Dynamic Spectrum Allocation:
The Impotency of Duration Notification

Bala Kalyanasundaram *
Computer Science Department
University of Pittsburgh
kalyan@cs.pitt.edu

Kirk R. Pruhs *
Computer Science Department
University of Pittsburgh
kirk@cs.pitt.edu

Abstract

For the classic dynamic storage allocation problem, we show that knowledge of the durations of the requests is of no great use to an online algorithm in the worst case. This answers an open question posed by Naor, Orda, and Petruschka [5]. More precisely, we show that the competitive ratio of every randomized algorithm against an oblivious adversary is $\Omega(\frac{\log x}{\log\log x})$, where x may be any of several different parameters used in the literature. It is known that First Fit, which does not require knowledge of the durations of the task, is logarithmically competitive in these parameters.

*Supported in part by NSF Grant CCR-9734927.

1 Introduction

The dynamic storage/spectrum allocation (DSA) problem is a classic combinatorial optimization problem in the computer science literature (for surveys see [2] or [9]). We consider the online version of DSA with known durations, which we now define.

Dynamic Storage/Spectrum Allocation with Known Durations: An online algorithm is equipped with a linear resource, for example memory or frequency spectrum, that it must use efficiently to satisfy a sequence of n requests for this resource. The ith request R_i is a pair (s_i, d_i) that is revealed to the online algorithm at some release time r_i. The parameter s_i denotes the bandwidth of the request R_i, and d_i denotes the duration of R_i. We number the requests by increasing order of release times, that is, $r_i \leq r_j$ for $i < j$. In response to request R_i , the online algorithm must allocate s_i units of contiguous resource to R_i during the time interval $[r_i, r_i + d_i]$. We say that R_i is *active* during $[r_i, r_i + d_i]$. Importantly, at no time may any two active requests share a common unit of resource. The objective function is to minimize the total resource size required to satisfy all of the given requests.

The standard measure used to compare online algorithms is competitiveness [1]. In the context of DSA, an online randomized algorithm A is *c-competitive* if for all input sequences I, the expected value of the bandwidth $A(I)$ used by A on input I is at most c times $\text{OPT}(I)$, the optimal bandwidth to satisfy I [1].

Most of the literature on online dynamic storage allocation deals with the unknown duration case, that is when the online algorithm does not learn the duration of a request R_i until R_i leaves the system at time $r_i + d_i$. There is no constant competitive algorithm in the unknown duration case [6, 7]. Every deterministic algorithm is $\Omega(\log \beta)$ competitive, where β is the ratio of the largest bandwidth of a request to the shortest bandwidth of a request [6]. Every randomized algorithm is $\Omega(\min(\frac{\log \beta}{\log\log \beta}, \frac{\log \alpha}{\log\log \alpha}))$ competitive against an oblivious adversary [4], here α is the maximum number of active requests at any one time. An *oblivious* adversary must specify all of the requests before the online algorithm begins execution [1]. The competitive ratio of First Fit is $O(\log \beta)$ [8], and $O(\log \alpha)$ [4]; The algorithm First Fit places each request in the lowest feasible location in the linear resource.

In [5], the dynamic storage allocation problem with known durations is considered. They show that in the case of known durations that an online algorithm may be logarithmically competitive in other parameters besides α and β. More precisely, an online algorithm may be $O(\log \Delta)$ competitive, here Δ is the ratio of the largest duration of a request to the smallest duration of a request, and may be $O(\log \tau)$ competitive, here τ is the number of distinct durations. In [5] the fundamental problem of determining whether knowledge of the durations of the requests is of significant benefit to the online algorithm, i.e. whether it is possible for an online algorithm to be constant competitive, is left open. Here we prove the following theorem.

Theorem 1 Assuming an oblivious adversary, the competitive ratio of every randomized online algorithm for the dynamic storage allocation with known durations is $\Omega(\frac{\log x}{\log\log x})$, where x may be any one of n, α, β, τ or $\log \Delta$.

We feel that the proper interpretation of these results is that knowledge of the durations does not greatly benefit the online allocator in the worst case. Although these results do leave open the possibility that knowing the durations may logarithmically improve the competitive ratio achievable by an online algorithm when the competitive ratio is measured in terms of Δ.

The offline version of DSA, where the algorithm has complete knowledge of future requests, is NP-hard [2], and a 3-approximation polynomial time algorithm is known [3]. Several average case analyses of various algorithms are known, for more information see [9].

For convenience, we will refer to the linear resource as memory for the rest of the paper.

2 The Lower Bound Construction

In this section we prove theorem 1 using the following statement of Yao's principle for online cost minimization problems [1].

Theorem 2 Suppose that there exists a input distribution on the inputs I such that for all deterministic algorithms A it is the case that

$$\liminf_{n\to\infty} \frac{E\left[A(I)\right]}{E\left[\textsc{Opt}(I)\right]} \geq c$$

and

$$\limsup_{n\to\infty} E\left[\textsc{Opt}(I)\right] = \infty$$

where the expectation is over the marginal distribution on inputs of size n, Then the competitive ratio of every randomized algorithm is at least c.

In order to apply Theorem 2, fix a deterministic online algorithm A, and fix some $c > 2$. Let $W = 12(12c)^{47c}$. Note that $c = \Theta(\frac{\log W}{\log\log W})$. We will give a request distribution that forces $E[A(I)] = \Theta(cW)$ while $E[\textsc{Opt}(I)] = \Theta(W)$.

Request Distribution: We partition the requests into $24c$ rounds. The bandwidth of each request in round i, $1 \leq i \leq 24c$, will be $w(i) = (12c)^{2i-2}$. Further, we partition round i, $1 \leq i \leq 24c$, into $\frac{W}{b(i)}$ stages, where $b(i) = (12c)^{2i-1}$. The number $\ell_{i,j}$ of requests in stage j, $1 \leq j \leq \frac{W}{b(i)}$, of round i, will be selected uniformly at random over the range $[1, 2j\frac{b(i)}{w(i)}]$. Note that we have set the value of W so that $\frac{W}{b(i)} \geq 12$ for any $1 \leq i \leq 24c$.

We associate an interval $I_{i,j} = [a_{i,j}, b_{i,j}]$ with the stage j of round i. Initially, $I_{1,1} = [a_{1,1}, b_{1,1}] = [0, 1]$. Each request in stage j of round i is released at time $a_{i,j}$.

The duration of the kth request, $1 \leq k \leq \ell_{i,j}$, in stage j of round i is then $(b_{i,j} - a_{i,j})(1 - 1/2^k)$. Notice that the durations increase throughout a stage. We call the last request in each stage a *fang*.

Finally, the interval associated with the next stage (whether it is in the same round or a different round is irrelevant) is $[a_{i,j} + (b_{i,j} - a_{i,j})(1 - 1/2^{\ell_{i,j}-1}), a_{i,j} + (b_{i,j} - a_{i,j})(1 - 1/2^{\ell_{i,j}})]$. That is, the interval for the next stage is between the end of the penultimate request in the previous stage and the end of the fang in the previous stage. Thus all fangs released during a stage will stay present till the arrival of the last request of the last round, while requests that are not fangs leave the system before requests in the next stage appear. For a graphical depiction of the construction see figure 1.

Intuitively, the adversary may allocate fangs consecutively in memory. However, since the online algorithm can not identify a fang when it arrives, it will not be able to allocate fangs to one part of memory, and thus will have a more fragmented resource when it proceeds to the next stage. We will first prove that $\textsc{Opt} = \Theta(W)$, and then show that the expected bandwidth used by A is $\Omega(cW)$.

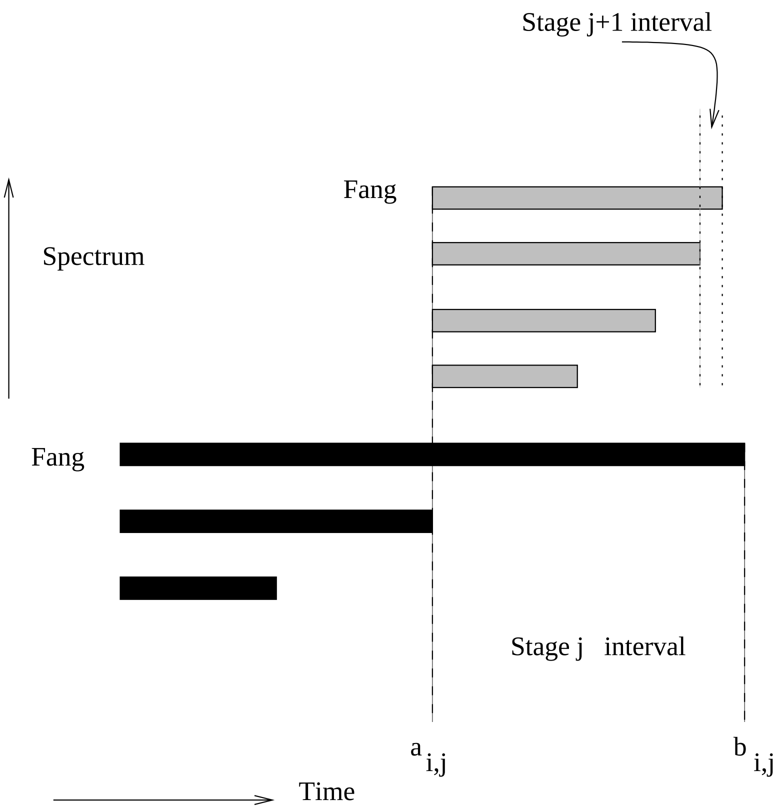

Figure 1: Lower Bound Construction

Lemma 3 With probability one, $\textsc{Opt}(I) \leq 4W$

Proof: One possible strategy to achieve $4W$ is to divide the memory into two pieces N and F, each of size $2W$. All non-fangs are allocated using to N using First Fit, and and all fangs are allocated to F using First Fit.

The total bandwidth of the non-fang requests in a stage j in round i is at most the product of the bandwidth $w(i)$ of the requests in round i, and the number of requests $2j\frac{b(i)}{w(i)}$. Since j is at most $W/b(i)$, the total bandwidth of the non-fang request in stage j of round i is at most

$$w(i) \cdot 2 \cdot \frac{W}{b(i)} \cdot \frac{b(i)}{w(i)} = 2W$$

Hence, all the non-fangs within any stage can be fit into N. Observe that any two non-fang requests not from the same stage do not overlap in time. Therefore, the same memory N can be reused for all of the stages.

We now calculate required bandwidth to satisfy fang requests. For round i, there are $\frac{W}{b(i)}$ fang requests of width $w(i)$ each. Therefore, the total bandwidth of the fangs is at most

$$\sum_{i=1}^{24c} \frac{W}{b(i)} w(i) = \sum_{i=1}^{24c} \frac{W}{12c} = 2W$$

Hence, all the fangs can fit into F. ∎

It order to analyze the performance of A it is convenient to grant A additional powers. At the end of each round, say round i, we allow A to reorganize its memory in the following manner. Assume that there are two fangs R_j and R_k such that the space between R_j and R_k is less than $w(i+1)$, the bandwidths of the requests in the next round. Notice that the space used by R_j and R_k, and the space between R_j and R_k, is of no use to A in future rounds. The online algorithm A may then delete the space used by R_j, the space used by R_k, and the space between these two requests from its memory, and meld the remaining two pieces of memory. We say that the remaining contiguous portion of the melded memory is *available* memory for the next round. Note that this is strictly to A's benefit since any feasible assignment on the original memory is also a feasible assignment in the modified memory.

We now wish to define what we call a fragmented round. Intuitively, a fragmented round is one in which A assigns many fangs to different parts of the available memory. Let i be some round. We partition A's available memory into contiguous *i-blocks* $B_{i,k}$ of size $b(i)$. We say that a fang from round i is assigned to an i-block $B_{i,k}$ if either the fang is contained entirely within $B_{i,k}$ or the fang crosses the boundary between $B_{i,k}$ and $B_{i,k+1}$. Let $f(i)$ be the number distinct i-blocks that have at least one fang assigned to them. We say that a round i is *fragmented* if $f(i) \geq \frac{W}{2b(i)}$.

We will now argue that:

- the expected number of fragmented rounds is at least one half of the total number of rounds, and
- the available memory decreases by $\Theta(W)$ after each fragmented round provided that A does not use too much memory.

Lemma 4 The expected number of fragmented rounds is at least $12c$.

Proof: Consider stage j of round i. Let $f_j(i)$ be the number of distinct i-blocks assigned at least one fang from stages 1 through j in round i. By our construction and the definition of fang requests, we have $f_{j-1}(i) \leq j-1$. Therefore, at most $(j-1)\frac{b(i)}{w(i)}$ of the requests in stage j of round i can occupy i-blocks that have already been assigned a fang in round i. Since the number of requests in stage j of round i is uniformly at random selected from the range $[1, 2j\frac{b(i)}{w(i)}]$, the probability that the fang of stage j of round i is assigned to an i-block that does not already contain a fang is at least $1/2$. Hence, the probability that half of the fangs in round i are assigned to an i-block that did not previously contain a fang from round i is at least $\frac{1}{2}$. The result then follows since there are $24c$ rounds. ∎

Lemma 5 During every fragmented round, either A deleted at least $\frac{W}{12}$ units of memory at the end of this round, or A used at least cW units of memory during this round.

Proof: Fix a fragmented round i, and consider the assignment of the fangs from round i. Within each i-block $B_{i,k}$ that contains a fang from round i, arbitrary pick a *canonical* fang from among those fangs assigned to $B_{i,k}$ during round i. Recall that there are at least $\frac{W}{2b(i)}$ canonical fangs since i is a fragmented round. We sort these canonical fangs according to their position in A's memory, and group them into $\frac{W}{6b(i)}$ groups, each containing 3 consecutive canonical fangs.

We say that a group is *sparse* if the gap between any two consecutive canonical fangs in the group is at least $w(i+1)$. Recall that we set W such that $\frac{W}{12b(i)} \geq 1$ for all $1 \leq i \leq 24c$. First, suppose that $\frac{W}{12b(i)}$ of the groups are sparse. Then the total memory used by A must be at least $\frac{W}{12b(i)}w(i+1)$, which by substitution is cW

On the other hand, suppose that $\frac{W}{12b(i)}$ of the groups are not sparse. Hence, the memory used by the canonical fangs in this group, as well as the memory in the gaps between these fangs, will be deleted by A at the end of the round. Since the gap between first and third requests of each group is at least $b(i)$, the total memory deleted by A during this round is at least $\frac{W}{12}$. ■

We are now ready to prove theorem 1.

Proof of Theorem 1: First, the fact that $\text{OPT} = \Theta(W)$ follows from lemma 3. By lemma 4 and lemma 5, the expected memory used by A is at least cW. Note that the expected memory used by OPT goes to ∞ as n goes to ∞. Hence, we get a lower bound of $\Omega(\frac{\log W}{\log\log W})$ on the competitive ratio.

Observe that we get a lower bound of $\Omega(\frac{\log \beta}{\log\log \beta})$ on the competitive ratio since $\beta \leq W$.

Observe that n is no smaller than α and τ. So in order to show a lower bound of $\Omega(\frac{\log x}{\log\log x})$ on the competitive ratio, for x equal to n, α, and τ, it is sufficient to show that $n = O(W^2)$. Since the number of stages in round i is $\frac{W}{b(i)}$, and the number of requests in stage j of round i is at most $2j\frac{b(i)}{w(i)}$, it follows that the number requests in round i is at most

$$\begin{aligned} & 2\frac{b(i)}{w(i)}\sum_{j=1}^{\frac{W}{b(i)}} j \\ = & \frac{b(i)}{w(i)}\frac{W}{b(i)}\left(\frac{W}{b(i)}+1\right) \\ \leq & \frac{b(i)}{w(i)}\frac{W}{b(i)}\left(2\frac{W}{b(i)}\right) \\ = & \frac{2W^2}{w(i)\cdot b(i)} \end{aligned}$$

Since $b(i)w(i) = (12c)^{4i-3} \geq 12c \geq 4$, the number of requests per round is going down by more than a factor of two each round. Hence, the total number of requests is at most

$$2\frac{2W^2}{w(1)\cdot b(1)} \leq 2W^2$$

Finally, that the competitive ratio is $\Omega(\frac{\log\log\Delta}{\log\log\log\Delta})$ follows from the fact that $\Delta = 2^n$ by construction. ■

References

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

[2] E. Coffman, "An introduction to combinatorial models of dynamic storage allocation", *SIAM Review*, **25**, 311 – 325, 1999.

[3] J. Gergov, "Algorithms for compile-time memory allocation", ACM/SIAM Symposium on Discrete Algorithms, S907-S908, 1999.

[4] M. Luby, J. Naor, A. Orda, "Tight Bounds for Dynamic Storage Allocation" *SIAM Journal of Discrete Mathematics*, **9**, 156 – 166, 1996.

[5] J. Naor, A. Orda, and Y. Petruschka, "Dynamic storage allocation with known durations", European Symposium on Algorithms, 378 – 387, 1997.

[6] J. Robson, "An estimate of the store size necessary for dynamic storage allocation", *Journal of the ACM*, **18**, 416 – 423, 1971.

[7] J. Robson, "Bounds on some functions concerning dynamic storage allocation", *Journal of the ACM*, **21**, 491 – 499, 1974.

[8] J. Robson, "Worst case fragmentation of first fit and best fit storage allocation strategies", *Computer Journal*, **20**, 242 – 244, 1977.

[9] P. Wilson, M. Johnstone, M. Neely, and D. Boles, "Dynamic storage allocation: a survey and critical review", International Workshop on Memory Management, *Lecture Notes in Computer Science*, **986**, 1 – 116, 1995.