CSC2421 Topics in Algorithms: Online and Other Myopic Algorithms Fall 2019

Allan Borodin

December 4, 2019

Announcements

- This is our last week. We start with Gregory discussing dynamic algorithms.
- Then I will discuss some additional stochastic input problems.
- I will then discuss some other adversarial and stochastic problems that can be viewed as an extension of the online model.

Stochastic probing and stochastic rewards

A different type of stochastic optimization problem concerns *stochastic probing*. In stochastic probing, input items come with a probability, and to reveal the actual item, an algorithm must probe the item.

In some probing variants, once the item is probed, the algorithm is *committed* to the item. In other variants, there may be some associated number of probes that are allowed,

Lets consider the BMM problem in the *stochastic-commit* model and call this the *stochastic-rewards* problem as it is sometimes referred to.

In the stochastic rewards with patience problem, when an online vertex u arrives all adjacent edges $e = (u, v)$ are labelled with a probability $p_e = p_{(u,v)}$ which denotes the probability that the edge exists. Furthermore, u has a patience constraint t_u.

Stochastic rewards continued

In order to attempt to match u to some available offline v, an online algorithm must probe the edges adjacent to u in some order. The patience constraint bounds the number of times edges adjacent to u can be probed.

Once a probed edge actually exists, the online algorithm is commtted to using that edge in the matching. The offline vertices may or may not be weighted.

When the patience parameter $t_u = 1$ for all online vertices, this is called the stochastic rewards problem without patience or simply the stochastic rewards problem..

What is the benchmark against which the online algorithm is competing? One urealistic benchmark is that the adversary-offline algorithm get to see the instance graph. Then consider an adversarial type graph with one online node having paience 1 and n offline vertices such that each edge has probability $1/n$. Then the graph will have an edge in the instance graph with probability $1 - [(1 - \frac{1}{n}]^n \approx 1 - \frac{1}{e}$ which is then the expected offline reward whereas the online algorithm only has expected reward $1/n$.

A better benchmark

One often uses an optimal LP fractional solution to some apprpriate relaxation of the matching problem to obtain an upper bound on a benchmark solution.

It seems like a good "natural" extension to the non-stochastc benchmark (i.e. an optimal solution) is the following: The adversary sets the probabilities of the edges and the benchmark can reveal edges in any order. In particular, it can look at the online nodes in any order. The only offline constraints are that the benchmark is also limited by the patience constraints of node and the commitment to any probed edge that exists.

The stochastic rewards without patience originated in Mehta and Panigraphi [2012]. (Their LP benchmark that can be argued is too powerful.)

I am following Brubach et al [2019] who consider the *stochastic rewards with patience problem with offline vertices weighted*. They articulated the "natural benchmark" above and use a more restricted LP to upper bound such a benchmark.

The Brubach et al results

Brubach et al follow a 2019 paper by Goyal and Udwani [2019] which claims to be using the same benchmark as in Brubach et al. However, it appears that the benchmark is even more restricted in making the vertices appear in the online order. That is a meaningful benchmark but I think the Brubach et al benchmark is more natural in the sense that it specializes to the usual offline OPT when all adjacent edges appear with probability 1.

First Brubach et al show that the "naive greedy" algorithm will not have any constant competitive ratio. The naive greedy algorithm will probe available edges in the order of non-increasing $p_{u,v} w_v$.

To see this, consider having one online vertex u and n offline vertices with weights $p_{v_1} = w_{v_1} = 1$ and $w_{v_i} = n - 2$ and $p_{v_i} = \frac{1}{n-1}$ for $i = 2, \ldots n$. The naive greedy algorithm gets reward 1 while probing all edges (u, v_i) for $i \geq 2$ first obtains expected reward
$(1 - \frac{1}{n-1})^{n-1} \cdot (n-2) + 1 \approx (1 - \frac{1}{e}) \cdot (n-1)$.

The Brubach et online algorithm for stochastic rewards with patience

We are considering an online algorithm and hence does *not* know future online nodes. So we only need to consider any given online node u with patience t_u and edges $(u, v_1), \ldots, (u, v_n)$ with corresponding edge probabilities $p(u, v_i)$. This is a star graph (also called a n-claw) with center u and "talons '' $\{v_i\}$. The intuition is that with large patience we can take a chance on a low probability (u, v_i) if the weight of v_i is big.

The optimal probing sequence (for each online vertex) will be calculated by dynamic programming (DP). It will essentially find an optimal subset $V' \subset V$ of size t_u and then probe the nodes in V' in order of non-increasing weights. Once an optimal subset V' is determined , it i optimal to probe thsat sequence in the specified order.

As is standard in DP, one computes the optimal (expected) value and then there are various ways to extend the DP to compute a corresponing solution.

The DP for computing the optimal probing sequence for an online node

Assume $w_{v_1} \geq w_{v_2}, \ldots \geq w_{v_n}$

The main idea is to define an apprpriate array and here we can define $V[i,t]$ the optimal expected value for probing the edges given patience t and starting from v_i.
The recursion for computing $V(i,t)$ is:
(1) $V(i,t) = p_{u,v_i} \cdot w_{v_i}$ if $t = 1$
(2) $V(i,t) = p_{u,v_i} w_{v_i} + (1 - p_{u,v_i}) \cdot \max_{j>i} V(j, t-1)$ for $t > 1$

The desired entry is $\max_i V[i,t]$

The adaptivity gap

Within TCS, stochastic probing was initiated in an article by Dean et al [2008] who considered a *stochastic knapsack problem*. The setting here is closer to that of priority (i.e, myopic) algorithms rather online algorithms. The inputs are pairs (v_i, s_i) where the size s_i is drawn from a distribution F_i.

The items are either read in a fixed order or an adaptive order, that is, an ordering depending on the previous drawn items. The main thing I want to mention here is that Dean et al define the *adaptivity gap*, the gap between the expected values obtaioned by a fixed order algorithm vs an optimal adaptive ordeer algoritihm. (The paper doesn't discuss the nature of what is alllowed as an ordering but one could at least argue that the fixed order should be a priority order as defined in fixed priority algorithms).

The adaptivity gap for stochastic rewards

The Brubach et al paper adopt and study the concept of the adaptivity gap with regard to the stochastic rewards with patience problem. Their adaptivity gap compares an online algorithm using a fixed ordering of edges for each online node vs their benchmark. That is, the ordering of edges cannot depend on the previous history of revealed edges. Hence, their fixed online algorithms will try to probe an edge even if it is adjacent to an already matched offline vertex. This seems too restrictve to me.

Here the adaptivity gap is comparing a non adaptive online algorithm that knows the stochastic type graph) to the benchmark, that is the adaptive offline algorithm that also knows the type graph.

Finally, as far as I know, the only inapproximation bounds for stochastic rewards is obtained by what we know for the i.i.d. setting.

The Pandora's box problem

Another interesting stochastic probing problem is the so-called Pandora's box problem. There are variants but here is the original offline problem as defined (and optimally solved) by Weitzman.

There are n boxes and each box i contains an independent (but not necessarily identical) random value v_i and a fixed cost c_i.

In the original problem, an offline algorithm knows the distributions and can choose the order of the boxes. The algorithm has the option to open any box i at cost c_i. At some point it stops and collects the maximum value in any box that has been opened having paid all the costs for opened boxes. The goal is to optimize the expected profit defined as $\mathbb{E}[\max_{i \in S} v_i] - \sum_{i \in S}$ where S is the set of opened boxes.

For the original problem, Weizman [1979] defined a reservation price v_i^* for each box i. The algorithm then sorts the boxes so that $v_1^* \geq v_2^* \ldots \geq v_n^*$ and stops whenever the maximum value of opened boxes exceeds the reservation price of any so far unopened box. This turns out to be an optimal stopping rule.

The online Pandora's box problem

In the online Pandora's box problem, the algorithm is given the boxes in an adversarial order. Like the prophet inequalities problem, it knows all the distributions in advance. It can decide to open a box (at a cost) or not open a box. At any time the algorithm can stop and receive the max value opened and pay the costs for all opened boxes.

What is the benchmark? We cannot use a fully clairvoyant benchmark that knows the best boxes to open (in terms of the expected net value obtained). Such a benchmark will result in an unbounded competitive ratio as observed by the following example: $c_i = 1$ for all i and all the distributions are identical such that $v_i = 0$ with probabiiity $1/2$ and $v_i = 2$ with probability $1/2$.

The expected value for opening any box is 1. Hence any oniine algorithm will have expected profit at most 1. The benchmark will see a box having $v_i = 2$ with probability $1 - 2^n$ and hence has expected profit that approaches 2 as $n \to \infty$.

A generalized online Pandora's box problem

While the original Pandora's box and prophet inequalities problems were only seeking a single item, these problems have been extended to consider selecting a set of items subject to some constraint (as we discussed for the generalized secretary problems).

In a recent paper, Esfandiari et al [2019] show how to reduce this generalized Pandora's box problem (for any constraint) to the problem of finding a threshold based algorithm for the correspnding generalized prophet inequalities problem (i.e. Pandora's box where all box costs are 0).

We will end the term with a listing of recent activity in the field of online algorithms

The field of online algorithms remains an active field of research where :

- Some existing problems have improved solutions often requiring substantial new ideas.
- Some existing problems are substantially generalized.
- New problems are being introduced.

Matching tasks to workers

Two interesting problems concern matching tasks (arriving online) to experts/workers. In each of these setting, a task consists of a subset of skills needed for completing the task.

Anagnostopoulos et al [2012] consider the following problem:
There are m skills and every task consist of some subset of required skills. That is a task T is a vector in $\{0,1\}^m$ where $T_i = 1$ iff the i^{th} skill is required for the task T.

There is a set $\mathcal{P}$ of experts who are in a social network $G = (\mathcal{P}, E, d)$ where d is a metric distance function on the edge representing a measure of closeness amongst pairs of experts. An expert posseses a subset of skills. That is, an expert P is also a vector in $\{0,1\}^m$ so that $P_i = 1$ iff P has the i^{th} skill.

Given an online sequence of tasks $(T^1, T^1, \ldots, T^n)$, each task T^j must be assigned to a team Q^j of experts so that the task is "covered" by the team. That is, for every skill i such that $T^j_i = 1$ there is an expert $P \in Q^j$ such that $P_i = 1$.

The objectives in assigning experts to tasks

There are two possibly conflicting objectives that we want the assignment of tasks to experts to satisfy: namely, we want to mimnimize a desired *allocation cost* (depending on the load assigned to each expert) while covering all tasks and keeping the *coordination cost* (depending on the social network distances) within some bound.

There are different ways to define allocation costs and coordination costs. But the basic idea is that the experts need to communicate and work well tegether in order to complete a task.

The paper provides competitive bunnds for different allocation and coordination costs. The paper also provides some experiments showing the tradeoffs between allocation and coordination (eg for diameter vs max load).

Another problem assigning workers to tasks

Anagnostopoulos et al [2018] (not quite the same set of authors) consider another setting where now there are tw0 types of workers, those that are hired and freelancers. As in the previous problem, online tasks consist of a subset of required skills and workers possess a subset of skills. Each online task must be covered by at least one worker, whether hired or a freelancer.

Hired workers have a salary (paid for each task) whether they are used or not as well as a hiring cost (including possible termination) whereas a freelancer is paid once for each task.

The objective is to minimize the total cost for covering each task.

The k-taxi problem

The k taxi problem is a generalization of the k-server problem. In this problem, a request is a pair of points (s, t) in a metric space which represents a passenger who wants to be driven from s to t.

As in the k-server problem, a request must be served by one of the k servers (i.e., taxis). In the easy version, the cost of serving a request is the total cost of the serverv travelling first to s and then to t. This problem can be reduced to the k-server problem.

In the hard version, the cost of serving a request is just the distance travelled without a passenger. In the hard version, the current best randomized competitive ratio for general n point metric spaces is $O(2^k \log n) where$.

For specific metric spaces (e.g. the line), one can do better. Specifically, there is a constant competitive ratio for the 3-taxi problem on the line.

Online bipartite matching variants relating to online advertising

As stated at the start of the course, one of the main reasons for the continuing research interest in online algorithms is due to online advertsing in search engines. This research has led to many variants of the basic online bipartite matching problem.

In this regard, problems are studied in both the adversarial (worst case instances) setting as well in stochastic settings. The settings that are arguably most appropriate for online advertising is for edge weighted bipartitie graphs which will preclude any constant competitive ratios for worst case instances in many settings.

Adwords and Display Ads

Two well studied online advertising models are the adwords problem and the display ads problem. We can think of advertisers being the offline nodes and online items being search terms, ads, etc.

- Adwords: There is a bid (i.e., profit) b_{ij} for matching item j with advertiser i. When $b_{ij} = b_i$ for all i, this is the vertex weighted bipartite mathching problem. In adwords, the usual assumption is that every advertisers has a budget B_i and that each bid $b_{i,j} << B_i$.
- Display ads: Every advertiser has a capacity C_i which is the a bound on the number of items that an advertiser can be allocated. In the case of "free dsposal", an advertiser can be assaigned any number of items but the profit is defined as sum of the C_i highest items assigned.

Table in Mehta surver updated by Chris Karavasilis

	ADV	ROM	Unknown IID	Known IID
Adwords (small bids)	$1 - 1/e$ (optimal) [8]	$1 - \epsilon$ [9]	$1 - \epsilon$	$1 - \epsilon$
Adwords (large bids)	1/2	0.51 [1]	$1 - 1/e$	$1 - 1/e$
Display Ads with Free Disposal (large capacities)	$1 - 1/e$ (optimal) [5]	$1 - \epsilon$ [7]	$1 - \epsilon$	$1 - \epsilon$
Display Ads with Free Disposal (general capacities)	0.5018 [2]	0.51 [1]	$1 - 1/e$	0.705
Display Ads without Free Disposal	0	$1/e$ (optimal) [6]	$1/e$	0.705 [4]
Submodular Welfare Maximization	1/2 (optimal) [3]	0.505 [1]	$1 - 1/e$ (optimal) [3]	$1 - 1/e$

Bipartite stochastic matching as an extension of prophet inequalities

Alaei, et al [2012] consider adwords and display ads in a setting where the online algorithm knows the independent (but not necessarily) identical distributions F_j of the j^{th} item.

They show the following:

- For display ads without free disposal, if $C_i \geq k$, then the competitive ratio is at least $1 - \frac{1}{\sqrt{k+3}}$
- For adwords, if $b_{ij} \leq \frac{B_i}{k}$, then the competitive ratio is at least $1 - \frac{1}{\sqrt{2\pi k}}$.

In both cases, the ratio approaches 1 as k gets large. The display ads result generalizes the original $\frac{1}{2}$ ratio for prophet inequalities.