提供整合服務之高速區域網路架構

計畫編號：NSC 88-2213-E-002-079
執行期限：87年8月1日至88年7月31日
主持人：蔡志宏 國立臺灣大學電機工程研究所

一、中文摘要

因為第三層交換技術的快速發展及服務品質的要求，如何在高速交換區域網路上提供服務品質保證已成為熱門的研究領域。本研究計畫將發展一頻寬保證精度較粗糙之排程演算法，以較低之硬體複雜度支援大量之連線，此特性在骨幹網路之路由器上非常重要。此排程演算法之硬體製作亦為研究之重點。此外，如何納入第四層之控制機制及利用骨幹之ATM服務與ATM服務品質需求之對應關係之研究亦包含於本計畫中。

英文摘要

Due to the rapid development of Layer 3 switching and the requirements of Quality of Service, providing Quality of Service guarantee over high speed and/or switched LAN is a promising new area. This project aims to develop a novel scheduling algorithm under loose bandwidth guarantee granularity. This scheduling algorithm has to support large number of connections, which is very important for the backbone switches or routers. Hardware implementation of this scheduling algorithm is also evaluated. Feasibility of including Layer 4 controlling mechanisms is studied. Last of all, the mapping between backbone ATM Service parameters and high layer QoS requirements is also studied.

二、計畫背景及目的

It is well known that providing real-time and multimedia communications over the Internet and high speed LAN is already a widely accepted technology trend. Many future applications of computer networks will rely on the ability of the network to provide Quality-of-Service (QoS) guarantees. These guarantees are usually in the form of bounds on end-to-end delay, bandwidth, delay jitter, packet loss rate, or a combination of these parameters. However, the unpredictable congestion level over Internet and LAN, which often involves packet losses as well as queueing delay, has made the quality of real-time multimedia communications unacceptable. Therefore, traffic scheduling algorithms are a necessary part of future integrated-services networks that will provide a broad range of QoS guarantees.

The most important academic which started the introduction of a whole new set of new scheduling algorithms for high speed network working should be the General Processor Sharing (GPS)[3][4] and Weighted Fair Queueing (WFQ)[5]. These papers, which are based on the same kind of philosophy and have been widely referenced, introduce new approaches to more fairly and proportionally allocate bandwidth among sessions with different bandwidth requirements. The scheduling algorithms of packet transmission, based either on GPS or WFQ, can be implemented over a network node. For example, Cisco System has successfully implemented the WFQ discipline in their router, for providing bandwidth control.

Originally, the VirtualClock scheduling algorithm aims to emulate a kind of time division multiplexing (TDM) server. Packet scheduling based on VirtualClock assigns a tag for each arriving packet, and packets are serviced in a non-decreasing order of tags. The operations of VirtualClock algorithm are detailed in [11] and [12]. Because the virtual finishing time of each packet is assigned based on the arriving pattern and the reservation of

the flow to which the packet belongs, the VirtualClock algorithm ensures each well-behaving flow good performance. However, the calculation of the virtual finishing time is independent of the behaviors of other flows. As a result, even if a flow sends packets at a rate slightly higher than specified, it still get *punished* by the VirtualClock server, regardless of fact that this behavior does not affect the performance of other flows and could effectively reduce its own backlog, as pointed out in [13]. By modifying the calculation of the virtual finishing time of a packet to be correlated with the active flows, we believe the *punishment* phenomenon of VirualClock would be relieved.

In the current Internet environment, when demands exceed the network capacity, instantly congestion occurs such that the ratio of packet loss may increase rapidly. The traditional TCP congestion control is implemented via a reactive, closed-loop, dynamic window control scheme: acknowledgments received by the sender are used to trigger the transmission of new segments. The TCP congestion control algorithm also tries to obtain as much bandwidth as possible from the network by continually increasing the send rate until packet losses occur. However the TCP transport protocol detects congestion only after a packet has been dropped or TCP timeout. It is possible that a congestion event has been happening, but the TCP will still increase congestion window before it detects packet losses. Unless the window sizes are optimally configures (i.e., the maximum congestion window is less than the queueing capacity of the network), usually the large number of TCP connections will eventually cause network congestion.

We also designed a new gateway congestion avoidance scheme that is referred as Pseudo Advertising Window (PAW) to achieve smaller buffer requirement, low queueing delay, no unnecessary packet loss and high bandwidth utilization.

三、研究方法及結果

I. VirtualClock Plus Algorithm

VirtualClock Plus scheduling algorithm is described as follows. First, all flows reserve their guaranteed bandwidth at connection establishment time. Secondly, the virtual finishing time of each arriving packet is calculated based on the system virtual time. Finally, the server is work-conserving, and transmits packets from all sessions in the increasing order of virtual finishing times. The queueing model of VirtualClock Plus algorithm is shown in Fig. 1. Suppose flow i becomes active after a_i^k, which is the arrival time of packet P_i^k, the k-th packet of flow i. Then the virtual finishing time of packet P_i^k, denoted as F_i^k, is defined by

$$F_i^1 = ST_{VCP}(a_i^1) + \frac{L_i^1}{r_i},$$

$$F_i^k = \max\{ST_{VCP}(a_i^k), F_i^{k-1}\} + \frac{L_i^k}{r_i}, k > 1$$

$$F_i^j = F_i^{j-1} + \frac{L_i^j}{r_i}, j > k$$

Here, the system virtual time, $ST_{VCP}(t)$, only needs to be calculated once a flow changes its backlogged state. Hence, the computation complexity of the system virtual time is $O(1)$, which is much less then WFQ[5].

II. Pseudo Advertising Window

In TCP original design, the receiver indicates the maximum total segment size it is willing to accept by specifying the ADVERTISING WINDOW field of the TCP-Header. The field contains a 16-bit unsigned integer in network-standard byte order. We treat the window advertisement as a specification of the receiver's current buffer size. In response to an increased window advertisement, the sender increases the size of its congesting window and sends more data. On the other hand, when the advertising window decreases, the sender decreases the congesting window. Hence the primary purpose of the ADVERTISING WINDOW field design was to avoid buffer overflow in end systems. However

in the current network, buffer overflow often happen more frequently rather in the gateway than in the end node.

The Pseudo Advertising Window (PAW) mechanism requires the router or the IP switch to compute the optimal congestion window size of some TCP sessions. The PAW mechanism then replaces the original *cwnd* value of the *ack* IP packet with the optimal *cwnd* size if necessary. That is, if the original value of Advertising Window filed was smaller than the optimal *cwnd* size the gateway computed, it would not do any change. In turn, by allowing the gateway to allocate all TCP session with the most appropriate *cwnd*, the buffer requirement would be reduced and ideally the congestion should be eliminated.

It will be easier to find the location of Advertising Window field in IPv4 because of the fixed IP header format. The HLEN of IP packet header is shown in Fig.2. Our simulation results also show that smaller queue length and better fairness can be achieved if PAW is used.

III. Computation of Optimal Congestion Window

Recently routers or IP switches show that there are still some restrictions in their hardware design regarding the support of IP Quality of Services, especially when large number of IP flows are to be with differential services. As a result, we discuss the computation of the optimal cwnd in two cases: Per-Flow Queueing case and Non-Per-Flow Queueing case. Here we assume TCP connections and UDP sessions are buffered in the different queues of the routers or IP switches, and the allocated bandwidth for T CP is acquired.

i. Per-Flow Queueing

The total operation of optimal congestion window size computing is presented in the Fig.3. When a new TCP session set up, the gateway records the interval between the arrival time of the first two packets as the *base_rtt* to compute the optimal congestion window size and enable PAWS. After the initial waiting, the session go into the OK State. The gateway might update *cwnd*, only when the queue is empty or bigger than the tolerant length.

ii. Non-Per-Flow Queueing

As we described before, restrictions exist in most routers if the large number of flows is to be supported with different classes of services via scheduler algorithm, like WFQ. As a result, one must take into account of the case that multiple flows with one single queue but yet require optimal congestion window control. The computation operation is shown in Fig.4.

四、結論與討論

The key contribution of the VirtualClock Plus algorithm is the elimination of "*punishment*" phenomenon of VirtualClock algorithm and the preservation of advantages in low complexity and low delay bound. The complexity of VirtualClock Plus is O(log N) but delay performance is similar to WFQ.

To support different QoS services, the next generation routers and switches will all implement some kind of scheduler algorithm. The complexity of an algorithm can become the key factor that determines whether an algorithm could be successfully deployed on current Internet. The PAW mechanism should be easy to implement in the current routers and IP switches based on the following feasibility study. We consider PAW might be realized to achieve better TCP congestion control in Internet.

五、參考文獻

[1] M. Li and Z. Tsai, "Design and Analysis of the GCRA Traffic Shaper for VBR Services in ATM Networks," *IEEE ICC'97*, Montreal, Canada, June 1997.

[2] M. Li and Z. Tsai, "A Traffic Shaper for Supporting CBR and VBR Services in ATM Networks," submitted to IEEE Trans. Networking.

[3] A. K. Parekh and R. G. Gallager, " A General Processor Sharing Approach to Flow Control in Integrated Services Networks - The Single Node Case,"

ACM/IEEE Trans. on Networking, Vol.1 1993.

[4] A. K. Parekh and R. G. Gallager, " A General Processor Sharing Approach to Flow Control in Integrated Services Networks - The Multiple Node Case," ACM/IEEE Trans. on Networking, Vol.2, April 1994.

[5] A. Demers, S. Keshav, and S. Shenkar, Analysis and Simulation of a Fair Queueing Algorithm, Internetworking Res.

[6] K. Thompson, G. J. Miller and R. Wilder, "Wide-Area Internet Traffic Patterns and Characteristics," IEEE Network, pp.~10--23, Nov./Dec. 1997.

[7] J. Martin and A. Nilsson, "The Evolution of Congestion Control in TCP/IP: from Reactive Windows to Preventive Rate Control"

[8] L. Barkmo, S. O'Milley and L. Peterson, "TCP Vegas: New Techniques for Congestion Detection and Avoidance," ACM SIGCOMM94, 1994.

[9] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Transaction on Networking, pp.~397--411, August 1993.

[10] J. Ahn, P. Danzing, Z. Liu, L. Yan, "Evaluation of TCP Vegas: Emulation and Experiment," ACM SIGCOMM95.

[11] N. R. Figueira and J. Pasquale, ``An Upper bound on Delay for the VirtualClock Service Discipline," IEEE/ACM Trans. Networking, vol. 3, no. 4, pp. 399-408, Aug. 1995.

[12] L. Zhang, ``VirtualClock: A New Traffic Control Algorithm for Packet Switching Networks," ACM Trans. Comput. Syst., vol. 9, pp. 101-124, May 1991.

[13] H. Zhang, ``Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks," Proc. IEEE, vol. 83, no. 10, pp. 1374-1396, Oct. 1995.

六、圖表

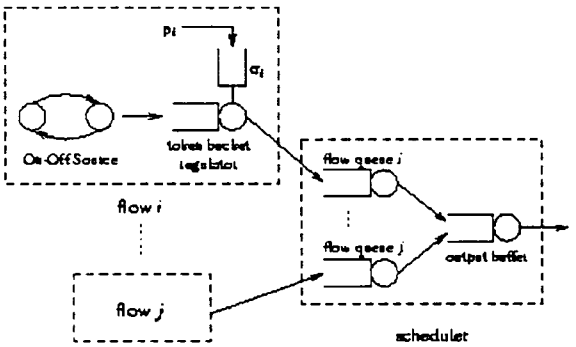

圖 1：Queueing model of VirtualClock Plus algorithm.

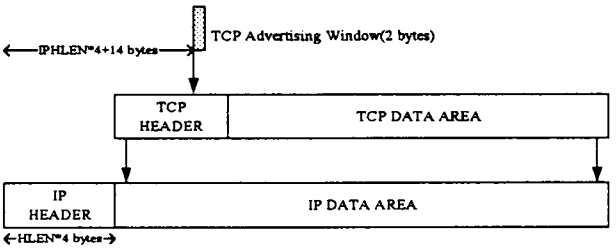

圖 2：The location of Advertising Window

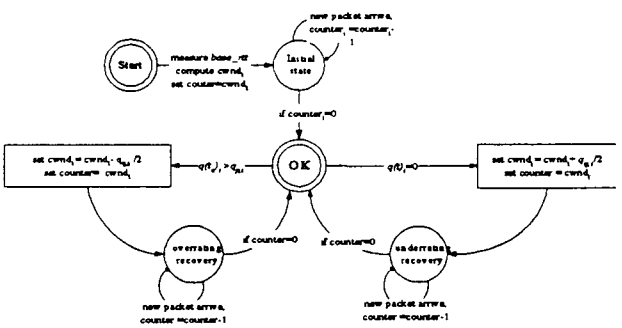

圖 3：Per-Flow Queueing

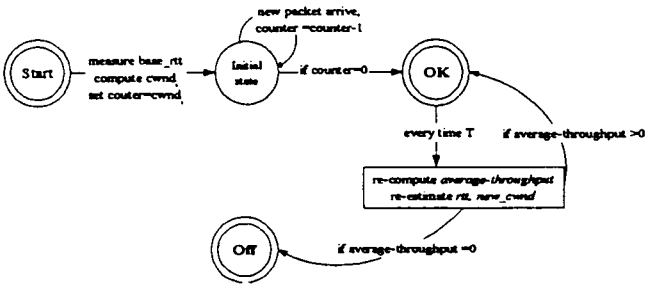

図 4：Non-Per-Flow Queueing