$;
      put the frame $p$ into $Q_{i'}$
      $S_{i'} \leftarrow S_{i'}$ + the length of frame $p$ and the inter-frame gap;
  At the target aggregated device:
    if receiving multiple frames simultaneously
      then first process the frame received from the link with the minimum number;
    end if

### 3.2 Frame disordering analysis

Since frame of different flows disordering has no effect on TCP performance, it is not strictly necessary to maintain the frame order from one flow to another. Therefore, we just analyze the frame disordering within the same flow.

When the frames in a flow are the same size, byte-counter frame splitting algorithm for link aggregation in data centers would not cause any frame disordering within the same flow.

For a given flow, its $j$th and ($j$+1)th frames are forwarded to links according the byte-counter algorithm. Let $T_{\text{arr}}^{(j)}$ and $T_{\text{arr}}^{(j+1)}$ be the time when two frames arrive at source aggregated device, clearly, $T_{\text{arr}}^{(j)} \leqslant T_{\text{arr}}^{(j+1)}$. Note that $T_{\text{arr}}^{(j)} = T_{\text{arr}}^{(j+1)}$ happens when frames are sent from multiple links. There can be any number of frames from other flows between frames $j$ and ($j$+1). The wait time $W$ a frame stays in switch port buffer satisfies $W^{(j)} \leqslant W^{(j+1)}$ because frame which come first always choose the shortest buffer owing to the byte-counter scheme. Since the frames in a flow are the same size and the links are identical both in capacity and delay, the transmission delay $T_{\text{trans}}$ and propagation delay $T_{\text{prop}}$ are same. The received time of frames $T_{\text{recv}}$ satisfies:

$$T_{\text{recv}} = T_{\text{arr}} + W + T_{\text{trans}} + T_{\text{prop}} \tag{1}$$

From above we can get $T_{\text{recv}}^{(j)} \leqslant T_{\text{recv}}^{(j+1)}$. When $T_{\text{recv}}^{(j)} < T_{\text{recv}}^{(j+1)}$, that is, the $j$th frame is received before the ($j$+1)th frame. For $T_{\text{recv}}^{(j)} = T_{\text{recv}}^{(j+1)}$, there would be no frame disordering within the flow because the link with minimum number is chosen when counters are equal at the source aggregated device and the $j$th frame is processed before the ($j$+1)th frame when two frames are received simultaneously at target aggregated device, as shown in Algorithm 1.

Moreover, the tail frames of TCP flows which are usually smaller than others may cause disordering. But the tail frame disordering does not affect the flow throughput since fast retransmission of TCP is triggered by three duplicated ACKs while only one duplicated ACK is generated in the disordering. In addition, ACK disordering would be caused when some ACKs are piggy-backed and others are not. But this does not affect TCP throughput since ACKs are cumulative.

## 4 Evaluation

In this section, we conduct extensive simulations with the network simulator NS2 (http://www.isi.edu/nsnam/ns/)

to evaluate the proposed byte-counter frame-level traffic splitting for link aggregation. The network model used in the simulations is shown in Fig. 4. Two 48-port gigabit Ethernet top-of-rack switches are connected by eight aggregated links and each switch connects 40 servers. The capacity of all links is 1 Gbit/s and the propagation delay is 20 μs. All the communications use TCP NewReno.

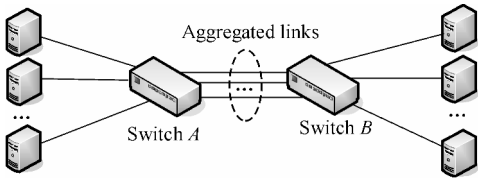



**Fig. 4** Network model used in the simulations

### 4.1 Benchmark suite

The goal of these tests is to determine the total throughput of aggregated links between two switches with various traffic patterns. In the absence of commercial data center network traces, we first generate a variety of communication pairs according the strategies similar to [1]:

1) Random ($k$): a server will send to other $k$ servers in the network with uniform probability.

2) Staggered (Pr): where a server will send to servers in the same switch with probability Pr, and to others with probability (1 – Pr). The specific server in the same switch or others is also chosen with uniform probability.

3) Stride ($k$): a server with index $x$ sends to the server with index ($x$ + $k$) mod (server_num). This pattern is common to high performance computation (HPC) applications.

Fig. 5 shows the total throughputs of aggregated links between switches with a variety of randomized, staggered and stride communication patterns.

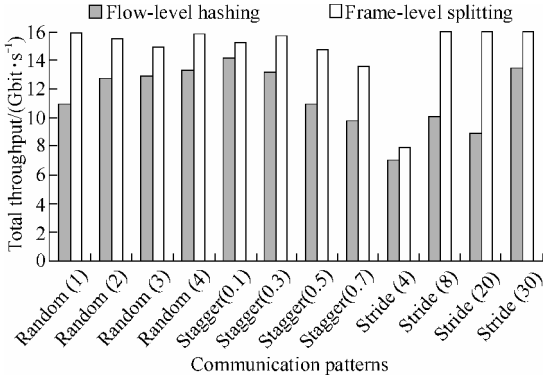


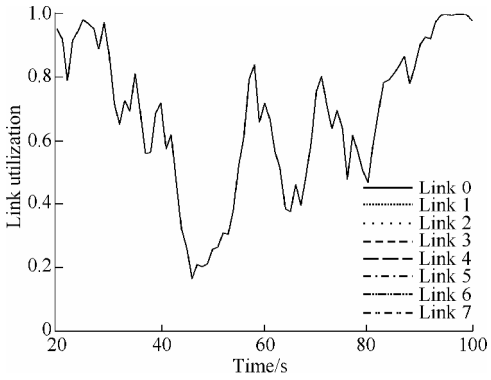

**Fig. 5** Total throughput of aggregated links between switches with various traffic patterns

As we can see, in all communication patterns explored, byte-counter frame-level traffic splitting significantly outperforms flow-level hashing with various traffic patterns. That is, our design can obtain the maximum benefits from aggregated capacity.

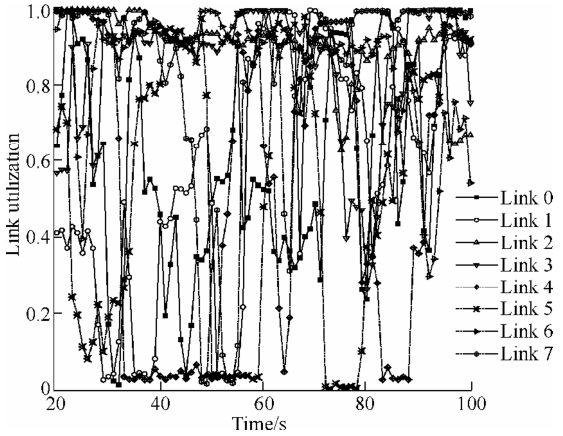
### 4.2 Generated traffic

In order to measure the performance of our design with the real traffic, we generate the traffic according to the characteristics presented by past study works, for instance, flow size is in heavy tail distribution and frame length (packet size) is in a bimodal distribution. Specifically, 90 percent of the generated flows are small flows whose size is uniformly distributed on the interval [100 kB, 1 MB] while other 10% are sized in uniform distribution on [100 MB, 1 GB]. The small-sized flows are used to send control messages and the frame length is usually small so we set frame length to be uniformly distributed on [150 B, 250 B]. Large-sized flows are usually used to transmit data so the frame size is set to be in normal distribution on [1 400 B, 1 500 B]. Frames of a flow are set to the same size. And the flow arrival rate follows the Poisson distribution. We just consider the traffic across switches since this is where the link aggregation is located. The simulation lasts 120 s and we just evaluate the stable interval from 20 s to 100 s. Our byte-counter frame-level traffic splitting and flow-level traffic hashing are evaluated under the same benchmark traffic.

Fig. 6 demonstrates the utilization of every physical aggregated links under two traffic splitting methods. The traffic is evenly balanced to every link and the utilization of each link is identical in frame-level traffic splitting, as shown in Fig. 6(a).




(a) Frame-level traffic splitting

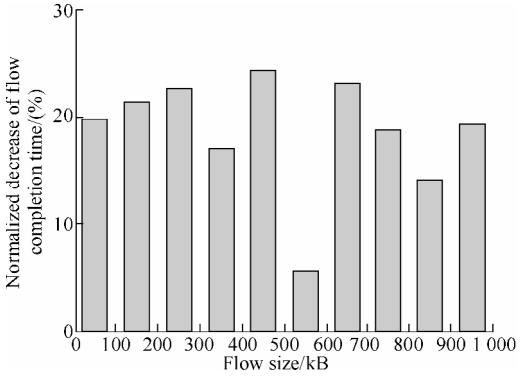



(b) Flow-level traffic hashing

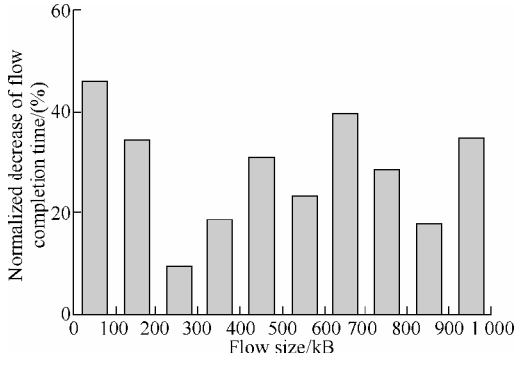
**Fig. 6** Utilization of every aggregated link between swiches under two traffic splitting methods

The utilization of each link under flow-level splitting varies widely from near 0 to 100%, as shown in Fig. 6(b). This reveals that our byte-counter frame-level traffic splitting can achieve optimal load balance.

Fig. 7 illustrates the normalized decrease of average flow completion time by frame-level traffic splitting comparing to flow-level method.




(a) Small-sized flows




(b) Large-sized flows

**Fig. 7** The normalized decrease of average flow completion time by frame-level traffic splitting comparing to flow-level method

As we can see, the average completion time of different sized flows is reduced by 24% on average and by up to 46% under frame-level traffic splitting. The flow completion time decrease of large-sized flow is bigger than small-sized flow because our design could increase the total throughput and large-sized flows are more sensitive to throughput. It should be noted that due to the dynamic utilization of aggregated links and dynamic queue occupation caused by the randomness of flow size and generating time, the benefits of flows with different size have some difference.

Finally, we evaluated the frame disordering with our design. The simulation results show that there was no any frame disordering in our byte-counter frame-level traffic splitting algorithm.

## 5 Conclusions and future work

After investigating the causes of frame disordering in frame-level traffic splitting, we find that all of them are no longer true or can be prevented for link aggregation in data centers. The only requirement is that frames in a flow are the same size which can be easily met in data centers. Then a byte-counter frame-level traffic splitting algorithm is presented which achieves the maximum benefits from aggregated capacity while causes no frame disordering. Simulation results show that the proposed frame-level traffic splitting method could achieve higher throughput and optimal load balance. The average completion time of different sized flows is reduced by 24% on average and by up to 46%.

The algorithm we proposed in this work focuses on the link aggregation between a pair of devices. In future work, we will extend it to a whole data center network.

**Acknowledgements**

This work was supported by the National Natural Science Foundation of China (61002011), the Open Fund of the State Key Laboratory of Software Development Environment (SKLSDE-2009KF-2-08), the National Basic Research Program of China (2009CB320505), the Hi-Tech Research and Development Program of China (2011AA01A102).

## References

1. Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture. Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications

(SIGCOMM'08), Aug 17–22, 2008, Seattle, WA, USA. New York, NY, USA: ACM, 2008: 63–74
2. IEEE Std 802.3ad-2000. Amendment to carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications -- Aggregation of multiple link. 2000.
3. Alexander T. Link aggregation in Ethernet frame switches. United States Patent. US 6553029 B1. 2003-04-22
4. Sato M, Nakajima S, Suzuki K. Ethernet link aggregation. United States Patent. US 2010/0215042 A1. 2010-04-26
5. Kandula S, Sengupta S, Greenberg A, et al. The nature of data center traffic: measurements and analysis. Proceedings of the 9th Internet Measurement Conference (IMC'09), Nov 4–6, 2009, Chicago, IL, USA. New York, NY, USA: ACM, 2009: 202–208
6. Benson T, Akella A, Maltz D A. Network traffic characteristics of data centers in the wild. Proceedings of the 10th Internet Measurement Conference (IMC' 10), Nov 1–3, 2010, Melbourne, Australia. New York, NY, USA: ACM, 2010: 267–280
7. Alizadeh M, Greenberg A, Maltz D, et al. Data center TCP (DCTCP). Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM'10), Aug 30–Sep 3, 2010, New Delhi, India. New York, NY, USA: ACM, 2010: 63–74
8. Cisco data center infrastructure 2.5 design guide. Cisco Validated Design I. 2007
9. Han Y, Makowski A M. Resequencing delays under multipath routing -- Asymptotics in a simple queuing model. Proceedings of the 25th Annual Joint Conference of the IEEE Computer and Communications (INFOCOM'06), Apr 23–29, 2006, Barcelona, Spain. Piscataway, NJ, USA: IEEE, 2006: 12p
10. Leung K C, Li V O K. Generalized load sharing for packet-switching networks I: theory and packet-based algorithm. IEEE Transactions on Parallel and Distributed Systems, 2006, 17(7): 694–702
11. Kandula S, Katabi D, Sinha S, et al. Dynamic load balancing without frame reordering. ACM SIGCOMM Computer Communication Review, 2007, 37(2): 53–62
12. Shi L, Liu B, Sun C, et al. Load-balancing multipath switching system with flow slice. IEEE Transactions on Computers, 2012, 61(3): 350–365
13. Sinha S, Kandula S, Katabi D. Harnessing TCP's burstiness with flowlet switching. Proceedings of the 3rd Workshop on Hot Topics in Networks (HotNets'04), Nov 15–16, 2004, San Diego, CA, USA. New York, NY, USA: ACM, 2004: 6p

(Editor: WANG Xu-ying)

## From p. 53

3. Li X M, Bai B M. Precoding based on minimum mean square error for multiuser MIMO downlinks. Journal of Chongqing University of Posts and Telecommunications: Natural Science, 2010, 22(1): 11−13(in Chinese).
4. Paulraj A, Nabar R, Gore D. Introduction to space-time wireless communications. London, UK: Cambridge University Press, 2003
5. Antonio P I, Daniel P P, Ana I P. A robust maximum approach for MIMO communications with imperfect channel state information based on convex optimization. IEEE Transactions on Signal Processing, 2006, 54(1): 346–360
6. Tong F, Glover I A, Pennock S R. Indoor distributed antenna experiments. Proceedings of the 14th IST Mobile and Wireless Communications Summit, Jun 19–22, 2005, Dresden, Germany
7. Hu H, Zhang Y, Luo J J. Distributed antenna systems: open architecture for future wireless communications. Boca Raton, FL, USA: Auerbach Publication, 2007
8. Xin L, Zhang J H, Xiong F, et al. Power coverage and downlink diversity for indoor distributed antenna system based on wideband measurement at 6 GHz. Proceedings of the 15th International Symposium on Wireless Personal Multimedia Communications (WPMC'12), Sep 24–27, 2012, Taipei, China
9. Dong D, Zhang J H, Zhang Y, et al. Large scale characteristics and capacity evaluation of outdoor relay channels at 2.35 GHz. Proceedings of the 70th Vehicular Technology Conference (VTC-Fall'09), Sep 20–23, 2009, Anchorage, AK,USA. Piscataway, NJ, USA: IEEE, 2009: 5p

(Editor: ZHANG Ying)