CSC2421: Online and other myopic algorithms Spring 2021

Allan Borodin

January 28, 2021

Week 3: agenda

We ended the week 2 discussion just mentioning the bin packing, time-series search, and one-way trading problems. We will discuss them a little further today following their presentation in Chapter 2.

We will then discuss the material in Chapter 3 (randomized online algorithms). In particular, we will discuss the following:

- The different types of adversaries when considering randomized algorithms.
- with respect to the randomized competitive ratio wrt. oblivious adversaries we will discuss:
  1. The one-way trading problem where randomization cannot help.
  2. The proportional knapsack problem where one random bit can be used to achieve a $\frac{1}{4}$ approiximation whereas without randomization no constant approximation is possible.
  3. The randomized time series problem where ranomization achieves a $O(\log \frac{U}{L})$ ratio whereas without randomization $\sqrt{\frac{U}{L}}$ is optimal.
  4. Paging where randomization can be used to achieve an (optimal) $H_k \approx \ln k$ ratio whereas without randomization we know that k is the optimal competitive ratio.

Week 3 agenda continued

- We will discuss the Yao Minmiax principle for deriving inapproximations for randomized algorithms, a technique used to show that the H_n competitive ratio is an inaqpproximation for any randomized paging algorithm.
- We will briefly introduce *de-randomzation*. We comment that the de-randomization of the randomized time-series algorithm is equivalent (in terms of the solution produced) the deterministic algorithm for one-way trading.
- Then I would like to see what possible topics are of interest for the course reading requirement. I may shorten the previous discussions to go over the index and suggest some active topics.

I think next week we can spend much of the class seeing who is willing to undertake any possible topics within the scope of the course. Lots of possibilities: classical results-improving or clarifying arguments, recent improved results, comparison of algorithms for a specific problem like *adwords*, new online applications, learning online algorithms, etc.

The bin packing problem

- Johnson in his 1974 JCSS paper (and perhaps in his PHD thesis) specifically asked if First Fit (and therefore Best Fit) was the best online algorithm (with respect to their $\frac{17}{10}$ competitive ratio) for bin packing. Johnsons thesis was preceded by a 1973 conference paper by Garey,Graham and Ullman and that paper was preceded by a 1971 technical report by Ullman. The culmination of all these works appears in a 1974 Johnson et al journal paper.
- In 1980, Yao explicitly considered Johnson's question as to the competitive analysis for the online bin packing problem (without using that terminology). He both provided an algorithm with an improved ratio as well as establishing an online inapproximation for bin packing.
- The early (1971,1973,1974) work by Johnson et al showed that the *asymptotic* ratio for the First Fit and Best Fit algorithms is $\frac{17}{10}$.
- Forty years later Dosa and Sgall show that $\frac{17}{10}$ is the strict competitive ratio. Yao (1980) exhibited an online algorithm with asymptotic competitive ratio $\frac{5}{3}$ and a $\frac{3}{2}$ asymptotic inapproximation for any online algorithm.

Bin packing continued

Bin packing is one of the most studied problem both with repect to online and offline algorithms. We want to pack a sequence (for online) or set (for offline) of items $I = \{x_1, \ldots x_n\}$ into the smallest number of bins of a fixed size. Without loss of generality we can assume the bin size is 1, and all $x_i \leq 1$.

In the offline world, Karmacher and Karp [1982] gave a polynomial time algorithm ALG such that $ALG(I) \leq OPT(I) + o(OPT(I))$ for all input sets I; that is, the asymptotic approximation ratio is 1.

It is not known if there is an algorithm satisfying $ALG(I) \leq OPT(I) + c$ for some consdtant c and as far as I know it is possible that there is an offline algorithm satisfying $ALG(I) \leq OPT(I) + 1$.

It is NP hard to determine if an input I can be packed into two bins or requires at least 3 bins.

Online bin packing

- There are three natural simple online bin packing algorithms, namely Next Fit, First Fit and Best Fit.
- The analysis of Next Fit is quite simple leading to a 2-competitive ratio which is tight for this algorithm.
- The analysis for First Fit and Best FIt basially are equivalent and non trivial.
- As mentioned, Yao gave an online algorithm (Refined First Fit) with a $\frac{5}{3}$ ratio which was followed by Lee and Lee [1983] who gave two algorithms, Harmonic with ratio 1.692 and Refined Harmonic with ratio 1.636.
- The Yao, and Lee and Lee algorithms both utilize an idea that occurs in other algorithms. Namely they explicitly partition the inputs into a finite (say M) number of classes (in contrast to the analysis in First Fit where a different weighting function is used for different classes of functions).
- They then deal with each input class by partitioning the bins into M classes used exclusively for the corresponding input classes.

The Lee and Lee Harmonic and Refined Harmonic algorithms

The Harmonic algorithm partitions the inputs $I = \bigcup_{k=1}^{M} I_k$ into M classes where $I_k = \{x \in (1/(k+1), 1/k]\}$ for $1 \leq k \leq M-1$ and $I_M = (0, 1/M]$.

A key idea is that we can tell when a bin of a certain class is *full* and no further items of that class can be packed into a full bin. Such an algorithm can then use the simpler Next Fit packing rule for each class.

An open bin that is not full is called *active*.
Harmonic will only have a constant number of active bins and such an algorithm is called a "constant space" bin packing algorithm. Harmonic achieves the optimal competitive ratio amongst constant space algorithms.

The Refined Harmonic recognizes that too much space is being wasted in the bins reserved for the big items in I_1 and I_2.

Refined Harmonic continued

Instead, it will further partition the inputs $I_1 = I_{1,1} \cup I_{1,2}$ and $I_2 = I_{2,1} \cup I_{2,2}$. Here $I_{1,1} = \{x \in (59/96, 1]\}, I_{2,1} = \{x \in 37/96, 1/2]\}$.

The items in $I_{1,1}, I_{2,1}$ and I_k for $3 \leq k \leq M$ will again be packed by Next Fit while the smaller items $I_{1,2}$ in I_1 will be more carefully packed with the smaller items $I_{2,2}$ in I_2.

- As stated Refined Harmonic results in a 1.6359 ... ratio for $M = 20$
- The Harmonic algorithm has been further refined and the latest refinement is ia 1.57829 ratio due to Balogh et al [2018]
- The current best inapproximation 1.54278 is due to Balogh et al [2012]

While I personally do not recommend working on these small "gaps" in proving the optimal ratio, it is the case that the Bin Packing problme has attracted attention and variants of the basic problem are still actively considered.

The most studied variant is for multi-dimendensional bin packing; for example, packing axis aligned rectangles in 1×1 bins.

Two maximization problems

Chapter 2 concludes with the analysis of the Time Series Search problem and the One-Way Trading problem.

Financial problems are natural problems for online analysis as often decisions must be made online and have important consequences.

The time series algorithm is an example of a simple *stopping rule* and stopping rules are a well studied topic (e.g., the prophet inequalities problem discussed in Chapter 16).

These two problems will also play an interesting role in our discussion of randomized algorithms in Chapter 3.

Both problems are parameterized by L and U, the minium and maximum exchange rates.

Time-series search and one-way trading

When L, U are known, we then obviously also know $\phi = \frac{U}{L}$ (but not conversely). The deterministic competitive ratios for both problems are stated in terms of ϕ. Without loss of generality lets take $U \geq L \geq 1$.

- In the time series problem we are trying convert all of our assets (say in US $ into Canadian $) in one day. Here we can achieve competitive ratio $\sqrt{\phi}$.
- In the one way trading problem, we are allowed to gradually trade the currency until some day when the trading ends. In this problem we can achieve competitive ratio $c(\phi) \log_2 \phi$ where $c(\phi) \rightarrow 1$ as $\phi \rightarrow \infty$.
- The algorthm indirectly uses the idea of partitioning the input space into a number of classes. We just saw the partitioning idea being used explicitly in the design and analysis of bin packing algorithms. In the one-way traading algorithm we have a logarithmic number of classes which is a commonly used idea.

The time-series search problem

- What if we don't know U and L? What if we only know ϕ? What if we do not have any a-priori information?

The time-series search problem

- What if we don't know U and L? What if we only know ϕ? What if we do not have any a-priori information?
- For determinstic time-series we need both U and L or else we will be forced to the trivial ratio ϕ. For one-way trading we can obtain an asymptotic $\log \phi$ ratio knowing only ϕ.
- Although not stated explictly anywhere (as far as I know), the time-series problem relates to the secretary problem as discussed in Chapter 16. In that chapter we consider stochastically generated input sequence and, in particular, the random order model (ROM). In ROM, we can achieve an absolute constant $\frac{1}{e}$ for time-series. (Note; here I am using our convention as to how constant maximization ratios are stated.)

Chapter 3: Randomized online algorithms

We have already seen that randomization is provably beneficial for online algorithms. We presented a randomized algorithmn for ski rental improving the constant from $2 - 1/b$ to $\frac{e}{e-1}$. In this chapter we will see more significant improvements as well as an example where randomization does not help.

We first need to provide a template for randomized algorithms. We will see that they can also be viewed as a probability mixture over deterministic algorithms. This is the same as how we view randomized strategies in game theory as a mixture over deterministic strategies.

When one is discussing randomized online algorithms we mean that the decisions are randomized (and not the order of the input items as in the ROM input sequencemodel).

In the case of deterministic algorithms there is one kind of adversary and the game between the algorithm and the adversary is a simple game since the adversary knows what the algorithm will do when given each input item.

Randomized algorithms and adversaries

For randomized online algorithms there are three kinds of adversaries. In order of increasing strength of the adversaries (and hence possibly decreasing how much randomization can help) we have:

1. An oblivious adversary
2. An adaptive online adversary
3. An adaptive offline adversary.

We will usually assume an oblivious adversary which is the only kind of adversary used for offline algorithms as far as I know.

The proportional knapsack problem

We will see a substantial improvemnent in the competitive ratio for the proporrtional knapsack even when using just one bit of randomization.

Without any randomization, no constant competitive ratio is possible.

But with one bit of randomization we obtain a 4-competitiive ratio.

In the time series problem, we can also see the same use of partitioning the inputs into classes and then randomly focusing on one class. In the time series problem we have a logarithmic (in ϕ) number of classes and we obtain a $O(\log \phi)$ ratio in contrast to the $\sqrt{\phi}$ ratio optimal ratio for deterministic algorithms.

Interestingly this simple randomized algorithm can be "de-randomized" and the de-randomized algorithm becomes the determininstic one-way trading algorithm.

It is an interesting research question to understand what can and what can't be de-randomized and what does this even mean.

The paging problem

As we recall, for deterministic paging (with k pages) the optimal competitive ration is k. With randomization we can achieve the ratio $H_k \approx \ln n$

The idea is to use randomization within a marking algorithm. The basic way to do this obtain $2H_n$ but H_n can be acheived by a somwwhat more involved algorithm.

The paging problem is a special case of the *k-server problem* for the case when the metric space is the uniform metric.

The fact that paging can enjoy an exponential improvement (from k to H_k) in the competitive ratio has led to a very significant stream of research ideas trying to achieve the same exponential improvement for every k-server problem.