行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※　　　　寬頻網際網路之允入控制與訊務排程　　　　※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：□個別型計畫　　■整合型計畫

計畫編號：NSC89-2219-E-009-001

執行期間：88 年 8 月 1 日至 89 年 7 月 31 日

計畫主持人：楊啟瑞

共同主持人：

計畫參與人員：

本成果報告包括以下應繳交之附件：

□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

執行單位：國立交通大學資訊工程研究所

中　華　民　國　89 年　10 月　25 日

行政院國家科學委員會專題研究計畫成果報告
寬頻網際網路之允入控制與訊務排程
Admission Control and Traffic Scheduling in Broadband Internet

計畫編號：NSC89-2219-E-009-001
執行期限：88 **年** 8 **月** 1 **日至** 89 **年** 7 **月** 31 **日**

主持人：**楊啟瑞　　國立交通大學資訊工程研究所**

1. 中文摘要

寬頻網際網路上的QoS (Quality-of-Service) **排程必須提供有限延遲與公平性的考量且保持最小之計算複雜度。**Prevailing weight-based **的排程原則提倡應用**multiple queues**和**engage **於**timestamp**的計算。這些原則在高複雜度的花費下，可以達到最佳之**QoS**效能，也可以為了計算簡化，達到次級之效能。在今年度的計畫中，我們設計了一個**weight-based Versatile QoS Scheduler (VQS) **和其可行之**VLSI**硬體實作架構。為了促進較適當之效能與複雜度間的平衡，**VQS**可以在寬頻網際網路內用不同之網路元件來實作。**VQS**利用較簡單之**single-queue**與不需**timestamp**計算之優點，以**weight**來控制封包插入於一分享式之資料結構。此資料結構包含了一系列固定大小之**windows。在一給定的**window**內，由一個**session**來的最大封包個數，是和此**session**的**weight**和**window**大小成正比。實驗結果證明，應用較小之**window**於**high-power **的網路元件，**VQS **可以在** throughput fairness、mean delay、**與** worst-case delay fairness**等項目上表現和**WF^2Q**一樣好。甚至在和**WF^2Q**相容與具備**traffic burstiness**的情況下，**VQS**較**WFQ**優越約**99%**之**delay bound**和** jitter。

關鍵字: **服務品質排程、產出公平性、**99% **延遲限制。**

Abstract

Quality-of-Service (QoS) scheduling for broadband Internet is aimed to provide bounded delay and fairness while retaining a minimum of computational complexity. Prevailing weight-based scheduling disciplines advocate the use of multiple queues and engage in timestamp computation. These disciplines achieve either superior QoS performance at the expense of higher complexity or degraded performance in return for computational simplicity. In the project of this year, we have designed a weight-based Versatile QoS Scheduler (VQS) and its feasible VLSI hardware implementation architecture. VQS is capable of being implemented in various network elements in broadband Internet facilitating proper trade-off balance between performance and complexity. Taking advantage of simpler single-queue management and lack

of timestamp computation, VQS governs packet insertion in a shared data structure comprising a sequence of fixed-size *windows* based on weights. Within a given widow, the maximum number of packets from a session is proportional to the session weight and the Window Size (*WS*). Simulation results demonstrate that, applying a smaller *WS* for high-power network elements, VQS performs as superior as WF^2Q with respect to throughput fairness, mean delay, and worst-case delay fairness. Moreover, compatible to WF^2Q, VQS outperforms WFQ with respect to 99-percentile delay bound and jitter in the presence of traffic burstiness.

Keywords: Quality-of-Service (QoS) scheduling, throughput fairness, 99% delay bound, jitter.

2. Approaches

2.1. Background and Concept

Scheduling disciplines proposed in the literature have been either single-queue [1-3] or multiple-queue-based [4-7]. Single-queue-based disciplines advocate the maintenance of a single shared queue for each output link. Different-session packets destined to the same output link are inserted in the shared queue in accordance with, for instance, the deadlines or priorities of packets. Packets are then transmitted in a FIFO manner. Consequently, scheduling complexity completely resides in the enqueueing process. Examples of this class include Earliest Deadline First (EDF) [1], Threshold Based Priority (TBP) [2], and Precedence with Partial Push-Out (PPP) [3]. The EDF discipline was shown to successfully support tight delay bound. However, it undergoes two major limitations- a priori deadline assumption and high implementation complexity due to packet sorting. Although TBP and PPP were justified effective for switches supporting two priorities, they fail to provide bounded delay and throughput fairness in the presence of malicious sessions.

Multiple-queue-based disciplines, on the other hand, adopt multiple queues maintained at each output link, one for each session. Packets arriving from different sessions are simply placed at the end of their corresponding queues. Scheduling complexity in this class resides in the dequeueing process instead. Prevailing disciplines in this class, which are weight-based, include Weighted Fair Queueing (WFQ) [4], Worst-case Fair Weighted Fair Queueing (WF^2Q) [5], Self-Clocked Fair Queueing (SCFQ) [6], and Frame-based Fair Queueing (FFQ) [7].

In this project, we aim to design a weight-based, highly versatile QoS scheduler, referred to as VQS, capable of being implemented in diverse network elements facilitating proper trade-off balance between performance and complexity. Taking advantage of simpler single-queue management and lack of timestamp computation, VQS governs the insertion of packets belonging to the same

output link in a shared data structure comprising a sequence of fixed-size *windows*. Within a given widow, the maximum number of packets from a session is proportional to the session weight and the Window Size (*WS*). Packets being placed at the same window are transmitted on a FIFO basis, limiting short-term unfairness to within a window interval. Packets being arranged outside of the window trigger new windows to be activated, enforcing weight-proportional service to be exerted.

2.2. The VQS System

VQS is assumed non-cut-through and non-preemptive. In other words, a packet is not served until its last bit has arrived, and once a packet is being served, no interruption is permitted until the whole packet is completely served. It is also work conserving in the sense that the server remains busy as long as there are packets in the queue. Packets are served under a normalized service rate of one cell/slot. Given a backlogged session, i, assigned with weight w_i, VQS allocates the session a minimum service rate of w_i / W (cells/slot), where $W = \sum_{i=1}^{N} w_i$ and N is the total number of sessions in the system. For ease of description, we assume the packet size is fixed (=L cells). The VQS algorithm, as will be shown, requires little modification for supporting variable-size packets.

For generalization, we consider two different VQS systems- a standalone VQS and an embedded VQS. While the former directly accepts input traffic from each session, the latter exerts a leaky-bucket regulator [8] between each session's input traffic and VQS. First, the input traffic from each session is generally modeled by a discrete-time Switched Bernoulli Process (SBP) [9]. The process alternates between the High and the Low states. Second, the leaky-bucket regulator for session i is defined by (ρ_i, σ_i), where ρ_i (cells/slot) is the token generation rate and σ_i (cells) is the maximum token bucket size. Thus, under the embedded system, traffic from session i exhibits a mean arrival rate of ρ_i / L

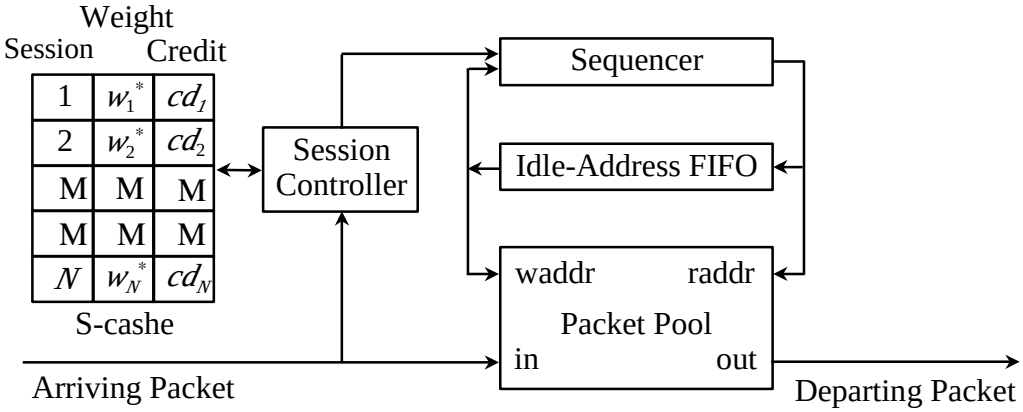

Figure 1. VQS implementation architecture.

(packets/slot) and burstiness which increases with $\pm_i$.

2.3. Implementation Architecture

The architecture (see Figure 1) includes a VLSI chip, called the Sequencer [10], as a key component. The Sequencer is essentially a sorting-memory chip. By cascading multiple Sequencer chips in series or in parallel, a large linked list type packet pool could be implemented. As depicted in Figure 2, when a packet arrives, the packet is stored in the packet pool at the address provided by the Idle-Address FIFO, which contains the addresses of unused space in the packet pool. Before the packet is written into the packet pool, its session identifier is extracted and used as an index into the Session (S)-cache. The S-cache maintains a separate entry for each session, including the normalized weight and credit. Notice that since we assume *WS*=1 in this architecture, the sum of normalized weights of all sessions is equal to 1. The Session Controller is responsible for determining the index of the window in which this arriving packet can be placed, based on the normalized weight of the session to which the packet belongs.

3. Results and Merit Review of the Project

The performance of VQS is evaluated via simulation. Simulation results demonstrate that, applying a smaller *WS* for network elements

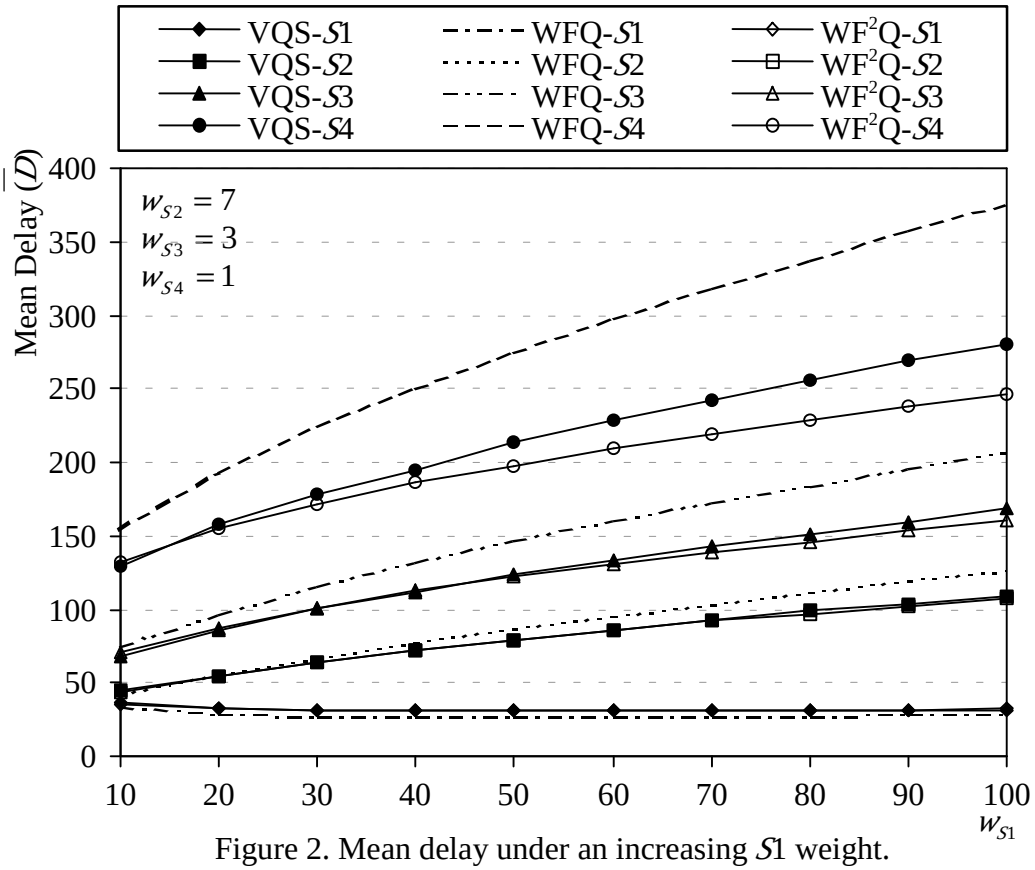

Figure 2. Mean delay under an increasing *S*1 weight.

with sufficient computation power, VQS performs as superior as the optimal scheme, WF^2Q, with respect to mean delay, throughput fairness, and worst-case delay fairness (see Figure 2). Moreover, compatible to WF^2Q, VQS outperforms WFQ with respect to 99-percentile delay bound and jitter in the presence of traffic burstiness. For network elements with limited power, VQS provides the best possible QoS using a larger window size.

The design and experimental results have been presented and demonstrated in various conferences and meetings, including IEEE ICC'00. Moreover, we have designed several networking control systems making use of the mechanism, which has been submitted to IEEE ICC 2001.

4. References

[1] J. Liebeherr, D. Wrege, and D. Ferrari, "Exact Admission Control for Networks with a Bounded Delay Service," *IEEE/ACM Trans. on Networking*, vol. 4, no. 6, Dec. 1996, pp. 885-901.

[2] D. Lee, and B. Sengupta, "Queueing Analysis of a Threshold Based Priority Scheme For ATM Networks," *IEEE/ACM Trans. on Networking*, vol. 1, no. 6, Dec. 1993, pp. 709-717.

[3] J. Hah, and M. Yuang, "A Delay and Loss Versatile Scheduling Discipline in ATM Switches," *Proc. IEEE INFOCOM*, 1998, pp. 939-946.

[4] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Proc. SIGCOMM*, 1989.

[5] J. Bennett, and H. Zhang, "WF^2Q: Worst-case Fair Weighted Fair Queueing," *IEEE INFOCOM*, 1996, pp. 120-128.

[6] S. Golestani, "A Self-Clocked Fair Queueing Scheme for Broadband Applications," *IEEE INFOCOM*, 1994, pp. 636-646.

[7] D. Stiliadis, and A. Varma, "Efficient Fair Queueing Algorithms for Packet-Switched Networks," *IEEE JSAC*, vol. 6, no. 2, April 1998, pp. 175-185.

[8] R. Cruz, "A Calculus for Network Delay, Part I: Network Elements in Isolation," *IEEE Trans. on Information Theory* , vol. 37, no. 1, Jan. 1991, pp. 114-131. .

[9] H. Saito, *Teletraffic Technologies in ATM Networks*, Artech House, 1994.

[10] H. Chao, and N. Uzun, "An ATM Queue Manager Handling Multiple Delay and Loss Priorities," *IEEE/ACM Trans. on Networking*, vol. 13, no. 6, Dec. 1995, pp. 652-659.