A COMPREHENSIVE MECHANISM TO IMPROVE TCP PERFORMANCE BY CONTROLLING ACKS

Runtong Zhang* and **Jian Ma****
Advanced Internet Technologies, Nokia (China) R&D Center
No. 11, He Ping Li Dong Jie, Beijing, 100013, PR China
runtong.zhang@nokia.com* and jian.j.ma@nokia.com**

Abstract: Being a widely used transport protocol in Internet, TCP has its own congestion control mechanism. However, TCP might be ineffective because it starts decreasing its rate just after it senses packet's loss, implying that the network has already been congested somewhere. Although several amendments have been developed to improved TCP performance over these years, some inefficiencies such as slow reaction and oscillatory flow still exist. A comprehensive mechanism of controlling ACKs in troublesome nodes is presented in this paper to improve TCP performance. This mechanism can be considered as an enhancement to the currently prevalent TCP and referred to ACK control. The basic idea of this mechanism arises from the fact that controlling ACK flows can indirectly affect the dynamics of TCP's behavior. In this paper, we mainly discuss two issues, i.e., flow smooth by delaying or accelerating backward ACKs, and precedence allowance by sequencing backward ACKs in differentiated services networks.

Key words: flow control, Internet, TCP, ACK, QoS, Diff-Serv

1. Introduction

Being a widely used transport protocol in Internet, TCP has its own congestion control mechanism. TCP operates in the following way [1,10 and 14]. It starts sending packet at a very slow rate (slow start) and watches the acknowledgments (ACK) from the destination to see if any packets are lost. If none is lost (source receives a consequent ACK), it speeds up. It keeps speeding up until a packet is lost, (if a packet does not arrive successfully, a time-out based approach is used to recover the lost packet,) at which time it decreases its rate. It keeps decreasing its rate until no packet is lost. It then increases its rate again, continually oscillating like this. It always losses packets in a round trip time and under-utilizing the network in another round trip time. The long recovery time for TCP causes a degradation of throughput. In addition, along with the invasion of new users and the rapid development of new applications, the requirement for the Internet to provide differentiated services (Diff-Serv, [6]) without complicated network functions is a new challenge to the current TCP.

Through these years, various amendments have been developed to improved TCP performance. Slow start, congestion avoidance, fast retransmit, and fast recovery [1] has been proven efficient and mature, and well applied. Among the recent research works, the Explicit Congestion Notification (ECN, [5]) is well known. It informs the source node of congestion occurrence by detection incipient congestion from the intermediate routers, which has the advantage of early congestion control. However, ECN needs some modifications to TCP behavior at end systems, which baffles its applications. Internet control message protocol (ICMP, [11]) source quench messages has the similar problems. Fast-TCP [15] is an enhancement to the current TCP by controlling backward ACKs in some specific cases, and it does not need to modify the implementation of either TCP senders or receivers. Diff-Serv is a new concept and hence little work has been done to include it in the TCP control mechanisms.

In this paper, a comprehensive mechanism of controlling ACKs in congested nodes is presented to improve TCP performance. This mechanism can be considered as an enhancement to the currently prevalent TCP and referred to ACK control. The basic idea of this mechanism arises from the fact that controlling ACK flows can indirectly affect the dynamics of TCP's behavior. ACK control consists of three main issues, which are flow smooth by delaying or accelerating backward ACKs, state detection by observing queue dynamics and precedence allowance by sequencing backward ACKs. A fuzzy approach is proposed to bridge the state detection and ACK control actions, and it also measures the degrees of off-normal system state. In addition, the concept of Diff-Serv is also allowed to this comprehensive ACK control mechanism. It is the hope that this mechanism achieves good TCP throughput, reduces the buffer requirement, lightens the flow congestion, adequately utilizes the bandwidth, and supports differentiated services. In addition, it is also simple to be implemented and does not need any modification of TCP behavior at end systems.

Two main issues of the TCK control mechanism, i.e., flow smooth by delaying or accelerating backward ACKs, and precedence allowance by sequencing backward ACKs, are discussed in sections 2 and 3, respectively, and some concluding remarks are given in section 4.

2. Controlling ACKs

To adequately utilize the bandwidth for a TCP connection, two contrary situations should try to be avoided. One is that too many packets congest at a node, which most likely results in packet loss. The opposite one is that a node is fully or partially idle while some source packets cannot be sent off due to limited congestion window [14] at that time, which normally leads to longer delay time and less system throughput. For easy reference, we call these two unexpected situations congestion and idle problems, respectively. This observation is valid for most kinds of networks, such as IP and ATM, and the nodes mentioned above might be routers, IP/ATM access nodes, ATM switches and etc., along the travelling paths. In this paper, we present the ACK control mechanism by studying the IP network and henceforce the node means router.

Traditionly, the congestion problem has attracted considerable attentions from Internet researchers and engineers, and various approachs have been proposed to solve this problem. For instance, most of the TCP control mechanisms mentioned in section 1 fall into the congestion control area. On the other hand, the idle problem is rarely mentioned in the Internet field.

We observe that controlling ACK flows can indirectly affect the dynamics of TCP's behavior, i.e., delaying backward ACKs may arrest the speeding up of sending rate and accordingly lighten the congestion problem, while accelerating ACKs may push the speeding up of sending rate and accordingly recover the idle problem. According to the observatoin, we devise the ACK control mechanism by dividing it into three phases, i.e., detection to the system state, identifying ACKs and controlling ACKs.

Detection to the sytem state actually means monitoring the level of buffer occupation at a network node. There are three typies of system state, i.e., congestion, normal and idleness. Congestion is notified if the buffer occupancy is larger than a prefixed high threshold. Idleness is notified if the buffer occupancy is smaller than a prefixed low threshold. And the normal state is that the buffer occupancy is between the high and the low prefixed thresholds, in the sense that the system is at an ideal and stable situation and no control actions should be taken. Whenever the system state is not at the normal state, i.e., at the congested or idle situation, some control actions will be taken and then identifying ACKs and controlling ACKs start to work, otherewise, no control actions needed. Some discussions concerning the choice of this high and low thresholds can be found in [7]. In addition, the changing trend of the size of the queueing packets in the node may affect the decision if a control action should be taken. For instance, if the the queue size is incresaing we should not likely take any extra actions to speedup the TCP sending rate; while if the queue size is decresaing we should not likely take any extra actions to slowdown the TCP sending rate. In other words, the control policy is of the hysteresis form. According to these observations, we devise the ACK control althorithm as shown in Figure 1.

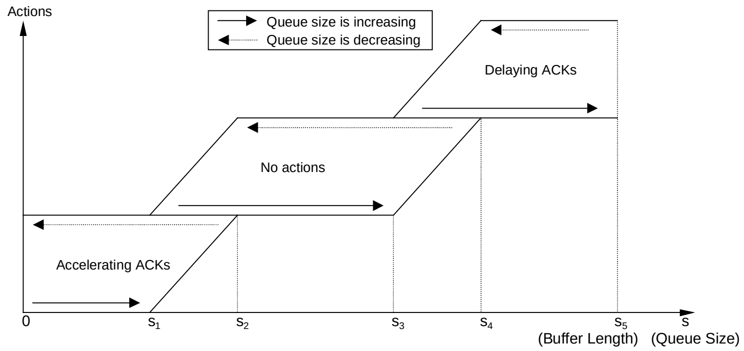

Figure 1. The illustration of the ACK control algorithm

Identifying ACKs has two meanings. One is that a router should check the ACK bits in TCP/IP packets so that ACK packets are found if the ACK bits are set. If yes, it is required to separate ACK flow from normal data traffic (refer to Figure 1), i.e. we could clear the ACK bits in the data packets and generate new ACK packets by copying the ACK information from the data packets. Hence, a backward ACK buffer is needed which is separate from the backward data buffer. Another one is that the router traverses its queue to see if any previous ACKs that belong to a same connection. The flow lable specification in IPv6 [3] packets will make this task easier. The former meanings is needed for control to both congestion and idleness, while the latter meanings is useful only to the idleness control. It is obvious that the ACKs control mechanism acts only when there are backword ACKs in the same node. This is not a drawback of ACK control. Such cases mean there need not any actions at all or there is no room to control the congestion for any TCP related mechanism. For instance, if a node is congested while there are currently no backword ACKs in the node, the congestion windows will be no long growing and congestion will finally be relieved. In such a case, except some drop methods (such as RED [4] and etc., which means that a loss of packet is unavoidable), any TCP related congestion control as mentioned in section 1 will have nothing to do. In other words, the ACK control mechanism has the fastest action time because its control loop is shorter, which is shown in Figure 2.

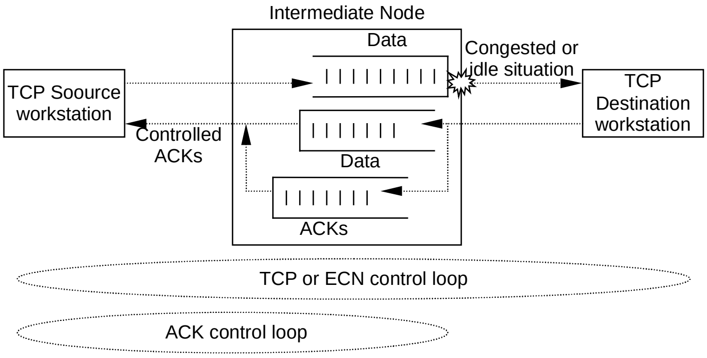

Figure 2. Control loop of ACK control vs. TCP

Controlling ACKs should be accordingly taken, if a congestion or idle situation is detected and there are backward ACKs in this node. Then, the rate of backward ACKs should be shaped according to the policy of ACK control. As previously stated, there are two opposite situations where ACK control should be applied, i.e., congestion and idleness. Generally, for the congested situation ACKs need to be delayed to block the speeding up of the data sending rate, while for the idle situation ACKs should be accelerated for the opposite purpose. However, the concrete algorithms or analysis on ACK shaping need further investigation. Theoretically speaking, two basic methods could be adopted here, i.e., rate-based and token-based. The rate-based method explictly calculates the leaking rate, and then ACKs are shaped according to the rate. The token-based method calculates the number of tokens for sending ACKs, and ACKs are allowed to leave only if there are tokens. These two methods are all based on the knowledge of resource conditions such as spare buffer space or the variation rate of buffer occupancy. Traditionally, the terms of leak and token are often mentioned in congestion control area. In this paper, there are used to two opposite deviant situations. Whenever the leaking rate or token generating rate is slower than the uncontrolled ACK rate, the ACKs is accordingly delayed which leading to longer stay in the backward ACK buffer. Whenever the leaking rate or token generating rate is faster than the uncontrolled ACK rate, the ACKs should be then accelerated which leading to shorter stay in the backward ACK buffer. The latter case can be realized by filtering some ACKs [2] by taking advantage of the fact that TCP ACKs are cumulative. Specifically, if any ACKs in the buffer belong to a same connection, some or all of them should be removed, and a latter ACK is forwarded to the sender and hence the ACKs' rate is several times as much as that no ACKs are removed. The policy that the filter used to drop packets is configurable and can either be deterministic or random.

ACK control mechanism can be easily implemented in various network nodes where information of link utilization or buffer occupancy may be employed to notify the congestion or idle situation. However, the most suitable position to implement ACK control mechanism is gateway where the tempestuous turbulence of system performance most likely takes place. We will give more technical remarks on this issue.

3. Differentiated Services in ACK Controlling

The Internet is at a phase of great changes. There are several stringent and new requirements for the networks because of two reasons: the invasion of new users, and the rapid development of new applications. These requirements mean that network capacity must rapidly be increased, and real-time service has to be fundamentally improved, which falls into the QoS management domain.

This has led to the founding of IETF Differentiated Services (Diff-Serv, [6]) Working Group (DSWG). Basically, Diff-Serv effort tries to provide a natural evolution path from the current best effort environment to a new environment capable to provide differentiated services without complicated network functions. According to Diff-Serv concept, it is not necessary to perform a unique QoS reservation for each flow. Within the Diff-Serv framework, the DS (Diff-Serv) byte is used to mark a packet to receive a particular forwarding treatment, or per-hop behavior (PHB) [8], at each network node. The DS byte overrides existing definitions of the IPv4 TOS byte overrides existing definitions of the IPv4 TOS octet and the IPv6 Traffic Class octet. Six bits are used as a DS codepoint (DSCP) to select the PHB that a packet experiences at each node whereas two bits are Currently Unused (CU), which is shown in Table 1.

Table 1: The DS byte structure

0 1 2 3 4 5	6 7
DSCP	CU

So far, several independent and different Diff-Serv themes are proposed. For example, we at least have Simple Differential Services, a Two Bit Differentiated Services Architecture for the Internet, Simple Integrated Media Access (SIMA, [12 and 13]), and Differentiated Service Scheme with Feedback Precedence Information (FPI, [16]). Although these schemes utilize the DS byte in different ways to define the PHB for their packets, one common point is that each packet in Diff-Serv networks is assigned with a unique priority in the sense of delay (such as real-time and non-real-time) or drop.

Within the ACK control mechanism, Diff-Serv requirement could be allowed based on the following observations. No matter if an ACK is an independent packet or one which is embed in an IP data packet, it should carry all TCP information including DS. Hence, Diff-Serv requirement could be taken into account during the ACK control course.

The ACK control with Diff-Serv is illustrated in Figure 3. Different from Figure 1, there are two or more backward ACK buffers. The ACKs are grouped according to their delay or discard priorities. Basically, such a classification of ACK types is set to guarantee that packets with higher delay priorities (such as real-time requirement) can be served sooner, in the meanwhile lower priorities (such as non-real-time) slower. However, due to that there are some internal relationship between the delay and discard priorities [17], the discard priority may also take effects on setting the backward ACK buffers. Theoretically speaking, the minimal number of the backward ACK buffers is the number of delay priorities, and the maximal number of the backward ACK buffers is the total number of all the priorities, i.e., the sum of all delay and discard priorities.

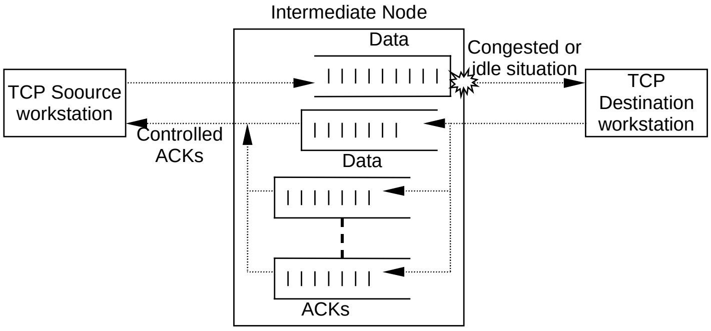

Figure 3. ACK control with differentiated services

Whenever the backward ACKs need to be delayed, (refer to the discussions in section 2), the ACKs with the lowest delay priority are first delayed. If there are no this kind of ACKs in the corresponding buffer, then the ACKs with one higher grade of delay priority should be checked and delayed, and so on, according to the strict queue policy or any relative policies. On the opposite side, whenever, the backward ACKs need to be accelerate, i.e. ACK filtering is needed, the ACKs with the highest delay priority are first checked and filtered. If there are no this kind of ACKs in the corresponding buffer, then the ACKs with one lower grade of delay priority should be checked and accelerated/filtered.

Although this ACK control mechanism with Diff-Serv is developed from the viewpoint of flow control, it has positive impact on the implements of the Diff-Serv concept. According to this mechanism, the concept of delay priority is better guaranteed than that in the normal TCP implement.

4. Concluding Remarks

This paper presents a comprehensive mechanism of controlling ACKs in troublesome nodes in order to improve TCP performance. This mechanism can be considered as an enhancement to the currently prevalent TCP and referred to ACK control. The basic idea of this mechanism arises from the fact that controlling ACK flows can indirectly affect the dynamics of TCP's behavior. In the mechanism, the concept of Diff-Serv is better implemented.

In the remainder of this concluding section, we would like propose some technical remarks.

- It is obvious that accelerating ACKs will likely shorten the round trip time (RTT). Then how is the effect of delaying ACKs? The answer should be the same based on the following observation. The ACK control mechanism acts only when an off-normal situation is detected, and it could lighten the congestion and adequately utilizes the bandwidth. In most time instants, the system is in the normal situation and no actions would be taken. Hence, the normal RTT will not be negatively affected. Because the unexpected problems can be overcome and no extra RTT is needed, hence it is the conclusion that, with the ACK control mechanism, TCP throughput can be improved.
- In this paper, ACK control solves not only the congestion problem, but also the idle problem. In fact, it seems to be the first time in the area of TCP study that the idle problem is explicitly proposed and examined. In order to smooth the system flows and improve the system throughput, more attention is needed on the study of

idle problem.

- IP networks are connectionless that ACK packets might travel along different paths from the forward data paths. It seems difficult for routers to determine whether data packets and its returning ACKs share the same path. However, in Internet there are actrually numerous situations that have a unique path between two routers. Many local networks or intranet are interconnected to Internet via a single router, where mechanism of delaying ACKs can be used. Particularly, this mechanism might be implemented in routers as an enhancing policy and can be optionally enabled depending on the location the routers deployed. This analysis is on the condition that ACK control is implemented at any troublesome nodes, it is not held when ACK control is only implemented at the gateways as previously discussed.
- Some transmission systems exhibit asymmetry with a larger data rate in one direction than the other. For example, some satellite systems are one way only and use a terrestrial return path. Terrestrial links might differ from the satellite links in terms of propagation delay, bit error rate, bandwidth, etc. Even in such kinds of asymmetry, ACK ocntrol could be of extra value because the ACK accelerating/filtering functions is helpful in saving bandwith. This character of ACK control is generated from its ability of adaptation.
- Since multiple TCP connections might share one link, it remains an open issue that whether ACK control can control ACKs for different TCP connections separately. We believe that classifying different types of TCP connections will produce an effective control because different TCP connections possess distinguishing traffic and different effect on networks. For example, TCP connections for file transfer services have greater impact on networks than those for telnet or rlogin. This is a research topic for further study.
- According to the above observation, we propose the ACK control with the allowance of Diff-Serv concept. However, to decrease the workload for routers, it would be better not to set too many backward ACK buffers. The suitable number of ACK buffers could be two (for real-time and non-real-time ACKs) or a little more.

References

[1] M. Allman, V. Paxson and W. Stevens, "TCP congestion control", *RFC 2581*, April, 1999.

[2] H. Balakrishnan, V.N. Padmanabhan, and R.H. Katz. "The effects of asymmetry on TCP performance", *Proc. 3rd ACM/IEEE Intl. MobiCom, Budapest, Hungary*, Sept. 1997.

[3] S. Deering and R. Hinden, "Internet protocol, version 6 specification", *Request for Comments* 1883, December 1995.

[4] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance", *IEEE/ACM Transactions on Networkking*, 1(4), pp. 397-413, Aug. 1993.

[5] S. Floyd, "TCP and explicit congestion notification", *ACM Computer Communication Review*, 24(5), pp. 10-23, Oct. 1994.

[6] Kalevi Kilkki, *Differentiated services for the Internet*, Macmillan Technical Publishing, Indianapolis, USA, July 1999.

[7] Y. Lai and Y. Lin, "Choice of high and low thresholds in the rate-based flow control scheme", *IEE Proceedings on Communications*, vol. 146, no. 2, April, 1999.

[8] K. Nichols, S. Blake, F. Backer and D. Black, "Definition of the differentiated services field (DS field) in the IPv4 and IPv6 Headers", *Request for comments* 2474, December 1998.

[9] J. Postel, et.al., "Internet protocol", *Request for Comments* 791, October 1981.

[10] J. Postel, "Transmission control protocol", *RFC 793, September*, 1981.

[11] K. K. Ramakrishnan, D. Chiu and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks", *ACM Transactions on Computer Systems*, 8(2), pp. 158-181, 1990.

[12] Jussi Ruutu and Kalevi Kilkki, "Simple integrated media access (SIMA) – a comprehensive service for future internet", *Proc PICS'98*, 1998.

[13] Jussi Ruutu and Kalevi Kilkki, "Performance of simple integrated media access (SIMA)", *Proc VVDC'97*, 1997.

[14] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*, Addison-Wesley, Reading, MA, Nov 1994.

[15] J. Wu, P. Zhang, T. Du, J. Ma, and S. Cheng. "Improving TCP performance in ATM network by the fast TCP flow control." *Proc. International Conference on Communication Technology (ICCT'98)*, pp. S46-07-1:S46-07-5, China Beijing. Oct. 1998.

[16] Runtong Zhang and Jian Ma, "On the enhancement to a differentiated services scheme", *Proc. 2000 IEEE/IFIP Network Operations and Management Symposium (NOMS'2000)*, Hulunono, USA, April 2000.

[17] Runtong Zhang and Jian Ma, "A fuzzy approach to the balance of drop and delay priorities in the differentiated services networks", *Proc. 4th World Conference on Systemsics, Cybernetics and Informatics (SCI'2000)*, Orlando, USA, July 2000.