Non-Preemptive Buffer Management for Latency Sensitive Packets

Moran Feldman
Technion
Haifa, Israel
Email: moranfe@cs.technion.ac.il

Joseph (Seffi) Naor
Technion
Haifa, Israel
Email: naor@cs.technion.ac.il

***Abstract*—The delivery of latency sensitive packets is a crucial issue in real time applications of communication networks. Such packets often have a firm deadline and a packet becomes useless if it arrives after its deadline. The deadline, however, applies only to the packet's journey through the entire network; individual routers along the packet's route face a more flexible deadline.**

We consider policies for admitting latency sensitive packets at a router. Each packet is tagged with a value and a packet waiting at a router loses value over time as its probability of arriving at its destination decreases. The router is modeled as a non-preemptive queue, and its objective is to maximize the total value of the forwarded packets. When a router receives a packet, it must either accept it (and possibly delay future packets), or reject it immediately. The best policy depends on the set of values that a packet can take. We consider three natural settings: unrestricted model, real-valued model, where any value above 1 is allowed, and an integral-valued model.

We obtain the following results. For the unrestricted model, we prove that there is no constant competitive ratio algorithm. The real valued model has a randomized 4-competitive algorithm and a matching lower bound. We also give for the last model a deterministic lower bound of $\phi^3 \approx 4.236$, almost matching the previously known 4.24-competitive algorithm. For the integral-valued model, we show a deterministic 4-competitive algorithm, and prove that this is tight even for randomized algorithms.

I. Introduction

A router in a communication network receives, buffers and transmits packets. Given that the router has only bounded output capacity, the router has to decide which packets to transmit now and which packets to keep buffered, hoping to transmit them later. Commonly studied router policies usually assume that either packets are indifferent to delays, or each packet has a firm deadline such that the packet must be forwarded before the deadline (or else it is deemed worthless). The first option corresponds to data packets, whereas the second option corresponds to real-time applications, *e.g.*, movie streaming.

In both settings the problem faced by a router can be modeled as an online problem in which each packet is associated with a value, and the router wants to maximize the total value of transmitted packets, subject to restrictions imposed by the architecture of the router and the requirements of the application. Different router architectures and network applications give rise to various issues, many of which have been extensively studied in the literature (see Section I-C for a few examples).

The assumption of a firm deadline for packets of real-time applications is justified from the network's perspective. The network gets credit only for packets arriving at the destination on time. However, since a packet goes through many routers on its way, an *individual router*'s perspective is quite different. The objective of a router is to maximize throughput, but without inducing significant delay on any of the forwarded packets. This observation has led Fiat et. al. [1] to define an online model where for each forwarded packet, the router gets a revenue equal to the value of the packet minus the delay that the packet incurs while waiting in the router. An online algorithm for this model faces a trade-off between buffering too many packets, which imposes a large delay and negligible revenue from each packet, and buffering too few packets.

We use *competitive analysis* as a performance measure for our online algorithms. The advantage of competitive analysis is that no assumption is made on how the input is generated. Instead, the performance of an online algorithm is compared against an optimal algorithm that knows the input ahead of time. Formally, let ALG be an online deterministic algorithm, and let OPT be the optimal *off-line* algorithm. Given an input sequence σ, we denote by $ALG(\sigma)$ and $OPT(\sigma)$ the value of the solutions that ALG and OPT output given σ. The competitive ratio of ALG is defined as $\sup_\sigma \frac{OPT(\sigma)}{ALG(\sigma)}$. If ALG is a randomized algorithm then the competitive ratio is

$$\sup_\sigma \frac{E_r[OPT(\sigma)]}{ALG(\sigma)},$$

where expectation is over the random choices r of ALG. This definition corresponds to an *oblivious adversary*[1]. Notice that for a maximization problem the competitive ratio is at least 1, as no algorithm can be better than OPT.

A. The Model

We consider a router model with a single non-preemptive FIFO buffer and continuous time. At time 0, the queue is empty. Packets arrive at the router at non-integral times. Each packet is either accepted to the buffer or rejected. At any integral time, the packet at the buffer head is dequeued and transmitted, unless the queue is empty at that time. As the queue is non-preemptive, packets may not leave the queue in any other way. For packet d, define $a(d)$ to be its arrival time and $w(d)$ to be its value.

In the HETEROGENOUS DELAY SENSITIVE PACKETS problem (HDSP), each packet d has a revenue depending on its transmission time. If d is never transmitted (*i.e.*, it is rejected),

[1] An oblivious adversary is familiar with ALG, but does not know in advance the results of the actual coin tosses of ALG.

then its revenue is 0. Otherwise, its revenue is equal to $w(d)$ minus the delay it suffers (*i.e.*, the number of integral times d spends in the queue without being transmitted). The objective is to decide which packets to accept in order to maximize the sum of revenues of all packets.

In the online setting, each time a packet arrives, the online algorithm has to make an irrevocable decision whether to accept the packet to the queue or reject it. The decision is made with no knowledge about future packet arrivals.

It turns out that the competitive ratio achievable for HDSP highly depends on the set of possible values that packets take. We consider three natural models:

Unrestricted model: *packets have positive real values.*
Real-valued model: *packets have real values* ≥ 1.
Integral-valued model: *packets have integer values* > 0.

Notice that the lower bound of 1 on the value of packets in the real-valued model is not an arbitrary choice. This lower bound comes up naturally, since for each unit of time that a packet is delayed, its value goes down by 1.

B. Results

We first consider the unrestricted model and show that, unfortunately, even a randomized algorithm cannot have a constant competitive ratio for this model. This improves upon the lower bound of 3 proved by [1] for the integral-valued model, which is currently the best lower bound for the unrestricted model.

We then give, for the real-valued model, a 4-competitive randomized algorithm and a matching lower bound. The best previous algorithm for this model was a deterministic 4.9-competitive algorithm given by [1] with only numerical proof for its competitive ratio. Extra computation time allowed [2] to fine tune the algorithm and improve its ratio to 4.24. We also give an analytical proof of a deterministic lower bound of $\phi^3 \approx 4.236$ (where $\phi = (\sqrt{5}+1)/2 \approx 1.618$ is the golden ratio) for the same model. This bound improves upon a deterministic lower bound of 3, and a lower bound of 4.1 for deterministic memory-less algorithms which was only proved numerically [1] and later improved to 4.23 by [2] using more computation time. Our result strengthens [1]'s conjecture that the "right" deterministic competitive ratio, for this model, is ϕ^3.

For the integral-valued model, we give a deterministic 4-competitive algorithm and a matching randomized lower bound. The previously known results for this model were deterministic lower bound of 3 [1] and upper bound of 4.24 [2].

Our positive results are achieved using the following technique. First, the set of possible inputs is reduced by showing a reduction from the general case to a more specialized set of inputs. Then, an algorithm is given for this set of inputs. The algorithms we present are combinatorial, yet are analyzed via a linear program using the dual-fitting technique.

Table I gives a short summary of known results and our improvements. Besides the three models that we consider, there is another natural model where all packets have equal value of R. For this model [1] gave a deterministic ϕ-competitive algorithm, and showed that this is the best possible for large values of R, even for randomized algorithms.

C. Related Work

There is a vast set of buffering models, capturing many different kinds of router architectures. We survey here only a few representative results. The model in which the algorithm is, perhaps, least restricted is the "bounded delay" problem, where the router buffers all packets that it receives, and it simply has to choose which packet to transmit at each time slot. Each packet has a deadline, and it must be deleted from the router's buffer if not transmitted before the deadline. The objective is to maximize the total value of transmitted packets.

The best known competitive ratios for the "bounded delay" problem are deterministic 1.828 and randomized $e/(e-1) \approx 1.582$ (see [3], [4]). On the negative side, the known lower bounds for this problem are $\phi \approx 1.618$ for deterministic algorithms and $5/4$ for randomized algorithms [5], [6]. See [3], [7], [8] for further results on variants of this problem.

The "preemptive FIFO" problem is a more restrictive problem, in which the buffer is of limited size B and is managed as a FIFO buffer, *i.e.*, packets can only be transmitted in the order in which they arrive. The most general variant of this problem has deterministic lower and upper bounds of 1.419 and $\sqrt{3} \approx 1.732$ [9]. If the packets are restricted to two values 1 and α, then the deterministic lower and upper bounds improve to 1.281 and 1.303 [9].

The most restrictive problem often considered is the "non-preemptive FIFO" model. In this model the setting is like in the previous problem, but the algorithm is only allowed to reject a packet when it arrives, not later. This restriction has no implications on OPT, hence, the competitive ratios for variants of this problem are much worse in comparison to their counterparts for the previous problem. For example, if the the input packets have only the values 1 and α, then there is only a $2 - 1/\alpha$ competitive algorithm, and this is tight for both deterministic and randomized algorithms [5]. Notice that HDSP can be viewed as a variant of "non-preemptive FIFO".

The rest of this paper is organized as follows. Section II gives an impossibility result for the unrestricted model. Sections III and IV give the positive results: a 4-competitive deterministic algorithm for the integral-valued model and a 4-competitive randomized algorithm for the real-valued model. Sections V and VI give the remaining lower bounds: a lower bound of ϕ^3 for deterministic algorithms in the real-valued model and a lower bound of 4 for randomized algorithms in both the real-valued and the integral-valued models. Due to space limitations, some of the proofs are sketched or omitted.

II. The Unrestricted Model

The unrestricted model is our most general model. We show that no constant competitive ratio randomized algorithm exists for this model even against an oblivious adversary.

Let ALG be any randomized algorithm for HDSP, and let $c \geq 1$ be any constant. The following is an oblivious adversary against which ALG is not c-competitive. The adversary gives ALG a series of increasing value packets; the last one of which has value of 1. ALG must accept each one of these packets with

TABLE I
SUMMARY OF KNOWN AND NEW RESULTS. ALL KNOWN RESULTS IN THE TABLE ARE INFERRED FROM [1] AND [2].

	Unrestricted Model		Real Valued Model		Integral-Valued Model	
	Known Result	New Result	Known Result	New Result	Known Result	New Result
Deterministic Lower Bound	3	∞	3	$\phi^3 \approx 4.236$	3	4
Deterministic Upper Bound	-	-	4.24	-	4.24	4
Randomized Lower Bound	$\phi \approx 1.618$	∞	$\phi \approx 1.618$	4	$\phi \approx 1.618$	4
Randomized Upper Bound	-	-	4.24	4	4.24	4

some positive probability in order to be competitive. However, no packet in the series increases the expected value of ALG by much because ALG already accepted the previous packets of the series with a positive probability. OPT, on the other hand, has the privilege of accepting only the last packet in the series and obtaining its entire value.

1. For i = 1 to $2c$ do:
2. Give ALG a packet of value $(1/2c)^{2c+1-i}$.
3. If the probability that ALG accepted any packet, so far, is less than $i/2c$, stop.
4. Give ALG a packet of value 1.

Remark: All packets arrive at non-integral times before the first integral time, *i.e.*, before any of them can be transmitted.

The following lemma shows that in order to be c-competitive ALG must pass the test in line 3 at each iteration, *i.e.*, it cannot delay the acceptance of the first packet by too much.

Lemma 2.1: If ALG is c-competitive, then the adversary does not terminate before reaching line 4.

Theorem 2.2: ALG is not c-competitive.

Remark: Theorem 2.2 holds only for constant values of c because the number of packets given to the algorithm depends on c. For deterministic algorithms, only 2 packets are needed to prove an unbounded competitive ratio for any algorithm.

III. A 4-COMPETITIVE ALGORITHM FOR THE INTEGRAL-VALUED MODEL

In this section we present a 4-competitive deterministic algorithm for the integral valued model. Section VI shows that this is the best possible bound, even for randomized algorithms.

A. Reduction

An input sequence is called *simple* if all packets arrive before the first integral time, *i.e.*, before any packet can be transmitted.

Reduction 3.1: If there is a deterministic R-competitive algorithm A on simple input sequences, then there exists a deterministic R-competitive algorithm B on all input sequences.

Proof Sketch: We first describe B and then sketch the analysis of its competitive ratio. Algorithm B uses a counter c (initially 0), and it uses algorithm A as a subroutine. When B receives a packet d, it feeds d to A and accepts d if and only if A accepts it. However, B makes some changes to d before feeding it to A. First, $a(d)$ is changed to be < 1; this way A "sees" a simple input sequence. Second, $w(d)$ is increased by c. In each integral time, if B transmits a packet, then it also increases c by 1. If B does not transmit a packet, then it sets c to 0 and resets A.

Consider any input sequence σ, and let $\mathcal{S}(\sigma)$ denote the set of input sequences that A has received from B between consecutive resets. By definition of A, for every $\sigma' \in \mathcal{S}(\sigma)$, $R \cdot A(\sigma') \geq OPT(\sigma')$.

On every $\sigma' \in \mathcal{S}(\sigma)$, B's queue is shorter than A's queue, and it can be shown that it is enough to counter the additional value A's packets have. Therefore, B gets at least the same revenue as all the simulations of A together. The last thing left to show is that OPT can gain from $\mathcal{S}(\sigma)$ at least as much as from σ. One can show that OPT can gain that much if it decides to accept from each $\sigma' \in \mathcal{S}(\sigma)$ the same packets it would have accepted from the corresponding sub-section of σ, had σ been the input sequence. ■

B. The Algorithm for Simple Input Sequences

Consider the following deterministic algorithm *Nearly Doubling Threshold* (NDT) (inspired by the 5.25-competitive DT algorithm of [1]): *Accept a packet d if and only if $w(d) \geq 2Q + 1$, where Q is the number of packets in the buffer.*

We analyze the competitive ratio of NDT on simple input sequences. For the sake of analysis, we use an LP formulation of the off-line version of HDSP (see LP1 in Figure 1). Variable $y(d,t)$ is an indicator to the event that packet d is transmitted in integral time t.

The coefficient of each variable $y(d,t)$ in the objective function is its value ($w(d)$) minus the number of integral times it spent in the buffer, if transmitted at time t. The packet constraints force a packet to be transmitted at most once. The time constraints prevent more than one packet from being transmitted at the same time. Notice that no constraint enforces FIFO transmission, however, such a constraint is not needed since opt is oblivious to the FIFO requirement.

The dual is LP2 (see Figure 1). Notice that every feasible solution for LP2 is an upper bound on the optimal solution for LP1, and therefore, also on opt. Thus, if A is an online algorithm for HDSP, and for any simple input sequence σ there is a feasible solution for LP2 of cost $\alpha \cdot A(\sigma)$, then A is α-competitive on simple input sequences. We now show how to construct a solution for LP2 of cost $4 \cdot \mathsf{NDT}(\sigma)$ for any input sequence σ, proving that NDT is 4-competitive for any simple input sequence. We abuse notation, and given an input sequence σ, we denote by LP2(σ) the constraints of LP2 when the input sequence is σ.

Reduction 3.2: Let Q_d denote the number of packets in the buffer before packet d in σ is received. We can assume that if d is accepted by NDT then its value is exactly $2Q_d + 1$.

Proof Sketch: Decreasing the value of d from a larger value to $2Q_d + 1$ does not change the set of packets NDT accepts. Therefore, it effects NDT at least as much as OPT, and can only increase the competitive ratio. ■

Fig. 1. LP formulation of the off-line version of HDSP (LP1) and its dual (LP2)

$$\begin{array}{lllll}
\text{(LP1)} \quad \max & \sum_d \sum_{t|a(d)<t<w(d)+a(d)} (w(d) - \lfloor t - a(d) \rfloor) \cdot y(d,t) & & & \\
& \sum_{t|a(d)<t<w(d)+a(d)} y(d,t) & \leq 1 & \forall\, d & \text{(packet constraint)} \\
& \sum_{d|a(d)<t<w(d)+a(d)} y(d,t) & \leq 1 & \forall\, t & \text{(time constaint)} \\
& y(d,t) & \geq 0 & \forall\, d, a(d) < t < w(d) + a(d) &
\end{array}$$

$$\begin{array}{lllll}
\text{(LP2)} \quad \min & \sum_t x_t + \sum_d z_d & & & \\
& x_t + z_d & \geq w(d) - \lfloor t - a(d) \rfloor & \forall\, d, w(d) + a(d) > t > a(d) & \\
& x_t, z_d & \geq 0 & \forall\, d, t &
\end{array}$$

Lemma 3.3: Let Q_f be the number of packets accepted by NDT(σ), then NDT(σ) $= Q_f(Q_f+1)/2$.

Proof: Since we are only dealing with a simple input sequences, when the i^{th} packet accepted by NDT is received there are $i-1$ packets in the buffer. By Reduction 3.2, the value of the i^{th} packet is $2(i-1)+1$. Therefore, the revenue NDT gets from the i^{th} packet is $2(i-1)+1-(i-1) = i$. ∎

Consider the following dual solution for LP2(σ). Variables of type z_d are assigned value of 0, and variables of type x_t are assigned value of $\max\{2Q_f - t, 0\}$.

Lemma 3.4: The above dual solution is feasible and costs $Q_f(2Q_f - 1)$.

Proof: Consider a constraint of the dual LP: $x_t + z_d \geq w(d) - \lfloor t - a(d) \rfloor$. Since σ is a simple input sequence $\lfloor t - a(d) \rfloor = t - 1$, so it is enough to show that $x_t \geq w(d) - t + 1$. By Reduction 3.2, $w(d) \leq 2Q_d - 1 \leq 2Q_f - 1$, and therefore, the last inequality holds due to the value assigned to x_t.

Note that only variables $x_1, x_2, \ldots, x_{2Q-1}$ get a non-zero assignment. The coefficients of all these variables in the objective function is 1, hence, the cost of the dual solution is $\sum_{t=1}^{2Q_f-1} 2Q_f - t = \frac{2Q_f(2Q_f-1)}{2} = Q_f(2Q_f-1)$. ∎

Corollary 3.5: NDT is a 4-competitive algorithm.

Proof: The ratio between the cost of the above dual solution and NDT(σ) is:

$$\frac{Q_f(2Q_f-1)}{Q_f(Q_f+1)/2} = 2 \cdot \frac{2Q_f+2-3}{Q_f+1} = 4 - \frac{6}{Q_f+1} < 4.$$

∎

IV. A 4-competitive Randomized Algorithm for the Real-Valued Model

Fiat et. al. [1] prove that randomization does not help when all packets have equal values. Section III shows that the same holds for the integral-valued model. Surprisingly, the real valued model is different. We show here a 4-competitive randomized algorithm for the real-valued model, bypassing the deterministic lower bound given in Section V. However, Section VI shows that this is the most randomization can do.

This section is structured similarly to Section III. First, a reduction to simple input sequences is presented, and then an algorithm for these sequences is given. Unlike Section III, the reduction here assumes the algorithm has a given property, and therefore, it is not a "real" reduction.

A. The Reduction

Consider a randomized algorithm for simple input sequences, and let Q be a random variable of the number of packets in the buffer of this algorithm at some point of time. The algorithm is called *good* if for every integer $k \geq 0$, $\Pr[Q = k] > 0 \Rightarrow \Pr[Q > k+1] = 0$. In other words, the algorithm is good if given the input sequence, there is a number Q, such that the algorithm currently has in its queue either Q or $Q+1$ packets, regardless of its actual coins tosses.

Reduction 4.1: If there is a good randomized R-competitive algorithm A for simple input sequences, then there exists a randomized R-competitive algorithm B for all input sequences. The proof is similar to the one of Reduction 3.1.

B. An Algorithm for Simple Input Sequences

In a sense, Reduction 4.1 requires an algorithm for simple input sequences with a very limited randomness. On the other hand, some randomness is required because of the lower bound for deterministic algorithms proved in Section V. The compromise between these requirements is an algorithm called *Randomized Nearly Doubling Threshold* (RNDT). We show RNDT is 4-competitive for simple input sequences. Together with Reduction 4.1, this implies a 4-competitive randomized algorithm for general input sequences.

RNDT uses a counter C and a variable p_C initialized to 0 and 1, respectivly. Let Q be a random variable of the number of packets already accepted by RNDT into the buffer. Consider a packet d which RNDT receives. A probability p_d is associated with d using the following rule. If d is the first packet, $p_d = 1/2$. Otherwise, p_d is defined as follows:

$$p_d = \begin{cases} 0 & w(d) - 2E(Q) \leq 0.5 \\ w(d)/2 - 0.25 - E(Q) & 0.5 \leq w(d) - 2E(q) \leq 2.5 \\ 1 & w(d) - 2E(Q) \geq 2.5 \end{cases}$$

Precisely one of the two cases below is applicable to RNDT:

- Case 1: $p_d \leq p_C$ - If $Q = C$, accept the packet d with probability p_d/p_C and update $p_C \leftarrow p_C - p_d$.
- Case 2: $p_d > p_C$ - If $Q = C$, accept the packet d, otherwise, accept the packet d with probability $(p_d - p_C)/(1 - p_C)$. Then update $C \leftarrow C + 1$ and $p_C = 1 + p_C - p_d$.

Lemma 4.2: RNDT maintains the invariant: $Q = C$ with probability p_C, and $Q = C+1$ with probability $1 - p_C$. Notice that this implies that the algorithm never divides by zero.

Corollary 4.3: RNDT is a good algorithm.

The analysis of RNDT goes along the same lines as the analysis of NDT in Section III, *i.e.*, we consider a simple input sequence σ and show that LP2(σ) has a solution of cost $4 \cdot E[\mathsf{RNDT}(\sigma)]$. Let us denote by E_f the final value of $E(Q)$ after all packets are received by RNDT.

Reduction 4.4: We can assume that no packet has value larger than $2E_f + 0.5$.

Lemma 4.5: The expected revenue of RNDT is at least $\frac{E_f^2+E_f+1/4}{2}$, unless no packet was received.

Consider the following dual solution for LP2(σ). Variables of type z_d are assigned value of 0, and variables of type x_t are assigned value of $\max\{2E_f + 1.5 - t, 0\}$.

Lemma 4.6: The above dual solution is feasible, and its cost is $2E_f^2 + 2E_f + 1/2$.

Corollary 4.7: RNDT is a 4-competitive algorithm for simple input sequences.

V. Lower Bound for Deterministic Algorithms in the Real Valued Model

The best known lower bound for deterministic algorithms in the real valued model is 3 [1]. In this section we show an improved lower bound of $\phi^3 \approx 4.236$. This lower bound is based on the technique of a lower bound of 4.1 proved by [1] for memory-less deterministic algorithms. Given $\beta > 1$, consider the following series. $b_{\beta,0}$ is 1, and for every $k \geq 1$:

$b_{\beta,k} = \min\left\{x \in \mathbb{R}^+ | \sum_{j=0}^{\lfloor x \rfloor}(x-j) \geq \beta \cdot \sum_{j=0}^{k-1}(b_{\beta,j} - j)\right\}$.

It is proved in [1] that if there is k such that $b_{\beta,k} < k$, then no memoryless deterministic algorithm can be better than β-competitive in the real valued model. We prove that this result also extends to non-memoryless algorithms.

Let A be any deterministic online algorithm for HDSP in the real valued model. Consider the following adversary:

1. For k = 0 to ∞ do:
2. Give packets of value $b_{\beta,k}$ to A till A accepts one of them, or till $\lfloor b_{\beta,k} \rfloor + 1$ packet have been given.
3. If A accepted no packets in the previous step, stop.

The idea is that if A is better than β competitive, then it must accept a packet in every iteration (otherwise, by the definition of $b_{\beta,k}$, OPT can accept only the $\lfloor b_k \rfloor + 1$ packets of value b_k and be at least β times better than A). However, if $k < b_{\beta,k}$ then the revenue A gets from the packet it accepts on the k^{th} iteration is negative ($b_{\beta,k} - k$), so it cannot help A anyway.

Theorem 5.1: If there is $k \geq 0$ such that $b_{\beta,k} < k$, then no deterministic algorithm is better than β-competitive in the real valued model.

Theorem 5.2: For any $\beta < \phi^3$ there is k such that $b_{\beta,k} < k$, hence, no deterministic algorithm is better than ϕ^3-competitive in the real valued model.

VI. Lower Bound for Randomized Algorithms in the Integral Valued Model

The best known randomized lower bound (of ϕ) was given by [1] for a model where all packets have equal values. In this section, we show that under the integral valued model, no randomized algorithm has a competitive ratio better than 4 against an oblivious adversary. Notice that the real valued model generalizes the integral valued model, and therefore, inherits this bound. In this section, we consider some randomized algorithm A and $\beta < 4$, and show that A cannot be β-competitive against the following adversary:

1. For k = 1 to ∞
2. Give k packets of value k to A
3. If the total expected revenue of $A < k(k+1)/(2\beta)$, stop

The idea behind the adversary is that A must accept enough packets to make its expected revenue $\geq k(k+1)/(2\beta)$, otherwise, it is not β-competitive because OPT can collect only the k packets of value k. However, over time, A has to add more and more packets to its buffer to keep up with this goal. The old packets in A's buffer diminish the effect of new packets, and eventually A fails to keep up with the above goal.

Due to space limitations, we omit the proof that indeed no A is β-competitive against this adversary.

VII. Conclusions

We considered three variants of the HDSP problem corresponding to three possible sets of allowed packet values. For the unrestricted model we showed that there is no constant competitive ratio algorithm. For the integral-valued model we gave a 4-competitive deterministic algorithm and showed that this is best possible even for randomized algorithms. For the real-valued model we gave a 4-competitive randomized algorithm and a matching lower bound. However, there is still a small gap in this model in terms of deterministic algorithms. We gave a lower bound of $\phi^3 \approx 4.236$ while the best upper bound known is 4.24. Closing this gap is an obvious open problem.

One can consider a general model in which all values larger than c are allowed, for some $c \geq 0$. Notice that this model generalizes both the real-valued and the unrestricted models, hence, the competitive ratio of this model must depend on c. It may be interesting to find the exact connection between c and the competitive ratio.

References

[1] A. Fiat, Y. Mansour, and U. Nadav, "Competitive queue management for latency sensitive packets," in *SODA*, 2008, pp. 228–237.

[2] ——, "Competitive queue management for latency sensitive packets," PEGG, 2007.

[3] M. Englert and M. Westermann, "Considering suppressed packets improves buffer management in QoS switches," in *SODA*, 2007, pp. 209–218.

[4] Y. Bartal, F. Y. L. Chin, M. Chrobak, S. P. Y. Fung, W. Jawor, R. Lavi, J. Sgall, and T. Tichy, "Online competitive algorithms for maximizing weighted throughput of unit jobs," in *STACS*, 2004, pp. 187–198.

[5] N. Andelman, Y. Mansour, and A. Zhu, "Competitive queueing policies for QoS switches," in *SODA*, 2003, pp. 761–770.

[6] F. Y. L. Chin and S. P. Y. Fung, "Online scheduling with partial job values: Does timesharing or randomization help?" *Algorithmica*, vol. 37, no. 3, pp. 149–164, 2003.

[7] F. Li, J. Sethuraman, and C. Stein, "An optimal online algorithm for packet scheduling with agreeable deadlines," in *SODA*, 2005, pp. 801–802.

[8] M. Chrobak, W. Jawor, J. Sgall, and T. Tichý, "Improved online algorithms for buffer management in QoS switches," *ACM Trans. Algorithms*, vol. 3, no. 4, p. 50, 2007.

[9] M. Englert and M. Westermann, "Lower and upper bounds on FIFO buffer management in QoS switches," in *ESA*, 2006, pp. 352–363.

Acknowledgement

The research of Seffi Naor is supported by ISF grant 1366/07.