Buffer Management of Multi-Queue QoS Switches with Class Segregation

TOSHIYA ITOH  
titoh@ip.titech.ac.jp  
Imaging Science & Engineering Laboratory  
Tokyo Institute of Technology

SEIJI YOSHIMOTO  
yoshimoto.s.aa@m.titech.ac.jp  
Department of Information Processing  
Tokyo Institute of Technology

Abstract: In this paper, we focus on buffer management of multi-queue QoS switches in which packets of different values are segregated in different queues. Our model consists of \(m\) queues and \(m\) packet values \(0 < v_1 < v_2 < \cdots < v_m\). Recently, Al-Bawani and Souza [IPL 113(4), pp.145-150, 2013] presented an online algorithm GREEDY for buffer management of multi-queue QoS switches with class segregation and showed that if queues have the same size, then the competitive ratio of GREEDY is \(1 + r\), where \(r = \max_{1 \leq i \leq m-1} v_i/v_{i+1}\). In this paper, we precisely analyze the behavior of GREEDY and show that it is \((1 + r)\)-competitive for the case that \(m\) queues do not necessarily have the same size.

Key Words: Online Algorithms, Competitive Ratio, Buffer Management, Class Segregation, Quality of Service (QoS), Class of Service (CoS).

1 Introduction

Due to the burst growth of the Internet use, network traffic has increased year by year. This overloads networking systems and degrades the quality of communications, e.g., loss of bandwidth, packet drops, delay of responses, etc. To overcome such degradation of the communication quality, the notion of Quality of Service (QoS) has received attention in practice, and is implemented by assigning nonnegative numerical values to packets to provide them with differentiated levels of service (priority). Such a Quality of Service (QoS) has received attention in practice, and is implemented by assigning nonnegative numerical values to packets to provide them with differentiated levels of service (priority). Such a packet value corresponds to the predefined Class of Service (CoS). In general, switches have several number of queues and each queue has a buffer to store arriving packets. Since network traffic changes frequently, switches need to control arriving packets to maximize the total values of transmitted packets, which is called buffer management. Basically, switches have no knowledge on the arrivals of packets in the future when it manages to control new packets arriving to the switches. So the decision made by buffer management algorithm can be regarded as an online algorithm. In general, the performance of buffer algorithms is measured by competitive ratio [10]. Online buffer management algorithms can be classified into two types of queue management (one is preemptive and the other is nonpreemptive). Informally, we say that an online buffer management algorithm is preemptive if it is allowed to discard packets buffered in the queues on the arrival of new packets; nonpreemptive otherwise (i.e., all packets buffered in the queues will be eventually transmitted).

1.1 Multi-Queue Buffer Management

In this paper, we focus on a multi-queue model in which packets of different values are segregated in different queues (see, e.g., [12], [13]). Our model consists of \(m\) packet values and \(m\) queues. Let \(V = \{v_1, v_2, \ldots, v_m\}\) be the set of \(m\) nonnegative packet values, where \(0 < v_1 < v_2 < \cdots < v_m\), and let \(Q = \{Q_1, Q_2, \ldots, Q_m\}\) be the set of \(m\) queues. A packet of value \(v_j \in V\) is referred to as a \(v_j\)-packet, and a queue storing \(v_j\)-packets is referred to as a \(v_j\)-queue. Without loss of generality, we assume that \(Q_j \in Q\) is a \(v_j\)-queue for each \(j \in [1, m]\). Each \(Q_j \in Q\) has a capacity \(B_j \geq 1\), i.e., each \(Q_j \in Q\) can store up to \(B_j \geq 1\) packets. Since all packets buffered in each queue \(Q_j \in Q\) have the same value \(v_j \in V\), the order of transmitting packets buffered in queue \(Q_j \in Q\) is irrelevant.

---

In general, we can consider a model of \(m\) packet values and \(n\) queues (with \(m \neq n\)), but in this paper, we deal with only a model of \(m\) packet values and \(m\) queues.

For any pair of integers \(a \leq b\), let \([a, b] = \{a, a + 1, \ldots, b\}\).
For convenience, we assume that time is discretized into slots of unit length. Packets arrive over time and each arriving packet is assigned with a (nonintegral) arrival time, a value $v_j \in V$, and its destination queue $Q_j \in Q$ (as we have assumed, $Q_j \in Q$ is a $v_j$-queue). Let $\sigma$ be a sequence of arrive events and send events, where an arrive event corresponds to the arrival of a new packet and a send event corresponds to the transmission of a packet buffered in queues at integral time (i.e., the end of time slot). An online (multi-queue) buffer management algorithm ALG consists of two phases: one is an admission phase and the other is a scheduling phase. In the admission phase, ALG must decide on the arrival of a packet whether to accept or reject the packet with no knowledge on the future arrivals of packets (if ALG is preemptive, then it may discard packets buffered in queues in the admission phase). In the scheduling phase, ALG chooses one of the nonempty queues at send event and exactly one packet is transmitted out of the chosen queue. Since all packets buffered in the same queue have the same value, preemption does not make sense in our model. Thus a packet accepted must eventually be transmitted.

We say that an (online and offline) algorithm is diligent if (1) it must accept a packet arriving to its destination queue when the destination queue has vacancies, and (2) it must transmit a packet when it has nonempty queues. It is not difficult to see that any nondiligent (online and offline) algorithm can be transformed to a diligent (online and offline) algorithm without decreasing its benefit (sum of values of transmitted packets). Thus in this paper, we focus on only diligent algorithms.

1.2 Main Results

Al-Bawani and Souza [2, Theorem 2.2] presented an online multi-queue buffer management algorithm GREEDY and showed that it is $(1+r)$-competitive for the case that $m$ queues have the same size, where

$$r = \max_{i \in [1,m-1]} \frac{v_i}{v_{i+1}}.$$  

In this paper, we remove the restriction that $m$ queue have the same size and show that the competitive ratio of GREEDY is $1+r$ for the case that $m$ queues do not necessarily have the same size (see Theorem 3.1). In addition, we construct a bad sequence $\sigma$ of events to show that the competitive ratio of GREEDY is at least $1+r$ for the case that $m$ queues do not necessarily have the same size (see Theorem 4.1).

1.3 Related Works

The competitive analysis for the buffer management policies for switches were initiated by Aiello et al. [1], Mansour et al. [19], and Kesselman et al. [17], and the extensive studies have been made for several models (for comprehensive surveys, see, e.g., [4], [13], [16], [11], [14]).

The model we deal with in this paper can be regarded as the generalization of unit-valued model, where the switches consist of $m$ queues of the same buffer size $B \geq 1$ and all packets have unit value, i.e., $v_1 = v_2 = \cdots = v_m$. The following tables summarize the known results (see Tables 1 and 2). On the other hand, the model we deal with in this paper can be regarded as a special case of the general $m$-valued multi-queue model, where each of $m$ queues can buffer at most $B$ packets of different values. For the preemptive multi-queue buffer management, Azar and Richter [6] showed a $(4+2\ln \alpha)$-competitive algorithm for the general $m$-valued case (packet values lie between 1 and $\alpha$) and a 2.6-competitive algorithm for the two-valued case (packet values are $v_1 < v_2$, where $v_1 = 1$ and $v_2 = \alpha$). For the general $m$-valued case, Azar and Righter [7] proposed a more efficient algorithm TRANSMIT-LARGEST HEAD (TLH) that is 3-competitive, which is shown to be $(3-1/\alpha)$-competitive by Itoh and Takahashi [15].

2 Preliminaries

2.1 Notations and Terminologies

Let $\sigma$ be a sequence of arrive and send events. Note that an arrive event corresponds to the arrival of a new packet (at nonintegral time) and a send event corresponds to the transmission of a packet buffered
Table 1: Deterministic Competitive Ratio (Unit-Valued Multi-Queue Model)

| Upper Bound | Lower Bound |
|-------------|-------------|
| 2 \[6\]   | 2 \(\frac{1}{m}\) \[6\] B = 1 |
| 1.889 \[3\] | 1.366 \(-\Theta(1/m)\) \[6\] B \(\geq 2\) |
| 1.857 \[3\] | 1.231 \[3\] large B |
| \(\frac{e}{e-1}\) \approx 1.582 \[5\] | \(\frac{e}{e-1}\) \approx 1.582 \[3\] — |

Table 2: Randomized Competitive Ratio (Unit-Valued Multi-Queue Model)

| Upper Bound | Lower Bound |
|-------------|-------------|
| \(\frac{e}{e-1}\) \approx 1.582 \[6\] | 1.46 \(-\Theta(1/m)\) \[6\] B = 1 |
| 1.231 \[9\] | 1.4659 \[3\] large m |
| \(\frac{1}{m}\) \[2\] \[3\] B \(\geq 2\) |
| m = 2 | m = 2 |

in queues at integral time. The algorithm GREEDY works as follows: At send event, GREEDY transmits a packet from the nonempty queue with the highest packet value, i.e., GREEDY transmits a \(v_h\)-packet if \(v_h\)-queue is nonempty and all \(v_{\ell}\)-queues are empty for \(\ell \in [h+1,m]\). At arrive event, GREEDY accepts packets in its destination queue until the corresponding queue becomes full.

For an online algorithm ALG and a sequence \(\sigma\), we use \(ALG(\sigma)\) to denote the benefit of the algorithm ALG on the sequence \(\sigma\), i.e., the sum of values of packets transmitted by ALG on \(\sigma\). For a sequence \(\sigma\), we also use \(OPT(\sigma)\) to denote the benefit of the optimal offline algorithm OPT on the sequence \(\sigma\), i.e., the sum of values of packets transmitted by OPT that knows the entire sequence \(\sigma\) in advance. For \(c \geq 1\), we say that an online algorithm ALG is \(c\)-competitive if \(OPT(\sigma)/ALG(\sigma) \leq c\) for any sequence \(\sigma\). Thus our goal is to design an efficient (deterministic) online algorithm ALG that minimizes \(OPT(\sigma)/ALG(\sigma)\) for any sequence \(\sigma\). For a sequence \(\sigma\), let \(A_j(\sigma)\) and \(A^*_j(\sigma)\) be the total number of \(v_j\)-packets accepted by GREEDY and OPT until the end of the sequence \(\sigma\), respectively. When \(\sigma\) is clear from the context, we simply denote \(A_j\) and \(A^*_j\) instead of \(A_j(\sigma)\) and \(A^*_j(\sigma)\), respectively.

2.2 Overview for GREEDY

For the case that \(B_j = B\) for each \(j \in [1,m]\), Al-Bawani and Souza \[2\] derived the following lemmas and showed that the competitive ratio of GREEDY is \(1 + r\) \[2\] Theorem 2.2], where

\[ r = \max_{i \in [1,m-1]} \frac{v_i}{v_{i+1}}. \]

**Lemma 2.1** \[2\] Lemma 2.3\]: \(A^*_m = A_m\).

**Lemma 2.2** \[2\] Lemma 2.4\]: For any \(i \in [1,m-1]\), \(\sum_{j=i}^{m-1} (A^*_j - A_j) \leq \sum_{j=i+1}^{m} A_j\).

**Lemma 2.3** \[2\] Lemma 2.6\]: \(\sum_{j=1}^{m-1} v_j(A^*_j - A_j) \leq \sum_{j=1}^{m-1} v_jA_{j+1}\).

**Lemma 2.4** \[2\] Lemma 2.7\]: \(\sum_{j=1}^{m-1} v_jA_{j+1} / \sum_{j=1}^{m-1} v_{j+1}A_{j+1} \leq r\).

\(^3\) Since \(Q_j \in Q\) is a \(v_j\)-queue, such a nonempty queue with highest packet value is unique if it exists.
In fact, the competitive ratio of the algorithm GREEDY can be derived as follows:

\[
\frac{\text{OPT}(\sigma)}{\text{GREEDY}(\sigma)} = \frac{\sum_{j=1}^{m} v_j A_j^*}{\sum_{j=1}^{m} v_j A_j} = 1 + \frac{\sum_{j=1}^{m-1} v_j (A_j^* - A_j)}{\sum_{j=1}^{m} v_j A_j} \leq 1 + \frac{\sum_{j=1}^{m-1} v_j A_{j+1}}{\sum_{j=1}^{m} v_j A_{j+1}} \leq 1 + r,
\]

where the second equality follows from Lemma 2.1, the first inequality follows from Lemma 2.3, and the second inequality follows from Lemma 2.4.

Lemmas 2.1 and 2.4 hold unless \( B_j = B \) for each \( j \in [1, m] \). On the other hand, Lemma 2.3 immediately follows from Lemma 2.2, however, Lemma 2.2 is shown only when \( B_j = B \) for each \( j \in [1, m] \). So for each \( i \in [1, m - 1] \), if \( \sum_{j=i}^{m-1} (A_j^* - A_j) \leq \sum_{j=i+1}^{m} A_j \) holds for general \( B_j \)'s (i.e., it is not necessarily the case that \( B_j = B \) for each \( j \in [1, m] \)), then we can show that the competitive ratio of GREEDY is \( 1 + r \) for general \( B_j \)'s. In the following section, we extend Lemma 2.2 to the case of general \( B_j \)'s, which implies that the competitive ratio of the algorithm GREEDY is \( 1 + r \) for general \( B_j \)'s.

### 3 Upper Bounds

In this section, we show the following theorem.

**Theorem 3.1:** For \( m \) packet values \( 0 < v_1 < v_2 < \cdots < v_m \), the competitive ratio of GREEDY is \( 1 + r \) for the case that \( m \) queues do not necessarily have the same size, where \( r = \max_{i \in [1, m-1]} v_i/v_{i+1} \).

As mentioned in Section 2.2, the following lemma is essential to show Theorem 3.1 and is an extension of Lemma 2.2 to the case that \( m \) queues do not necessarily have the same size.

**Lemma 3.1:** For each \( i \in [1, m - 1] \), \( \sum_{j=i}^{m-1} (A_j^* - A_j) \leq \sum_{j=i+1}^{m} A_j \) holds for general \( B_j \)'s (i.e., it is not necessarily the case that \( B_j = B \) for each \( j \in [1, m] \)).

#### 3.1 Proof of Lemma 3.1

For an arbitrarily fixed \( i \in [1, m - 1] \), let \( V_i = \{v_i, v_{i+1}, \ldots, v_m\} \subseteq V \) and \( V_i^\prime = \{v_1, v_2, \ldots, v_{i-1}\} \subseteq V \). The notion of time intervals is defined as follows: A time interval \( \text{ITV} \) ends with a send event and the next time interval starts with the first arrive event after the end of \( \text{ITV} \). We say that \( \text{ITV} \) is an \textit{i-red interval} (or \( r_i \)-interval) if the value of any packet sent by GREEDY during \( \text{ITV} \) is in \( V_i \), and we say that \( \text{ITV} \) is an \textit{i-green interval} (or \( g_i \)-interval) if the value of any packet sent by GREEDY during \( \text{ITV} \) is in \( V_i^\prime \) or \( \text{ITV} \) contains send events at which GREEDY sends no packets. Partition sequence \( \sigma \) of events into \( r_i \)-intervals and \( g_i \)-intervals such that no two consecutive intervals are of the same color. It is easy to see that this partition is feasible. From the definition of GREEDY, we have the following observation:

**Observation 3.1 [2] Observation 2.5:** For any \( g_i \)-interval and any \( j \in [i, m] \), each \( v_j \)-queue of the algorithm GREEDY is empty and no \( v_j \)-packets arrive.

For any \( j \in [i, m] \), let \( A_j(\text{ITV}) \) and \( A_j^*(\text{ITV}) \) be the total number of \( v_j \)-packets accepted by GREEDY and \( \text{OPT} \) in \( \text{ITV} \), respectively. Let \( \mathcal{R}_i \) be the set of all \( r_i \)-intervals. From Observation 3.1 it follows that

\[
A_j = \sum_{\text{ITV} \in \mathcal{R}_i} A_j(\text{ITV}); \quad A_j^* = \sum_{\text{ITV} \in \mathcal{R}_i} A_j^*(\text{ITV}).
\]

So it suffices to show Lemma 3.1 for each \( r_i \)-interval \( \text{ITV} \in \mathcal{R}_i \), i.e., for an arbitrarily fixed \( \text{ITV} \in \mathcal{R}_i \),

\[
\sum_{j=i+1}^{m} \{A_j^*(\text{ITV}) - A_j(\text{ITV})\} \leq \sum_{j=i}^{m-1} A_j(\text{ITV}).
\]

Let \( e_1, e_2, \ldots, e_h \) be events in an arbitrarily fixed \( \text{ITV} \in \mathcal{R}_i \). For GREEDY, we use \( \delta_j(e_h) \) to denote the total number of \( v_j \)-packets sent by GREEDY until the event \( e_h \) of \( \text{ITV} \) and \( b_j(e_h) \) to denote the number of
packets contained in \(v_j\)-queue of \textsc{greedy} just after the event \(e_k\) of \(\text{itv}\). For \textsc{opt}, we use \(\delta_j^+(e_k)\) to denote the total number of \(v_j\)-packets sent by \textsc{opt} until the event \(e_k\) of \(\text{itv}\) and \(b_j^+(e_k)\) to denote the number of packets contained in \(v_j\)-queue of \textsc{opt} just after the event \(e_k\) of \(\text{itv}\). Note that \(\delta_j^+(e_k)\) denotes the total number of send events until the event \(e_k\) at which \textsc{opt} sends no packets. For each \(j \in [i, m]\), it is immediate from Observation 3.1 that for \textsc{greedy}, \(\text{itv}\) starts with \(v_j\)-queue empty and ends with \(v_j\)-queue empty. Since no further \(v_j\)-packets arrive in \(\text{itv}\) after the (final) event \(e_k\) of \(\text{itv}\), we have that

\[
A_j(\text{itv}) = \delta_j(e_k) + b_j^+(e_k) = \delta_j(e_k).
\]  

(2)

Let \(r_j(e_k|\text{g}, \text{o})\) be the total number of \(v_j\)-packets that are accepted by \textsc{greedy} and \textsc{opt} until the event \(e_k\) of \(\text{itv}\), \(r_j(e_k|\text{g}, \overline{\text{g}}, \text{o})\) be the total number of \(v_j\)-packets that are accepted by \textsc{greedy} and are rejected by \textsc{opt} until the event \(e_k\) of \(\text{itv}\), \(r_j(e_k|\overline{\text{g}}, \overline{\text{g}}, \text{o})\) be the total number of \(v_j\)-packets that are rejected by \textsc{greedy} and are accepted by \textsc{opt} until the event \(e_k\) of \(\text{itv}\), and \(r_j(e_k|\overline{\text{g}}, \overline{\text{g}}, \text{o})\) be the total number of \(v_j\)-packets that are rejected by \textsc{greedy} and \textsc{opt} until the event \(e_k\) of \(\text{itv}\). Then from the facts that \(A_j(\text{itv}) = r_j(e_k|\text{g}, \text{o}) + r_j(e_k|\text{g}, \overline{\text{g}}, \text{o})\) and \(A_j^+(\text{itv}) = r_j(e_k|\text{g}, \text{o}) + r_j(e_k|\text{g}, \overline{\text{g}}, \text{o})\), it follows that

\[
A_j(\text{itv}) - A_j^+(\text{itv}) = r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}) - r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}).
\]  

(3)

Thus to prove that Equation (1) holds, it suffices to show that

\[
\varphi(e_k) = \sum_{j=i+1}^{m} A_j(\text{itv}) + \sum_{j=i}^{m-1} \{ A_j(\text{itv}) - A_j^+(\text{itv}) \} = \sum_{j=i+1}^{m} \delta_j(e_k) + \sum_{j=i}^{m-1} \{ r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}) - r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}) \} \geq 0.
\]  

(4)

where the second equality follows from Equations (2) and (3).

For each \(j \in [1, m]\), we say that send event \(e\) is \((i, j)\)-selecting if \textsc{greedy} sends a \(v_i\)-packet and \textsc{opt} sends a \(v_j\)-packet at the send event \(e\), and say that send event \(e\) is \((i, 0)\)-selecting if \textsc{greedy} sends a \(v_i\)-packet and \textsc{opt} sends no packets at the send event \(e\). For each \(j \in [0, m]\), let \(\Delta_{i,j}(e_k)\) be the total number of \((i, j)\)-selecting send events until the event \(e_k\) of \(\text{itv}\). To show that Equation (4) holds, the following claims are crucial. Let \(N = \sum_{j=i}^{m} \delta_j(e_k)\) be the total number of send events in \(\text{itv}\) in \(\mathcal{R}_i\).

Claim 3.1: For each \(j \in [i, m-1]\), \(r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}) - r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}) \geq \Delta_{i,j}(e_k) - \delta_j^+(e_k)\).

Claim 3.2: \(N \geq \sum_{j=0}^{i-1} \Delta_{i,j}(e_k) + \sum_{j=0}^{m-1} \delta_j^+(e_k) + \Delta_{i,m}(e_k)\).

The proofs of Claims 3.1 and 3.2 are given in Sections 3.2.1 and 3.2.2 respectively. From Claims 3.1 and 3.2 we can immediately derive Equation (1) as follows:

\[
\varphi(e_k) = \sum_{j=i+1}^{m} \delta_j(e_k) + \sum_{j=i}^{m-1} \{ r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}) - r_j(e_k|\text{g}, \overline{\text{g}}, \text{o}) \} \geq \sum_{j=i+1}^{m} \delta_j(e_k) + \sum_{j=i}^{m-1} \{ \Delta_{i,j}(e_k) - \delta_j^+(e_k) \} = N - \delta_i(e_k) + \sum_{j=i}^{m-1} \{ \Delta_{i,j}(e_k) - \delta_j^+(e_k) \},
\]  

(5)

where the first inequality follows from Claim 3.1 and the second equality follows from the fact that \(N = \sum_{j=i}^{m} \delta_j(e_k)\). Note that \(\delta_i(e_k) = \sum_{j=0}^{i-1} \Delta_{i,j}(e_k)\). Then from Equation (5), it follows that

\[
\varphi(e_k) \geq N - \sum_{j=0}^{i-1} \Delta_{i,j}(e_k) + \sum_{j=i}^{m-1} \{ \Delta_{i,j}(e_k) - \delta_j^+(e_k) \} = N - \sum_{j=0}^{i-1} \Delta_{i,j}(e_k) - \Delta_{i,m}(e_k) - \sum_{j=i}^{m-1} \delta_j^+(e_k) \geq 0,
\]
3.2 Proofs of Claims

3.2.1 Proof of Claim 3.1

For each $h \in [1, k]$, we use $\alpha_j(e_h) \geq 0$ to denote the margin of $v_j$-queue at the event $e_h$, i.e.,

$$\alpha_j(e_h) = \max \left\{0, b_j(e_h) - b_j^+(e_h)\right\} = \begin{cases} b_j(e_h) - b_j^+(e_h), & b_j(e_h) > b_j^+(e_h); \\ 0, & b_j(e_h) \leq b_j^+(e_h). \end{cases}$$

(6)

Note that $\alpha_j(e_h) \geq 0$ by definition. Since $b_j(e_k) = 0$ by Observation 3.1, we have that $\alpha_j(e_k) = 0$. Then to prove that $r_j(e_k|G, \overline{O}) - r_j(e_k|\overline{G}, O) \geq \Delta_{i,j}(e_k) - \delta_j^+(e_k)$, it suffices to show that for each $h \in [1, k]$,

$$r_j(e_h|G, \overline{O}) - r_j(e_h|\overline{G}, O) \geq \Delta_{i,j}(e_h) - \delta_j^+(e_h) + \alpha_j(e_h).$$

(7)

For an arbitrarily fixed $j \in [i, m - 1]$, we derive Equation (7) by induction on $h \in [1, k]$.

Base Step: From the definition of $\mathbf{RT} \in R_i$, it follows that $e_1$ is arrive event, and from Observation 3.1 it follows that $v_j$-queue of GREEDY is empty just before the event $e_1$ for each $\ell \in [i, m]$. Assume that a $v_s$-packet arrives at the event $e_1$. Let us consider the following cases: (a) $s = j$ and (b) $s \neq j$.

(a) $s = j$: Since $v_j$-queue of GREEDY is empty just before the event $e_1$, GREEDY accepts a $v_j$-packet at the event $e_1$. So it is obvious that $r_j(e_1|G, \overline{O}) \geq 0$, $r_j(e_1|\overline{G}, O) = 0$, and $b_j(e_1) = 1$. Since $e_1$ is arrive event, we have that $\Delta_{i,j}(e_1) = \delta_j^+(e_1) = 0$. We claim that $\alpha_j(e_1) = 0$. If OPT accepts a $v_j$-packet at the event $e_1$, then we have that $b_j^+(e_1) \geq 1 = b_j(e_1)$, and if OPT rejects a $v_j$-packet at the event $e_1$, then we have that $b_j^+(e_1) = B_j \geq 1 = b_j(e_1)$. Thus in Case (a), it follows that Equation (7) holds for $h = 1$.

(b) $s \neq j$: Since $v_j$-queue of GREEDY is empty just before the event $e_1$ and no $v_j$-packets arrive at the event $e_1$, we have that $r_j(e_1|G, \overline{O}) = r_j(e_1|\overline{G}, O) = b_j(e_1) = 0$. From the fact that $e_1$ is arrive event, it follows that $\Delta_{i,j}(e_1) = \delta_j^+(e_1) = 0$. Since $b_j(e_1) = 0$, we have that $b_j^+(e_1) \geq b_j(e_1)$, i.e., $\alpha_j(e_1) = 0$. Thus in Case (b), it follows that Equation (7) holds for $h = 1$.

Induction Step: For any $\ell \in [2, k]$, we assume that Equation (7) holds for $h = \ell - 1$, i.e.,

$$r_j(e_{\ell-1}|G, \overline{O}) - r_j(e_{\ell-1}|\overline{G}, O) \geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1}).$$

(8)

For the event $e_{\ell}$, let us consider the following cases: (c) $e_{\ell}$ is arrive event and (d) $e_{\ell}$ is send event.

(c) $e_{\ell}$ is arrive event: Assume that a $v_s$-packet arrives at the event $e_{\ell}$. Since $e_{\ell}$ is arrive event, it is immediate that $\Delta_{i,j}(e_{\ell}) = \Delta_{i,j}(e_{\ell-1})$ and $\delta_j^+(e_{\ell}) = \delta_j^+(e_{\ell-1})$. If $s \neq j$, then $r_j(e_{\ell}|G, \overline{O}) = r_j(e_{\ell-1}|G, \overline{O})$, $r_j(e_{\ell}|\overline{G}, O) = r_j(e_{\ell-1}|\overline{G}, O)$, and $\alpha_j(e_{\ell}) = \alpha_j(e_{\ell-1})$ hold. Thus from Equation (8), it follows that Equation (7) holds for $h = \ell$. So we assume that $s = j$ and let us consider the following cases: (c-1) both GREEDY and OPT accept the $v_j$-packet; (c-2) both GREEDY and OPT reject the $v_j$-packet; (c-3) GREEDY rejects and OPT accepts the $v_j$-packet; (c-4) GREEDY accepts and OPT rejects the $v_j$-packet.

For Case (c-1), GREEDY and OPT accept the $v_j$-packet at the event $e_{\ell}$. So we have that $r_j(e_{\ell}|G, \overline{O}) = r_j(e_{\ell-1}|G, \overline{O})$, $r_j(e_{\ell}|\overline{G}, O) = r_j(e_{\ell-1}|\overline{G}, O)$, $b_j(e_{\ell}) = b_j(e_{\ell-1}) + 1$, and $b_j^+(e_{\ell}) = b_j^+(e_{\ell-1}) + 1$. This implies that $\alpha_j(e_{\ell}) = \alpha_j(e_{\ell-1})$. Thus from Equation (8), it follows that

$$r_j(e_{\ell}|G, \overline{O}) - r_j(e_{\ell}|\overline{G}, O) = r_j(e_{\ell-1}|G, \overline{O}) - r_j(e_{\ell-1}|\overline{G}, O) \geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1}) = \Delta_{i,j}(e_{\ell}) - \delta_j^+(e_{\ell}) + \alpha_j(e_{\ell}).$$

For Case (c-2), GREEDY and OPT reject the $v_j$-packet at the event $e_{\ell}$. Then we have that $r_j(e_{\ell}|G, \overline{O}) = r_j(e_{\ell-1}|G, \overline{O})$, $r_j(e_{\ell}|\overline{G}, O) = r_j(e_{\ell-1}|\overline{G}, O)$, $b_j(e_{\ell}) = b_j(e_{\ell-1})$, and $b_j^+(e_{\ell}) = b_j^+(e_{\ell-1})$. This immediately
implies that \( \alpha_j(e_\ell) = \alpha_j(e_{\ell-1}) \). Thus from Equation (S), it follows that
\[
\begin{align*}
r_j(e_\ell|G, \overline{\Theta}) - r_j(e_\ell|\overline{\Theta}, o) &= r_j(e_{\ell-1}|G, \overline{\Theta}) - r_j(e_{\ell-1}|\overline{\Theta}, o) \\
&\geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1}) \\
&= \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + \alpha_j(e_\ell).
\end{align*}
\]

For Case (c-3), GREEDY rejects and OPT accepts the \( v_j \)-packet at the event \( e_\ell \). So it is easy to see that \( r_j(e_\ell|G, \overline{\Theta}) = r_j(e_{\ell-1}|G, \overline{\Theta}) \), \( r_j(e_\ell|\overline{\Theta}, o) = r_j(e_{\ell-1}|\overline{\Theta}, o) + 1 \), \( b_j(e_\ell) = b_j(e_{\ell-1}) = B_j \), \( b_j^+(e_\ell) = b_j^+(e_{\ell-1}) + 1 \leq B_j \), and \( \alpha_j(e_{\ell-1}) = b_j(e_{\ell-1}) - b_j^+(e_{\ell-1}) \geq 1 \). This implies that
\[
\alpha_j(e_\ell) = b_j(e_\ell) - b_j^+(e_\ell) = b_j(e_{\ell-1}) - b_j^+(e_{\ell-1}) - 1 = \alpha_j(e_{\ell-1}) - 1 \geq 0.
\]

Thus from Equation (S), it follows that
\[
\begin{align*}
r_j(e_\ell|G, \overline{\Theta}) - r_j(e_\ell|\overline{\Theta}, o) &= r_j(e_{\ell-1}|G, \overline{\Theta}) - r_j(e_{\ell-1}|\overline{\Theta}, o) - 1 \\
&\geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1}) - 1 \\
&= \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + \alpha_j(e_\ell).
\end{align*}
\]

For Case (c-4), GREEDY accepts and OPT rejects the \( v_j \)-packet at the event \( e_\ell \). So it is immediate to see that \( r_j(e_\ell|G, \overline{\Theta}) = r_j(e_{\ell-1}|G, \overline{\Theta}) + 1 \), \( r_j(e_\ell|\overline{\Theta}, o) = r_j(e_{\ell-1}|\overline{\Theta}, o) \), \( b_j(e_\ell) = b_j(e_{\ell-1}) + 1 \leq B_j \), \( b_j^+(e_\ell) = b_j^+(e_{\ell-1}) = B_j \), and \( b_j(e_{\ell-1}) - b_j^+(e_{\ell-1}) \leq -1 \). This implies that
\[
b_j(e_\ell) - b_j^+(e_\ell) = b_j(e_{\ell-1}) + 1 - b_j^+(e_{\ell-1}) \leq 0,
\]
and we have that \( \alpha_j(e_{\ell-1}) = \alpha_j(e_\ell) = 0 \). Thus from Equation (S), it follows that
\[
\begin{align*}
r_j(e_\ell|G, \overline{\Theta}) - r_j(e_\ell|\overline{\Theta}, o) &= r_j(e_{\ell-1}|G, \overline{\Theta}) + 1 - r_j(e_{\ell-1}|\overline{\Theta}, o) \\
&\geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1}) + 1 \\
&= \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + \alpha_j(e_\ell).
\end{align*}
\]

Hence in Case (c), we have that Equation (A) holds for \( h = \ell \).

(d) \( e_\ell \) is send event: Let \( v_x \) and \( v_y \) be the values of packets sent by GREEDY and OPT at the event \( e_\ell \), respectively. We consider the following cases: (d-1) \( y \neq j \); (d-2) \( y = j \) and \( x \neq i \); (d-3) \( y = j \) and \( x = i \). Since \( e_\ell \) is send event, we have that \( r_j(e_\ell|G, \overline{\Theta}) = r_j(e_{\ell-1}|G, \overline{\Theta}) \) and \( r_j(e_\ell|\overline{\Theta}, o) = r_j(e_{\ell-1}|\overline{\Theta}, o) \).

For Case (d-1), OPT does not send a \( v_j \)-packet at the event \( e_\ell \). It is obvious that \( b_j^+(e_\ell) = b_j^+(e_{\ell-1}) \), \( \Delta_{i,j}(e_\ell) = \Delta_{i,j}(e_{\ell-1}) \), \( \delta_j^+(e_\ell) = \delta_j^+(e_{\ell-1}) \), and \( b_j(e_\ell) \leq b_j(e_{\ell-1}) \). This implies that \( \alpha_j(e_\ell) \leq \alpha_j(e_{\ell-1}) \). Thus from Equation (S), it follows that
\[
\begin{align*}
r_j(e_\ell|G, \overline{\Theta}) - r_j(e_\ell|\overline{\Theta}, o) &= r_j(e_{\ell-1}|G, \overline{\Theta}) - r_j(e_{\ell-1}|\overline{\Theta}, o) \\
&\geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1}) \\
&= \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + \alpha_j(e_\ell).
\end{align*}
\]

For Case (d-2), OPT sends a \( v_j \)-packet at the event \( e_\ell \). It is obvious that \( \delta_j^+(e_\ell) = \delta_j^+(e_{\ell-1}) + 1 \), \( b_j^+(e_\ell) = b_j^+(e_{\ell-1}) - 1 \), and \( b_j(e_\ell) \leq b_j(e_{\ell-1}) \), and it follows that \( \alpha_j(e_\ell) \leq \alpha_j(e_{\ell-1}) + 1 \). Since GREEDY does not send a \( v_i \)-packet at the event \( e_\ell \), we have that \( \Delta_{i,j}(e_\ell) = \Delta_{i,j}(e_{\ell-1}) \). From Equation (S), it follows that
\[
\begin{align*}
r_j(e_\ell|G, \overline{\Theta}) - r_j(e_\ell|\overline{\Theta}, o) &= r_j(e_{\ell-1}|G, \overline{\Theta}) - r_j(e_{\ell-1}|\overline{\Theta}, o) \\
&\geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1}) \\
&= \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + 1 + \alpha_j(e_{\ell-1}) \\
&\geq \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + \alpha_j(e_\ell).
\end{align*}
\]
For Case (d-3), we further consider the following cases: (d-3.1) $i = j$ and (d-3.2) $i < j$. For Case (d-3.1), both GREEDY and OPT sends a $v_j$-packet at the event $e_\ell$. Then it is immediate that $b_j(e_\ell) = b_j(e_{\ell-1}) - 1$, $b_j^+(e_\ell) = b_j^+(e_{\ell-1}) - 1$, $\delta_j^+(e_\ell) = \delta_j^+(e_{\ell-1}) + 1$, and $\Delta_{i,j}(e_\ell) = \Delta_{i,j}(e_{\ell-1}) + 1$. This implies that $\alpha_j(e_\ell) = \alpha_j(e_{\ell-1})$ by definition. Thus from Equation (7), it follows that

$$r_j(e_\ell|G, \overline{\sigma}) - r_j(e_\ell|G, \overline{o}) = r_j(e_{\ell-1}|G, \overline{\sigma}) - r_j(e_{\ell-1}|G, \overline{o})$$

\[
\geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1})
\]

\[
= \Delta_{i,j}(e_\ell) - 1 - \delta_j^+(e_{\ell-1}) + \alpha_j(e_\ell)
\]

\[
= \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + \alpha_j(e_\ell).
\]

For Case (d-3.2), GREEDY sends a $v_i$-packet and OPT sends a $v_j$-packet at the event $e_\ell$. It is immediate that $\delta_j^+(e_\ell) = \delta_j^+(e_{\ell-1}) + 1$ and $\Delta_{i,j}(e_\ell) = \Delta_{i,j}(e_{\ell-1}) + 1$. Since $i < j$, we have that $b_j(e_{\ell-1}) = 0$ by definition (if $b_j(e_{\ell-1}) > 0$, then $v_i$ is not the highest packet value among the packets residing in queues just after the event $e_{\ell-1}$ and GREEDY does not send a $v_i$-packet at the event $e_\ell$). So it follows that $b_j(e_\ell) = b_j(e_{\ell-1}) = 0$ and this implies that $\alpha_j(e_\ell) = 0 \leq \alpha_j(e_{\ell-1})$. Thus from Equation (8), it follows that

$$r_j(e_\ell|G, \overline{\sigma}) - r_j(e_\ell|G, \overline{o}) = r_j(e_{\ell-1}|G, \overline{\sigma}) - r_j(e_{\ell-1}|G, \overline{o})$$

\[
\geq \Delta_{i,j}(e_{\ell-1}) - \delta_j^+(e_{\ell-1}) + \alpha_j(e_{\ell-1})
\]

\[
= \Delta_{i,j}(e_\ell) - 1 - \delta_j^+(e_{\ell-1}) + \alpha_j(e_\ell)
\]

\[
= \Delta_{i,j}(e_\ell) - \delta_j^+(e_\ell) + \alpha_j(e_\ell).
\]

Hence in Case (d), we have that Equation (7) holds for $h = \ell$.

### 3.2.2 Proof of Claim 3.2

Since $\Delta_{i,j}(e_k)$ is the total number of $(i, j)$-selecting send events in ITV and $\delta_j^+(e_k)$ is the total number of $v_j$-packets sent by OPT in ITV, we have that $\Delta_{i,j}(e_k) \leq \delta_j^+(e_k)$ for each $j \in [0, m]$. Thus it follows that

$$\sum_{j=0}^{i-1} \Delta_{i,j}(e_k) + \sum_{j=i}^{m-1} \delta_j^+(e_k) + \Delta_{i,m}(e_k) \leq \sum_{j=0}^{i-1} \delta_j^+(e_k) + \sum_{j=i}^{m-1} \delta_j^+(e_k) + \delta_m^+(e_k)$$

$$= \sum_{j=0}^{m} \delta_j^+(e_k) = N,$$

where the second equality follows from the fact that $N$ is the total number of send events in ITV.

### 4 Lower Bounds

In this section, we derive lower bounds for the competitive ratio of the algorithm GREEDY, which shows that the competitive ratio of GREEDY cannot improve any more.

**Theorem 4.1:** For $m$ packet values $0 < v_1 < v_2 < \cdots < v_m$ and any $\varepsilon > 0$, the competitive ratio of the algorithm GREEDY cannot be less than $1 + r - \varepsilon$ for the case that $m$ queues do not necessarily have the same size, where $r = \max_{i \in [1, m-1]} v_i/v_{i+1}$.

**Proof:** To derive lower bounds for the competitive ratio of GREEDY for the case that $m$ queues do not necessarily have the same size, define a sequence $\sigma$ as follows: The sequence $\sigma$ consists of $m$ phases. The phase $P_1$ includes $B_m$ time slots. In the 1st time slot of the phase $P_1$, $B_1$ copies of $v_1$-packet arrive, $B_2$ copies of $v_2$-packet arrive, $\ldots$, and $B_m$ copies of $v_m$-packet arrive. For each $i \in [2, B_m]$, a $v_{m-1}$-packet arrives in the $i$th time slot of the phase $P_1$. For each $j \in [2, m]$, the phase $P_j$ includes $B_{m+1-j}$ time slots.
In the 1th time slot of the phase $P_j$, a $v_{m+1-j}$-packet arrives. For each $i \in [2, B_{m+1-j}]$, a $v_{m-j}$-packet arrives in the $i$th time slot of the phase $P_j$. Regard $v_0$-packet as a null packet and this implies that no packets arrive in the $i$th time slot of the phase $P_m$ with $i \in [2, B_1]$.

On the sequence $\sigma$, the behavior of GREEDY is given in Figure[1]. From the definition of GREEDY, it is immediate that $B_m$ copies of $v_m$-packets are sent in the phase $P_1$, $B_{m-1}$ copies of $v_{m-1}$-packets are sent in the phase $P_2$, . . . , and $B_1$ copies of $v_1$-packets are sent in the phase $P_{m}$. For the queues of GREEDY, we observe that for each $j \in [1, m]$, $v_1$-queue, . . . , $v_{m-j}$-queue are full and $v_{m-j+1}$-queue, . . . , $v_m$-queue are empty at the end of the phase $P_j$. Thus for the benefit $\text{GREEDY}(\sigma)$, it follows that

$$\text{GREEDY}(\sigma) = B_1v_1 + B_2v_2 + \cdots + B_{m-1}v_{m-1} + B_mv_m.$$  

We consider the following offline algorithm Adv (on the sequence $\sigma$, the behavior of Adv is given in Figure[2]. For each $j \in [1, m-1]$ and each $i \in [1, B_{m+1-j}]$, Adv sends a $v_{m-j}$-packet at the end of the $i$th time slot of the phase $P_j$. For the queues of Adv, we observe that for each $j \in [1, m]$, every queue is full just before the send event in the 1st time slot of the phase $P_j$. Then it follows that Adv sends $B_m$ copies of $v_{m-1}$-packets in the phase $P_1$, $B_{m-1}$ copies of $v_{m-2}$-packets in the phase $P_2$, . . . , and $B_2$ copies of $v_1$-packets in the phase $P_{m-1}$. In particular, we have that just after the arrive event $e_s$ in the 1st time slot of the phase $P_m$, every queue of Adv is full and no further packets arrive. This implies that after the arrive event $e_s$ in the 1st time slot of the phase $P_m$, Adv sends $B_1$ copies of $v_1$-packets, $B_2$ copies of $v_2$-packets, . . . , and $B_m$ copies of $v_m$-packets. Thus for the benefit $\text{OPT}(\sigma)$, we have that

$$\text{OPT}(\sigma) \geq \frac{\text{Adv}(\sigma)}{\text{GREEDY}(\sigma)} \geq \frac{B_1v_1 + B_2v_2 + \cdots + (v_\ell + v_{\ell+1}) + \cdots + B_{m-1}v_{m-1} + B_mv_m}{B_1v_1 + B_2v_2 + \cdots + \ell + \ell + \cdots + B_{m-1}v_{m-1} + B_mv_m}.$$  

Assume that $r = v_\ell/v_{\ell+1} = \max_{i \in [1, m-1]} v_i/v_{i+1}$ for some $\ell \in [1, m-1]$. Note that

$$\lim_{B_{\ell+1} \to \infty} \frac{B_1v_1 + B_2v_2 + \cdots + (v_\ell + v_{\ell+1}) + \cdots + B_{m-1}v_{m-1} + B_mv_m}{B_1v_1 + B_2v_2 + \cdots + \ell + \ell + \cdots + B_{m-1}v_{m-1} + B_mv_m} = \frac{\ell + \ell + \cdots + B_{m-1}v_{m-1} + B_mv_m}{\ell + \ell + \cdots + B_{m-1}v_{m-1} + B_mv_m} = 1 + \frac{v_\ell}{v_{\ell+1}} = 1 + r.$$  

This implies that for any $\varepsilon > 0$, the competitive ratio of GREEDY cannot be less than $1 + r - \varepsilon$. ■

References

[1] W.A. Aiello, Y. Mansour, S. Rajagopalan, and A. Rosen. Competitive Queue Policies for Differential Services. In Proc. of IEEE INFOCOM, pp.431–440 (2000).

[2] K. Al-Bawani and A. Souza. Buffer Overflow Management with Class Segregation. Information Processing Letters 113(4), pp.145-150, 2013.
[3] S. Albers and M. Schmidt. On the Performance of Greedy Algorithms in Packet Buffering. In *Proc. of the 36th Annual ACM Symposium on Theory of Computing* pp.35–44 (2004).

[4] Y. Azar. Online packet Switching. In *Proc. of the 2nd Workshop on Approximation and Online Algorithms*, Lecture Notes in Computer Science 3351, pp.1-5 (2004).

[5] Y. Azar and A. Litichevskey. Maximizing Throughput in Multi-Queue Switches. In *Proc. of the 12th European Symposium on Algorithms*, Lecture Notes in Computer Science 3221, pp.53-64 (2004).

[6] Y. Azar and Y. Richter. Management of Multi-Queue Switches in QoS Networks. In *Proc. of the 35th Annual ACM Symposium on Theory of Computing*, pp.82–89 (2003).

[7] Y. Azar and Y. Richter. The Zero-One Principle for Switching Networks. In *Proc. of the 36th Annual ACM Symposium on Theory of Computing*, pp.64–71 (2004).

[8] M. Bienkowski. An Optimal Lower Bound for Buffer Management in Multi-Queue Switches. In *Proc. of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms*, pp.1295–1305 (2011).

[9] M. Bienkowski and A. Mardy. Geometric Aspects of Online Packet Buffering: An Optimal Randomized Algorithm for Two Buffers. In *Proc. of the 8th Latin American Symposium on Theoretical Informatics*, Lecture Notes in Computer Science 4857 pp.252-263 (2008).

[10] A. Borodin and R. El-Yaniv. Online Computation and Competitive Analysis. Cambridge University Press (1998).

[11] M. Chrobak. Online Algorithms Column 13. *ACM SIGACT News*, 39(3), pp.96-121 (2008).

[12] Cisco Systems, Inc. Quality of Service (QoS). Internetworking Technology Handbook, Chapter 49, 2006.

[13] L. Epstein and R. van Stee. Buffer Management Problems. *ACM SIGACT News*, 35(3), pp.58-66 (2004).

[14] M.H. Goldwasser. A Survey of Buffer Management Policies for Packet Switches. *ACM SIGACT News*, 41(1), pp.100-128 (2010).

[15] T. Itoh and N. Takahashi. Competitive Analysis of Multi-Queue Preemptive QoS Algorithms for General Priorities. *IEICE Trans. on Fundamentals of Electronics, Communications, and Computer Science*, E89-A(5), pp.1186-1197 (2006).

[16] W. Jawor. Three Dozen Papers on Online Algorithms. *ACM SIGACT News*, Volume 36, Issue 1, pp.71-85 (2004).

[17] A. Kesselman, Z. Lotker, Y. Mansour, B. Patt-Shamir, B. Schieber, and M. Sviridenko. Buffer Overflow Management in QoS Switches. In *Proc. of the 33rd Annual ACM Symposium on Theory of Computing*, pp.520–529 (2001).

[18] G. Li and H. Liu. Dynamic Resource Allocation with Finite Buffer Constraints in Broadband OFDMA Networks. In *Proc. of IEEE Wireless Communications and Networking Conference*, pp.1037-1042, 2003.

[19] Y. Mansour, B. Patt-Shamir, and O. Lapid. Optimal Smoothing Schedules for Real-Time Streams. In *Proc. of the 19th ACM Symposium on Principles of Distributed Computing*, pp.21.29 (2000).
A Behavior of GREEDY

The following figure shows the behavior and the queue state of GREEDY on the sequence $\sigma$.

$P_1$
- time slot 1
- arrival: $v_1$-packet $\times B_1$, $v_2$-packet $\times B_2$, ..., $v_m$-packet $\times B_m$
- send: $v_m$-packet

$P_2$
- time slot 1
- arrival: $v_{m-1}$-packet
- send: $v_{m-1}$-packet

$P_3$
- time slot 1
- arrival: $v_{m-2}$-packet
- send: $v_{m-2}$-packet

$P_{m-1}$
- time slot 1
- arrival: $v_2$-packet
- send: $v_2$-packet

$P_m$
- time slot 1
- arrival: $v_1$-packet
- send: $v_1$-packet

Figure 1: Behavior of GREEDY on $\sigma$
B Behavior of ADV

The following figure shows the behavior and the queue state of ADV on the sequence $\sigma$.

\begin{figure}[h]
\centering
\begin{align*}
P_1 & \quad \begin{cases}
\text{time slot} & 1 \\
\text{arrival: } v_1\text{-packet} \\
\text{send: } v_{m-1}\text{-packet}
\end{cases} \\
\text{time slots} & 2 \sim B_m \\
\text{arrival: } v_{m-1}\text{-packet} \\
\text{send: } v_{m-1}\text{-packet} \\

P_2 & \quad \begin{cases}
\text{time slot} & 1 \\
\text{arrival: } v_{m-1}\text{-packet} \\
\text{send: } v_{m-2}\text{-packet}
\end{cases} \\
\text{time slots} & 2 \sim B_{m-1} \\
\text{arrival: } v_{m-2}\text{-packet} \\
\text{send: } v_{m-2}\text{-packet} \\

P_3 & \quad \begin{cases}
\text{time slot} & 1 \\
\text{arrival: } v_{m-2}\text{-packet} \\
\text{send: } v_{m-3}\text{-packet}
\end{cases} \\
\text{time slots} & 2 \sim B_{m-2} \\
\text{arrival: } v_{m-3}\text{-packet} \\
\text{send: } v_{m-3}\text{-packet}
\end{align*}

\vdots

\begin{align*}
P_{m-1} & \quad \begin{cases}
\text{time slot} & 1 \\
\text{arrival: } v_2\text{-packet} \\
\text{send: } v_1\text{-packet}
\end{cases} \\
\text{time slots} & 2 \sim B_2 \\
\text{arrival: } v_1\text{-packet} \\
\text{send: } v_1\text{-packet} \\

P_m & \quad \begin{cases}
\text{time slot} & 1 \\
\text{arrival: } v_1\text{-packet} \\
\text{send: } v_1\text{-packet}
\end{cases} \\
\text{time slots} & 2 \sim B_1 \\
\text{arrival: } \_ \\
\text{send: } v_1\text{-packet}
\end{align*}

\begin{align*}
P_1^* & \quad \begin{cases}
\text{time slots} & 1 \sim B_2 \\
\text{arrival: } \_ \\
\text{send: } v_2\text{-packet}
\end{cases} \\
\end{align*}

\begin{align*}
P_2^* & \quad \begin{cases}
\text{time slots} & 1 \sim B_3 \\
\text{arrival: } \_ \\
\text{send: } v_2\text{-packet}
\end{cases} \\
\end{align*}

\vdots

\begin{align*}
P_{m-1}^* & \quad \begin{cases}
\text{time slots} & 1 \sim B_m \\
\text{arrival: } \_ \\
\text{send: } v_2\text{-packet}
\end{cases} \\
\end{align*}

\end{figure}

Figure 2: Behavior of ADV on $\sigma$