ON THE PROVISION OF INTEGRATED QOS GUARANTEES OF UNICAST AND MULTICAST TRAFFIC IN INPUT-QUEUED SWITCHES *

Ge Nong and Mounir Hamdi
Department of Computer Science
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
Email: nong@ieee.org and hamdi@cs.ust.hk

Abstract

In this paper, we address the problem of providing QoS guarantees for multiple input-queued switching architecture. Stable matching of inputs and outputs is used for scheduling the enqueued packets to be transmitted across the switching fabric. With an in-depth theoretic analysis on the properties of stable matching in the context of a multiple input-queued switch, we propose efficient schemes to guarantee the QoS of both unicast and *multicast* traffic with fixed-length or *variable-length* packets. Using these schemes, the QoS of both fixed-length and variable-length packets can be guaranteed by *independently* employing suitable service disciplines at the packets' destined outputs like what is being done in an output queueing switch.

1 Introduction

Some efforts have been recently reported on how to provide QoS guarantees on Virtual Output Queueing (VOQ) and Combined Input and Output Queueing (CIOQ) switches [2, 3, 5, 8] [1]. All of the scheduling algorithms developed in these proposals are based on the "stable matching" concept because of its potential in bounding a packet's maximum switching delay. While these efforts are a step in the right direction, they have at least a few shortcomings: (1) The methods used to provide QoS guarantees for these switches are developed in the context of unicast traffic only and will not be valid any more through the addition of multicast traffic; (2) They achieve QoS guarantees through "exact emulation" of OQ switches [3] which is a conservative methodology under most realistic traffic patterns; (3) The algorithms proposed to provide QoS for these switches are too complex to be implemented in real-time, especially at high speeds. In this paper, we expand upon these previous research efforts and we address some of their shortcomings.

The rest of this paper is organized as follows: the architectures of VOQ and CIOQ switches are described in section 2. In Section 3, a brief review on the existing scheduling algorithms based on stable matching of inputs and outputs to guarantee QoS in VOQ or CIOQ switches is given. In section 4, our schemes using stable matching for providing QoS guarantees to both unicast and multicast traffic in VOQ or CIOQ switches are presented. Finally, a conclusion is given.

2 Architectures of VOQ and CIOQ Switches

The switches under investigation are assumed to be non-blocking $N \times N$ switches. The switch operates at discrete time slots and the (unicast or multicast) packets arrive only at the beginning of time slots and depart only at the end of time slots. The inputs and outputs of the VOQ switch are connected via a nonblocking interconnection network such as a crossbar. Each input port maintains a separate queue for packets destined to each output port. Only the HOL packet of each queue can be selected for transmission across the switch in each time slot, with the constraints that each input/output can transmit/receive at most one HOL packet from any of its N queues, in each time slot. If one buffer per output port is also provided on the output side of a VOQ switch, we have a CIOQ switch. Hence, the results es-

*This work was supported in part by the Hong Kong Research Grant Council under the Grant RGC/HKUST 6026/97E.

[1] There is a problem with the algorithm in [8]. Please refer to the web site http://www.cs.cmu.edu/ stoica/IWQoS98-fix.html for more details, and how the algorithm is corrected.

0-7803-5796-5/99/$10.00 © 1999 IEEE

tablished for the VOQ switches are also applicable to the CIOQ switches unless specified otherwise. When output queueing is needed on the output side, the term CIOQ is used.

Along with an $N \times N$ VOQ switch, we define a referred $N \times N$ OQ switch. Each output in the referred OQ switch is denoted as a *referred OQ server*. The referred OQ switch of a VOQ switch is simulated in parallel with the VOQ switch in order to produce informations needed for guaranteeing the traffic's QoS in the VOQ switch, as will be shown later.

The following notations are used to simplify and clarify our presentation in this paper.

ω : the speed-up of the switching fabric normalized by an input/output link speed equals to 1.

P_{ij} : a packet destined for output j from input i.

$D(P_{ij})$: the delay of the packet P_{ij} in the referred OQ server. The delay of a packet includes both the waiting time and the service time.

$D_{VOQ}(P_{ij})$: the delay of the packet P_{ij} in the VOQ/CIOQ switch.

$D_{QoS}(P_{ij})$: the maximum delay of the packet P_{ij} in the referred OQ server for guaranteeing P_{ij}'s QoS, i.e., P_{ij}'s QoS can be guaranteed if $D(P_{ij}) \leq D_{QoS}(P_{ij})$.

$B_{in}(P_{ij})$: the maximum number of packets that can block P_{ij} and arrive at input i.

$B_{out}(P_{ij})$: the maximum number of packets that can block P_{ij} and are destined for output j.

3 Existing Scheduling Algorithms Based On Stable Matching

It was proved in [3] that an OQ switch scheduled by *monotonous* work-conserving service disciplines can be *exactly emulated* by a CIOQ switch with an internal speed-up of 2, for any switch size and any *unicast* traffic. The departure process of packets in a CIOQ switch exactly emulating an OQ switch is the same as that in the emulated OQ switch. A service principle is said to be *monotonous* if and only if any new arrival doesn't change the relative scheduling order of the packets already enqueued [8]. In this paper, we also assume that **the service disciplines are monotonous.**

Instead of using PIM-like algorithms [1] to find matching of inputs and outputs, the Gale-Shapley algorithm (GSA), which was first introduced by Gale and Shapley to solve the stable marriage problem, was employed by the CIOQ switch [3]. The GSA used in the CIOQ switch finds *stable* matching of inputs and outputs based on the defined input and output preference lists. Each input/output preference list ranks the outputs/inputs packets in order of preferences. A matching is said to be *unstable* if there is at least a pair of input and output which aren't matched, but which each prefer the other to their currently matched mates. A matching which is not unstable is called *stable*. The stable matching of inputs and outputs have an important property [3] which forms the basis of designing schemes for providing QoS guarantees in a VOQ switch: *A packet P_{ij} in a VOQ switch will be scheduled by a stable matching algorithm to be transmitted across the switching fabric* if and only if *no packet ahead of P_{ij} in the preference lists of input i and output j is switched.*

The results[2] established in [3, 8] were developed in the context of *unicast traffic only* and won't be valid any more in case multicast traffic is included. However, the ability to support multicast traffic is indispensable for high-speed packet switches designed for the future Internet. In the following section, we will address the problem of providing QoS guarantees for both unicast and multicast traffic in VOQ and CIOQ switches.

4 Guaranteeing Delay-bounded QoS in VOQ/CIOQ Switches Using Stable Matching

The required QoS is supposed to be delay-bounded, i.e., the QoS required by any packet P_{ij} can be guaranteed if $D(P_{ij}) \leq D_{QoS}(P_{ij})$. Herewith in this paper, an arriving multicast packet is *treated* as a batch arrival of unicast packets with different destined outputs. However, please note that we are not making any assumption about how the multicasting must be performed. In other words, our proposed method will work whether multicasting is being performed using complete input packet replication, partial input packet replication, or no input packet replication at all [4]. Moreover, without explicit specification, the scheduling of multicast packets is assumed to be fan-out splitting (also known as nonatomic multicasting) such that copies of the multicast packet can be scheduled independently (at different time slots) [4].

Once the input and the output preference lists have been constructed, the stable matching algorithm is performed to schedule the enqueued packets to be trans-

[2] A CIOQ switch with an internal speed-up of 2 can exactly emulate an OQ switch by using stable matching.

mitted across the switching fabric. Provided that the switching fabric is fast enough to transmit all packets to their destined outputs promptly, the specified QoS of packets are guaranteed. The question is: How fast should the switching fabric be in order to guarantee that all packets can be transmitted to their destined outputs promptly? Our answers for this question are illustrated using two schemes: one is based on the packets delays in the referred OQ switch and another is based on the packets maximum delays that can be tolerated for guaranteed QoS.

4.1 Schemes Based on $D(P_{ij})$

We first introduce the definition for output preference lists.

Definition 1 *An output preference list is defined according to the order that packets receive service in the referred OQ server.*

With this definition of output preference list, it is obvious that $B_{out}(P_{ij}) = D(P_{ij}) - 1$ when the service discipline employed by output j is monotonous. Consequently, we have the following theorem[3].

Theorem 1 *In a VOQ/CIOQ switch with the output preference lists as defined by Definition 1, if the service discipline employed by output j is* monotonous *and the speed-up $\omega \geq 1 + \frac{B_{in}(P_{ij})}{D(P_{ij})+X}$ for any packet P_{ij}, where $X \geq 0$ is a constant and $\forall i \in [1, N]$, then any packet P_{ij} can be transmitted across the switching fabric within a delay of $D(P_{ij}) + X$.*

In Theorem 1, the definition of $B_{in}(P_{ij})$ was not explicitly specified, which enables us to define the input preference lists in various ways. The various definitions of input preference list may lead to different values of $B_{in}(P_{ij})$ – which may result in different time complexities of finding stable matchings. For purpose of demonstration, we will introduce two examples in the sequel for different definitions of the input preference lists.

4.1.1 Example 1: LIFO Input Preference Lists

In [3], the problem of exactly emulating an OQ switch by a CIOQ switch was investigated, where the input preference lists are defined as follows:

[3]All proofs for the results present in this paper are omitted due to the space limit.

Definition 2 *The packets in an input preference list are sorted in the reverse order of packets' arrivals, i.e., LIFO.*

Let the input and the output preference lists be defined by Definition 2 and Definition 1, respectively. It is easy to see that under the environment of unicast traffic only, a speed-up of 2 is sufficient to guarantee that all packets are transmitted across the switching fabric when they should be served in the referred OQ switch (which was called "exactly emulating" or "behaving identically" in [3, 7]). This can be shown as follows.

With Definition 2, in order to exactly emulate an OQ switch by a CIOQ switch, we have $X = 0$ and $B_{in}(P_{ij}) = B_{out}(P_{ij}) = D(P_{ij}) - 1$, for any packet P_{ij}. From Theorem 1, we have $\omega \geq 2 - \frac{1}{D(P_{ij})}$ which produces $\omega \geq 2$ when $D(P_{ij}) \to \infty$. Hence, a speed-up of 2 is immediately obtained.

4.1.2 Example 2: Serving-Time-based Input Preference Lists

Let $L(P_{ij})$ be the number of packets before P_{ij} at output j's preference list. The input preference lists are defined as:

Definition 3 *An input preference list orders the packets by their preference values defined as $CPV(P_{ij}) = L(P_{ij})$ for any packet P_{ij}.*

With such input preference lists, we have the following lemma.

Lemma 1 *With the input and the output preference lists being defined by Definition 1 and Definition 3, a speed-up of $\omega = 3$ is sufficient for any packet P_{ij} in a VOQ/CIOQ switch under unicast traffic only to be transmitted across the switching fabric within a delay of $D(P_{ij})$.*

A greedy algorithm can be easily designed to find a stable matching based on $CPV(P_{ij})$ of each packet P_{ij}: a packet P_{ij} with the currently least $CPV(P_{ij})$ is selected until no conflict-free packet can be added. Such a greedy algorithm has a time complexity of $O(N^2)$ instead of $\Omega(N^2)$ for the GSA.

Next, using LIFO input preference lists as an example, we show how to lower the speed-up required for individual input/output buffers and efficiently support multicast traffic with guaranteed QoS.

4.1.3 Reducing The Required Internal Speed-up

With Definition 2 for the input preference lists, $B_{in}(P_{ij})$ is a linear function of $D(P_{ij})$ for any packet P_{ij}. Fig. 1 shows a sample architecture in which the required internal speed-up is 1.5. As we notice from the figure, two (independently) parallel switching fabrics are provided. The buffer at each input port is halved into two (equal size) sub-buffers and each sub-buffer can be accessed (read/written) independently. It is assumed that the maximum arrival rate at each input port is 1 cell per time slot. The demultiplexer at each input port dispatches the incoming cells to its two attached sub-buffers alternately so that the maximum arrival rate at each sub-buffer is 0.5 cell per time slot. It is not difficult to show that under the given conditions, $B_{in}(P_{ij}) = 0.5D(P_{ij}) - 1$ for any enqueued packet P_{ij}. Recalling the results stated in Theorem 1, the required speed-up for each sub-buffer/switching fabric is 1.5. As a result, the maximum throughput of the switch shown in Fig. 1 is $1/1.5 = 67\%$. Following the same direction, a switch with four parallel switching fabrics running at a speed-up of 1.25 will achieve a maximum throughput of $1/1.25 = 80\%$.

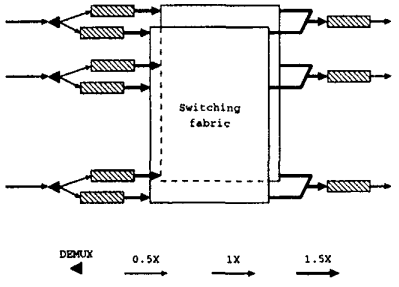

Figure 1: A sample architecture with the required internal speed-up of 1.5

4.1.4 Support of Multicast Traffic with Guaranteed QoS

As can be seen from our derivation of $B_{in}(P_{ij})$ and $B_{out}(P_{ij})$, the assumption of *unicast traffic only* is essential to the correctness that a speed-up of 2 suffices for a CIOQ switch to exactly emulate an OQ switch. With unicast traffic only, at most one (unicast) packet can arrive at any input at each time slot so that there are at most $D(P_{ij}) - 1$ packets enqueued in input i that can block P_{ij}. In case multicast traffic is included, an arriving multicast packet can be *viewed* as a burst of unicast packets simultaneously arriving at an input link, which can result in temporarily overloading that input link[4]. Consequently, it won't be possible to bound $B_{in}(P_{ij})$ for any packet P_{ij} without controlling the arriving traffic, i.e., traffic control is needed in order to carry the emulation with a small required speed-up. The following lemma gives the required speed-up for supporting multicast traffic under certain traffic shaping scheme.

Lemma 2 *For a VOQ/CIOQ switch with an internal speed-up $\omega \geq 2$ under multicast (and unicast) traffic, if*

1. *Inputs and outputs preference lists are defined by Definition 2 and Definition 1; and*
2. *There are at most $\frac{X}{\omega - 1}$ packets arriving at an input port within any time interval of X slots;*

then $D_{VOQ}(P_{ij}) \leq D(P_{ij}) + X - 1$.

So far, we have discussed the problem of bounding $D_{VOQ}(P_{ij})$ of a packet P_{ij} in the VOQ switch with reference to $D(P_{ij})$, its delay in the referred OQ switch. However, simulating the referred OQ switch required extra efforts. We proceed to consider the problem of bounding $D_{QoS}(P_{ij})$ of a packet P_{ij} without aid of the referred OQ switch.

4.2 Schemes Based on $D_{QoS}(P_{ij})$

In case the input and the output preference lists are ordered by a different criterion, the counting of P_{ij}'s blocking packets over the time interval $D_{QoS}(P_{ij})$ (since its arrival) is difficult. To deal with this problem, CPV input and output preference lists should be defined. We notice that there is a class of so-called *deadline-ordered* service disciplines, in which packets are assigned deadlines and transmitted in increasing order of the deadlines [6]. A packet's QoS can be guaranteed if it was transmitted by its assigned deadline. A number of known service disciplines are deadline-ordered [9], for example, Delay-EDD, PGPS, Rate-proportional servers, Service curve approaches, VituralClock and WFQ, etc... Therefore, the packets deadlines assigned by the referred OQ switch are a natural candidate for CPV. In view of this, we assume that the service disciplines are not only *monotonous* but also *deadline-ordered*. With such an assumption, the input and the output preference lists are defined as follows.

[4]Again, please note that we are not making any assumption about how the multicasting must be performed. In other words, our proposed method will work whether multicasting is being performed using complete input packet replication, partial input packet replication, or no input packet replication at all [4].

Definition 4 *The input and the output preference list orders the enqueued packets by their assigned* deadlines.

With input and output preference lists being defined by Definition 4, a greedy algorithm can be executed to find stable matchings of inputs and outputs.

- **Determining The QoS guarantees**

Let λ_{ij} denote all the packets from a traffic stream destined to output j from input i and queueing on the input side of the VOQ switch when a packet P_{ij} is queueing at its input buffer. Furthermore, let $\Lambda_{in}(i) = \cup_{j=1}^{N}\lambda_{ij}$/ $\Lambda_{out}(j) = \cup_{i=1}^{N}\lambda_{ij}$ denote the aggregate traffic offered to input i/output j when a packet P_{ij} is queueing at its input buffer. With the definition of input and output preference lists as given by Definition 4, we derive the theorem below for the determination of guaranteed QoS in the VOQ switches. For the purpose of notation simplicity, we first cast the following term.

Definition 5 *Deadline-ordered server: a server is deadline-ordered if and only if its service discipline is deadline-ordered.*

Let OS denote a work-conserving deadline-ordered OQ server with service capacity of ω packet(s) per time slot. In addition, let $MOUT(P_i)$ denote the destined outputs of a multicast packet which arrived at input i. We have the following two Theorems for supporting fan-out and non-fan-out splitting multicast traffic in a VOQ switch, respectively.

Theorem 2 *A VOQ switch can guarantee the delay-bounded QoS of a fan-out splitting packet* P_{ij} if *the following conditions are satisfied:*

1. *the service discipline of the referred OQ server* j *is monotonous and deadline-ordered;*
2. P_{ij}*'s delay is upper bounded by* $D_{QoS}(P_{ij})$ *in* OS *with the offered traffic* $\Lambda_{in}(i) \cup \Lambda_{out}(j)$.

Theorem 3 *A VOQ switch can guarantee the delay-bounded QoS of a non-fan-out splitting* multicast *packet* P_i if *the following conditions are satisfied:*

1. *The service discipline of all outputs in* $MOUT(P_i)$ *are monotonous and deadline-ordered;*
2. P_i*'s delay is upper bounded by* $D_{QoS}(P_i)$ *in* OS *with the offered traffic* $\Lambda_{in}(i) \cup (\cup_{j \in MOUT(P_i)}\Lambda_{out}(j))$.

5 Conclusion

A wide class of QoS guaranteed service disciplines originated in the context of OQ switches has been identified to be capable of providing QoS guarantees, for both unicast and multicast traffic, using VOQ or CIOQ switches. Similar to an OQ switch, QoS of packets in a VOQ or CIOQ switch can be guaranteed by *independently* employing suitable service disciplines for the packets' destined outputs, provided that the conditions given in this paper are satisfied. The ability of *isolating* the schedulings of service disciplines employed by different outputs render the present method general.

References

[1] T. E. Anderson, S. S. Owicki, J. B. Saxe, and A. P. Thacker. High-speed switch scheduling for local-area networks. *ACM Transactions on Computer Systems*, 11(4):319–352, Nov 1993.

[2] A. Charny, P. Krishna, N. Patel, and R. Simcoe. Algorithms for providing bandwidth and delay guarantees in input-buffered crossbars with speed up. In *IWQOS '98*, 1998.

[3] S. T. Chuang, A. Goel, N. McKeown, and B. Prabhakar. Matching output queueing with a combined input output queued switch. Technical report, Stanford CSL-TR-98-758, Mar. 1998.

[4] M. H. Guo and R. S. Chang. Multicast ATM switches: survey and performance evaluation. *Computer Communication Review*, 28(2):98–131, Apr. 1998.

[5] P. Krishna, N. Patel, A. Charny, and R. Simcoe. On the speedup required for work-conserving crossbar switches. In *IWQOS '98*, 1998.

[6] R. F. Norival and P. Joseph. A schedulability condition for deadline-ordered service disciplines. *IEEE/ACM Transactions on Networking*, 5(2):232–44, Apr. 1997.

[7] B. Prabhakar and N. McKeown. On the speedup required for combined input and output queued switching. Technical report, Stanford CSL-TR-97-738, Nov. 1997.

[8] I. Stoica and H. Zhang. Exact emulation of an output queueing switch by a combined input and output queueing switch. In *IWQoS'98*, 1998.

[9] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proceedings of the IEEE*, 83(10):1374–96, Oct. 1995.