社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
COMP2007-26(2007-06)

2ポート共有メモリ型スイッチにおけるオンラインバッファ管理アルゴリズムの厳密な競合比解析

小林 浩二† 宮崎 修一†† 岡部 寿男††

† 京都大学 情報学研究科 〒606–8501 京都府京都市左京区吉田本町
†† 京都大学 学術情報メディアセンター 〒606–8501 京都市左京区吉田本町
E-mail: †kobaya@net.ist.i.kyoto-u.ac.jp, ††{shuichi,okabe}@media.kyoto-u.ac.jp

あらまし オンラインバッファ管理問題は, 近年のネットワーク運用における主要な論点となっている QoS (Quality of Service) 保証実現のための, スイッチなどのキュー管理をオンライン問題として定式化した問題であり, 様々なモデルが考案されている. 本論文ではその中の 1 つである共有メモリ型スイッチを扱ったモデルを取り上げる. 我々は, スイッチの出力ポートの数 N が 2 であるときに、アルゴリズム Longest Queue Policy (LQD) の競合比が $\frac{4M-4}{3M-2}$ であることを示した。ここで, M はスイッチのバッファのサイズである. これは、Hahne らによって示された下限に一致する. また、任意の N の場合に、LQD の競合比を 2 から $2 - \frac{1}{M}\min_{K=1,2,\ldots,N}\{\lfloor\frac{M}{K}\rfloor + K - 1\}$ に改良した.

キーワード 競合比解析, 共有メモリ型スイッチ, バッファ管理

A Tight Upper Bound on Online Buffer Management for Two-port Shared-Memory Switches

Koji KOBAYASHI†, Shuichi MIYAZAKI††, and Yasuo OKABE††

† Graduate School of Informatics, Kyoto University, Yoshida-honmachi, Sakyo-ku, Kyoto, 606–8501 Japan
†† Academic Center for Computing and Media Studies, Kyoto University, Yoshida-Hommachi, Sakyo-ku Kyoto 606–8501, Japan
E-mail: †kobaya@net.ist.i.kyoto-u.ac.jp, ††{shuichi,okabe}@media.kyoto-u.ac.jp

Abstract The online buffer management problem formulates the problem of queueing policies of network switches supporting QoS (Quality of Service) guarantee. For this problem, several models are considered. In this paper, we focus on shared memory switches with preemption. We prove that the competitive ratio of the Longest Queue Drop (LQD) policy is $\frac{4M-4}{3M-2}$ in the case of $N = 2$, where N is the number of output ports in a switch and M is the size of the buffer. This matches the lower bound given by Hahne, Kesselman and Mansour. Also, in the case of arbitrary N, we improve the competitive ratio of LQD from 2 to $2 - \frac{1}{M}\min_{K=1,2,\ldots,N}\{\lfloor\frac{M}{K}\rfloor + K - 1\}$.

Key words Competitive analysis; Shared memory switches; Buffer management

1. Introduction

When we consider the performance of the Internet traffic, one of the crucial problems is a buffer management for routers or switches. The task of a switch is to receive a packet, find its destination, and transmit it from an appropriate output port. However, when the arrival rate of packets exceeds the transmission capacity of a switch, some packets will be lost. To ease this situation, we introduce buffers; when burst of packets happens, we temporary store those packets to buffers, and process them when available. One of the key strategies in managing buffers is to decide an acceptance of packets. For example, we are to decide whether to accept the current packet, or to reject it for more important ones that may arrive in the future.

Recently, this kind of problem is modeled as online problems, and a great amount of work has been done. There have been proposed many kind of models, and the most basic model is the following [1]: A switch has a buffer (FIFO queue) of bounded size B. An input is a sequence of events.

Each event is an arrival event or a send event. At an arrival event, one packet arrives at an input port. Each packet has a value and a size. A switch can store packets provided that the total size of stored packets does not exceed B. At an arrival event, if there is a room for the new packet, an online policy determines, without knowing the future, to accept it or not. At each send event, the first packet of the queue is transmitted. (If there are a lot of arrival events between two send events, we can interpret it as a burst.) The goal of the problem is to maximize the sum of the values of transmitted packets. A goodness of an online policy is evaluated by the competitive analysis [7], [18]. If, for any input σ, an online policy A obtains value at least $1/c$ of the optimal offline policy for σ, then we say that A is c-competitive.

Up to the present, several models have been considered. Among them, Hahne et al. have introduced a shared memory switch model [11]. In this model, a switch has one input port and N output ports, and each packet has a destination port. Each output port has a buffer (FIFO queue), where the length of each queue is variable but the total size of N output queues is M. When a packet arrives, an online policy determines to accept it (if the buffer has room for the new packet), reject it, or preempt (namely, drop packets already in the buffer to make space) and accept the new packet. At each time unit, the switch sends first packet of each non-empty queue. Hence it can send up to N packets at the same time. When the value of any packet is 1, and the size of any packet is 1, they showed that the Longest Queue Drop (LQD) policy is 2-competitive [11]. They also showed that no online algorithm can have competitive ratio better than $\frac{4M-4}{3M-2}$.

Our Results. In this paper, we improve the upper bound on the competitive ratio of LQD from 2 to $2-\frac{1}{M}\min_{K=1,2,\ldots,N}\{\lfloor\frac{M}{K}\rfloor+K-1\}$; $2-\frac{1}{M}(\lfloor\frac{M}{N}\rfloor+N)$ if $N<\sqrt{M}$, and $2-\frac{1}{M}(2\sqrt{M}-1)$, otherwise. Moreover, when $N=2$, we give a matching upper bound $\frac{4M-4}{3M-2}$. Usually, in the computational complexity theory, performance of algorithms is evaluated by its asymptotic behavior, e.g., when N goes infinity. However, in our current case, it is natural to assume that the number of output ports N is bounded, and hence, it is significant to improve the competitive ratio in the case that N is constant.

Let us briefly explain an idea of improvement. In [11], the authors introduced an *extra packet*, which is defined as a packet sent from the ith queue of optimal offline policy (OPT, for short) when LQD does not send a packet from the ith queue at the same time slot. They showed that each extra packet can be 1-to-1 matched (say, F-matching) with a packet sent by LQD. Hence, the number of packets sent by OPT is at most twice that of LQD. To guarantee this matching, they showed the following. At time t, a packet p located at the ℓth position in OPT's ith queue is called a *charged packet* if at the same time slot t, LQD's ith queue holds less than ℓ packets. They proved that at any time, every charged packet can be 1-to-1 matched (say, T-matching) with a packet in LQD's queue. Since an extra packet is a charged packet when it is sent, it is easy to see that if we can construct a T-matching at any time, then we can construct an F-matching.

To give a better upper bound, we introduce three new ideas. First, we prove that in a T-matching, each charged packet can be 1-to-$X(>1)$ matched with the packets sent by LQD (say, b-matching), namely, X packets sent by LQD are assigned for one extra packet (say, o-extra packet). Second, in order to reduce the number of o-extra packets to be b-matched, we introduce the matching between an o-extra packet and a LQD's extra packet (say, ℓ-extra packet), which is defined as a packet sent from the ith LQD'queue when OPT does not send a pcket from the ith queue at the same time slot. However, only these are not sufficient to give a tight bound when $N=2$ since the number of packets in the current LQD buffer is not sufficient to construct a matching. Finally, to resolve this problem, we "borrow" packets from the future: We prove that, in the case where the packets current LQD buffer holds is insufficient to construct our matching, we can guarantee that enough number of packets will arrive in the future.

Related Results. The first and the most basic model of online buffer management problem is a single non-preemptive queue model introduced by Aiello, et al. [1]. In this model, each packet has a unit size 1, and a packet can have one of two values 1 and $\alpha(\geqq 1)$ (which we call two-values model). They showed that no online algorithm can be better than $(2-1/\alpha)$-competitive. Later, Andelman, et al. [4] gave a $(2-1/\alpha)$-competitive online algorithm, obtaining a tight bound. In a model where a packet can take any value in $[1,\alpha]$ (multi-values model), an upper bound $\ln(\alpha)+2+O(\ln^2(\alpha)/B)$ [3] and a lower bound $\ln(\alpha)+1$ [4] are presented. When preemption is allowed, the current best upper and lower bounds are 1.282 [10] and 1.28 [19] for two-values model, respectively, and 1.732 [10] and 1.419 [15] for multi-values model, respectively.

In a bounded delay model [4], [6], [8], [9], [12], [13], [16], each packet has a deadline. A packet disappears when the deadline comes. Another popular model is a multiple FIFO queue model [2], [5], [17]. In this model, at each send event, only one packet located at the top of some queue can be transmitted. So, the task of online algorithm is to select which queue to send the packet at each send event, as well as a conventional accept/reject/preempt decision at an arrival event. Finally, for the shared memory switch model, treated in this paper, there are some other variations; non-preemptive queue model, multi-values model, and so on [11], [14].

2. Preliminaries

In this section, we formally define a problem studied in this paper, and *LQD* policy introduced in [11].

2.1 Online Buffer Management Problem for Shared Memory Switches

A switch has one input port and N output ports. Each output port has its own buffer (FIFO queue) whose size is variable but the total size of N output queues is M. Each packet has its destination port (1 through N). The size and the value of each packet is one. Hence, the number of packets switch can store simultaneously is at most M. We assume that no more than one packets arrive at the same time.

An input is a sequence of events. An *event* is an *arrival event* or a *send event*. At an arrival event, a packet (say, p) arrives at an input port, and the task of an online algorithm (or an online policy) is to select one of the following actions: *accept* p, *reject* p, or drop a packet p' existing in the current buffer and accept p (*preempt* p'). If a packet is accepted, it is stored at the tail of the corresponding output queue. When a send event occurs, the first packets of all output queues are sent. Without loss of generality, and for the simplicity of analysis, we assume that a send event occurs at an integer time, and arrival event occurs at non-integer time. The *cost* of an algorithm is the number of sent packets. Therefore, our goal is to maximize the number of packets eventually transmitted. The cost of an online algorithm A for an input σ is denoted by $V_A(\sigma)$. If $V_A(\sigma) \geqq V_{OPT}(\sigma)/c$ for an arbitrary input σ, we say that A is *c-competitive*, where OPT is an optimal offline policy for σ. Without loss of generality, we can assume OPT never preempts packets (See [11], e.g.).

2.2 Longest Queue Drop (*LQD*) Policy

Longest Queue Drop (*LQD*) policy acts in the following way when a packet p arrives: If the buffer is not full, it accepts p. If the buffer is full, it drops a packet at the tail of the longest queue and accepts p. (If there are more than one longest queues, it selects one arbitrarily.) Hence, if p is destined for a longest queue, then it is rejected.

3. $2-\frac{1}{M}\min_{K=1,2,\ldots,N}\left\{\lfloor\frac{M}{K}\rfloor+K-1\right\}$ Upper Bound

For analyses, we give some definitions. For a time t when an event occurs, $t-$ represents a moment before t and after the previous event occurred. Similarly, $t+$ is the moment after t and before the next event occurs. The jth queue of the switch is denoted as $Q^{(j)}\,(1 \leqq j \leqq N)$. For a policy A, $h_A^{(j)}(t)$ denotes the number of packets $Q^{(j)}$ holds at time t. For a policy A and a packet p arriving at an A's queue, if p is sent at time t, then we write $s_A(p) = t$. If p is rejected or preempted, $s_A(p)$ is defined to be the time when the next send event happens for convenience, namely, $s_A(p) = \lceil t \rceil$. For a policy A and a packet p, if p is located at the kth position in a queue, then we write $\ell_A(p,t) = k$, and if p is not stored in a queue, we write $\ell_A(p,t) = 0$.

3.1 Overview of the Analysis

In order to improve an upper bound, let us classify packets sent by LQD or OPT into three categories: A packet sent from $Q^{(j)}$ of OPT when LQD does not send a packet from $Q^{(j)}$ is called an *o-extra packet*. A packet sent from $Q^{(j)}$ of LQD when OPT does not send a packet from $Q^{(j)}$ is called an *ℓ-extra packet*. A packet sent from $Q^{(j)}$ of LQD when OPT also sends a packet from $Q^{(j)}$ is called a *common packet*. Let T_{OE}, T_{LE} and T_{CM} be the numbers of o-extra packets, ℓ-extra packets and common packets, respectively. Then, it is easy to see that $V_{OPT}(\sigma) = T_{CM} + T_{OE}$ and $V_{LQD}(\sigma) = T_{CM} + T_{LE}$ for any σ. In the following sections, we evaluate the upper bound of $T_{OE} - T_{LE}$. For this purpose, we will show that each o-extra packet is *a-matched* or *b-matched* at the end of input. a-matching is 1-to-1 matching of an OPT packet p with an ℓ-extra packet p'. If p is a-matched with p', we write $m(p,a) = p'$. If p is not a-matched with p', we write $m(p,a) = \phi$. b-matching is 1-to-$M/S(N)$ matching of an OPT packet p with an LQD packets, where $S(N) = \max_{K=1,2,\ldots,N}\{M - \lfloor\frac{M}{K}\rfloor - K + 1\}$. In this case, we divide each LQD packet into $S(N)$ fragments which have $1/S(N)$ size for descriptive purposes. The fragments of an LQD packet p are denoted by $p^{(1)}, p^{(2)}, \ldots, p^{(S(N))}$. Then, if an OPT packet p is b-matched with M fragments $q_1, q_2, \ldots, q_M$ of LQD packets, we write $m(p,b) = \{q_1, q_2, \ldots, q_M\}$. (If $i \neq j$, q_i differs from q_j.) If p is not b-matched with p', we write $m(p,b) = \phi$. We write $\forall\epsilon, m(p,\epsilon) = \phi$ when p is neither a-matched or b-matched. Note that an ℓ-extra packet can be both a-matched and b-matched. Let $\tilde{T}_{OE}$ be the number of o-extra packets a-matched. Then $T_{OE} - \tilde{T}_{OE}$ packets are b-matched. Therefore, $\tilde{T}_{OE} \leqq T_{LE}$ and $T_{OE} - \tilde{T}_{OE} \leqq S(N)/MV_{LQD}$ hold, from which, we have $T_{OE} - T_{LE} \leqq S(N)/MV_{LQD}$. So, $V_{OPT} = T_{CM} + T_{OE} \leqq T_{CM} + T_{LE} + S(N)/MV_{LQD} =$

$(1+S(N)/M)V_{LQD}$. Hence, we have the following theorem:

[Theorem 3.1] The competitive ratio of LQD is at most $\left(2-\frac{1}{M}\min_{K=1,2,\ldots,N}\left\{\lfloor\frac{M}{K}\rfloor+K-1\right\}\right)$.

3.2 The Matching Routine

We will show that each o-extra packet sent at time t is a-matched or b-matched at t, in order to guarantee that each o-extra packet is a-matched or b-matched at the end of input. We give the following definition. For a queue $Q^{(j)}$ such that $h^{(j)}_{OPT}(t)-h^{(j)}_{LQD}(t)>0$, a packet p in OPT's queue satisfying $h^{(j)}_{LQD}(t)+1 \leqq \ell_{LQD}(p,t) \leqq h^{(j)}_{OPT}(t)$ is called a *charged* packet at time t. Packets in OPT's queues that are not charged are called *uncharged packets*. Note that for an integer t, if a charged packet is located at the head of an OPT's queue at $t-$, it is an o-extra packet that is sent at t. Now, we execute the following Matching Routine at time t when a send event occurs. This routine matches each charged packet which does not construct any matching at $t-$ by a-matching or b-matching. Here we will give some definitions for the routine. $g(t)$ denotes the number of charged packets at time t, namely, $g(t)=\sum_{j=1,\ldots,N}\max\{h^{(j)}_{OPT}(t)-h^{(j)}_{LQD}(t),0\}$. At time t when a send event occurs, let $f_{OE}(t)$, $f_{LE}(t)$ and $f_{CM}(t)$ be the numbers of o-extra packets sent by OPT at t, ℓ-extra packets sent by LQD at t and common packets sent by LQD at t, respectively. $\mathcal{P}(t)$ is defined as follows,

$$\mathcal{P}(t)=\begin{cases}0 & (t=0)\\ \max\{\mathcal{P}(t-1)-f_{OE}(t-1) & \\ \qquad +f_{LE}(t-1),0\} & (t=1,2,\ldots).\end{cases}$$

The intuitive meaning of $\mathcal{P}(t)$ is as follows, which will be proved later in Lemma 3.4: In Matching Routine at time t, there are at least $\mathcal{P}(t)$ packets of LQD available for a-matching.

Matching Routine (at time t when a send event occurs)

Step 1: For each OPT packet p which is a uncharged packet in $Q^{(i)}$

If $\exists\epsilon\in\{a,b\}$ $m(p,\epsilon)\neq\phi$ and a charged packet p' exists in $Q^{(i)}$ such that $\forall\gamma\in\{a,b\}$ $m(p',\gamma)=\phi$, then $m(p',\epsilon):=m(p,\epsilon)$ and $m(p,\epsilon):=\phi$.

Step 2: For each LQD packet p dropped at time between $(t-1)+$ and $t-$

If $\exists\epsilon$ $p\in m(p',\epsilon)$, then $m(p',\epsilon):=\phi$.

Let $\mathcal{K}(t)$ be the set of charged packets that are b-matched at this moment.

Step 3:

If there is a charged packet such that $\forall\epsilon$ $m(p,\epsilon)=\phi$, then we execute one of the following steps depending on the value of $g(t-)$.

Case 3.1: $g(t-)\leqq S(N)$

We construct β-matching for $g(t-)-|\mathcal{K}(t)|$ unmatched charged packets.

Case 3.2: $g(t-)>S(N)$

Step 3.2.1:

We construct α-matching for $g(t-)-S(N)$ unmatched charged packets.

Step 3.2.2:

We construct β-matching for $S(N)-|\mathcal{K}(t)|$ unmatched charged packets.

Constructing α-matching:

Let cp be an unmatched charged packet to be α-matched. By Lemma 3.5, there is an a-unmatched ℓ-extra packet such that $\ell_{LQD}(sp,t-)=1$. We a-match cp with sp, i.e. $m(cp,a):=\{sp\}$.

Constructing β-matching:

Let x be the number of unmatched charged packets to be β-matched.

Step β.a:

We construct a-matching for $\min\{x,\mathcal{P}(t)\}$ unmatched charged packets.

Let cp be an unmatched charged packet to be a-matched, and let sp be an a-unmatched ℓ-extra packet such that $\ell_{LQD}(sp,t-)=0$.

We a-match cp with sp i.e. $m(cp,a):=\{sp\}$ (say, $\beta.a$-matching).

Step β.b:

We construct b-matching for $\max\{x-\mathcal{P}(t),0\}$ unmatched charged packets.

Let cp be an unmatched charged packet to be b-matched.

By the proof of Lemma 3.7, there are $M/S(N)$ fragments of packets R such that $s_{OPT}(cp)\geqq s_{LQD}(p)$ $(p\in R)$.

We b-match cp with R i.e. $m(cp,b):=\{R\}$ (say, $\beta.b$-matching).

Constructing matchings is executed only in the routine. Therefore, if an o-extra packet p is matched with R when p is sent, p remains matched R until the end of input. Hence, if we show the feasibility of the routine at any send event time t, then we can show that each o-extra packet can be a-matched or b-matched at the end of input.

Now, we show the following lemma concerning the routine.

[Lemma 3.2] If an OPT packet p_1 is matched with p_2 i.e. $p_2\in m(p_1,\epsilon)$, then $s_{OPT}(p_1)\geqq s_{LQD}(p_2)$.

Proof. First we will show that the statement holds when matching is constructed, and then show that the condition of the statement is not broken in other steps. In the routine, the steps constructing matchings are only Step 3.2.1 (α-matching), Case 3.1 and Step 3.2.2 (β-matching). In constructing α-matching, since a charged packet p' becomes a-matched with an LQD packet p located at the top of a queue, $s_{OPT}(p') \geqq s_{LQD}(p)$. In constructing $\beta.a$-matching, since a charged packet p' becomes a-matched with an LQD packet p which has already been sent at the time executing the routine, $s_{OPT}(p') \geqq s_{LQD}(p)$. In constructing $\beta.b$-matching an OPT packet p' with an LQD packet p, $s_{OPT}(p') \geqq s_{LQD}(p)$ by the condition in constructing β.b-matching. Therefore, in constructing matchings the statement holds. Next, we consider other steps in the routine. At Step 1, we analyze the case where $m(p', \epsilon) := m(p, \epsilon)$. Since p is uncharged and p' is charged in the same queue, $\ell(p, t-) \leqq h^{(i)}_{LQD}(t-) < \ell(p', t-)$. Hence, since the statement holds in constructing matchings, the statement still holds after the execution of Step 1. At Step 2, the statement holds since we only break existing matchings. □

Using Lemma 3.2, we prove the following lemma.

[Lemma 3.3] Let p_1 be an OPT packet which has been sent before t and matched with an LQD packet p_2. Then, p_2 has been sent before t.

Proof. Since $s(p_1) \geqq s(p_2)$ by Lemma 3.2 and p_1 has been sent before t, the statement is true. □

Then, we show the property of $\mathcal{P}(t)$.

[Lemma 3.4] The number of ℓ-extra packets which have been sent before time $t-$ and are not a-matched or a-matched with charged packets at time $t-$ is at least $\mathcal{P}(t)$.

Proof. We prove the lemma by induction on time. At the beginning, the statement is true since $\mathcal{P}(0) = 0$. We assume that the statement is true at time k and show that it is true at $k+1$. By induction hypothesis, there are at least $\mathcal{P}(k)$ ℓ-extra packets that satisfy the statement of the lemma at $k-$. Also, there are $f_{LE}(k)$ ℓ-extra packets sent at k. We classify those ℓ-extra packets sent at k and the $\mathcal{P}(k)$ ℓ-extra packets into three types according to their statuses in a-matchings at k: An ℓ-extra packet a-matched with an o-extra packet sent at k is called an *a-packet*. An ℓ-extra packet a-matched with a charged packet is called an *a′-packet*. An ℓ-extra packet a-unmatched is called an *n-packet*. Since the number of a-packets is at most $f_{OE}(k)$, there are at least $\mathcal{P}(k) + f_{LE}(k) - f_{OE}(k)$ a'-packets and n-packets, which is clearly at least $\mathcal{P}(k+1)$. We have shown that the statement holds at $k+1$. □

a'-packets at time t and n-packets at time t are the packets used to construct a-matching in the routine executed at t. Hence, this lemma shows that $\mathcal{P}(t)$ is the lower bound on the number of ℓ-extra packets that can be used to construct a-matching in the routine executed at t.

3.3 Feasibility of the Matching Routine

In this section, we guarantee the feasibility of Matching Routine. In particular, we show that α-matching (at Step 3.2.1) and β-matching (at Case 3.1 and Step 3.2.2) can be constructed at any time. First, we prove the feasibility of α-matching. Let $\mathcal{H}(t)$ denote the set of indices j such that LQD has at least one packet in $Q^{(j)}$ at time t, namely, $\mathcal{H}(t) = \{j \mid h^{(j)}_{LQD}(t) > 0, \ \ 1 \leqq j \leqq N\}$.

[Lemma 3.5] Let t' be a time when a send event happens. Suppose that between $(t'-1)+$ and $t'-$, there arises an unmatched charged packet. Then, $g(t'-) - f_{LE}(t') \leqq S(N)$.

Proof. Let $t \in (t'-1, t')$ be the time when the arrival event occurs, and suppose that an unmatched charged packet arises in $Q^{(1)}$, and no such event happens in (t, t'). Note that such event occurs when OPT accepts a new packet and LQD rejects it, or LQD preempts a packet. In any case, OPT has at least $\lfloor \frac{M}{|\mathcal{H}(t+)|} \rfloor$ uncharged packet by the definition of LQD. Hence, $g(t+) \leqq M - \lfloor \frac{M}{|\mathcal{H}(t+)|} \rfloor$. Since LQD does not drop any packet between t and t', the number of charged packets does not increase, namely $g(t+) \geqq g(t'-)$. By the above two inequalities, we have that $g(t'-) \leqq M - \lfloor \frac{M}{|\mathcal{H}(t+)|} \rfloor$. First, suppose that $N \leqq M$. By the definition of LQD, it does not drop a packet located at the head of a queue. Hence, $|\mathcal{H}(t+)| \leqq |\mathcal{H}(t'-)|$. On the other hand, if $N > M$, a packet p located at the head of a queue at time $t+$ may be dropped between t and t'. However, in such a case, a new packet is accepted at the head of another queue instead of p by the definition of LQD. So, $|\mathcal{H}(t+)| \leqq |\mathcal{H}(t'-)|$. Hence, we have that $g(t'-) \leqq M - \lfloor \frac{M}{|\mathcal{H}(t'-)|} \rfloor$. Now, for $Q^{(j)} \quad (j \in \mathcal{H}(t'-), j \neq 1)$, let y be the number of queues such that $h^{(j)}_{OPT}(t'-) > 0$, and let x be the number of queues such that $h^{(j)}_{OPT}(t'-) = 0$. By the definition of x and y, $x + y = |\mathcal{H}(t'-)| - 1$. By the definition of y, OPT has at least y uncharged packets at $t'-$ and these packets do not include packets in $Q^{(1)}$. Hence, $g(t'-) \leqq M - \lfloor \frac{M}{|\mathcal{H}(t'-)|} \rfloor - y$. By the definition of x and t', LQD holds x ℓ-extra packets at the head of queues, namely $f_{LE}(t') = x$. By the above two inequalities, we have that

$$
\begin{aligned}
g(t'-) - f_{LE}(t') &\leqq M - \lfloor \frac{M}{|\mathcal{H}(t'-)|} \rfloor - y - x \\
&= M - \lfloor \frac{M}{|\mathcal{H}(t'-)|} \rfloor - |\mathcal{H}(t'-)| + 1 \\
&\leqq \max_{K=1,2,\ldots,N} \{M - \lfloor \frac{M}{K} \rfloor - K + 1\} \\
&= S(N).
\end{aligned}
$$

□

Now, we prove that α-matching can be constructed using Lemma 3.5.

[Lemma 3.6] We can construct α-matching.

Proof. We show that α-matching can be constructed at time t' when a send event occurs. When α-matching needs to be constructed, $g(t'-) > S(N)$ holds at Step 3.2.1 in the routine. Therefore, the number of packets to be a-matched in constructing α-matching is at most $g(t'-) - S(N)$. On the other hand, since $g(t'-) - f_{LE}(t') \leqq S(N)$ by Lemma 3.5, LQD holds at least $g(t'-) - S(N)$ ℓ-extra packets at the head of queues at time $t'-$. In addition, they are not a-matched with o-extra packets which have been sent before $t'-$ by Lemma 3.3. Hence, we have shown that the statement is true. □

Next, we will show that β-matching can be constructed. Let $g_\beta(t-)$ denote the number of packets to be matched in constructing β-matching at time t when a send event occurs.

[Lemma 3.7] We can construct β-matching.

Proof. We show that β-matching can be constructed at time t' when a send event occurs. According to the rule of Matching Routine, $g_\beta(t'-)$ is at most $S(N) - |\mathcal{K}(t')|$. Then, we consider LQD packets available for constructing $\beta.b$-matching. Let p be an arbitrary LQD packet in a queue at $t'-$. Then, by the definition of LQD, $\ell_{OPT}(cp, t'-) \geqq \ell_{LQD}(p, t'-)$ holds for each packet cp becoming charged between $t'-1$ and t', and hence $s_{OPT}(cp) \geqq s_{LQD}(p)$. In addition, by Lemma 3.3, p is not matched with a packet which has been sent before $t'-$. Hence, $\sum_{i=1}^{N} h_{LQD}^{(i)}(t'-) - |\mathcal{K}(t')|M/S(N)$ b-unmatched packets in the queues at $t'-$ can be used for constructing $\beta.b$-matching. Since LQD drops at least one packet between $(t'-1, t')$ when there exist charged packets to be matched at t', $\sum_{i=1}^{N} h_{LQD}^{(i)}(t'-) = M$ by the definition of LQD. Secondly, we consider LQD packets for constructing $\beta.a$-matching. At least $\mathcal{P}(t')$ ℓ-extra packets exist for constructing $\beta.a$-matching by Lemma 3.4. Now, we check that $M/S(N)$ LQD packets can be assigned to each charged packet to be matched in constructing $\beta.b$-matching at time t'. Note that we can use $\mathcal{P}(t')$ ℓ-extra packets exists for constructing $\beta.a$-matching, whose existence is guaranteed as mentioned above. We have that

$$\begin{aligned}&\sum_{j=1}^{N} h_{LQD}^{(j)}(t'-) - \frac{|\mathcal{K}(t')|M}{S(N)}\\&\quad - \frac{M}{S(N)}(g_\beta(t'-) - |\mathcal{K}(t')| - \mathcal{P}(t'))\\&\quad \geqq M - \frac{M}{S(N)}(S(N) - \mathcal{P}(t')) \geqq 0,\end{aligned}$$

which completes the proof. □

4. $\frac{4M-4}{3M-2}$ Upper Bound for $N=2$

4.1 Overview of the Analysis

In this section, we assume that M is even for simplicity. In the case of $N = 2$, we can give a better bound. To this end, we use c-matching instead of b-matching. The definition of a-matching is the same to that of previous section. c-matching denotes the 1-to-$(3M-2)/(M-2)$ matching an OPT packet p with LQD packets. Therefore, we have that $T_{OE} - T_{LE} \leqq \frac{M-2}{3M-2} V_{LQD}(\sigma)$. Hence, we have the following theorem:

[Theorem 4.1] When $N = 2$, the competitive ratio of LQD is at most $\frac{4M-4}{3M-2}$.

Though the lower bound on the competitive ratio of LQD given in [11] is at least 4/3, we can show that the lower bound is $\frac{4M-4}{3M-2}$ if we analyze it in more detail. Similarly to the previous section, we use the matching routine in order to prove that each o-extra packet can be a-matched or c-matched at the end of input. Let t be a send event time. The routine in this section (Routine 2) is the same to Matching Routine in the previous section by replacing $\mathcal{P}(t)$ by $\mathcal{P}(t')$ ($t'(> t)$ is specified later) in Step 3.2.2 and changing the number of packets b-matched in Step $\beta.b$ from $M/S(N)$ to $(3M-2)/(M-2)$.

4.2 Feasibility of the Matching Routine

In this section, we will show that the feasibility of Routine 2. Since the feasibility of α-matching can be proved by the same proof as in the previous section, we only prove the feasibility of β-matching. If we try to apply the same proof as the previous section for $N = 2$, then $\mathcal{P}(t)$ cannot be a lower bound on the number of LQD packets available for constructing $\beta.a$-matching because we have replaced b-matching by c-matching. Hence, we prove the relation between $\mathcal{P}(t)$ and the number of packets OPT and LQD hold. Using this relation, we can evaluate the number of packets available for matching, by a technique stronger than ones in the previous section.

[Lemma 4.2] At any time t, $\sum_{i=1}^{2} h_{LQD}^{(i)}(t) + \mathcal{P}(\lceil t \rceil) \geqq \sum_{i=1}^{2} h_{OPT}^{(i)}(t)$.

Proof. The proof is by induction on time. At the beginning, obviously the statement is true. Next, for a time t when an event happens, we assume that the statement is true at time $t-$, and show that it is true at time $t+$, namely, we assume that $\sum_{i=1}^{2} h_{LQD}^{(i)}(t-) + \mathcal{P}(\lceil t- \rceil) \geqq \sum_{i=1}^{2} h_{OPT}^{(i)}(t-)$, and prove $\sum_{i=1}^{2} h_{LQD}^{(i)}(t+) + \mathcal{P}(\lceil t+ \rceil) \geqq \sum_{i=1}^{2} h_{OPT}^{(i)}(t+)$.

Case 1. OPT and LQD send packets. We have that

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t+) = \sum_{i=1}^{2} h_{LQD}^{(i)}(t-) - f_{CM}(t) - f_{LE}(t),$$

and

$$\sum_{i=1}^{2} h_{OPT}^{(i)}(t+) = \sum_{i=1}^{2} h_{OPT}^{(i)}(t-) - f_{CM}(t) - f_{OE}(t).$$

From the above equations and by the definition of $\mathcal{P}(\lceil t+\rceil)$,

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t+) + \mathcal{P}(\lceil t+\rceil) - \sum_{i=1}^{2} h_{OPT}^{(i)}(t+)$$
$$= \sum_{i=1}^{2} h_{LQD}^{(i)}(t-) + \max\{\mathcal{P}(\lceil t-\rceil) - f_{OE}(t) + f_{LE}(t), 0\}$$
$$- \sum_{i=1}^{2} h_{OPT}^{(i)}(t-) - f_{LE}(t) + f_{OE}(t).$$

Now, we have two cases. First, if $\mathcal{P}(\lceil t-\rceil) - f_{OE}(t) + f_{LE}(t) \geqq 0$,

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t-) + \mathcal{P}(\lceil t-\rceil) - f_{OE}(t) + f_{LE}(t)$$
$$- \sum_{i=1}^{2} h_{OPT}^{(i)}(t-) - f_{LE}(t) + f_{OE}(t)$$
$$= \sum_{i=1}^{2} h_{LQD}^{(i)}(t-) + \mathcal{P}(\lceil t-\rceil) - \sum_{i=1}^{2} h_{OPT}^{(i)}(t-) \geqq 0,$$

and the statement holds. Next, if $\mathcal{P}(\lceil t-\rceil) - f_{OE}(t) + f_{LE}(t) < 0$,

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t-) - \sum_{i=1}^{2} h_{OPT}^{(i)}(t-) - f_{LE}(t) + f_{OE}(t)$$
$$> \sum_{i=1}^{2} h_{LQD}^{(i)}(t-) + \mathcal{P}(\lceil t-\rceil) - \sum_{i=1}^{2} h_{OPT}^{(i)}(t-) \geqq 0.$$

Again, the statement holds.

Next, suppose that the event at t is an arrival event. Let p be the packet arrived at t. Note that, in this event, $\mathcal{P}(\lceil t-\rceil)$ never changes i.e. $\mathcal{P}(\lceil t-\rceil) = \mathcal{P}(\lceil t+\rceil)$.

Case 2. ***LQD*** **accepts.** Since OPT accepts or rejects p, $\sum_{i=1}^{2} h_{OPT}^{(i)}(t+) \leqq \sum_{i=1}^{2} h_{OPT}^{(i)}(t-) + 1$. On the other hand, LQD accepts p and hence, $\sum_{i=1}^{2} h_{LQD}^{(i)}(t+) = \sum_{i=1}^{2} h_{LQD}^{(i)}(t-) + 1$. We have that

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t+) + \mathcal{P}(\lceil t+\rceil) - \sum_{i=1}^{2} h_{OPT}^{(i)}(t+)$$
$$\geqq (\sum_{i=1}^{2} h_{LQD}^{(i)}(t-) + 1) + \mathcal{P}(\lceil t-\rceil) - (\sum_{i=1}^{2} h_{OPT}^{(i)}(t-) + 1)$$
$$\geqq 0$$

Case 3. ***LQD*** **rejects or preempts.** Since LQD is greedy, when it rejects or preempts, its buffer is full, i.e., $\sum_{i=1}^{2} h_{LQD}^{(i)}(t+) = M$. On the other hand, by the definition of OPT, $\sum_{i=1}^{2} h_{OPT}^{(i)}(t+) \leqq M$. We have that

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t+) + \mathcal{P}(\lceil t+\rceil) - \sum_{i=1}^{2} h_{OPT}^{(i)}(t+)$$
$$\geqq M + \mathcal{P}(\lceil t-\rceil) - M \geqq 0.$$

In each case, we have shown that the statement holds at $t+$. □

Now, we prove the following lemma using Lemma 4.2.

[Lemma 4.3] Let t be a time when a event does not happen, and $B(\lceil t\rceil)$ be the number of packets which LQD does not hold in queues at t and sends between $\lceil t\rceil$ and $\lceil t\rceil + \max_{k_1}\{h_{OPT}^{(k_1)}(t)\} - 1$. Let $t' = \lceil t\rceil + \max_{k_2}\{h_{LQD}^{(k_2)}(t)\} - 1$. Then $B(\lceil t\rceil) + \mathcal{P}(t'+1) \geqq \max\{\max_{k_1}\{h_{OPT}^{(k_1)}(t)\} - \max_{k_2}\{h_{LQD}^{(k_2)}(t)\}, 0\}$.

Proof. If $\max_{k_2}\{h_{LQD}^{(k_2)}(t)\} \geqq \max_{k_1}\{h_{OPT}^{(k_1)}(t)\}$, it is not hard to see that the statement is true since the right hand side of the inequality is 0. Then, in the following, we consider the case $\max_{k_2}\{h_{LQD}^{(k_2)}(t)\} < \max_{k_1}\{h_{OPT}^{(k_1)}(t)\}$ holds. Without loss of generality, we assume $h_{OPT}^{(1)}(t) = \max_{k_1}\{h_{OPT}^{(k_1)}(t)\}$.

Since OPT never preempts a packet,

$$\sum_{i=1}^{2} h_{OPT}^{(i)}(t'+) \geqq h_{OPT}^{(1)}(t'+) \geqq h_{OPT}^{(1)}(t) - \max_{k_2}\{h_{LQD}^{(k_2)}(t)\}.$$

Using Lemma 4.2,

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t'+) + \mathcal{P}(t'+1) \geqq \sum_{i=1}^{2} h_{OPT}^{(i)}(t'+).$$

By the above two inequalities, we have that

$$\sum_{i=1}^{2} h_{LQD}^{(i)}(t'+) + \mathcal{P}(t'+1) \geqq h_{OPT}^{(1)}(t) - \max_{k_2}\{h_{LQD}^{(k_2)}(t)\}.$$

Let C be the number of packets which LQD holds at $t'+$ and sends after $(\lceil t\rceil + \max_{k_1}\{h_{OPT}^{(k_1)}(t)\} - 1)+$. If $C = 0$, it is easy to see that the statement holds since $\sum_{i=1}^{2} h_{LQD}^{(i)}(t'+) \leqq B(\lceil t\rceil)$. Next, if $C > 0$, LQD holds more than $h_{OPT}^{(1)}(t) - \max_{k_2}\{h_{LQD}^{(k_2)}(t)\}$ packets at time $t'+$. Since all packets LQD holds at t have been sent before $t'+$ by the definition of t', we have that $B(\lceil t\rceil) \geqq h_{OPT}^{(1)}(t) - \max_{k_2}\{h_{LQD}^{(k_2)}(t)\}$, and since $\mathcal{P}(t'+1) \geqq 0$, the statement is true. □

We are ready to prove the main lemma.

[Lemma 4.4] When $N = 2$, we can construct β-matching.

Proof. We show that β-matching can be constructed at time t' when a send event happens. $g_\beta(t'-)$ is at most $S(2) - |\mathcal{K}(t')|$ by the condition of Step 3.2.2. Similarly to the proof of Lemma 3.7, LQD holds at least $M - |\mathcal{K}(t')|M/S(2)$ b-unmatched packets in the queues at $t'-$ which can be used for constructing $\beta.b$-matching. Since LQD is greedy, it cannot be the case that new charged packets arise in both queues at the same time. Therefore, we assume that there are charged packets in $Q^{(2)}$ at $t'-$. Let $t \in (t'-1, t')$ be the time when the arrival event occurs and an unmatched charged packet arises and no such event happens in $(t'-1, t')$. Then, $h^{(2)}_{OPT}(t+) - h^{(2)}_{LQD}(t+) = g(t+) \geqq g(t'-) \geqq g_\beta(t'-)$ holds. And using Lemma 4.3, if we let $t'' = t' + h^{(2)}_{LQD}(t+)$, then $B(t') + \mathcal{P}(t'') \geqq h^{(2)}_{OPT}(t+) - h^{(2)}_{LQD}(t+)$ holds. We consider packets in $B(t')$. If we let p be a charged packet at $t+$, an LQD packet p' exists in $B(t')$ such that $s_{LQD}(p') \leqq s_{OPT}(p)$. Next, let us consider $\mathcal{P}(t'')$. Using Lemma 3.4, $\mathcal{P}(t'')$ denotes the number of ℓ-extra packets which can be a-matched at t''. Since the charged packets in queues at $t'-$ are not sent until t'', we can use ℓ-extra packets in $\mathcal{P}(t'')$ for constructing $\beta.a$-matching at t'. Now, we check if $\frac{3M-2}{M-2}$ LQD packets can be assigned to each charged packet to be matched in constructing $\beta.b$-matching at time t'. Note that we can use $\mathcal{P}(t'')$ ℓ-extra packets for constructing $\beta.a$-matching. We have that

$$\begin{aligned}
&\sum_{j=1}^{2} h^{(j)}_{LQD}(t'-) - \frac{3M-2}{M-2}|\mathcal{K}(t')| + B(t') \\
&- \frac{3M-2}{M-2}(g_\beta(t'-) - |\mathcal{K}(t')| - \mathcal{P}(t'')) \\
&\geqq M + g_\beta(t'-) - \mathcal{P}(t'') - \frac{3M-2}{M-2}(g_\beta(t'-) - \mathcal{P}(t'')) \\
&\geqq M - \frac{2M}{M-2}(M - \frac{1}{2}M - 1 - 0) = 0,
\end{aligned}$$

which completes the proof. □

5. Concluding Remarks

In this paper, we studied the performance of LQD policy in buffer management for shared memory switches. We improved the competitive ratio of LQD to $\frac{4M-4}{3M-2}$ when $N = 2$, which matches the lower bound. Also, in the case of arbitrary N, we gave an upper bound on the competitive ratio $2 - \frac{1}{M}(\lfloor \frac{M}{N} \rfloor + N)$.

References

[1] W. Aiello, Y. Mansour, S. Rajagopolan, and A. Rosen, "Competitive queue policies for differentiated services," *IEEE INFOCOM*, pp. 431–440, 2000.

[2] S. Albers and M. Schmidt, "On the performance of greedy algorithms in packet buffering," *SIAM J. Comput.*, Vol. 35, No. 2, pp. 278–304, 2005.

[3] N. Andelman and Y. Mansour, "Competitive management of non-preemptive queues with multiple values," *In Proc. of 17th International Symposium on Distributed Computing*, pp. 166–180, 2003.

[4] N. Andelman, Y. Mansour and A. Zhu, "Competitive queueing policies for QoS switches," *In Proc. of 14th annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 761–770, 2003.

[5] Y. Azar and Y. Richter, "Management of multi-queue switches in QoS networks," *In Proc. of 35th annual ACM Symposium on Theory of Computing*, pp. 82–89, 2003.

[6] Y. Bartal, F. Chin, M. Chrobak, S. Fung, W. Jawor, R. Lavi, J. Sgall and T. Tichý, "Online competitive algorithms for maximizing weighted throughput of unit jobs," *In Proc. of 21st International Symposium on Theoretical Aspects of Computer Science*, pp. 187–198, 2004.

[7] A. Borodin and R. El-Yaniv, "Online Computation and Competitive Analysis," *Cambridge University Press*, 1998.

[8] F. Chin and S. Fung, "Online scheduling for partial job values: Does timesharing or randomization help?," Algorithmica, Vol.37, pp. 149–164, 2003.

[9] M. Chrobak, W. Jawor, J. Sgall and T. Tichý, "Improved online algorithms for buffer management in QoS switches," *In Proc. of 12th annual European Symposium on Algorithms*, pp. 204–215, 2004.

[10] M. Englert and M. Westermann, "Lower and Upper Bounds on FIFO Buffer Management in QoS Switches," *In Proc. of 14th annual European Symposium on Algorithms*, pp. 352–363, 2006.

[11] E. Hahne, A. Kesselman and Y. Mansour, "Competitive buffer management for shared-memory switches," *In Proc. of 13th annual ACM Symposium on Parallel Algorithms and Architectures*, pp. 53–58, 2001.

[12] B. Hajek, "On the competitiveness of on-line scheduling of unit-length packets with hard deadlines in slotted time," *In Proc. 35th annual Conference on Information Sciences and Systems*, pp. 434–439, 2001.

[13] A. Kesselman, Z. Lotker, Y. Mansour, B. Patt-Shamir, B. Schieber, and M. Sviridenko, "Buffer overflow management in QoS switches," *In Proc. of 33rd annual ACM Symposium on Theory of Computing*, pp. 520–529, 2001.

[14] A. Kesselman and Y. Mansour, "Harmonic buffer management policy for shared memory switches," *Theoretical Computer Science*, Vol. 324, No. 2-3, pp. 161–182, 2004.

[15] A. Kesselman, Y. Mansour and R. Stee, "Improved competitive guarantees for QoS buffering," *In Proc. of 11th annual European Symposium on Algorithms*, pp. 361–372, 2003.

[16] F. Li, J. Sethuraman, and C. Stein, "An optimal online algorithm for packet scheduling with agreeable deadlines," *In Proc. of 16th annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 801–802, 2005.

[17] M. Schmidt, "Packet buffering - randomization beats deterministic algorithms," *In Proc. of 22nd International Symposium on Theoretical Aspects of Computer Science*, pp. 293–304, 2005.

[18] D. Sleator and R. Tarjan, "Amortized efficiency of list update and paging rules," *CACM 28*, pp. 202–208, 1985.

[19] M. Sviridenko, "A lower bound for on-line algorithms in the FIFO model," unpublished manuscript, 2001.