

Competitive Queue Management for Latency Sensitive Packets

Amos Fiat*

Yishay Mansour†

Uri Nadav‡

October 17, 2007

For who knows most, him loss of time most grieves

Dante Alighieri (1265-1321)
The Divine Comedy, Purgatory, Canto III

Abstract

We consider the online problem of non-preemptive queue management. An online sequence of packets arrive, each of which has an associated intrinsic value. Packets can be accepted to a FIFO queue, or discarded. The profit gained by transmitting a packet diminishes over time and is equal to its value minus the delay. This corresponds to the well known and strongly motivated Naor's model in operations research.

We give a queue management algorithm with a competitive ratio equal to the golden ratio ($\phi \approx 1.618$) in the case that all packets have the same value, along with a matching lower bound. We also derive $\Theta(1)$ upper and lower bounds on the competitive ratio when packets have different intrinsic values (in the case of differentiated services). We can extend our results to deal with more general models for loss of value over time.

Finally, we re-interpret our online algorithms in the context of selfish agents, producing an online mechanism that approximates the optimal social welfare to within a constant factor.

1 Introduction

This paper deals with latency issues arising in the context of network devices such as switches and routers. Given multiple streams of incoming packets which are to be merged and sent along the same outgoing link, limited bandwidth on the outgoing link may result in packet delay and/or packet loss.

We consider the online problem of active queue management. In our problem the input consists of an online sequence of packets arriving over time to a non-preemptive FIFO queue. Any packet placed in the queue will eventually be sent, and packets are sent in order of arrival.

We consider queuing models where the actual benefit derived from sending a packet degrades over time. The goal is to model transmission of packets bearing time-dependent data such as audio, video, and more general latency sensitive applications. The latency of a packet is a major parameter in the assessment of its true utility (e.g., TCP, where delayed messages may result in a reduction of the overall transmission protocol throughput).

Section 1.1 describes related work, including the bounded delay model, [9]. In the bounded delay model, incoming packets have step function describing the value loss, past the deadline the packet is useless, prior to the deadline it has lost no value. Unfortunately, online queue policies for FIFO queues and heterogeneous packets with deadlines result in unbounded competitive ratios. Quite likely, this is why only non-FIFO queue regimes have been studied for service with deadlines.

Our paper shows that online policies for FIFO queues are much better if one assumes more gradual value loss, (constant competitive ratio vs. unbounded ratios). So, the question arises, which is the more relevant model? One obvious reason in support of the gradual loss of value lies within human nature — people are, generally, not entirely oblivious to the time “wasted” while waiting for service, even if such service is eventually given.

Consider the example of streaming video, there is an inherent deadline imposed by the timestamp within an MPEG video packet. However, such packets will

*School of Computer Science, Tel Aviv University. Supported in part by a grant from the German-Israel Foundation and by the Israel Science Foundation. Email: fiat@cs.tau.ac.il.

†School of Computer Science, Tel Aviv University. Email: mansour@cs.tau.ac.il. The research was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, by a grant no. 1079/04 from the Israel Science Foundation, by a grant from BSF and an IBM faculty award.

‡School of Computer Science, Tel Aviv University. Supported in part by a grant from the German-Israel Foundation and by the Israel Science Foundation. Email: urinadav@cs.tau.ac.il.

typically proceed through a large network with multiple routers along the transmission path. Thus, although there is a true deadline, this deadline is only relevant with regard to the entire transmission path taken, not in the context of any specific router. Furthermore, taking no value loss prior to the deadline may imply that the packet will indeed pass through (some) of the routers, but will be discarded further down the line. A reasonable rule of thumb could be to assume that value is lost over time, and this may produce much better overall performance.

We consider several versions of this problem. The simplest case is that of homogenous packets, where all packets have the same intrinsic value upon arrival, and packets lose value linearly over time. Our results also hold for “growing impatience” rather than linear value loss, *i.e.*, the longer the packet is in the system, the more value is lost per time unit. In particular, this includes commonly considered “soft real time restrictions” such as quadratic or exponential value loss (in terms of delay).

We next deal with the more general case of heterogeneous packets, where different packets may have different intrinsic values upon arrival, bounding the competitive ratio to be some constant $3 \leq c < 5$.

Our results include:

1. For homogenous packets (equal valued) we give a lower bound of ϕ on the competitive ratio (even for randomized algorithms). We also show that the competitive ratio is $\leq \phi$ for arbitrary “growing impatience” value loss (where the 2nd derivative of the value loss function is ≥ 0).
2. For heterogeneous packets and linear value loss:
 - (a) We give a simple threshold queue policy, “doubling threshold”, with a competitive ratio of 5.25 (tight for the algorithm).
 - (b) We observe that this problem has an $O(n \log n)$ time optimal offline algorithm (improving upon the obvious matching approach).
 - (c) We show a lower bound of 3 on the competitive ratio of any deterministic online algorithm.
 - (d) For deterministic and memoryless algorithms (includes all threshold algorithms) we show a lower bound of 4.1 on the competitive ratio.
 - (e) We then study a more “sophisticated” threshold algorithm, “increasing thresholds”, and prove that it has a competitive ratio of $c < 5$.
3. Finally, we relate the issue of the online competitive ratio to an online mechanism design problem

for packets generated by selfish agents. In this case there is little reason to trust the “intrinsic value” claimed by the owner. We reinterpret our online algorithms as yielding an incentive compatible online pricing scheme for heterogeneous packets, that guarantees a constant fraction of the optimal social welfare (defined as the sum of agent utilities). To place this in context, the VCG mechanism is not applicable here in this online setting.

1.1 Related Work

1.1.1 Stochastic models Our model is a worst case analogue and generalization of Naor’s stochastic model [12] from 1969. This model has been extensively studied in operations research [8]. The model has customers (utility maximizing decision makers) arrive at a FIFO queue, inspect the queue size and decide whether to join or balk. Customer benefit from completed service is R , but there is a fixed cost per unit of delay. This literature, as in much of the queuing theory, assumes stochastic arrival process of customers (e.g., Poisson distribution) and a stochastic service time (e.g., exponential distribution).

In such settings, selfish customers will enter the queue even if their benefit from doing so is negligible. Individuals decisions are not socially optimal. Under appropriate stochastic assumptions, Naor [12] devises tolls to be levied on the selfish agents, thus effectively regulating the queue size, and hence achieving the (expected) optimal social welfare. Hassin [7] shows that optimal social welfare (sum of utilities or benefits) can be attained without tolls if the queue regime is LIFO and customers are allowed to renege from the queue after joining it.

1.1.2 Worst case models Worst case analysis of buffer control algorithms has been primarily studied in the context of limited buffer size [1, 3, 2, 9]. Related to our work, packets with expiration times have also been considered, [9, 3, 4, 11], with limited sized buffers and when packets may be sent in arbitrary order.

Aiello et al. [1] suggest and analyze a queuing model for differentiated services. In their model a queue of limited size faces a stream of packets of one of two (unchanging) values, low or high. They analyze and compare different queue policies for this problem using the competitive analysis. They show a constant competitive ratio for this problem. Andelman et al. [2] study the same model with a wide range of possible packet values. They show a competitive ratio which is polylogarithmic in the range, and their policy requires a pre-knowledge of the range.

Kesselman *et al.* [9] and Andelman *et al.* [3] study bounded delay queues where packets may be sent in arbitrary order, but each packet arrives with an expiration time. Packets that are not sent by this deadline expire and are lost. Recently, Li *et al.* [11] showed a competitive ratio of 1.854. Englert and Westermann [6] showed a 1.893-competitive memoryless algorithm and 1.828 competitive history dependent algorithm. Chin *et al.* [4] present a randomized algorithm that achieves a competitive ratio of $e/(e-1) \approx 1.582$. Chin and Fung [5] present a lower bound of $5/4$ on the competitive ratio of any randomized algorithm.

If packets have deadlines, and the deadline of a subsequent packet must be past the deadline of an earlier packet, then Li *et al.* [10] give an optimal online algorithm with a competitive ratio equal to the golden ratio.

2 Model and Notations

We consider a single, non-preemptive, queue of packets. Time is partitioned into unit length time slots, integrally aligned.

Packets arrive over time at non-integral times. Multiple packets may arrive during a single time slot, but every packet has its own unique arrival time. At the end of every time slot, and should the queue be non-empty, one (and exactly one) packet is extracted from the queue and transmitted.

A packet p is identified by its value, denoted $\text{val}(p)$ and its (distinct, non-integer) time of arrival, denoted $\text{arrive}(p)$. A packet is either rejected or inserted into the queue. A packet placed in the queue must eventually be transmitted, at the end of some time slot, which we call $\text{send}(p)$. The delay a packet experiences is the number of full time slots between arrival and transmission times of the packet, *i.e.*, $\text{delay}(p) = \lfloor \text{send}(p) - \text{arrive}(p) \rfloor$.

The delay of a packet is equal to the number of packets in the queue when it arrived (which does not include itself):

$$\begin{aligned} \text{delay}(p) &= \text{buffer size at time } \text{arrive}(p) \\ &= \lfloor \text{send}(p) - \text{arrive}(p) \rfloor. \end{aligned}$$

In general, the loss of value of a packet is some function of the packet delay. In particular, linear loss means loss proportional to the delay, we will generally assume linear loss to mean a constant of one, but our results can be easily extended to other constants of proportionality.

The adversary determines the timing of packet arrivals and the value associated with the packet. Packets are transmitted from the head of the FIFO queue at integral times. We do not explicitly assume a restriction on the capacity of the queue. However, one may assume without loss of generality that no packet will be inserted

into the queue if the delay is such that the loss is greater than the initial value.

An online queue policy specifies, for every incoming packet, as to whether to enqueue the packet or to reject it. For an online policy, the decision to accept or reject a packet may not depend on future events. An offline queue policy does the same but has prior knowledge of the entire event sequence in advance. The benefit of a queue policy on a given event sequence is the sum of the benefits of the packets that it transmits.

A memoryless queue policy determines whether to accept or reject an incoming packet based only upon the current contents of the buffer, the delay that these packets have had, and the value of the incoming packet. A special case of memoryless policies, called *threshold policies*, determine if to accept or reject while only considering the number of packets currently in the buffer, and the value of the incoming packet.

Let $B(t)$ denote the number of packets in the queue at time t . If a packet p arrives at time t , $B(t)$ is the number of packets in the queue, not counting p , (irrespective of whether p is to be accepted or not). Similarly, let $B^*(t)$ denote the number of packets in the queue of the optimal (offline) schedule.

As there may be several different optimal schedules (giving the maximal benefit), we fix a specific optimal policy, OPT, defined in the appendix.

3 Homogenous Packets

In this section we consider only sequences of homogenous packets, all having the same value $R > 0$.

OBSERVATION 3.1. *The benefit of a policy from a finite event sequence of homogenous packets with value R is*

$$\begin{aligned} & \left| \{i \in \mathbb{N} \mid i \leq t_{\max} \text{ and } B(i) > 0\} \right| \cdot R - \\ & \sum_{i \in \{1, \dots, t_{\max}\}} (B(i) - 1), \end{aligned}$$

where t_{\max} is the transmission time of the last packet transmitted (*i.e.*, the transmission time of the last packet accepted from the sequence).

In Section 3.1 we present a deterministic algorithm (queue policy) with a competitive ratio of $\phi \approx 1.618$. In Section 3.2 we show that ϕ is a lower bound on the competitive ratio of all online algorithms, even for randomized algorithms against an oblivious adversary.

3.1 An optimal online algorithm Our online policy for homogenous packets is a threshold policy.

As a motivating example, consider the following: Let B be the current size of the online buffer, consider

the $2B$ -threshold policy, i.e., accept a packet if its value is $\geq 2B$. Alternately, as all packets have value R in the homogenous case, accept a packet only if the current size of the buffer is no more than $\lfloor R/2 \rfloor$. It can be argued that the competitive ratio is close to 4. A $2B$ -threshold policy accepts about half the number of packets accepted by an optimal policy (no optimal policy would ever have more than R packets in its queue, otherwise packets with 0 or negative benefit would be queued). For every packet accepted, the $2B$ -threshold policy collects a benefit of at least $\lceil R/2 \rceil$, which is at least half the maximum benefit collected by OPT from any packet. We next show a more refined analysis of the competitive ratio.

THEOREM 3.1. *Let B be the current size of the queue. The competitive ratio of the $(B\phi^2)$ -threshold algorithm is at most $\phi \approx 1.618$.*

In our proof we use ON to denote the $(B\phi^2)$ -threshold policy while OPT denotes the optimal offline. We first perform a simple relaxation on the event sequence, defined as follows: After ON's buffer is emptied, there are no more packet arrival events until OPT's buffer is emptied as well. We derive the following claim for such restricted event sequences:

LEMMA 3.1. *For any queue policy, limiting the event sequences to be restricted event sequences does not decrease the competitive ratio.*

Proof. We consider an arbitrary event sequence σ which does not meet the "restricted" criteria, i.e., there is some time t_0 at which ON's queue is empty and the next packet arrival is at time t_1 such that the OPT queue is non-empty, $B^*(t_1) > 0$.

We change event sequence σ to σ' by adding $B^*(t_0)$ send events following time t_0 , and delaying all other events in σ by an extra $B^*(t_0)$ time units. The purpose of the additional send events is to empty OPT's buffer. We show that the ratio between OPT's benefit and the benefit for ON on σ is no smaller than the same ratio on σ' .

ON's decisions remain unchanged, since the additional send events are scheduled when ON's queue is empty. On the other hand, the optimal offline benefit cannot decrease. Keeping the same schedule produces a benefit of at least $\text{OPT}(\sigma)$, as the queue size at every send event does not increase comparing to the original. Therefore $\text{OPT}(\sigma') \geq \text{OPT}(\sigma)$ and the claim follows. ■

Once both ON and OPT have empty buffers, they are both back at their initial state. Thus, it suffices to prove an upper bound on the competitive ratio by

restricting attention to those restricted event sequences that have no arriving packets after ON empties its buffer for the first time. We assume that the sequence begins with at least one arrive event before the first send event. In the remainder of this section we restrict attention to such event sequences.

Thus, for our event sequences, the online algorithm sends one packet at every send event until its queue is empty.

We now prove Theorem 3.1. We define a potential function that measures the difference between twice the benefit of ON, and the benefit of OPT. We show inductively that the potential function is always non-negative.

We define the functions $f : R^+ \mapsto Z^+$ and $v : R^+ \mapsto Z^+$ as follows:

$$\begin{aligned} f(t) &= |\{i \in \mathbb{N} | i < t \text{ and } B(i) > 0\}| \cdot R - \\ &\quad \sum_{i \in \{1, \dots, \lfloor t \rfloor\}} (B(i) - 1), \\ v(t) &= \sum_{j=1}^{B(t)} R - (j - 1). \end{aligned}$$

Consider an event sequence where the last packet arrival event is at time t_{last} , it follows from Observation 3.1 and the definition of f and v above that for any $t > t_{\text{last}}$ the total benefit of the queue policy on the event sequence is equal to $f(t) + v(t)$. Similarly, we define the functions $f^*(\cdot), v^*(\cdot)$ where $B^*(\cdot)$ replaces $B(\cdot)$. It then follows that when both ON and OPT buffers are empty then $f(t)$ and $f^*(t)$ are respectively equal to the benefit of ON and OPT on the event sequence, and both v and v^* are zero.

CLAIM 3.1. *If $B(t) > 0$ then $\phi \cdot f(t) \geq f^*(t)$.*

Proof. Obviously $f^*(t) \leq \lfloor t \rfloor \cdot R$. If $B(t) > 0$ then for every $t' < t$ we have $B(t') > 0$, since the online buffer empties only once. Hence, $|\{i \in \mathbb{N} | t > i \text{ and } B(i) > 0\}| = \lfloor t \rfloor$. Also, ON's queue length never exceeds $(1 - 1/\phi)R$. Therefore,

$$f(t) \geq \lfloor t \rfloor \cdot R - \lfloor t \rfloor (1 - 1/\phi)R \geq 1/\phi \cdot f^*(t). \quad \blacksquare$$

We now define a potential function $\psi : R^+ \mapsto Z$ as follows:

$$\psi(t) = \phi(f(t) + v(t)) - (f^*(t) + v^*(t)).$$

Proving that $\psi(t) \geq 0$ for all $t \geq 0$ concludes the proof of Theorem 3.1: Note that the functions $f(\cdot), v(\cdot), B(\cdot)$

(and similarly, $f^*(\cdot), v^*(\cdot), B^*(\cdot)$), change only when events occur. We now use the notation that for the i 'th event in the event sequence, occurring at time t , we write $f(i)$ to indicate the value of f just after the queue policy responded to the event (instead of $f(t+\epsilon)$, for an arbitrary small ϵ). (Similarly, we use notation $B(i), v(i), f^*(i)$, etc.).

Following Lemma 3.1 we consider only restricted sequences. We prove $\psi(t) \geq 0$ for all t by induction on the number of events. The basis of the induction is at time 0 before any event occurs. Both queues are empty and the claim trivially holds. Assume the claim holds for $i-1$ events, and consider the i 'th event.

Assume the i th event is a send event. The value $f(i) = f(i-1) + R - (B(i-1) - 1)$. The queue size decreases by 1, hence, $v(B(i)) = v(B(i-1)) - (R - (B(i-1) - 1))$, and we get that $f(i) + v(i) = f(i-1) + v(i-1)$. Similarly, $f^*(i) + v^*(i) = f^*(i-1) + v^*(i-1)$, and therefore $\psi(i) = \psi(i-1)$.

Assume the i th event is packet arrival. The function f changes only after send events, hence $f(i) = f(i-1)$, and $f^*(i) = f^*(i-1)$. We consider now 3 relevant cases:

1. Both OPT and ON accept the packet,
2. ON accepts and OPT rejects, and
3. ON rejects and OPT accepts.
4. A packet which both ON and OPT reject does not affect the potential function and can be ignored.

In case 1, $B(i) = B(i-1) + 1$ and $B^*(i) = B^*(i-1) + 1$. When a packet is accepted, the queue size after packet acceptance is at most $(1-1/\phi)R+1$, hence, $v(i) = v(i-1) + R - (B(i+1) - 1) \geq v(i-1) + R/\phi$. OPT's queue size is at least 1 after the packet is accepted, so $v^*(i) = v^*(i-1) + R - (B^*(i+1) - 1) \leq v^*(i) + R$. Consequently,

$$\begin{aligned} \psi(i) - \psi(i-1) &= \\ \phi(v(i) - v(i-1)) - (v^*(i) - v^*(i-1)) &\geq 0. \end{aligned}$$

In case 2, ON's queue size increases by 1, hence, $v(i) > v(i-1)$, while OPT's queue size does not change, so $v^*(i) = v^*(i-1)$. Again, $\psi(i) > \psi(i-1)$.

In case 3, in which a packet is rejected by ON must have occurred prior to the first time ON's queue is emptied and therefore, by Claim 3.1

$$(3.1) \quad \phi f(i) \geq f^*(i).$$

Additionally, we can tell that $B(i) = B(i-1) = \lfloor (1-1/\phi)R \rfloor + 1$ since ON just rejected a packet. OPT's queue size $B^*(i) \leq R$ and therefore

$$(3.2) \quad \phi v(i) \geq v^*(i).$$

Combining equations 3.1 and 3.2 we get that $\psi(i) \geq \psi(i-1)$. We conclude that $\psi(i) \geq 0$ after every event i . This completes the proof of Theorem 3.1.

3.2 Lower Bound The following theorem shows that any online algorithm for homogenous packets has a competitive ratio of at least $\phi \approx 1.618$ and thus the $(B\phi^2)$ -threshold algorithm achieves the best competitive ratio possible.

THEOREM 3.2. *The competitive ratio of any online algorithm (deterministic or randomized) is at least $\phi \approx 1.618$.*

Proof. We derive the proof for a deterministic online algorithm and at the end sketch the extensions for the randomized case.

Let π be an online policy for queue management. We construct an input sequence $\sigma(\pi)$ in the following way. We first set a threshold at αR where $\alpha = 1/\phi^2$. At each time slot R packets arrive until the first send event after which π 's queue size is $(1/\phi^2)R = \alpha R$ or below. Subsequently, no further packets arrive. Let t_0 denote the time of the send event at which this occurs.¹

The optimal schedule accepts one packet in every slot before slot number t_0 . In the last slot the optimal schedule accepts all R packets. The optimal benefit for this sequence is therefore:

$$\text{OPT}(\sigma(\pi)) = t_0 \cdot R + R(R+1)/2.$$

We give an upper bound on the benefit of policy π on event sequence $\sigma(\pi)$ using Observation 3.1,

$$\begin{aligned} \pi(\sigma(\pi)) &\leq t_0 R - t_0 \alpha R + \alpha R \cdot R - \frac{\alpha R}{2}(\alpha R - 1) \\ &= t_0 \cdot R(1 - \alpha) + \frac{\alpha R}{2}(2R - \alpha R + 1) \\ &< t_0 \cdot R(1 - \alpha) + \frac{\alpha R}{2}(2(R+1) - \alpha(R+1)) \\ &= (1 - \alpha)t_0 R + \alpha(2 - \alpha)R(R+1)/2. \end{aligned}$$

Since $\alpha = 1/\phi^2$, we have $1 - \alpha = \alpha(2 - \alpha)$ and so,

$$\begin{aligned} \pi(\sigma(\pi)) &\leq (1 - \alpha)(t_0 R + R(R+1)/2) \\ &= (1 - \alpha)\text{OPT}(\sigma(\pi)). \end{aligned}$$

The competitive ratio of every deterministic online algorithm is therefore at least $1/(1 - \alpha) = \phi$.

For an online randomized algorithm we use a similar construction, however now the value $B(t)$ is a random

¹Notice that t_0 may be infinite if π 's queue size never gets below αR . In such a case we can take t_0 to be arbitrarily large, and a similar argument would derive the lower bound.

variable. We now define t_0 to be the first time t when $E[B(t)]$ is less than αR . The analysis of the optimum remains the same, while for the analysis of the randomized online we have a dependence on various values of $B(t)$.

In each time slot earlier than t_0 the expected queue size is at least αR and therefore the expected benefit is at most $(1 - \alpha)R$.

$$\begin{aligned}
E[\pi(\sigma(\pi))] &\leq t_0 R - t_0 \alpha R + \\
&\quad E[B(t_0) \cdot R - \frac{B(t_0)}{2}(B(t_0) - 1)] \\
&= t_0 \cdot R(1 - \alpha) + E[B(t_0)] \cdot R - \\
&\quad \frac{1}{2}(E[B^2(t_0)] - E[B(t_0)]) \\
&< t_0 \cdot R(1 - \alpha) + E[B(t_0)] \cdot R - \\
&\quad \frac{1}{2}(E^2[B(t_0)] - E[B(t_0)]) \\
&< t_0 \cdot R(1 - \alpha) + \alpha R \cdot R - \\
&\quad \frac{1}{2}((\alpha R)^2 - \alpha R) \\
&< (1 - \alpha)t_0 R + \alpha(2 - \alpha)R(R + 1)/2.
\end{aligned}$$

The first inequality follows since $E[B(t)] \geq \alpha R$ for $t < t_0$ and the second inequality follows since $E[B^2(t_0)] > E^2[B(t_0)]$. ■

Along with Theorem 3.1, these results are tight.

4 Heterogenous Packets

In this section we consider event sequences where every packet may have a different intrinsic value. The first policy we present, *doubling threshold (DT)* is a dynamic threshold policy: when the queue size is B , packets are accepted if their value is at least $2B$. I.e., if the packet benefit is at least the queue size at the time of arrival.

Later, we consider a threshold policy, *increasing threshold (IT)*, that also determines a threshold value as a function of the queue size — but this is not a linear function. Rather, it seeks to optimize the threshold so as to balance different adversarial scenarios. For clarity of exposition we primarily describe the analysis of the doubling threshold algorithm; the incremental threshold analysis is much more complicated.

We first restrict the set of event sequences, defined as follows: If DT accepts a packet p , then we reduce $\text{val}(p)$ to $2B(\text{arrive}(p))$. I.e. the value of p is reduced to the lowest possible value for which p is still accepted by DT. We derive the following claim for such restricted inputs (proof omitted).

LEMMA 4.1. *The competitive ratio attainable on restricted event sequences is equal to the competitive ratio on arbitrary event sequences.*

THEOREM 4.1. *The competitive ratio of algorithm DT is equal to 5.25.*

The proof of Theorem 4.1 is rather complex. We first give a considerably nicer proof that gives an upper bound of 8 on the competitive ratio. Then, we give a sketch of the full proof of Theorem 4.1.

We regard each packet accepted by DT as two half packets. The idea behind this proof is to map every packet accepted by OPT to ‘half’ a packet accepted by DT. We use a greedy mapping rule: an incoming packet is mapped to the first ‘available’ half packet in DT’s queue, according to the FIFO sending order. A half packet is available if and only if no packet enqueued by OPT is mapped to it, otherwise it is unavailable. If p is mapped to one of the half packets that comprise packet q , we say p is associated with q . We next show that a packet enqueued by OPT cannot be transmitted by OPT before the packet with which it is associated with is transmitted by DT.

LEMMA 4.2. *Let p be a packet in OPT’s queue and let q be a packet in DT’s queue with which p is associated. Then $\text{send}(p) \geq \text{send}(q)$.*

Proof. We prove the claim by induction on the number of packets accepted by OPT. The first packet p in a sequence is accepted by DT (the size of the buffer is 0 at this time, so every packet with positive value must be accepted). OPT also accepts a packet during this slot (otherwise it is not optimal). The first packet accepted by OPT, q , is associated with p (it may be the same packet). Both p and q are first in a FIFO buffer and are scheduled to be transmitted at the next send event, therefore $\text{send}(p) = \text{send}(q)$.

Assume the induction hypothesis for $i - 1$ packets and consider the i ’th packet p , accepted by OPT. Let B^* denote the length of OPT’s queue just prior to p being enqueued. By the induction hypothesis, there are at most B^* unavailable half packets in DT’s queue, so there are at most $\lfloor B^*/2 \rfloor$ packets scheduled to be sent before the first available packet q . Hence, $\text{send}(p) \geq \text{send}(q)$. ■

We now argue that this mapping process is well defined, i.e., it cannot be that a packet enqueued in OPT’s queue has no available packet in DT’s queue — the mapping never runs a deficit of half packets.

LEMMA 4.3. *Every packet enqueued by OPT has an available half packet in DT’s queue at the time of its arrival.*

Proof. Consider a packet p accepted at time t by OPT. If p is accepted by both OPT and DT then both

halves of p are available. Otherwise, p is accepted by OPT and rejected by DT. Its value is $\text{val}(p) < 2B(t)$ and consequently OPT's queue size before accepting p , $B^*(t) < 2B(t)$, as buffer lengths are integral $B^*(t) \leq 2B(t) - 1$. Thus, of the $2B(t)$ half packets in the DT queue, at most $2B(t) - 1$ of them are unavailable and there remains at least one available half packet. ■

Mapping each packet of OPT to a half of a packet in the DT queue is, in and of itself, insufficient to guarantee a constant competitive ratio. We could be associating two packets of large benefit with a packet of small benefit. To overcome this issue, part of the benefit from a packet p enqueued by DT is distributed to all packets currently in the DT queue at time $\text{arrive}(p)$. Instead of benefit, we now consider "credit", where the credit of a packet is sum of all benefits distributed to it. Clearly, the total benefit equals the sum of all the credits given to packets.

More precisely, when a packet p is accepted by DT, and $\text{arrive}(p) = t$, then $\text{val}(p) = 2B(t)$, and $\text{benefit}(p) = B(t)$. We shall "redistribute" some of p 's benefit as credit given to other packets. A packet p keeps only half its credit ($1/2 \cdot B(\text{arrive}(p))$) and the rest is equally distributed among the $B(\text{arrive}(p))$ packets that are already waiting in the queue, so that each gets an extra $1/2$ unit of credit.

LEMMA 4.4. *At a time t , the credit of a packet in DT's buffer, is at least $1/2 \cdot B(t)$.*

Proof. Consider a packet p in DT's FIFO queue at position i . When it entered, the queue size was at least i , so it has an initial credit of at least $1/2 \cdot i$. Also, half a credit point was given to p from each of the $B(t) - i$ packets on top of p queued afterwards. Overall, its credit is at least $1/2 \cdot B(t)$. ■

To conclude the proof that the competitive ratio of DT is ≤ 8 , consider a packet r , accepted by DT, and denote its credit by $\text{credit}(r)$. At most two packets p, q are associated with r . r 's credit at time $\text{arrive}(p)$ is at least $B(\text{arrive}(p))/2$, while the value of p , $\text{val}(p) \leq 2B(\text{arrive}(p))$, by assumption on the event sequences. Likewise, r 's credit at time $\text{arrive}(q)$ is at least $B(\text{arrive}(q))/2$ and $\text{val}(q) \leq 2B(\text{arrive}(q))$. Hence,

$$\begin{aligned} \text{credit}(r) &= \max(B(\text{arrive}(p))/2, B(\text{arrive}(q))/2) \\ &\geq (\text{val}(p) + \text{val}(q))/8. \end{aligned}$$

I.e., the credit of a packet is at least $1/8^{\text{th}}$ of the sum of values of the packets associated with it and so equal to at least $1/8^{\text{th}}$ of their benefit. Comparing the sum over packets accepted by DT and over packets accepted by OPT, completes the proof.

We derive a tight bound of 5.25 on the competitive ration of the DT algorithm using the following, more general theorem, regarding online threshold policies. This theorem is later used to give a better online algorithm for heterogenous packets.

THEOREM 4.2. *1. Upper bounds: let β be some positive real constant, and let a_0, a_1, a_2, \dots be a sequence of real values such that $a_0 = 1$, for all $i \geq 0$: $a_i \geq 2i$, and*

$$\sum_{i=0}^{\lfloor a_n \rfloor} (a_n - i) \geq \beta \sum_{i=0}^{n-1} (a_i - i).$$

Then, the threshold policy that, given a queue currently of size b , sets a threshold of a_b , has a competitive ratio $\leq \beta$.

2. Lower bounds: let β be some positive real constant, and define the sequence of real values b_0, b_1, \dots , as follows: $b_0 = 1$, and for all $i \geq 1$:

$$b_i = \min \left\{ x \in \mathbb{R}^+ \mid \sum_{j=0}^{\lfloor x \rfloor} (x - j) = \beta \sum_{j=0}^{i-1} (b_j - j) \right\}.$$

Then, if there exists some ℓ such that $b_\ell - \ell < 0$ then no memoryless competitive algorithm can achieve a competitive ratio of $\leq \beta$.

Proof. [Sketch of proof:] The 2nd claim of Theorem 4.2 is the simpler claim. Consider a sequence of packet arrivals, all of which arrive within the first time slot, and of values equal to the b_i sequence. The sequence has the property that any online algorithm *must* accept these packets so as to be $\leq \beta$ competitive. If the packet valued b_j is not accepted then the adversary can pack the buffer with packets of this value. However, if $b_j - j < 0$ then the online algorithm is only losing value by accepting this b_j valued packet.

Regarding the first claim of Theorem 4.2, one can interpret the profit to the online algorithm as being

$$\sum_t (a_{B(t)} - B(t)),$$

where $B(t)$ is the size of the online buffer at the times, t , of online packet transmissions. This follows from an analysis similar to that performed for the ϕ competitive algorithm for homogenous packets. It follows from the threshold policy that the adversary cannot accept a packet of value $\geq a_{B(t)+1}$ during time slot t . If it so happens that the most valuable packet in the adversary buffer has value $\leq a_{B(t)+1}$, then — even if the adversary

Input sequence σ	ON(σ)	OPT(σ)	Ratio
1	1	3	3
1,2,2	2	6	3
1,2,2,3,3,3	3	10	10/3
1,2,2,3,3,3,4,4,4,4	4	15	15/4
1,2,2,3,3,3,4,4,4,4,5,5,5,5,5	5	15	3

Table 1: Attaining a lower bound using the input sequence of packets with values: (1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5)

were to pack his buffer with such packets — adversary profit could not exceed

$$\leq \sum_{j=0}^{\lfloor a_{B(t)+1} \rfloor} (a_{B(t)+1} - j).$$

This implies that if all packets arrive during the 1st time slot with values equal to the a_i values, then the competitive ratio of such an algorithm is no more than β . It also follows from the construction that $a_j \leq \beta(j-1)$.

We construct a sequence, T_t , that gives an upper bound on the value of the highest valued packet within the adversary buffer. Either $T_{t+1} = T_t - 1$ (as the packets have aged by 1), or a new packet has been accepted into the adversary buffer, with

$$T_{t+1} \leq a_{B(t)+1} \leq \beta(B(t)).$$

Thus, for each time slot t where the adversary accepts a packet, we know that $T_t \leq \beta B(t)$, yet we can also associate a profit of $B(t)$ with the online algorithm at time t . For time slots t where the adversary accepts no packets, the total sum of such T_t is bounded by β times the sum of the appropriate B_t . ■

THEOREM 4.3. *The competitive ratio of any deterministic online algorithm for heterogenous packets is at least 3. The competitive ratio of any memoryless online algorithm for heterogenous packets is at least 4.1*

Proof. To prove a lower bound of 4.1 on the competitive ratio of any memoryless algorithm for heterogenous packets, we make use (again) of Theorem 4.2, and try to find the largest β for which there exists an ℓ such that $b_\ell - \ell < 0$ holds. Solving the recurrence with $\beta = 4.1$ gives $b_{46} = 45.7$.

To prove a lower bound of 3 on the competitive ratio of any online algorithm, we fix an online policy π and consider an event sequence $\sigma(\pi)$. The sequence

$\sigma(\pi)$ includes arrivals only during the first slot, and is as follows:

The first arrival is of a packet with value 1. If π doesn't accept this packet then the sequence stops. Notice that in this case, the competitive ratio is infinite, since $\text{OPT}(\sigma(\pi)) = 1$ and $\pi(\sigma(\pi)) = 0$. If the first packet is accepted, then we have two more arrivals of packets with value 2. Again, if π rejects both of them then the sequence stops. Without loss of generality, one may assume that π does not accept both the packets with value 2, otherwise it has a benefit of 0 for the second such packet.

We proceed similarly, and if one of the 2-valued packets is accepted then we add 3 arrivals of packets with value 3 and likewise 4 arrivals of packets with value 4 and 5 arrivals of packets with value 5. The sequence stops once π rejects all the packets from a certain class of values, or after the last packet with value 5.

Generally, π does not accept two or more packets with the same value. Otherwise, when the second packet with value i is accepted, there are already i packets in π 's buffer, and the benefit for this packet is 0. The optimal solution, on the other hand, accepts all and only packets with the last value offered in the sequence $\sigma(\pi)$.

Every deterministic online algorithm π must behave in one of the 5 possible scenarios, summarized in Table 1. It's easy to verify that in each of them, the ratio between $\text{OPT}(\sigma(\pi))$ and $\pi(\sigma(\pi))$ at least 3. ■

4.1 A better threshold algorithm. To obtain a competitive ratio < 5 , we use the equations of Theorem 4.2. Choose some β , fix $a_0 = 1$, and solve the set of equalities

$$\sum_{i=0}^{\lfloor a_n \rfloor} (a_n - i) = \beta \sum_{i=0}^{n-1} (a_i - i),$$

for the sequence $\{a_n\}_{n \geq 1}$

One can show that for all $\beta > 4.9$ the solution to the a_i sequence obeys $a_i > 2i$. Therefore, it follows from Theorem 4.2 that the competitive ratio is no more than β .

Our analysis can clearly be refined, and even the condition that $\alpha_i > 2i$ is not really required. We suspect that the analysis can be improved and that one can come up with an $\{a_i\}$ sequence of thresholds that result in a competitive ratio of ϕ^3 .

We call the resulting family of algorithms *incremental thresholds*, as the thresholds increase superlinearly.

5 Regulating Selfish Packets

We now point to a connection between our work and Naor’s model [12], and state our results in terms of social welfare given selfish customers. If incoming packets are regarded as selfish decision makers, then the simple threshold policy is a dominating strategy. Customers join the queue if and only if its length is less than their value. In the worst case, the stream is such that the queue size is always slightly less than the value of the incoming packet, and hence, the profit for a customer is always 0. From the perspective of [12], Theorem 3.2 can be interpreted as a proposition to levy a toll of $1/\phi \cdot R$ at the entry point of a queue. This ensures a $1/\phi$ approximation of the optimal social welfare. Theorem 3.2 tells us no other online payment scheme can achieve a better guarantee in the worst case.

In the case of heterogenous customers, the value of a packet can be regarded as its type, and the problem of levying tolls can be seen as a mechanism design issue. Theorem 4.1 can be interpreted as a payment mechanism, where the price for a customer to enter the queue equals the queue size on his arrival. Clearly, this mechanism is truthful. Moreover, customers make no bid, only decide on whether to join the queue or to balk. It follows from the competitive ratio of 5.25 for DT that this mechanism guarantees an approximation ratio of 5.25 on the optimal social welfare, in the worst case.

6 Discussion and Open Problems

Directions for future research include the following:

1. We conjecture that the competitive ratio of deterministic memoryless algorithms for heterogeneous packets is $\phi^3 \approx 4.23$.
2. Although randomness did not help in the homogeneous case, it seems possible that randomized algorithms can beat the deterministic lower bound of 3 for heterogeneous packets.
3. A major goal would be to control a queue with intrinsic packet values and value loss functions.

References

- [1] W. Aiello, Y. Mansour, S. Rajagopalan, and A. Rosen. Competitive queue policies for differentiated services. *Journal of Algorithms*, 55:113–141, 2005.
- [2] N. Andelman and Y. Mansour. Competitive management of non-preemptive queues with multiple values. In *DISC*, pages 166–180, 2003.

- [3] N. Andelman, Y. Mansour, and A. Zhu. Competitive queueing policies for QoS switches. In *Proc. of 14th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 761–770, 2003.
- [4] F. Y. L. Chin, M. Chrobak, S. P. Y. Fung, W. Jawor, J. Sgall, and T. Tichy. Online competitive algorithms for maximizing weighted throughput of unit jobs. *Journal of Discrete Algorithms*, 4(2):255–276, 2006.
- [5] F. Y. L. Chin and S. P. Y. Fung. Online scheduling with partial job values: Does timesharing or randomization help? *Algorithmica*, 37(3):149–164, 2003.
- [6] M. Englert and M. Westermann. Considering suppressed packets improves buffer management in qos switches. In *Proc. of 18th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007.
- [7] R. Hassin. On the optimality of first come last served queues. *Econometrica*, 53(1):201–202, 1985.
- [8] R. Hassin and M. Haviv. *To Queue or not to Queue: Equilibrium Behavior in Queueing Systems*. Kluwer Academic Publishers, 2003.
- [9] A. Kesselman, Z. Lotker, Y. Mansour, B. Patt-Shamir, B. Schieber, and M. Sviridenko. Buffer overflow management in QoS switches. *SIAM J. on Computing*, 33(3):563–583, 2004.
- [10] F. Li, J. Sethuraman, and C. Stein. An optimal online algorithm for packet scheduling with agreeable deadlines. In *Proc. of 16th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2005.
- [11] F. Li, J. Sethuraman, and C. Stein. Better online buffer management. In *Proc. of 18th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007.
- [12] P. Naor. The regulation of queue size by levying tolls. *Econometrica*, 37(1):15–24, 1969.

A Appendix: An Optimal Offline Algorithm

In this section we describe efficient offline algorithms for maximizing the sum of packet benefits (alternately, in the context of utilities and selfish agents, “maximizing the social welfare given the true agent utilities”). We describe algorithms for heterogenous packets. Ergo, they are trivially applicable in case of homogenous packets as well. We observe that the offline problem can be solved via bipartite matching in $O(n^3)$ time complexity, where n is the number of arriving packets in the event sequence. We also show a greedy algorithm that also gives the optimal sum of benefits, and requires only $O(n \log n)$ time complexity.

A natural approach to solving the offline problem of maximizing benefit from heterogenous packets is to use weighted bipartite matching. One could introduce a vertex for every packet and every time slot. Add edges from packets p to time slots $t > \text{arrive}(p)$, of weight $\text{val}(p) - [t - \text{arrive}(p)]$. Now, compute a maximal weighted matching. Packets that are matched to time slots will be accepted by the queue policy, packets that

are not matched will be rejected. We could interpret an edge matching packet p to time slot t as though this means that packet p is to be transmitted at time t and this implies some arbitrary non-preemptive queue regime. Fortunately, it follows from Observation A.2 that using the FIFO queue regime will result in optimal benefit².

Given an event sequence, a transmission schedule is a mapping, m , from the arriving packets of the event sequence to the integers ≥ 1 or the special symbol \emptyset , such that

1. If packet p is mapped to $m(p) \neq \emptyset$ then $\text{arrive}(p) < m(p)$, this can be interpreted as saying that packet p is to be transmitted at time slot $m(p)$. We interpret $m(p) = \emptyset$ as indicating that packet p is rejected.
2. No two packets are mapped to the same integer.

Given an event sequence σ , we define the total benefit of the transmission schedule m to be $\sum_{p \in S} \text{benefit}(p)$, where S is the set of all packets p in σ such that $m(p) \neq \emptyset$. Recall that if $m(p) \neq \emptyset$ then $\text{benefit}(p) = \text{val}(p) - [m(p) - \text{arrive}(p)]$. Given an event sequence σ , a queue policy π , and a queue regime \mathcal{R} , the pair (π, \mathcal{R}) jointly determines a unique transmission schedule $m : \sigma \mapsto Z^+ \cup \{\emptyset\}$ ³.

OBSERVATION A.1. *Given an event sequence σ , and a transmission schedule m , choose any two packets, p and q , such that $m(p) \neq \emptyset$ and $m(q) \neq \emptyset$, and such that the arrival times of both p and q are prior to the earlier transmission (i.e., $\max(\text{arrive}(p), \text{arrive}(q)) < \min(m(p), m(q))$). Now, define a new transmission schedule $m' = m$, except in that $m'(p) = m(q)$ and $m'(q) = m(p)$. (I.e., m' switches the transmission times of packets p and q). Then, the total benefit from transmission schedules m and m' is the same.*

Given an event sequence σ and a transmission schedule m , let m' be another transmission schedule reachable from m by successive applications of transmission time swaps as described in Observation A.1. Let π be a queue policy that determines m (in conjunction with some queue regime \mathcal{R}). Note that if some packet p has negative benefit in schedule m' (i.e., $\text{benefit}(p) = \text{val}(p) - [m'(p) - \text{arrive}(p)] < 0$), then the

queue policy π is clearly non optimal, since π would have benefited by rejecting packet p .

To derive an optimal offline policy for non-preemptive FIFO queues, we start by allowing preemptive and arbitrary queue regimes. The following observation states that while preemption may add considerable strength to online algorithms, it does not impact the optimal (offline) solution. It also states that the queue regime is irrelevant in the context of non-preemptive queue regimes. (Proof in the Appendix).

OBSERVATION A.2. *1. The maximal total benefit is the same irrespective of whether the queue regime allows preemption or not.*

2. *For non-preemptive queue regimes, the total benefit from an event sequence depends only on the admission policy and not on queue regime.*

We consider the following algorithm, GREEDY, which defines both the queue policy and regime: initially, GREEDY accepts every incoming packet. On a send event, GREEDY transmits the packet that currently possesses the highest benefit (ties are broken arbitrarily), and preempts packets with negative or zero benefit.

THEOREM A.1. *For any event sequence σ , $\text{GREEDY}(\sigma) = \text{OPT}(\sigma)$, GREEDY has $O(n \log n)$ time complexity, where n is the number of arriving packets in σ .*

GREEDY defines both a queue policy and a queue regime, the queue regime used by GREEDY is not FIFO. However, it follows from Observation A.2 that we could define a queue policy, OPT, that accepts the same packets eventually transmitted by GREEDY, along with any non-preemptive queue regime, and achieve maximal benefit.

²We remark that this bipartite matching approach can be used for much more general latency sensitivity scenarios, but then Observation A.2 may become inapplicable.

³Likewise, given an event sequence σ and a transmission schedule $m : \sigma \mapsto Z^+ \cup \{\emptyset\}$, there exist (possibly many) pairs of (queue policy, queue regime) that determine m .