ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.10, No.12, 1–11

TCP congestion control for wireless datacenters

Jingyuan Wang[1a)], Yunjing Jiang[1], Yuanxin Ouyang[1], Chao Li[2], Zhang Xiong[1,2], and Junxia Cai[3]

[1] *State Key Laboratory of Software Development Environment, School of Computer Science and Engineering, Beihang University, Beijing 100191, China*

[2] *Research Institute of Beihang University in Shengzhen, Shenzhen 518057, China*

[3] *State Information Center, Beijing 100045, China*

a) *jywang@buaa.edu.cn*

Abstract: TCP is a low cost and easy-to-use transport layer protocol widely used in datacenter based applications and web services. Many TCP congestion control algorithms have been proposed to improve the performance of TCP in datacenter networks. However, the emerging wireless technologies in datacenter networks create new problems for TCP congestion control. On the one hand, TCP algorithms must suit the special application patterns (such as incast and Partition/Aggregate) of the datacenters. On the other hand, TCP must perform well within the random packet losses caused by unreliable wireless links. Although there are many datacenter/wireless oriented TCP algorithms for either datacenter or wireless networks, designing a TCP algorithm that can meet both "datacenter" and "wireless" requirements of wireless datacenters still remains a great challenge. In this paper, a novel congestion control algorithm is proposed to improve the performance of TCP over wireless datacenter environments. The proposed approach combines an ACK-based wireless network bandwidth estimate with an ECN-based datacenter TCP congestion control that can perform well in both the datacenter environments and wireless links. The simulated experiments validate the performance of our proposed approach.

Keywords: wireless datacenter, TCP congestion control, network bandwidth estimation

Classification: Electron devices, circuits, and systems

References

[1] M. Al-Fares, A. Loukissas and A. Vahdat: ACM SIGCOMM Computer Communication Review (2008) 63.

[2] K. Wu, J. Xiao and L. Ni: Frontiers of Computer Science **6** [5] (2012) 596.

[3] H. Chen, P. Lu, P. Xiong, C. Z. Xu and Z. Wang: Frontiers of Computer Science **6** [4] (2012) 373.

[4] V. Jacobson: ACM SIGCOMM Computer Communication Review **25** [1] (1995) 157.

[5] H. Tang, A. Gulbeden, J. Zhou, W. Strathearn, T. Yang and L. Chu: Proc. ACM/IEEE 2004 Conference Supercomputing (2004) 53.

© IEICE 2013

DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

[6] D. Zats, T. Das, P. Mohan, D. Borthakur and R. Katz: Proc. ACM SIGCOMM 2012 (2012) 139.
[7] M. Alizadeh, A. Greenberg, D. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta and M. Sridharan: Proc. ACM SIGCOMM 2010 (2010) 63.
[8] B. Vamanan, J. Hasan and T. Vijaykumar: Proc. ACM SIGCOMM 2012 (2012) 115.
[9] H. Wu, Z. Feng, C. Guo and Y. Zhang: Proc. ACM CoNEXT 2010 (2010) 13.
[10] D. Halperin, S. Kandula, J. Padhye, P. Bahl and D. Wetherall: Proc. ACM SIGCOMM 2012 (2011) 38.
[11] Y. Cui, H. Wang, X. Cheng and B. Chen: IEEE Wireless Commun. **18** [6] (2011) 46.
[12] X. Zhou, Z. Zhang, Y. Zhu, Y. Li, S. Kumar, A. Vahdat, B. Zhao and H. Zheng: Proc. ACM SIGCOMM 2012 (2012) 443.
[13] W. Zhang, X. Zhou, L. Yang, Z. Zhang, B. Zhao and H. Zheng: Proc. 10th ACM Workshop on Hot Topics in Networks (2011).
[14] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi and R. Wang: Proc. ACM MobiCom 2001 (2001) 287.
[15] C. Fu and S. Liew: IEEE J. Sel. Areas Commun. **21** [2] (2003) 216.
[16] Y. Gu and R. Grossman: Computer Networks **51** [7] (2007) 1777.
[17] ns-2 network simulator: http://www.isi.edu/nsnam/ns
[18] Y. Cui, H. Wang, X. Cheng and B. Chen: IEEE Wireless Commun. **18** [6] (2011) 46.
[19] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat and M. Yasuda: NSDI (2012) 19.

1 Introduction

Datacenter networks (DCNs) are becoming an increasingly important topic for both research and industry in recent years [1, 2, 3]. Using TCP as the transport layer protocol of a DCN is attractive because of its low cost and ease of use. The TCP congestion control algorithms that target Internet environments (such as TCP Reno [4]) achieved great success in previous decades but are found to perform poorly in modern datacenter networks. The special network environments and application patterns that datacenters now face create many new TCP performance challenges such as the TCP incast [5] problem and the ultra-low latency requirement [6]. In recent years, several TCP variants have been proposed to meet the TCP congestion control requirement of datacenter environments, including DCTCP [7], D^2TCP [8], ICTCP [9], etc. Recent research on TCP congestion controls in DCNs have attracted significant attention to the networking community.

Wireless datacenter networking [10] is now emerging as the next generation of datacenter network architecture. Some recent works propose using license-free 60 GHz RF wireless technology to overcome the wiring complexity and traffic balance challenges of scale-explosion datacenters, including wired/wireless DCN architectures proposed by [10, 11, 12], and a completely wireless DCN proposed by [13]. However, it is well known that the lossy channels of wireless networks can cause significant performance degradation

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

of TCP congestion control. Traditional TCP algorithms assume that all packet losses are caused by overdriving the network. Therefore, any random packet loss event that is caused by random physical layer artifacts (e.g., multi-path interferences) and has nothing to do with network congestion will also cause TCP to unnecessarily slow down throughput, which significantly degrades the performance of TCP. Although many TCP variants (such as TCP Westwood [14] and TCP Veno [15]) have been proposed to improve TCP performance in the wireless links over the Internet, the solutions for wireless links in the datacenter environment are still lacking.

Due to the "datacenter" and "wireless" specificities of wireless datacenter networks, neither existing TCP variants for datacenter networks nor for wireless links can be directly used in wireless datacenters. Designing a TCP algorithm that performs gracefully in both datacenter environments and wireless links remains a great challenge.

This paper focuses on improving TCP performance in wireless datacenter networks and proposes a novel wireless datacenter oriented TCP congestion control algorithm–Wireless Datacenter TCP (WDCTCP). WDCTCP combines an ACK-based network bandwidth estimation and an ECN-based congestion control. WDCTCP uses the ACK-based network bandwidth estimation to protect against random packet losses in wireless links, and uses ECN to accurately detect the congestion event of datacenter networks. Experiments over an ns-2 emulator evaluate the improved performance of the proposed approach compared to other TCP variants. To the best of our knowledge, WDCTCP is the first work that targets TCP congestion control in wireless datacenter networks.

The paper is organized as follows: In Section 2, we analyze the performance of the state-of-the-art datacenter TCP algorithm in a wireless environment. The proposed algorithm is presented in Section 3. The performance evaluation is provided in Section 4. Conclusions are offered in Section 5.

2 Problem of DCTCP in wireless networks

We use DCTCP [7] as an example to explain the problem of a wired datacenter oriented TCP algorithm in wireless datacenters. In DCTCP, the modified switch compares the current in-flight packet queue length in the buffer with a threshold K and marks the IP ECN bit of the in-flight packet if the queue length is greater than K. The DCTCP receivers echo the ECN flag in the ACKs when they receive packets carrying an ECN flag. The DCTCP senders calculate the fraction F of packets with an ECN flag in a window:

$$F := \frac{\#\ of\ packets\ with\ ECN\ flag}{Total\ \#\ of\ packets\ in\ a\ window}. \tag{1}$$

The DCTCP sender uses an exponentially smoothed F as the indicator of network congestion level, called α:

$$\alpha := (1-g) \times \alpha + g \times F, \text{ where } 0 < g < 1. \tag{2}$$

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

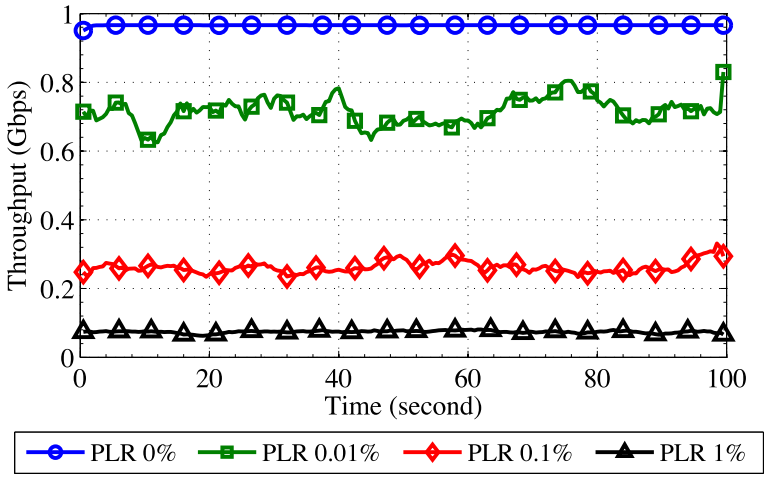


Fig. 1. Throughput of DCTCP in wireless networks

The DCTCP adjusts its window according to the indicator α in each RTT (Round-Trip Time) as:

$$w := \begin{cases} w - w\dfrac{\alpha}{2}, & \text{if } F > 0, \\ w + 1, & \text{if } F = 0. \end{cases} \tag{3}$$

The DCTCP also responds to network packet losses. When a DCTCP sender detects a packet loss event by receiving 3 duplicate ACKs, it halves the window size, slowing down the packet sending rate. This feature causes the DCTCP to perform poorly in wireless links with random packet losses. Figure 1 plots the throughput of a DCTCP flow in the wireless links with a 0% to 1% random packet loss rate (PLR). As shown in Figure 1, DCTCP performs poorly with wireless random packet losses. Because packet losses are severe link transmission error events, we cannot tackle the wireless link challenge by directly disabling the packet loss response function.

The wireless link challenge of using DCTCP cannot be tackled simply by using a wireless TCP variant (such as TCP Westwood [14] or TCP Veno [15]) to replace the DCTCP, because the datacenters have special requirements for TCP algorithms, for example, the requirement of incast scenarios. Incast is a kind of popular traffic pattern in datacenters, which exists widely in search engines, social network servers and other datacenter based TCP applications. Incast scenarios let a group of TCP severs synchronously send small data segments to a single client, and the servers can not send new data until all data segments have been received by the client. As reported by H. Tang et al. [5], due to the influence by special network environment of datacenters and TCP timeout mechanism, with the increase of server number, the incast traffic pattern can cause TCP flows to suffer throughput degradations by one/two orders of magnitude. Only datacenter targeted TCP variants, such as DCTCP, can avoid throughput degradation. The existing wireless TCPs perform poorly in incast scenarios, because they are not designed for the special datacenter environments. We demonstrated the performance of TCP Westwood and Veno in incast scenarios in Figure 2. As shown in the figure,

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

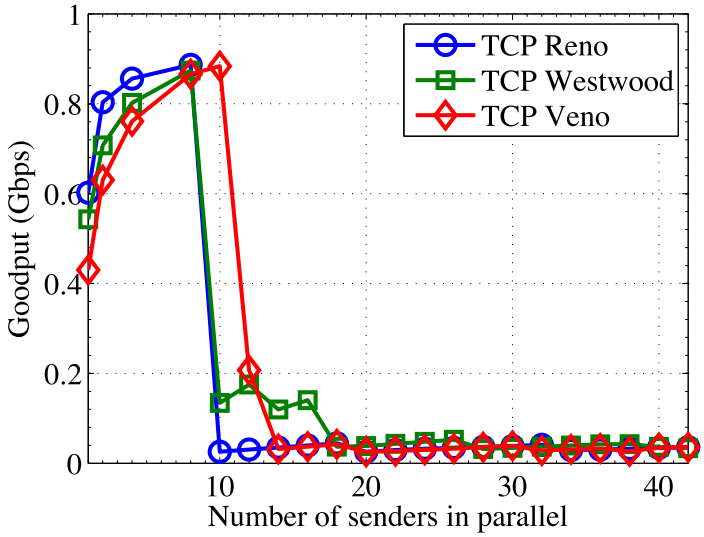


Fig. 2. Performance of wireless TCPs in incast scenarios

the performance of TCP Westwood and Veno are very close to traditional TCP Reno. The performance is fading obviously when the sender number is beyond some threshold. TCP variants for wireless link degrade their performance in wireless data centers, hence, a well-designed TCP variant that can achieve graceful performance **both** over wireless links and in datacenters is pretty necessary.

3 Wireless datacenter TCP

3.1 Network bandwidth estimation

In this section, we describe the algorithm design of WDCTCP. WDCTCP employs an ACK-based network bandwidth estimate approach to mitigate the impact of wireless packet losses to TCP performance. Figure 3 plots the principle of ACK-based network bandwidth estimation. As shown in Figure 3, WDCTCP sends out back-to-back packets at randomized intervals. Once the packets pass through a narrow link bottleneck, there will be a gap between the two packets. The bottleneck bandwidth can be estimated at the receiving end by comparing the inter-packet space. A similar estimating principle has been used in TCP Westwood [14] and UDT [16] to protect against wireless random packet losses with very good performance results. According to the self-clock feature of TCP, the inter-packet space can be echoed to the sender side by ACKs. Therefore, WDCTCP calculates network bottleneck bandwidth when the ith ACK arrived using

$$BW'_i = \frac{D_i}{t_i - t_{i-1}}, \tag{4}$$

where D_i is a corresponding amount of data that has been ACKed by the ith ACK and t_i is the arrival time of the ith ACK. WDCTCP uses a filter to reduce the noise influence of the bottleneck bandwidth estimation in each of the kth RTT:

$$BW_k = (1-\beta)\cdot BW_{k-1} + \beta \cdot \overline{BW_k}, \tag{5}$$

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

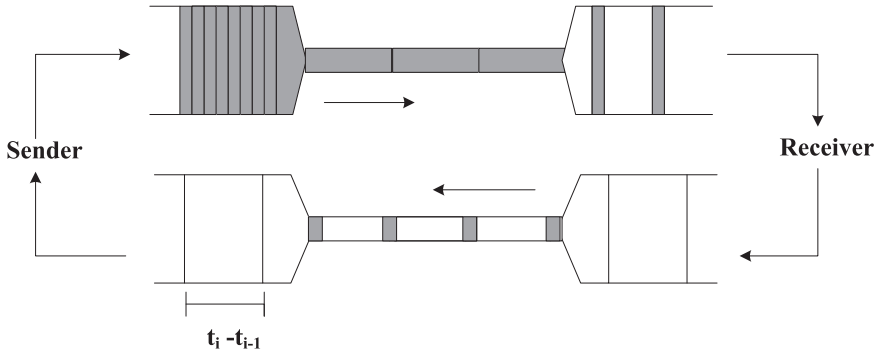


Fig. 3. Estimate the network bandwidth using ACKs [4]

where $\overline{BW_k}$ is the average of BW_i' in last kth RTT. BW_k is the network available bandwidth estimation of kth RTT.

3.2 Congestion control window adjustment

WDCTCP reacts to two types of network congestion signals: ECN and packet loss. The ECN signal is used for the wired links and switches of datacenters that support IP ECN marking. When an ACK carrying an ECN flag arrives at a TCP sender, WDCTCP calculates the network congestion indicator α the same as DCTCP in Expressions (1) and (2). WDCTCP adjusts window size in each RTT by

$$w := \begin{cases} \left(1 - \dfrac{\alpha}{2}\right) \cdot w + \dfrac{\alpha}{2} \cdot \dfrac{BW_k \times RTT_{min}}{S}, & \text{if } F > 0, \\ w + 1, & \text{if } F = 0. \end{cases} \tag{6}$$

where RTT_{min} is the minimal observed RTT value used as a reasonable estimate of the propagation delay (i.e., the time actually required for a packet to be transmitted from the sender to the receiver). The physical meaning of (6) is letting the transmission rate of the TCP sender converge with the network bandwidth.

WDCTCP also reacts to the packet loss signals in networks. When the WDCTCP sender detects a packet loss event by receiving 3 duplicate ACKs, the congestion control window is set to

$$w =: \max\left\{2, \frac{w}{2} + \frac{BW_k \times RTT_{min}}{2S}\right\}. \tag{7}$$

where S is the average TCP packet size used by WDCTCP senders. When the WDCTCP sender detects a packet loss by the timeout expiration, the sender sets the congestion control window to 2 and sets the slow start threshold to

$$TH_s =: \max\left\{2, \frac{w}{2} + \frac{BW_k \times RTT_{min}}{2S}\right\}. \tag{8}$$

When WDCTCP detects a packet loss, the sender decreases the transmission rate to the mid-value between the current sending rate and BW_k. Because BW_k has no relation to the network random packet loss rate, the window size adjustment of WDCTCP is immune to random packet losses.

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

As shown in Eqs. (6), (7), and (8), we can summarize that the target of window size adjustment of WDCTCP is letting the network bandwidth operate at a critical saturation state. This window size adjustment target is independent of network random packet losses. Moreover, the ECN-driven congestion control window adjustment in (6) allows WDCTCP to incorporate the datacenter friendly features of DCTCP. Therefore, WDCTCP can meet the requirements of both datacenter environments and wireless links in wireless datacenter networks.

4 Experiments

We tested the performance of WDCTCP by using the ns-2 [17] simulator. We use ns-2 to simulate a typical wireless datacenter introduced by Y. Cui [18]. The topology of the wireless datacenter is plotted in Figure 4. The servers in the datacenter are grouped by racks. Servers in the same rack are connected by a ToR (Top of Rack) switch. The aggregation switches connect racks and aggregate traffics of different racks to core switches. To balance traffics of racks among different aggregation switches, a wireless interface is installed to each ToR switch. Servers of different aggregation domains can communicate by using wireless interfaces, as a result, workloads of the aggregation and core switches can be reduced. Because wireless radio medium is unreliable, the wireless links have random packet losses rate. The parameters setup of wired and wireless links in the wireless datacenter is listed in Table I. We tested the performance of WDCTCP over the simulated wireless datacenter. The compared algorithms include DCTCP, TCP Reno and wireless link targeted TCP variant: TCP Westwood and Veno.

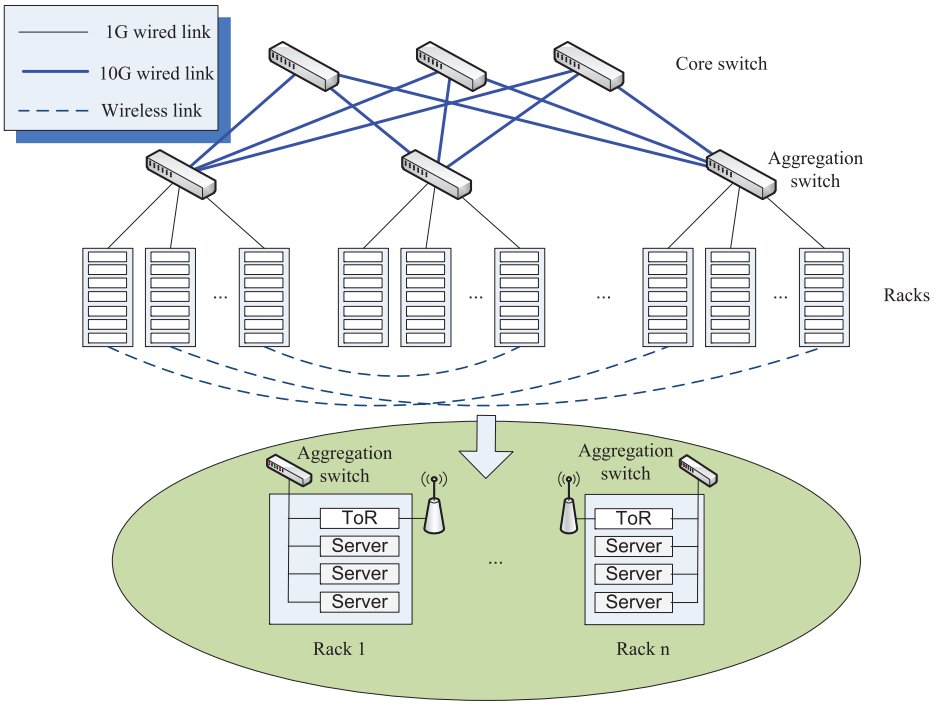
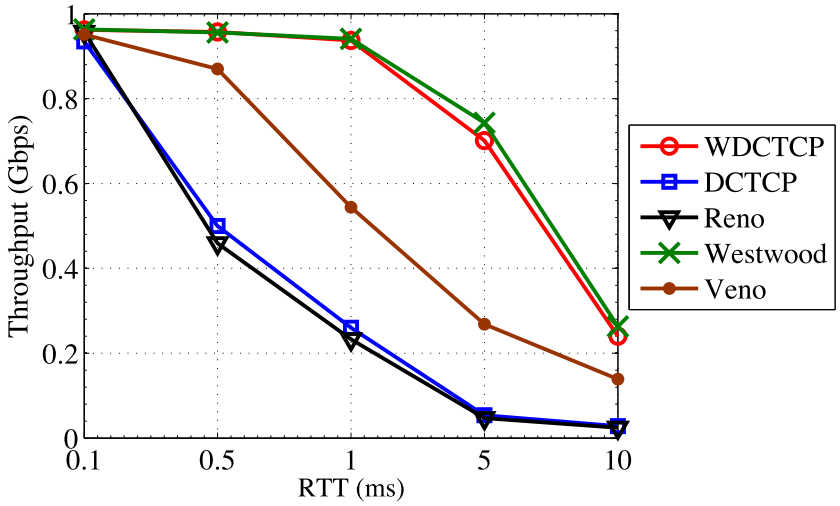


Fig. 4. Topology of the simulated wireless datacenter [18]

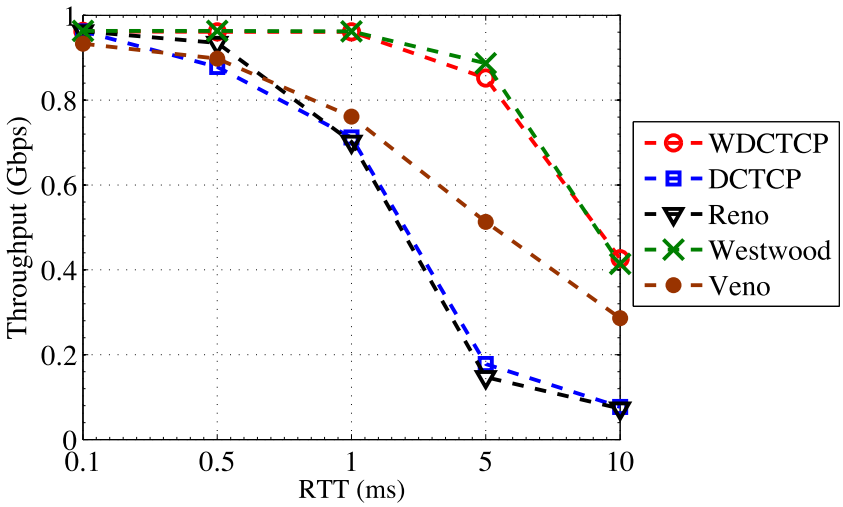
© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

Table I. The parameters setup of wireless datacenter links

	Bandwidth	Delay	Packet Loss Rate
1G wired links	1 Gbps	0.05 ms	0%
10 wired links	10 Gbps	0.1 ms	0%
wireless links	1 Gbps	0.1 - 10 ms	0.1% - 0.01%



(a) Packet Loss Rate = 0.1%



(b) Packet Loss Rate = 0.01%

Fig. 5. Performance of WDCTCP over wireless links

4.1 Performance in wireless links

We first investigate the performance of WDCTCP over wireless links. In our wireless datacenter testbed, the wireless link bandwidth is set as 1 Gbps, the packet loss rate as 0.1% and 0.01%, and the propagation delay of the links are varied from 0.1 to 10 ms. Figure 5 plots the experiment results. As shown in Figure 5, WDCTCP achieves similar throughput to TCP Westwood and much higher throughput than other TCP algorithms. Compared with DCTCP, TCP Reno and Veno, WDCTCP is more robust in dealing with

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

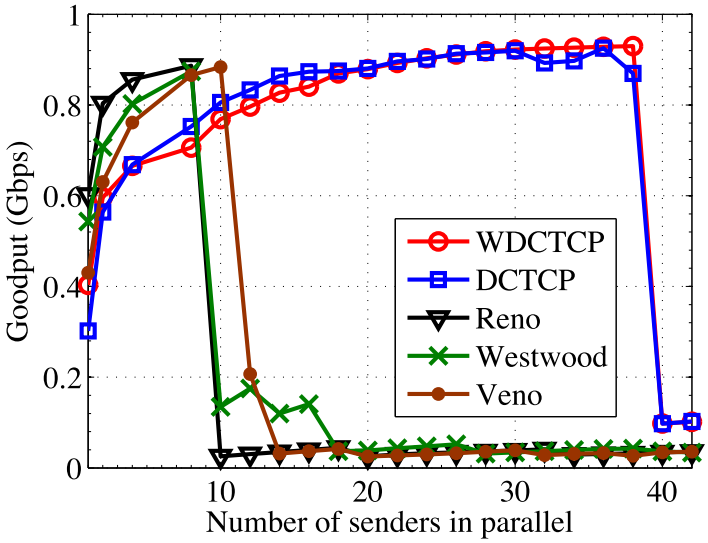


Fig. 6. Performance of WDCTCP in the incast scenario

wireless link random packet losses in the experiments.

4.2 Performance in datacenter networks

As reported in [7], the primary performance challenges for TCP in datacenters include TCP incast problems and large queuing delays caused by switch buffer queue buildup. We initially set an incast scenario to investigate the WDCTCP performance. In this setup, a group of servers in a rack simultaneously sends data blocks to a client in a rack of other aggradation domain. The data block size sent by each sender is 64 KB, and the servers do not send a new data block until the client receives the data blocks of all senders. The bottleneck bandwidth of the incast scenario is 1 Gbps, the propagation RTT is 1 ms. The number of servers in the experiment increases from 1 to 42. The experiment results are plotted in Figure 6. As shown in Figure 6, the throughput collapse threshold of WDCTCP is very close to DCTCP, and significantly higher than TCP Reno, Veno and Westwood. This result means that WDCTCP achieves comparable performance with DCTCP in incast scenarios.

Datacenter applications also requires the network queuing delay of TCP flows to be as short as possible [19]. We tested the queuing delay of WDCTCP in the simulated wireless datacenter setups. In this experiment, we let 2 and 20 long term TCP flows connect two severs belong to two racks in different aggregation domains (pass through a ToR switch, a Aggregation switch and a core switch). The bottleneck bandwidth of the flows passed links is 1 Gbps and the propagation RTT is 1 ms. We sample end-to-end RTTs for each second and plot the Cumulative Distribution Function (CDF) of one hour of samples of different TCP variants in Figure 7. We can see from Figure 7 that the RTTs of 2 and 20 WDCTCP flows are respectively stable around 120 and 250 μs (which is close to DCTCP), while TCP Reno, Westwood and Veno are two orders of magnitude larger than either WDCTCP or DCTCP. The queuing delay performance of WDCTCP in datacenter net-

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

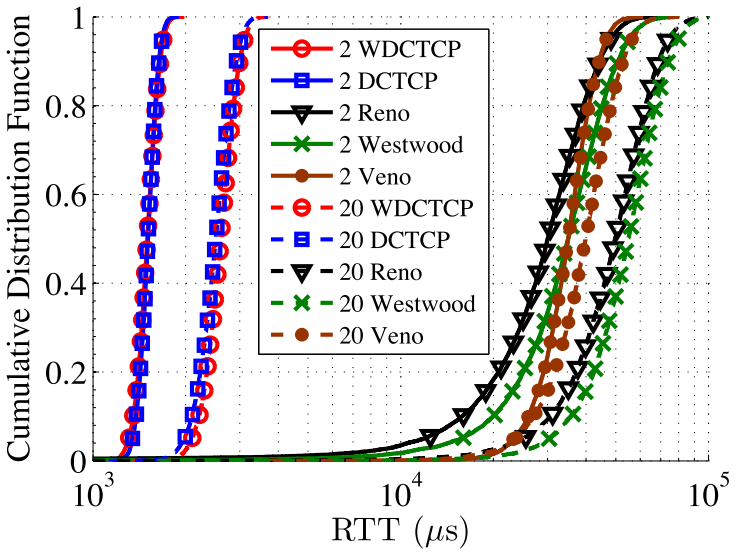


Fig. 7. RTT CDF comparison of different TCP variants

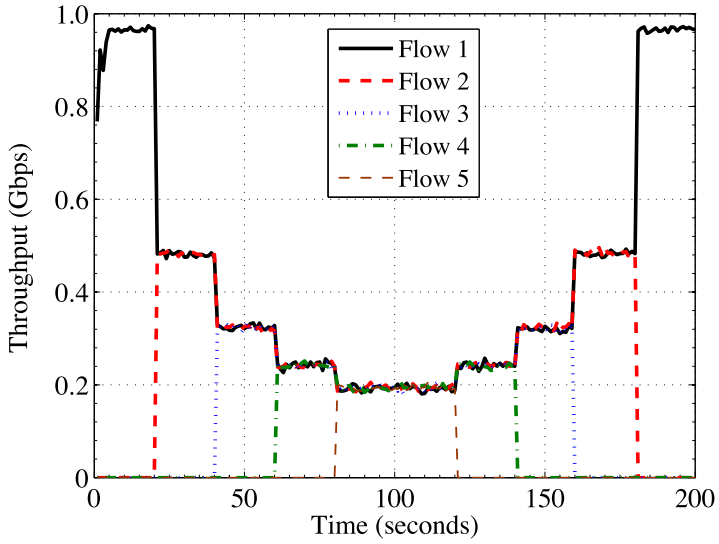


Fig. 8. Fairness and convergence of sequential starting experiments

works is comparable to DCTCP as shown in this experiment, which can meet the low latency requirement of datacenter applications.

4.3 Fairness and convergence

To show that WDCTCP flows quickly converge to their fair share, we use a sequential starting experiment to demonstrate the fairness and convergence feature of WDCTCP. We use 6 servers in a rack in the experiments. One of the servers acts as a receiver, while the others act as senders. The senders are sequentially started and stopped at 20 second intervals. The bandwidth of the ToR switch is 1 Gbps, and propagation RTT is 0.4 ms. The time series depicting the overall flow throughput is shown in Figure 8. As shown in Figure 8, WDCTCP quickly converges to fair share as the flows come and go. The Jain's Fairness Index is 0.99 throughout the experiment. This experi-

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013

ment shows that WDCTCP achieves graceful fairness and convergence in the datacenter network setup of our experiment. According to the experimental conclusion of DCTCP in [7], the fairness and convergence of WDCTCP can satisfy the common needs of datacenter applications.

5 Conclusions

In this paper, a novel TCP congestion control algorithm, designated as WDCTCP, is proposed to improve the performance of datacenter-oriented TCP algorithms over lossy links in wireless datacenters. WDCTCP introduces an ACK-based network bandwidth estimation into a DCTCP ECN-based congestion control framework. WDCTCP use the ACK-based network bandwidth estimation to mitigate the damage of wireless random packet losses and uses the ECN mechanism to achieve high throughput and low queuing delay in datacenter applications. The performance of our proposed approach in both wireless links and typical datacenter scenarios is validated by simulation experiments. To the best of our knowledge, there are no algorithms that specially target wireless datacenter networks. WDCTCP represents a beneficial first step in TCP congestion control algorithm design in wireless datacenter networks.

Acknowledgments

The work was supported by the National Natural Science Foundation of China (Grant No. 61202426, 61272350), and the State Key Laboratory of Software Development Environment (SKLSDE-2012ZX-20).

© IEICE 2013
DOI: 10.1587/elex.10.20130349
Received May 13, 2013
Accepted May 17, 2013
Publicized May 28, 2013
Copyedited June 25, 2013