Implementing Distributed Weighted Fair Queuing Schedule in CICQ Switches

Wang Rong, Chen Yue, Wu Jiangxing

NDSC, PLA Information Engineering University, Zhengzhou, Henan province, 450002, China

***Abstract* — Traditional input-queued switches based on crossbar are insufficient in providing good QOS performance. As a contrast, the CICQ(combined input and crosspoint queuing) switches can provide almost 100% throughput under different uniform and nonuniform input traffic, the performance of which is very closed to OQ(output-queued) switch, and has the potentials to support good QOS. Based on the CICQ switches, we put forward a new scheme which can realize distributed weighted fair queuing schedule for the packets of variable length, and have both the scalability of input-queued switches and QOS performance of output-queued switches. We also discussed and solved the problem of updating the virtual time function of back-pressured queues. Simulation results show the scheme is very effective and have good performance.**[1]

***Key words* — CICQ switch, Distributed WFQ(weighted fair queuing) Schedule, Input-queued Switch.**

I. INTRODUCTION

The internet has become a fundamental infrastructure of the world, the bone line rate has reached 10Gbps. With the increase of bandwidth, the key problem of IP networks—QOS is not solved very well yet. If we hope the video, audio and data will be carried in the IP networks, we must resolve the problem of QOS.

According to IETF's Integrated service, guaranteed service can ensure low loss rate and strict upper bound of end-to-end delay for a single flow. To realize the integrated service, packet weighted fair queuing scheduler must be realized in routers. Packet weighted fair scheduling algorithms include WFQ[1], WF^{2}Q, SCFQ[2], etc., all of which are realized in output-queued switches. Although output-queued switch can support good QOS, it is not scalable and hard to be used in high-speed networks. On the other hand, it is very hard for the input-queued switch to provide good QOS performance.

In this paper we discuss CICQ(combined input and crosspoint queuing) switch[5]. The CICQ switch can not only realize 100% throughput under i.i.d Bernoulli uniform traffic, but can also realize close 100% throughput under various bursty and non_uniform traffic[6], the performance of which is very close to output-queued switch. Base on the CICQ switch, we put forward a new scheme which realizes distributed weighted fair queuing schedule for the packets of variable length in CICQ switch. The scheme takes advantage of both the scalability of input-queued switch and QOS support of output-queued switch. The rest of the paper is organized as follows. Section IIdescribes the structure and implementing of CICQ switch. The scheme of distributed weighted fair queuing schedule is presented in Section III. Section IV discussed the update of virtual time function of back-pressured queues. The simulation results are shown in Section V. Finally Section VI concludes the paper.

II. THE STRUCTURE AND IMPLEMENTING OF CICQ SWITCH

In traditional crossbar, the buffers are located in input or output ports, there is no buffer in switch fabric. In CICQ switch(as in Fig.1), in addition to buffers in input port, there are small buffers in every crosspoint of crossbar, called crosspoint buffer. For a N×N switch, the number of crosspoint and crosspoint buffer is N^2. We also call this kind of crossbar buffered crossbar[6]. In the input port of CICQ switch the VOQ(virtual output port queue) structure is used to eliminate the HOL blocking.

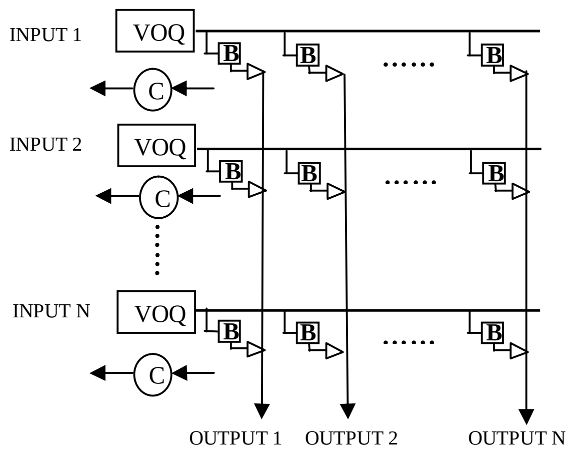

Fig. 1 The structure of CICQ switch

CICQ switch has many advantages over traditional

[1] The author is Dr. candidate in The PLA Information Engineering University. Address: 57th team, Yard 2 in Huaishuling, Changxindian, Fengtai District, Beijing. PC:100072. email: wang6802@sohu.com.

crossbar switch:

1) The crossbar must switch the cells in all input ports synchronously. Because different input ports belong to different clock domain, the signals from different clock domain need to be synchronized. In buffered crossbar, due to the crosspoint buffers in every crosspoint, the packets in every input port are switched independently with crosspoint buffers.

2) The traditional crossbar needs a matching algorithm to calculate the maximum matching between input and output ports. As the line rate increases, complicated algorithm achieving good performance is hard to realize. In buffered crossbar switch, every input and output port operate independently, it is unnecessary to realize the matching algorithm, it becomes possible that the switch operate at higher speed and realize sophisticated QOS.

3) In traditional crossbar, the variable length packet must be segmented into fixed-size cells before entering the crossbar. A part of bandwidth is wasted by the segmentation. In buffered crossbar the variable length packet is switched directly without segmenting, the bandwidth of switch fabric is fully used.

4) Traditional input-queued crossbar can not provide good QOS, the weighted fair queuing schedule algorithm is hard to realize in input-queued switch. In contrast, buffered crossbar has potential to support good QOS. In this paper we put forward a scheme which realize distributed weighted fair queuing schedule in CICQ switch.

If we put all the packets in crosspoint buffers, because the number of the crosspoints is N^2, the crossbar will become too large, it is hard to realize and not scalable. Practically we can consider to put a small number of buffers in crosspoints, and put the main buffers in input ports. Appropriate flow control mechanism is used to prevent the overflow of the crosspoint buffer(Fig.1). When a crosspoint buffer is full, it will give a back-pressure signal to indicate the corresponding input queue to stop the transmission.

III. IMPLEMENTING THE DISTRIBUTED WEIGHTED FAIR QUEUING SCHEDULE IN CICQ SWITCH

A. *Weighted Fair Queuing Scheduler*

GPS(Generalized processor sharing) discipline is an idealized server. It assume that the server can serve multiple sessions simultaneously and the traffic is infinitely divisible. It is characterized by N positive numbers $\Phi_1, \Phi_2, \ldots, \Phi_N$. The server operates at a fixed rate r and is work-conserving. Let $W_i(t_1,t_2)$ denote the amount of service session i served in the interval $[t_1,t_2]$, then a GPS server is defined as one for which

$$\frac{W_i(t_1,t_2)}{W_j(t_1,t_2)} \geq \frac{j_i}{j_j} \quad j=1,2,\ldots,N \qquad (1)$$

holds for any session i that is backlogged during the interval $[t_1,t_2]$. It follows that the service rate of session i is

$$r_i(t_1,t_2) = \frac{j_i}{\sum_j j_j} r \qquad (2)$$

In a packet system, only one session could be served at the same time, and the service of the whole packet must be completed before the next one can be served. There are different ways of emulating GPS service in a packet system. According to the packet selection policy, the GPS-related schedulers can be categorized into 3 classes, SSF(Smallest virtual Starting time First), SFF(Smallest virtual Finish time First), and SEFF(Smallest Eligible virtual Finish time First). The SEFF policy has the best performance, where the server selects the packet with minimum virtual finishing time among all eligible packets. A packet is eligible for transmission if the system virtual time is greater than its virtual starting time. The GPS-related schedulers include SCFQ, WF^2Q, WF^2Q+[4] etc..

B. *The Structure of Distributed Weighted Fair Scheduler*

In OQ switch the contention point exists in output port, the scheduler only need to be realized in every output port. In CICQ switch the packets are buffered in input port, and there are also packets in every crosspoint buffer. To emulate the OQ structure the packets must be scheduled in multiple contention points[5]: in VOQs of every input port, in input ports and output ports. The structure is shown in Figure 3.

In every VOQ queue, $V_{ij,}$ there is a scheduler S_{ij}, which serves the flows in V_{ij}. In every input port there is a scheduler S_i^{in}, which takes every VOQ as a single flow, and serves all the VOQs in the input port. In every output port there is also a scheduler, S_j^{out}, which takes charge of scheduling all the crosspoint buffers which output to the port. The fully distributed schedule means that there are no complicated communication and coordination between VOQ schedulers, input port schedulers and output port schedulers. To realize it we adopt the following policy: the scheduler S_{ij} in VOQ_{ij} only serves all the flows in this VOQ; the scheduler S_i^{in} in input port i takes N VOQs in the port as N sessions, and schedules the N VOQs. The weight of each VOQ equals to the sum of weights of reserved flows in this VOQ. If the bandwidth of the VOQ as a whole is guaranteed, the bandwidth of every flow in the VOQ can be also guaranteed. In output port j the scheduler S_j^{out} takes as N sessions the N crosspoint buffers which output to this port, and serves the N crosspoint buffers. The weight of each crosspoint buffer equals to the sum of weights of reserved flows in the VOQ that corresponds to the buffer. Similarly the bandwidth of the flows in crosspoint buffer

Top

is guaranteed on the condition that the bandwidth of the crosspoint buffer is guaranteed. The schedulers above can be WFQ, SCFQ or WF^{2}Q+ etc., we assume the scheduler used in the scheme is WF^{2}Q+.

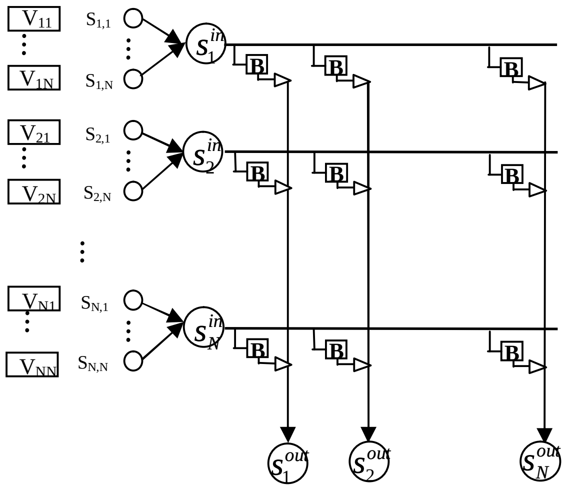

Fig. 3 Distributed weighted fair queuing schedule in CICQ

The distributed structure has obvious advantages. The flow-based schedule involves large amount of computation, the computation-consuming work can be distributed to line cards, because the VOQ schedulers could be realized in line cards. In the switch fabric, the scheduled sessions are the VOQ queues and crosspoint buffers which have fixed number and the number of which is much less than the flows. The above merit makes it easy to realize the buffered crossbar and its schedulers on a single chip.

IV. THE UPDATE OF THE VIRTUAL TIME OF BACK-PRESSURED QUEUES

In CICQ switch, crosspoint buffer could not be very large, great amount of packets are buffered in input port queues. Crosspoint buffers and input queues are coordinated through back-pressure signals. When a crosspoint buffer is full, it gives a back-pressure signal, the VOQ received the signal will not be scheduled until the signal is cancelled. When the queue are temporally not scheduled, the virtual time function of the queue will remain unchanged; when back-pressure signal is cancelled, how to update the virtual time function of the queue is a question. During the back-pressured period, the virtual time of other queues will increase. After the back-pressure is cancelled, the virtual time of the back-pressured queue will be smaller than others, as a result the back-pressured queue is scheduled preferably. We give an example to analyze the problem.

Assume that there are three queues in one input port, the weights of queue 0, queue 1 and queue 2 are r_0, r_1, r_2 respectively. Queue 0 is destined to output port 0. We define V(t) the system virtual time function of the scheduler, $S_i(t)$ the virtual start time function of queue i, $F_i(t)$ the virtual finish time function of queue i. Let the packet length of each queue be L. Assume the queue 0 is back-pressured at time t_1, according to the policy of WF^{2}Q+, we have

$$S_0(t_1) < V(t_1) \qquad (3)$$

Assume that during time $[t_1,t_2]$ there are packets waiting in queue 0 and queue 1, and the queue 2 is empty. Because the queue 1 is not back-pressured, it is served during time $[t_1,t_2]$. And suppose at time t_2, the back-pressure of queue 0 is cancelled, and during time $[t_1,t_2]$ only one packet is serviced in queue 1, and the service is just completed at time t_2. Then the change of system virtual time during time $[t_1,t_2]$ is

$$V(t_2) - V(t_1) = L/r_1 \qquad (4)$$

Assume at time t_2, queue 1 becomes empty, and there are still packets in queue 0, and a new packet arrive in queue 2, we have

$$S_2(t_2) \geq V(t_2) \qquad (5)$$

from (4), (5), we have

$$S_2(t_2) \geq V(t_2) \Rightarrow F_2(t_2) \geq V(t_2) + L/r_2 \Rightarrow$$

$$F_2(t_2) \geq V(t_1) + L/r_1 + L/r_2 \qquad (6)$$

because(3), we have,

$$F_2(t_2) \geq S_0(t_1) + L/r_1 + L/r_2 \geq F_0(t_1) + L/r_1 + L/r_2 - L/r_0 \qquad (7)$$

During time $[t_1,t_2]$ the queue 0 is back-pressured, the virtual time function of queue 0 is unchanged, i.e. $F_0(t_1) = F_0(t_2)$, so

$$F_2(t_2) - F_0(t_2) \geq L(1/r_1 + 1/r_2 - 1/r_0) \qquad (8)$$

which indicates the delay of queue 2 is related to r_1. When the packet arrives in queue 2 at time t_2, only queue 0 and queue 2 are not empty and served, so the delay of queue 2 should be related to r_0 and r_2, not r_1. So after the back pressure is cancelled, the virtual time function of back-pressured queue remains unchanged, which increases the delay of other queues.

To solve the problem above, we update the virtual time function as follows. When the back pressure is cancelled, the virtual finish time of back-pressured queue is compared with the system virtual time. If the virtual finish time is less than the system virtual time, we set the queue's virtual start time to the system virtual time, and change the queue's virtual finish time accordingly; if the virtual finish time of back-pressured queue is equal to or greater than the system virtual time, the queue's virtual finish time remain unchanged. So we have

$$F_0(t_2) > V(t_2) \qquad (9)$$

from $F_2(t_2) > V(t_2) + L / r_2$, then

$$F_2(t_2) - F_0(t_2) > L / r_2 \qquad (10)$$

from which we can see the delay of queue 2 is not related to r_1.

V. THE SIMULATION SCENARIO AND SIMULATION RESULT

We simulate a 4×4 switch with port rate of 1Gbps. The input flows are buffered in VOQs. Let f(i,j) denote the flow from port i to port j.

Simulation scenario 1:

Assume that the reserved bandwidths of flows f(1,1), f(2,1), f(3,1), f(4,1) are 40%, 30%, 20% and 10% respectively., and arrival rates of 4 flows are identical. The arrival rate of the rest flows is 5%. We change the load of four flows to view the bandwidth assignment. As shown in Fig. 4, when the load of input flow is under 25%, the assigned bandwidth of each flow is able to keep up with the input load and is identical. For load beyond 40%, every flow receives its reserved bandwidth. Let's look at the situation when input load is between 25% and 40%. For instance, at 30%, flow f (1, 1) shares 30% of the bandwidth with 10% unused. Flows f (3, 1) receives 26.8 %(versus a reservation of 20%) with 6.8% over-allocated bandwidth, and flow f (4, 1) (versus a reservation of 10%) receives 13.4% with 3.4% over-allocated bandwidth. The assignment of the rest bandwidth conforms to the weighted fair requirement.

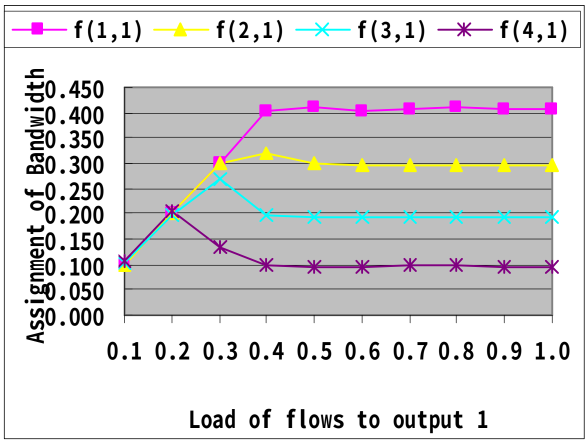

Fig. 4. The bandwidth assignment of flows to output 1

Simulation scenario 2:

Assume that the reserved bandwidths of flows f(1,1), f(1,2), f(1,3), f(1,4) are 40%, 30%, 20% and 10% respectively, the reserved bandwidths of flows f(2,1), f(3,1) and f(4,1) are the same as above. Except that the arrival rate of f(1,1) is 300 Mbps, the arrival rates of the other flows are 400Mbps. As shown in table 1, among the contending flows in input port 1, the unused bandwidth of flow f(1,1) is distributed fairly according to the reservation of each flow.

Table 1: over-allocated bandwidth of each flow

Traffic flow	Reserved bandwidth (Mbps)	throughput(Mbps)	Over-allocated bandwidth(Mbps)
f(1,2)	300	350.26	50.26
f(1,3)	200	233.10	33.10
f(1,4)	100	116.62	16.62
f(2,1)	300	350.28	50.28
f(3,1)	200	233.12	33.12
f(4,1)	100	116.57	16.57

VI. CONCLUSION

The traditional input-queued crossbar is extensively used in various routers, but it is inefficient in supporting good QOS. Based on the traditional crossbar, we discussed a new switch fabric—CICQ switch in detail, which can provide high throughput under various uniform and nonuniform traffic. The throughput of CICQ is closed to OQ (output queued switch), and can support good QOS. In this paper, we put forward a new scheme which realize distributed weighted fair queuing schedule in packet systems. The simulation result shows that the scheme is effective. We also discuss the issue of updating the virtual time function of back-pressured queue, and give the solution.

REFERENCE

[1] A. Parekh and R. Gallager, A generalized processor sharing approach to flow control-The single node case[**A**]. *Proc. IEEE InfoCom*[**C**], 1992. pp. 915-924.

[2] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks[**J**]. *IEEE Commun. Mag.*, vol. 83, pp. 1374-96, Oct. 1995.

[3] Bennett J,ZhangHui. Hierarchical packet fair queueing algorithms. In: Proc ACM SIGCOMM' 96, Stanford,CA,1 996.1 43 – 156.

[4] D. Stephens and H. Zhang, "Implementing Distributed Packet Fair Queueing in a Scalable Switch Architecture," in Proc. INFOCOM, 1998.

[5] M. Katevenis, G. Passas, D. Simos, I. Papaefstathiou, N. Chrysos: "Variable Packet Size Buffered Crossbar (CICQ) Switches", Proc. IEEE International Conference on Communications (ICC 2004), Paris, France.