Journal of Zhejiang University SCIENCE A
ISSN 1673-565X (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

EVCP: a convergence time improved high-speed transport congestion control protocol

LU Guang[†1], WANG Yong-chao[2], ZHU Miao-liang[1]

([1]*Institute of Artificial Intelligence, Zhejiang University, Hangzhou 310027, China*)

([2]*Network Center, Zhejiang University, Hangzhou 310027, China)*

[†]E-mail: luguang@zju.edu.cn

Received July 12, 2006; revision accepted Jan. 17, 2007

Abstract: The Internet evolves to incorporate very-high-bandwidth optical links and more large-delay satellite links. TCP faces new challenges in this unique environment. Theory and experiments showed that TCP becomes inefficient and is prone to be unstable as the per-flow product of bandwidth and latency increases, regardless of the queuing scheme. Variable-structure congestion Control Protocol (VCP) is proposed to address these problems. However, VCP has problem in terms of convergence time, i.e., it takes a long time for a new VCP flow to achieve fair bandwidth allocation if the existing VCP flows have large congestion windows. This paper proposed an Extended Variable-structure congestion Control Protocol (EVCP), which adopted a convergence controller. The basic idea of convergence controller is that if a flow has larger window than its fair window, its congestion window should be decreased more aggressively than usual in Multiplicative Decrease (MD) phase. Simulations showed that EVCP has better performance in terms of convergence time while keeping the advantages of VCP.

Key words: Congestion control, Variable-structure congestion Control Protocol (VCP), High-speed network
doi:10.1631/jzus.2007.A0849 **Document code:** A **CLC number:** TP393.04

INTRODUCTION

As the Internet continues to thrive, a large number of very-high-bandwidth optical links and more large-delay satellite links will be deployed. For example, the backbone links' capacity of the next generation Internet of China reaches 20 Gbps. Many applications that require global distribution for the data to be analyzed effectively, such as high-energy physics, bioinformatics and high performance grid can all benefit from the development of the Internet. Define the supposed loss rate for a connection to be the maximum packets loss rate that a congestion control algorithm will tolerate to sustain a given level of throughput. Let the packet loss recovery time for a connection with given transfer rate be the length of time required by a congestion control algorithm to return to its initial sending rate following the detection of a packet loss. Due to the need of the long packet loss recovery time and low supposed loss rate, current TCP, referred to as Reno (Jacobson, 1990), and its enhanced versions (Jacobson *et al*., 1992; Mathis *et al*., 1996; Hoe, 1996) cannot achieve high throughput in high-speed wide area networks.

Many protocols were proposed to approach this problem in a number of ways. One kind of approach that tried to be compatible with current TCP for a peaceful coexistence required changes only at the end-hosts for effective use. There are several works on such modifications of TCP for high speed, including: High-Speed TCP (Floyd, 2003), Fast TCP (Jin *et al*., 2004), BIC (Xu *et al*., 2004), TCP-RAB (Tang *et al*., 2004) and Scalable TCP (which is built on High-Speed TCP) (Kelly, 2003). Fast TCP uses buffer delay as congestion signal, on which TCP Vegas is based (Brakmo and Peterson, 1995). BIC views Congestion Control as a searching problem, in which the system gives yes/no feedback through packet loss. It consists of two parts: binary search increase and additive increase. The other kind of approach is that

explicit congestion feedback from routers enables a congestion control protocol to prevent queuing. Depending on whether the explicit feedback consumes few bits per packet or more, explicit congestion control protocol can be classified as limited-feedback and rich-feedback. Rich-feedback designs include eXplicit Control Protocol (XCP) (Katabi *et al*., 2002), Rate Control Protocol (RCP) (Dukkipati *et al*., 2005), and JetMax (Zhang *et al.*, 2006). Examples of limited feedback protocols are Explicit Congestion Notification (ECN) (Ramakrishnan and Floyd, 1999) and Variable-structure congestion Control Protocol (VCP) (Xia *et al*., 2005). XCP requires multiple bits to encode the congestion-related information exchanged between routers and end-hosts. Unfortunately, there is not enough space in the IP header for these bits. VCP leverages only the existing two ECN bits for network congestion feedback, and yet achieves performance comparable to XCP.

However, VCP has problem in terms of convergence time. That is, it takes a long time for a new VCP flow to achieve fair bandwidth allocation if the existing VCP flows have large congestion windows. Consider a certain scenario where load factor of bottleneck link is higher than 80%, flows have entered into the Additive Increase (AI) phase. As soon as a new flow starts, this new flow also enters into the AI phase. Its congestion window is increased by one packet per RTT. It has no chance to increase fast its congestion window through the Multiplicative Increase (MI) mechanism. When bottleneck link is overloaded, all senders of involved flows receive load factor from receivers piggybacked on ACK packets. Then, all flows enter into the Multiplicative Decrease (MD) phase. Hereafter all flows experience AI and MD in turn. So the newcoming flow need long time to increase its congestion window. That is why VCP framework needs long time to reach the point of fairly sharing bandwidth between contenting VCP flows.

This paper proposed an Extended Variable-structure congestion Control Protocol (EVCP), which adopted convergence controller. The basic idea of convergence controller is that if a flow has larger window than its fair window, its congestion window should be decreased more aggressively than usual in MD phase. Simulations showed that EVCP has higher performance in terms of fairness convergence time while keeping the advantage of VCP.

OVERVIEW OF VCP

VCP leverages only the existing two ECN bits for network congestion feedback, and achieves performance comparable to XCP, i.e., high utilization, low persistent queue length, negligible packet loss rate, and reasonable fairness. On the downside, VCP converges significantly slower onto a fair share of bandwidth than XCP.

VCP is built around two design guidelines: (1) Decouple efficiency control and fairness control. VCP routers compute only a congestion level, and end-hosts run one of the three algorithms (AI, MI and MD) as a function of the congestion level. More precisely, VCP classifies the network utilization into different utilization regions and determines the controller suitable for a given region. When network utilization is low, the goal of VCP is to improve efficiency more than fairness. On the other hand, when utilization is high, VCP accords higher priority to fairness than efficiency. (2) Use link load factor as the congestion signal. XCP uses spare bandwidth as a measure of the degree of congestion, while VCP uses load factor as the congestion signal, i.e., the relative ratio of demand and capacity.

In summary, VCP chooses the following three ranges to encode the load factor ρ_l: Low-load region: $\hat{\rho}_l = 80\%$ when $\rho_l \in [0\%, 80\%]$; High-load region: $\hat{\rho}_l = 100\%$ when $\rho_l \in (80\%, 100\%]$; Overload region: $\hat{\rho}_l > 100\%$ when $\rho_l \in (100\%, +\infty)$.

During every time interval t_p each router estimates a load factor ρ_l for each of its output links l as:

$$\rho_l = \frac{\lambda_l + k_q \tilde{q}_l}{\gamma_l C_l t_p}, \tag{1}$$

where, λ_l is the amount of input traffic during the period t_p, $\tilde{q}_l$ is the persistent queue length during this period, k_q controls how fast the persistent queue drains, γ_l is the target utilization, and C_l is the link capacity.

At any time t, a VCP sender performs one of the three actions based on the value of the encoded load factor sent by the network:

MI, for $\hat{\rho}_l = 80\%$:

$$cwnd(t + t_{RTT}) = cwnd(t) \cdot (1 + \xi). \tag{2}$$

ξ is calculated from the following equation: for k= 0.25 and $\hat{\rho}_l = 80\%$,

$$\xi(\hat{p}_l) = k(1-\hat{p}_l)/\hat{p}_l = 0.0625. \quad (3)$$

AI, for $\hat{\rho}_l$=100%:

$$cwnd(t+t_{RTT}) = cwnd(t) + \alpha. \quad (4)$$

MD, for $\hat{\rho}_l$>100%:

$$cwnd(t+\delta t) = cwnd(t)\cdot\beta. \quad (5)$$

To offset the impact of the RTT heterogeneity, the scaled MI/AI parameters ξ_s, α_s are calculated as follows:

For MI:

$$\xi_s \leftarrow (1+\xi)^{t_{RTT}/t_p} - 1. \quad (6)$$

For AI:

$$\alpha_s \leftarrow \alpha\cdot(t_{RTT}/t_p)^2. \quad (7)$$

SIMULATION AND ANALYSIS

VCP conjoins two bits of ECN in the IP header to convey congestion information. According to the congestion information from the receiver, the sender of VCP flow chooses one of congestion window control algorithms (MI, AI, MD) to respond to the congestion signal. Two bits can express four congestion signals. In VCP, $(00)_2$, $(01)_2$, $(10)_2$ and $(11)_2$ are interpreted as follows: ECN disable, low congestion level, high congestion level and overload, respectively. Congestion transition point from low level to high level is set as 80%.

Consider a certain scenario where load factor of bottleneck link is higher than 80%, flows have entered into the AI phase. As soon as a new flow starts, this new flow also enters into the AI phase. Its congestion window is increased by one packet per RTT. It has no chance to increase fast its congestion window through MI mechanism. When bottleneck link is overloaded, all senders of involved flows receive load factor from receivers piggybacked on ACK packets. Then, all flows enter into MD phase. Hereafter all flows experience AI and MD in turn. So the newcoming flow needs long time to increase its congestion window. We show a simulation result to confirm the long convergence time of VCP. Details of the simulation configurations and parameters used are described in Section 4. We assumed that there were only two VCP flows. The start time of flow 1 and flow 2 were 0 second and 100 second, respectively. Fig.1 shows the congestion window size of the two flows. We can see that it takes a long time for flow 2 to achieve fair bandwidth allocation.

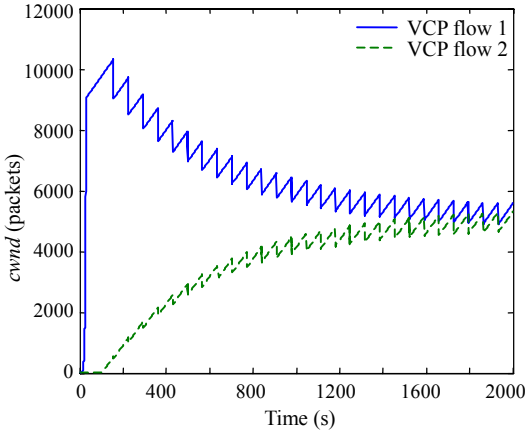

Fig.1 Congestion window size vs time for the VCP protocol

ALGORITHM OF EVCP

EVCP protocol is based on VCP and intends to reduce VCP's convergence time. EVCP protocol mainly adopts a convergence controller. It does not need any modifications in the intermediate route side and receiver side to support EVCP. Main modifications are at the sender side. Detailed description is given in the following subsections.

Convergence controller

Given $F=\{f_i\}$ a set of EVCP flows sharing the same bottleneck link on their paths, in which f_i is an EVCP flow, i=1,2,…,N. N is the amount of EVCP flows. W_i is the congestion window of f_i, W_{low} is the low-bound size of congestion window of flows in high speed network environment. Then, $L=\{f_i\in F|W_i<W_{low}\}$ is a subset of low-speed EVCP flows. $H=\{f_i\in F|W_i\geq W_{low}\}$ is a subset of high speed EVCP flows, and $F=L\cup H$.

The basic idea of convergence controller of EVCP protocol is that if a flow has larger window than its fair window, its congestion window should be decreased more aggressively than usual in MD phase. When an EVCP flow $f_l\in L$, and the sender of f_l receives load factor ρ_l>100%, the sender of f_l uses the

same algorithm as VCP protocol in MD phase. If an EVCP flow $f_l \in H$, and the sender of f_l receives load factor $\rho_l > 100\%$, the convergence controller will check whether the congestion window of f_l is in the downward trend, then try to identify whether f_l belongs to the set D. $D=\{f_i \in H | W_i$ is in the downward trend$\}$. In order to improve the convergence speed, if $f_l \in D$, the EVCP flow f_l should decrease its congestion window more aggressively and spare more available bandwidth for low-speed flows, else if $f_l \in \overline{D}$, f_l adopts the same MD algorithm as the VCP protocol's MD algorithm.

The convergence controller uses three variables W_{prev}, W_{max} and *NumDec* to identify whether an EVCP flow belongs to D. The mechanism of identification is like the work of (Nabeshima and Yata, 2004). As soon as the sender of an EVCP flow f_l receives congestion signal, the convergence controller of this flow is enabled. Details of its algorithm are as follows:

(1) if $f_l \in L$, then the parameter β of MD algorithm is set to 0.875;

(2) if $f_l \in H$, then:

I. If W_{prev} of f_l is less than or equal to W_l, $f_l \in \overline{D}$, β is set to 0.875, W_{max} and W_{prev} are set to W_l, *NumDec* is set to 0;

II. If W_{prev} of f_l is greater than W_l, *NumDec* is increased by 1, and to check whether the following two conditions are fulfilled: *NumDec* is more than or equal to N_1; ($W_{max}-cwnd$) is more than or equal to S.

(i) If it is satisfied with I and II, $f_l \in D$, β is set to 0.7, W_{max}, W_{prev}, and *NumDec* are set to 0, and f_l is not allowed to enter into MI phase until next MD phase.

(ii) If I and II are not fulfilled, $f_l \in \overline{D}$, W_{prev} is set to W_l. In addition, if *NumDec* equals N_2, then, W_{max} and *NumDec* are updated to W_l and 0, respectively.

Where, W_{max} is the maximal value of W_l during the downward trend; W_{prev} is the value of W_l before the last MD phase; *NumDec* is used to count the times of executing MD algorithm when an EVCP flow f_l is suspected to be in downward trend; N_1 is the low-bound times of executing MD algorithm to identify whether an EVCP flow f_l is truely in the downward trend; N_2 is the upper-bound times of executing MD algorithm to identify whether W_{max} is out of date.

Fig.2 shows the state-machine of the EVCP protocol. When one flow enters into MD phase, its convergence controller is called firstly. The convergence controller will make the decision whether or not the congestion window is in the downward trend and set the corresponding suitable value to the parameter β of MD algorithm. The flowchart of the convergence controller is shown in Fig.3.

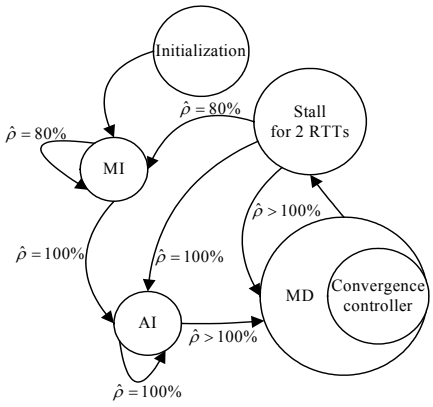

Fig.2 The state-machine of the EVCP protocol

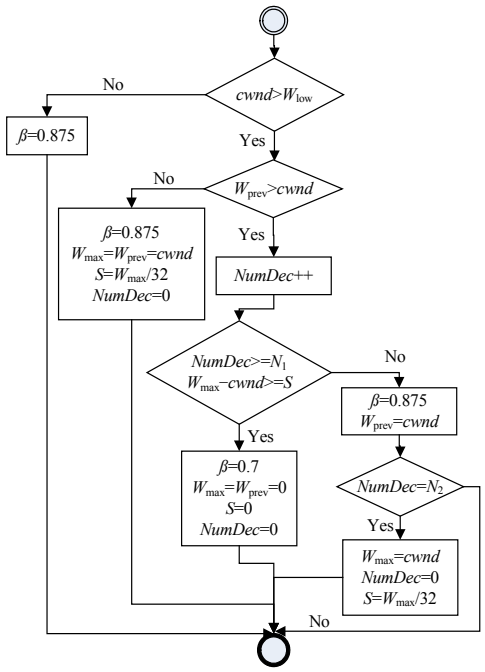

Fig.3 Flowchart of the convergence controller

Analysis

The VCP protocol takes $O(P\log\Delta P)$ RTTs to converge onto fairness for any link, where P is the per-flow bandwidth-delay product, and $\Delta P>1$ is the largest congestion window difference between flows sharing that link. Note that during AI phase all VCP flows grow the same amount of congestion window,

while only during MD phase the congestion window difference will be reduced. For any two flows i and j, at the mth epoch, we have:

$$\Delta w_{ij}(m)=\beta\cdot\Delta w_{ij}(m-1)=\ldots=\beta^m\cdot\Delta w_{ij}(0), \quad (8)$$

where $\Delta w_{ij}(m)=w_i(m)-w_j(m)$.

Compared with the VCP protocol, the EVCP protocol's convergence controller can quickly reduce congestion window difference between flows sharing that link. Suppose congestion windows of some EVCP flows are identified in the downward trend after N_1 epochs. For example, $f_i\in D$ and $f_j\in\bar{D}$. Given β_c the MD parameter $f_i\in D$, at the (N_1+1)th epoch, we have

$$\Delta w_{ij}(N_1+1)=\beta_c\cdot w_i(N_1)-\beta\cdot w_j(N_1)$$
$$=\beta\cdot\Delta w_{ij}(N_1)-(\beta-\beta_c)\cdot w_i(N_1), \quad (9)$$
$$\beta-\beta_c=0.175. \quad (10)$$

Then we have

$$\Delta w_{ij}(N_1+1)=\beta\cdot\Delta w_{ij}(N_1)-0.175w_i(N_1). \quad (11)$$

During the process when flows converge onto fairness, the measure to punish flows belonging to D is taken by convergence controller several times. So convergence controller of EVCP can improve the convergence speed.

PERFORMANCE EVALUATION

In this section, we use extensive ns2 simulations to evaluate the performance of EVCP. We compare EVCP protocol with VCP protocol in terms of the speed of converging onto fairness.

Simulation environment

Unless otherwise stated, the following parameter values were used as default. We used the single congestion link topology as shown in Fig.4, where S_i was sending packets to D_i (i=1, 2, ..., N). All traffic passed through the bottleneck link. The bottleneck link bandwidth was 1 Gbps. The link delay was 50 ms. Each router supported EVCP. The buffer size was 100% BDP. The packet size was 1500 bytes. For EVCP, N_1, N_2 and S were 2, 10, and $W_{max}/32$, respectively. As background traffic, a set of web traffic flows and a set of standard TCP flows whose maximum window size was limited to 20 packets were generated in both directions.

When the total number of flows was N, we assumed that flows 1~(N−1) started sending packets randomly. Specifically, their start time was randomized in the range 0~10 second in each simulation run. Flow N was assumed to start sending packets at 100 second. We then calculated the time from 100 second until the measured throughput of flow N exceeded the fair one. The time was used as the convergence time of flow N.

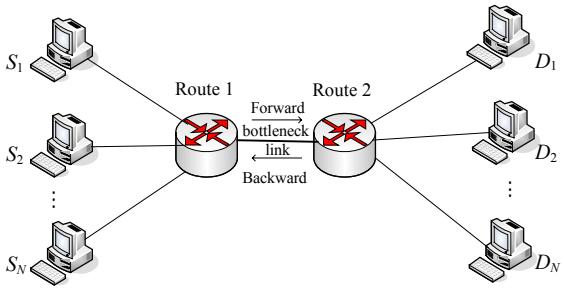

Fig.4 Simulation topology of the testbed

Convergence time of EVCP

In this subsection, we evaluated the convergence time in an environment with no background traffic. Fig.5 shows the congestion window size for two EVCP flows. After the EVCP flow 1 was started at 0 second, its congestion window was increased very quickly during MI region. The EVCP flow 1 entered into AI phase as soon as the load factor of bottleneck was higher than 80%. The EVCP flow 2 was started at 100 second and had to enter into AI phase to slowly increase its congestion window. After three MD epochs, the EVCP flow 1 was identified as a flow which should spare more bandwidth for other flows. The EVCP flow 2 got a chance to quickly increase its congestion window through MI algorithm at times such as 300 second and 540 second, while the EVCP

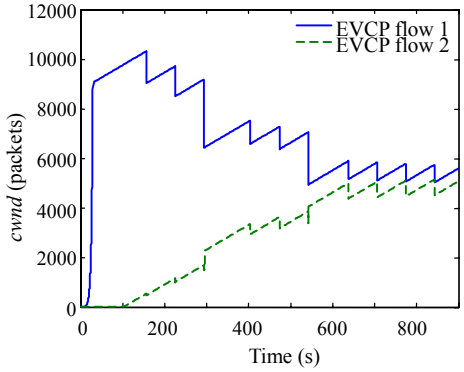

Fig.5 Congestion window size vs time for the EVCP protocol

flow 1, which was constrained by convergence controller, was not allowed to enter into MI phase. These measures of convergence controller can effectively reduce the congestion window difference between EVCP flow 1 and EVCP flow 2. Compared with Fig.1, we can see that EVCP protocol has better performance in convergence time than VCP protocol.

Fig.6 shows the convergence time of flow N when the total number of flows was varied from 2 to 10. We can see that the convergence time of EVCP flows is shorter than that of VCP flows regardless of the total number of flows. When VCP is used, decreasing the total number of flows increases the convergence time. The reason is that the fair share rate increases as the total number of flows is decreased. Thus, it takes longer time for the measured throughput of flow N to exceed the fair share rate as the total number of flows is decreased. On the other hand, the convergence time of EVCP protocol varies little in all cases.

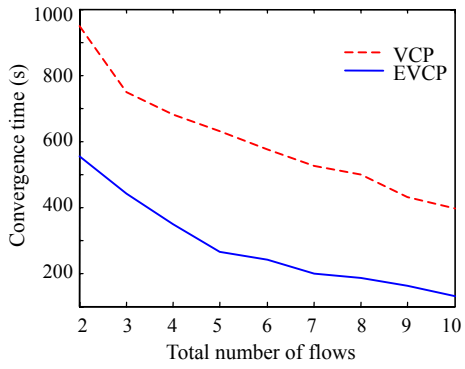

Fig.6 Convergence time vs total number of flows with background traffic

Fig.7 shows the convergence time of flow N when the total number of flows was 2, and the bottleneck link bandwidth was varied from 100 Mbps to 1 Gbps. We can see that the convergence time of EVCP is shorter than that of VCP regardless of the bandwidth.

Fairness of EVCP

To study the behavior of EVCP flows converging onto fairness and to compare with VCP protocol, we revert to the single bottleneck link with a bandwidth of 45 Mbps where we introduce 5 flows into the system one by one every 100 second. We also set the RTT values of the five flows to 40 ms, 50 ms, 60 ms, 70 ms and 80 ms, respectively. The reverse path has 5 flows that are always active. This scenario is the same as that in (Xia *et al*., 2005).

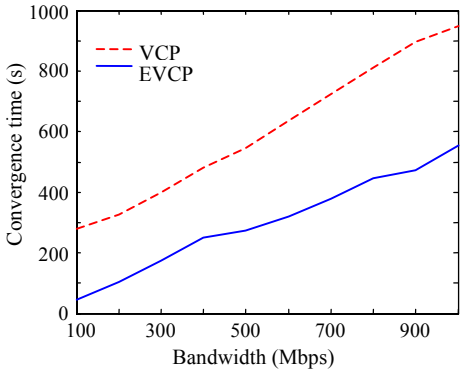

Fig.7 Convergence time vs bandwidth with background traffic

Fig.8a shows that EVCP reallocates bandwidth to new flows, compared with VCP, EVCP has shorter convergence time than that of VCP. When the bandwidth of the bottleneck link increased to 120 Mbps, Fig.8b shows EVCP has significantly shorter convergence time than VCP, e.g., 2 EVCP flows can converge onto fairness before the third EVCP flow was introduced into the system. The simulation scenario in Fig.9 is that the bandwidth of the bottleneck link was increased to 250 Mbps, and 5 flows and background traffic were sharing the same bottleneck link. Fig.9 shows that the convergence controller of EVCP protocol can effectively improve the performance in converging onto fairness where larger congestion window differences between flows existed.

In order to validate the applicability of EVCP protocol's convergence controller, we designed more wild simulation scenarios. As shown in Fig.10a and Fig.10b, the bottleneck link's bandwidth was increased to 1.4 Gbps and 3 Gbps, respectively. Ten flows and background traffic were introduced to share the same bottleneck link. The first 2 flows were randomly started at between 0 second and 1 second. Then 2 flows were introduced every 200 s. Fig.10 shows that convergence controller of the EVCP protocol speeded up flows converging onto fairness.

CONCLUSION

In this paper, we proposed EVCP, an Extended Variable-structure congestion Control Protocol for high BDP networks. Using extensive ns2 simulations,

we showed that EVCP not only keeps good properties such as high utilization, reasonable fairness, low persistent bottleneck queue, and negligible packet loss rate, but also significantly shortens the convergence time between contending flows.

EVCP protocol is based on VCP protocol and adopts convergence controller to help speed up converging onto fairness share between contending flows. The convergence controller is used to identify whether flows must aggressively decrease their congestion window to spare more bandwidth for the other contending flows. We used simple mathematical analysis to show the reason why convergence controller can effectively improve the performance in convergence time. We also carried out many simulation scenarios to evaluate the new introduced parameters of EVCP protocol. The results showed that the default values of these parameters are applicable for wide scenarios.

In the future, we will further do research to estimate the optimum values of N_1, N_2 and S. we will also study using the active queue management mechanism to solve the synchronous decrease issue to better improve the performance of EVCP.

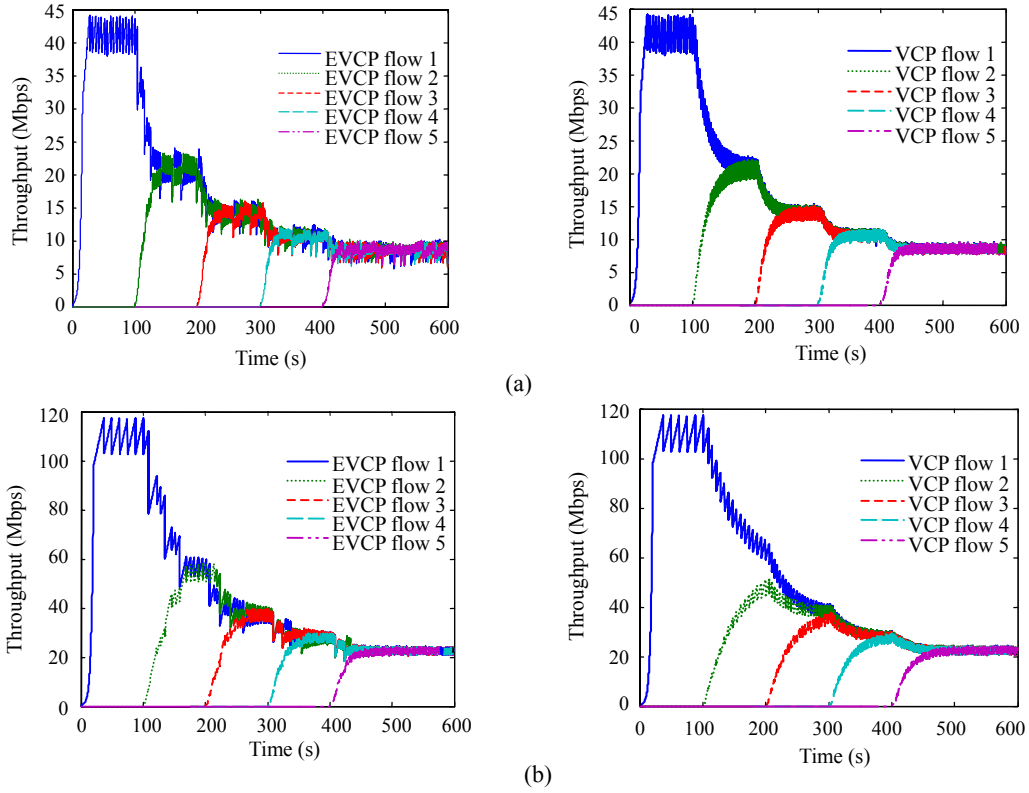

Fig.8 Flows converging onto fairness where the bandwidth of the bottleneck link is 45 Mbps (a) and 120 Mbps (b)

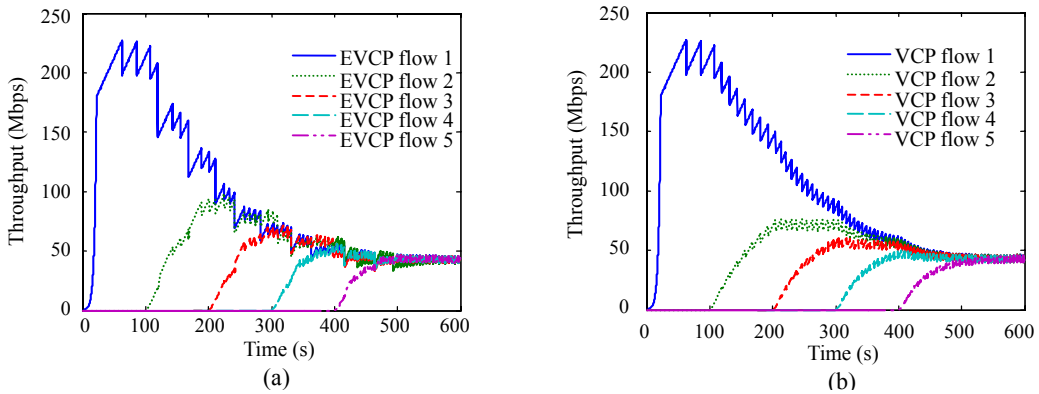

Fig.9 Five flows and background traffic sharing a bottleneck link whose bandwidth is 250 Mbps. (a) EVCP; (b) VCP

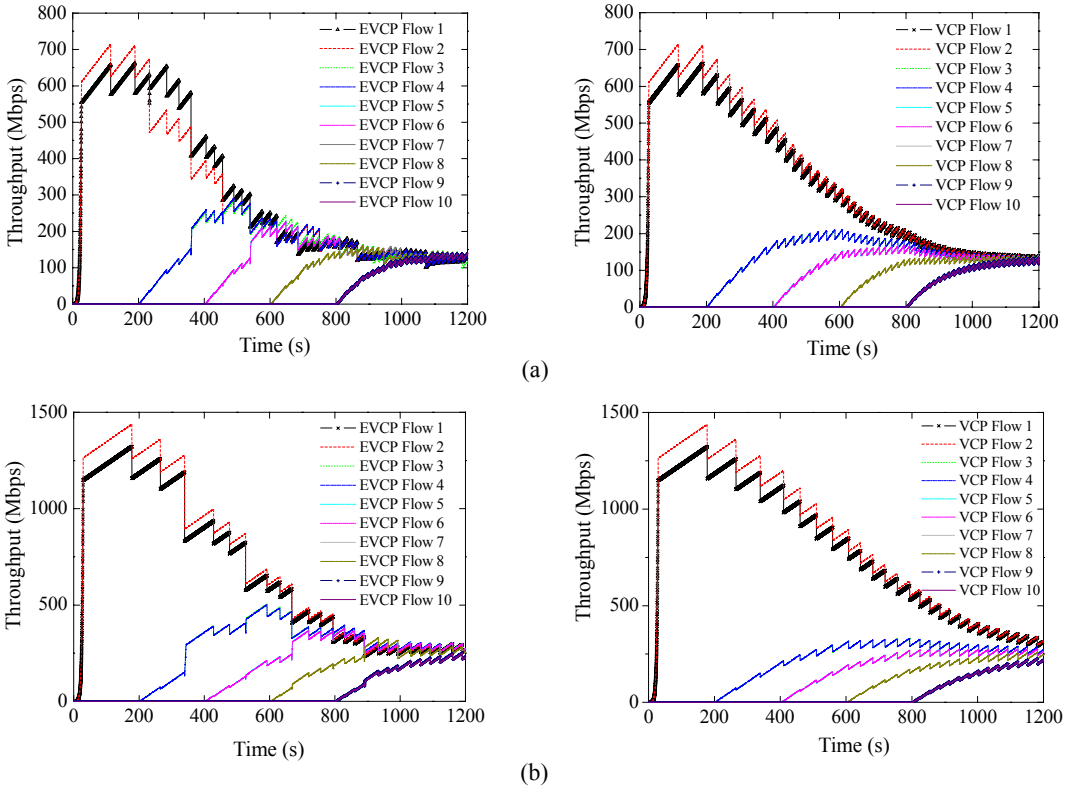

Fig.10 Ten flows and background traffic sharing a bottleneck link whose bandwidth is 1.4 Gbps (a) and 3 Gbps (b)

References

Brakmo, L., Peterson, L., 1995. TCP Vegas: end to end congestion avoidance on a global Internet. *IEEE J. Selected Areas Commun.*, **13**(8):1465-1480. [doi:10.1109/49.464716]

Dukkipati, N., Kobayashi, M., McKeown, N., 2005. Processor Sharing Flows in the Internet. Proc. 13th International Workshop on Quality of Service. Passau, p.286-297.

Floyd, S., 2003. High Speed TCP for Large Congestion Windows. RFC 3649.

Hoe, J., 1996. Improving the startup behavior of a congestion control scheme for TCP. *ACM SIGCOMM Computer Commun. Rev.*, **26**(4):270-280. [doi:10.1145/248157.248180]

Jacobson, V., 1990. Berkeley TCP Evolution from 4.3-Tahoe to 4.3 Reno. Proc. 18th Internet Engineering Task Force. University of British Colombia, Vancouver, p.365-376.

Jacobson, V., Braden, R., Borman, D., 1992. TCP Extensions for High Performance. RFC 1323.

Jin, C., Wei, D., Low, S., 2004. FAST TCP: Motivation, Architecture, Algorithms, Performance. Proc. INFOCOM'04. Hong Kong, p.2490-2501.

Katabi, D., Handley, M., Rohrs, C., 2002. Congestion control for high bandwidth-delay product networks. *ACM SIGCOMM Computer Commun. Rev.*, **32**(4):89-102. [doi:10.1145/964725.633035]

Kelly, T., 2003. Scalable TCP: improving performance in high-speed wide area networks. *ACM SIGCOMM Computer Commun. Rev.*, **33**(2):83-91. [doi:10.1145/956981.956989]

Mathis, M., Mahdavi, J., Floyd, S., Romanow, A., 1996. TCP Selective Acknowledgement Options. RFC 2018.

Nabeshima, M., Yata, K., 2004. Improving the Convergence Time of High Speed TCP. Proc. 12th IEEE International Conference on Networks. Singapore, p.19-23.

Ramakrishnan, K., Floyd, S., 1999. A Proposal to Add Explicit Congestion Notification (ECN) to IP. RFC 2481.

Tang, X.H., Liu, Z.L., Zhu, M.L., 2004. TCP-Rab: a receiver advertisement based TCP protocol. *J. Zhejiang Univ. Sci.*, **5**(11):1352-1360. [doi:10.1631/jzus.2004.1352]

Xia, Y., Subramanian, L., Stoica, I., Kalyanaraman, S., 2005. One More Bit is Enough. Proc. ACM SIGCOMM'05. Philadelphia, p.37-48.

Xu, L., Harfoush, K., Rhee, I., 2004. Binary Increase Congestion Control (BIC) for Fast Long-Distance Networks. Proc. INFOCOM'04. Hong Kong, p.2514-2524.

Zhang, Y., Leonard, D., Loguinov, D., 2006. JetMax: Scalable Max-Min Congestion Control for High-Speed Heterogeneous Networks. Proc. INFOCOM'06. Barcelona, Spain, p.1-13.