CSC2420: Algorithm Design, Analysis and Theory Fall 2017

Allan Borodin and Nisarg Shah

October 25, 2017

Lecture 7

Announcements:

- The first two questions for assignment 2 have been posted.
- I hope to add one or two more questions this week.
- This week is my last lecture and then Nisarg Shah will do the remaining lectures.
- In case I mentioned it, there was a recent paper claiming a proof that $P \neq NP$. However, that proof now seems to have been refuted.

Todays agenda

- Continue randomized algorithms for max-sat
    1. Review of naive randomized algorithm for weghted max-sat.
    2. Johnson's deterministic Max-Sat algorithm
    3. Online randomized $\frac{3}{4}$ approximation for max-sat
    4. Yannakakis randomized LP rounding for max-sat
    5. The KVV algorithm for online unweighted bipartite matching
    6. The Buchbinder et al two sided online greedy algorithm and application to max-sat.
- More ways to de-randomize
    1. Buchbinder and Feldman De-randomization using parallel streams
    2. Online with advice and relation to randomized online algorithms
    3. De-randomization using two and multi pass algorithms
- ROM, i.i.d. online models
- The landscape for vertex and edge weighted online bipartite matching.

Review: The naive randomized algorithm for exact Max-k-Sat

We continue our discussion of randomized algorithms by considering the use of randomization for improving approximation algorithms. In this context, randomization can be (and is) combined with any type of algorithm. We will mainly focus today on using randomization for online algorithms and how that can possibly lead to a deterministic algorithm.

- We recall the weigted exact Max-k-Sat problem where we given a CNF propositional formula in which every weighted clause has exactly k literals. The goal is to find a satisfying assignment that maximizes the size (or weight) of clauses that are satisfied.
- Since exact Max-k-Sat generalizes the exact k- SAT decision problem, it is clearly an NP hard problem for $k \geq 3$. It is interesting to note that while 2-SAT is polynomial time computable, Max-2-Sat is still NP hard.
- The naive randomized (online) algorithm for Max-k-Sat is to randomly set each variable to *true* or *false* with equal probability.

Analysis of naive Max-k-Sat algorithm continued

- Since the expectation of a sum is the sum of the expectations, we just have to consider the probability that a clause is satisfied to determine the expected weight of a clause.
- Since each clause C_i has k variables, the probability that a random assignment of the literals in C_i will set the clause to be satisfied is exactly $\frac{2^k-1}{2^k}$. Hence $\mathbf{E}$ [weight of satisfied clauses] $= \frac{2^k-1}{2^k}\sum_i w_i$
- Of course, this probability only improves if some clauses have more than k literals. It is the small clauses that are the limiting factor in this analysis.
- This is not only an approxination ratio but moreover a "totality ratio" in that the algorithms expected value is a factor $\frac{2^k-1}{2^k}$ of the sum of all clause weights whether satisfied or not.
- We can hope that when measuring against an optimal solution (and not the sum of all clause weights), small clauses might not be as problematic as they are in the above analysis of the naive algorithm.

Derandomizing the naive algorithm

We can derandomize the naive algorithm by what is called the method of conditional expectations. Let $F[x_1, \ldots, x_n]$ be an exact k CNF formula over n propositional variables $\{x_i\}$. For notational simplicity let *true* $= 1$ and *false* $= 0$ and let $w(F)|\tau$ denote the weighted sum of satisfied clauses given truth assignment τ.

- Let x_j be any variable. We express $\mathbf{E}[w(F)|_{x_i \in_u \{0,1\}}]$ as $\mathbf{E}[w(F)|_{x_i \in_u \{0,1\}} | x_j = 1] \cdot (1/2) + \mathbf{E}[w(F)|_{x_i \in_u \{0,1\}} | x_j = 0] \cdot (1/2)$
- This implies that one of the choices for x_j will yield an expectation at least as large as the overall expectation.
- It is easy to determine how to set x_j since we can calculate the expectation clause by clause.
- We can continue to do this for each variable and thus obtain a deterministic solution whose weight is at least the overall expected value of the naive randomized algorithm.
- NOTE: The derandomization can be done so as to achieve an online algorithm. Here the (online) input items are the propostional variables. What input representation is needed/sufficient?

The derandomization of the naive algorithm is Johnson's algorithm

In [1974] Johnson published a determinstic algorithm for Weighted Max-Sat. Johnson showed that the algorithm achieves an approximation of at least $1/2$.

As mentioned, the analysis of the de-randomization of the naive algorithm does not immdiately yield a better bound because of the possible existence of clauses containing only one literal.

Twenty years after Johnson's algorithm, Yannakakis [1994] presented the naive algorithm and showed that Johnson's algorithm is the derandomized naive algorithm.

Yannakakis also observed that for arbitrary Max-Sat, the approximation of Johnson's algorithm is at best $\frac{2}{3}$. For example, consider the 2-CNF $F = (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge \bar{y}$ when variable x is first set to true.

Chen, Friesen, Zheng [1999] showed that Johnson's algorithm achieves approximation ratio $\frac{2}{3}$ for arbitrary weighted Max-Sat.

Johnson's Max-Sat Algorithm

Johnson's [1974] algorithm

```
For all clauses $C_i$, $w'_i := w_i/(2^{|C_i|})$
Let $L$ be the set of clauses in formula $F$ and $X$ the set of variables
**For** $x \in X$ (or until $L$ empty)
   Let $P = \{C_i \in L$ such that $x$ occurs positively$\}$
   Let $N = \{C_j \in L$ such that $x$ occurs negatively$\}$
   **If** $\sum_{C_i \in P} w'_i \geq \sum_{C_j \in N} w'_j$
     $x := true; L := L \setminus P$
     **For all** $C_r \in N$,   $w'_r := 2w'_r$   **End For**
   **Else**
     $x := false; L := L \setminus N$
     **For all** $C_r \in P$,   $w'_r := 2w'_r$   **End For**
   **End If**
   Delete $x$ from $X$
**End For**
```

Aside: This reminds me of boosting (Freund and Shapire [1997])

Modifying Johnson's algorithm for Max-Sat

- In proving the $(2/3)$ approximation ratio for Johnson's Max-Sat algorithm, Chen et al asked whether or not the ratio could be improved in the random order model (ROM); that is, by using a random ordering of the propositional variables.
- The question asked by Chen et al was answered by Costello, Shapira and Tetali [2011] who showed that in the ROM model, Johnson's algorithm achieves approximation $(2/3 + \epsilon)$ for $\epsilon \approx .003653$
- Poloczek and Schnitger [same SODA 2011 conference] show that the approximation ratio for Johnsons algorithm in the ROM model is at most $2\sqrt{157} \approx .746 < 3/4$, the ratio first obtained by Yannakakis' IP/LP approximation that we will soon present.
- Moreover, Poloczek and Schnitger presented an online randomized algorithm (called SLACK) that achieves a $\frac{3}{4}$ approximation.

The idea of the SLACK algorithm

Poloczek and Schnithger first consider a "canonical randomization" of Johnson's algorithm"; namely, the canonical randomization sets a variable $x_i = \textit{true}$ with probability $\frac{w'_i(P)}{w'_i(P)+w'_i(N)}$ where $w'_i(P)$ (resp. $w'_i(N)$) is the current combined weight of clauses in which x_i occurs positively (resp. negatively).

Their substantial additional idea is to adjust the random setting so as to better account for the weight of unit clauses in which a variable occurs.

For arbitrary Max-Sat (resp. Max-2-Sat), the current best approximation ratio is .7968 (resp. .9401) using semi-definite programming and randomized rounding.
Note: While existing combinatorial algorithms do not come close to these best known ratios, it is still interesting to understand simple and even online algorithms for Max-Sat.

What are the limitations of deterministic simple max-sat algorithms?

- To precisely model the Max-Sat problem within the online, ROM, and priority frameworks, we need to specify the input model.
- In increasing order of providing more information (and possibly better approximation ratios), the following input models can be considered:

Model 0 Each propositional variable x is represented by the names of the positive and negative clauses in which it appears.

Model 1 Each propositional variable x is represented by the length of each clause C_i in which x appears positively, and for each clause C_j in which it appears negatively. This is sufficient for the naive algorithm and its derandomization.

Model 2 In addition, for each C_i and C_j, a list of the other variables in that clause is specified.

Model 3 The variable x is represented by a complete specification of each clause it which it appears.

Can the Slack algorithm be de-randomized?

- SLACK is a randomized online algorithm (i.e. adversary chooses the ordering) where the variables can be represented within input model 1.
- This approximation ratio is in contrast to Azar et al [2011] who prove that no randomized online algorithm can achieve approximation better than 2/3 when the input model is the weakest (i.e. Model 0) of the input models.
- Poloczek [2011] shows that no deterministic priority algorithm can achieve a 3/4 approximation within the input model 2. This provides a sense in which to claim that the Poloczek and Schnitger Slack algorithm "cannot be derandomized".
- The best deterministic priority algorithm in the third (most powerful) model remains an open problem as does the best randomized priority algorithm and the best ROM algorithm.

Revisiting the “cannot be derandomized comment”

Spoiler alert: we will be discussing how algorithms that cannot be derandomized in one sense can be derandomized in another sense.

- The Buchbinder et al [2012] online randomized $1/2$ approximation algorithm for Unconstrained Submodular Maximization (USM) cannot be derandomized into a “similar” deterministic algorithm by a result of Huang and Borodin [2014].
- However, Buchbinder and Feldman [2016] show how to derandomize the Buchbinder et al algorithm into a polynomial time online deterministic algortihm.
- The Buchbinder et al USM algorithm is the basis for a randomized $3/4$ approximation online MaxSat (even Submodular Max Sat) algorithm.
- Pena and Borodin show how to derandomize this $3/4$ approximation algorithm following the approach of Buchbinder and Feldman.
- Poloczek et al [2017] de-randomize an equivalent Max-Sat algorithm using a 2-pass online algorithm.

Yannakakis' IP/LP *randomized rounding* algorithm for Max-Sat

- We will formulate the weighted Max-Sat problem as a $\{0,1\}$ IP.
- Relaxing the variables to be in $[0,1]$, we will treat some of these variables as probabilities and then round these variables to 1 with that probability.
- Let F be a CNF formula with n variables $\{x_i\}$ and m clauses $\{C_j\}$. The Max-Sat formulation is :
  maximize $\sum_j w_j z_j$
  subject to $\sum_{\{x_i \text{ is in } C_j\}} y_i + \sum_{\{\bar{x}_i \text{ is in } C_j\}} (1 - y_i) \geq z_j$
  $y_i \in \{0,1\}$; $z_j \in \{0,1\}$
- The y_i variables correspond to the propositional variables and the z_j correspond to clauses.
- The relaxation to an LP is $y_i \geq 0$; $z_j \in [0,1]$. Note that here we cannot simply say $z_j \geq 0$. Why?

Randomized rounding of the y_i variables

- Let $\{y_i^*\}, \{z_j^*\}$ be the optimal LP solution,
- Set $\tilde{y}_i = 1$ with probability y_i^*.

Theorem

Let C_j be a clause with k literals and let $b_k = 1 - (1 - \frac{1}{k})^k$. Then $Prob[C_j$ is satisifed $]$ is at least $b_k z_j^*$.

- The theorem shows that the contribution of the j^{th} clause C_j to the expected value of the rounded solution is at least $b_k w_j$.
- Note that b_k converges to (and is always greater than) $1 - \frac{1}{e}$ as k increases. It follows that the expected value of the rounded solution is at least $(1 - \frac{1}{e})$ LP-OPT $\approx .632$ LP-OPT.
- Taking the max of this IP/LP and the naive randomized algorithm results in a $\frac{3}{4}$ approximation algorithm that can be derandomized. (The algorithm can be de-randoimized but the de-randomized algorithm will still be solving LPs.)

Submodular maximization problems; An important diversion before returning to MaxSat

- A set function $f : 2^U \rightarrow \Re$ is submodular if $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$ for all $S, T \subseteq U$.
- Equivalently, f is submodular if it satisfies decreasing marginal gains; that is,
  $f(S \cup \{x\}) - f(S) \geq f(T \cup \{x\}) - f(T)$ for all $S \subseteq T \subseteq U$ and $x \in U$
- We will always assume that f is *normalized* in that $f(\varnothing) = 0$ and non-negative.
- Submodular functions arise naturally in many applications and has been a topic of much recent activity.
- Probably the most frequent application of (and papers about) submodular functions is when the function is also monotone (non-decreasing) in that $f(S) \leq f(T)$ for $S \subseteq T$.
- Note that linear set functions (also called modular) functions (i.e. when $f(S \cup \{x\}) - f(S) = f(T \cup \{x\}) - f(T)$) are a special case of monotone submodular functions.

Submodular maximization continued

In the submodular maximization problem, we want to compute S so as to maximize $f(S)$.

- For monotone functions, we are maximizing $f(S)$ subject to some constraint (otherwise just choose $S = U$).
- For the non monotone case, the problem is already interesting in the unconstrained case. Perhaps the most prominent example of such a problem is Max-Cut (and Max-Di-Cut).
- Max-Cut is an NP-hard problem. Using an SDP approach for Max-Cut (and Max-2-Sat) yields the approximation ratio $\alpha = \frac{2}{\pi} \min_{\{0 \leq \theta \leq \pi\}} \frac{\theta}{(1-cos(\theta))} \approx .87856$. Assuming UGC, this is optimal.
- For a submodular function, we may be given an explicit representation (when a succinct representation is possible as in Max-Cut) or we access the function by an oracle such as the *value oracle* which given S, outputs the value $f(S)$; such an oracle call is considered to have $O(1)$ cost. Other oracles are possible (e.g. given S, output the element x of U that maximizes $f(S \cup \{x\}) - f(S)$).

Unconstrained (non monotone) submodular maximization

- Feige, Mirrokni and Vondrak [2007] began the study of approximation algorithms for the unconstrained non monotone submodular maximization (USM) problem establishing several results:
  1. Choosing S uniformly at random provides a $1/4$ approximation.
  2. An oblivious local search algorithm results in a $1/3$ approximation.
  3. A non-oblivious local search algorithm results in a $2/5$ approximation.
  4. Any algorithm using only value oracle calls, must use an exponential number of calls to achieve an approximation $(1/2 + \epsilon)$ for any $\epsilon > 0$.
- The Feige et al paper was followed up by improved local search algorithms by Gharan and Vondrak [2011] and Feldman et al [2012] yielding (respectively) approximation ratios of .41 and .42.
- The $(1/2 + \epsilon)$ inapproximation was augmented by Dobzinski and Vondrak showing the same bound for an explicitly given instance under the assumption that $RP \neq NP$.

The Buchbinder et al (1/3) and (1/2) approximations for USM

In their FOCS [2012] (and SICOMP [2015]) paper , Buchbinder et al gave an elegant linear time deterministic $1/3$ approximation and then extend that to a randomized $1/2$ approximization. The conceptually simple form of the algorithm is (to me) as interesting as the optimality (subject to the proven inapproximation results) of the result. Let $U = u_1, \ldots u_n$ be the elements of U *in any order*.

The deterministic 1/3 approximation for USM

```
$X_0 := \varnothing$; $Y_0 := U$
For $i := 1 \ldots n$
  $a_i := f(X_{i-1} \cup \{u_i\}) - f(X_{i-1})$; $b_i := f(Y_{i-1} \setminus \{u_i\}) - f(Y_{i-1})$
  If $a_i \geq b_i$
    then $X_i := X_{i-1} \cup \{u_i\}$; $Y_i := Y_{i-1}$
    else $X_i := X_{i-1}$; $Y_i := Y_{i-1} \setminus \{u_i\}$
  End If
End For
```

The randomized 1/2 approximation for USM

- Buchbinder et al show that the “natural randomization” of the previous deterministic algorithm achieves approximation ratio $1/2$.
- That is, the algorithm chooses to either add $\{u_i\}$ to X_{i-1} with probability $\frac{a'_i}{a'_i+b'_i}$ or to delete $\{u_i\}$ from Y_{i-1} with probability $\frac{b'_i}{a'_i+b'_i}$ where $a'_i = \max\{a_i, 0\}$ and $b'_i = \max\{b_i, 0\}$.
- If $a_i = b_i = 0$ then add $\{u_i\}$ to X_{i-1}.
- Note: Part of the proof for both the deterministic and randomized algorithms is the fact that $a_i + b_i \geq 0$.
- This fact leads to the main lemma for the deterministic case:

$$f(OPT_{i-1} - f(OPT_i) \leq [f(X_i - f(X_{i-1}] + [f(Y_i) - f(Y_{i-1}]$$

Here $OPT_i = (OPT \cup \{X_i\}) \cap Y_i$ so that OPT_i coincides with X_i and Y_i for elements $1, \ldots i$ and coincides with OPT on elements $i+1, \ldots, n$. Note that $OPT_0 = OPT$ and $OPT_n = X_n = Y_n$. That is, the loss in OPTs value is bounded by the total value increase in the algorithm's solutions.

Applying the algorithmic idea to Max-Sat

Buchbinder et al are able to adapt their randomized algorithm to the Max-Sat problem (and even to the Submodular Max-Sat problem). So assume we have a monotone normalized submodular function f (or just a linear function as in the usual Max-Sat). The adaption to Submodular Max-Sat is as follows:

- Let $\phi : X \to \{0\} \cup \{1\} \cup \varnothing$ be a standard partial truth assignment. That is, each variable is assigned exactly one of two truth values or not assigned.
- Let $\mathcal{C}$ be the set of clauses in formula Ψ. Then the goal is to maximize $f(\mathcal{C}(\phi))$ where $\mathcal{C}(\phi)$ is the sat of formulas satisfied by ϕ.
- An extended assignment is a function $\phi' : X \to 2^{\{0,1\}}$. That is, each variable can be given one, two or no values. (Equivalently $\phi' \subseteq X \times \{0,1\}$ is a relation.) A clause can then be satisfied if it contains a positive literal (resp. negative literal) and the corresponding variable has value $\{1\}$ or $\{0,1\}$ (resp. has value $\{0\}$ or $\{0,1\}$.
- $g(\phi') = f(\mathcal{C}(\phi'))$ is a monotone normalized submodular function. '

Buchbinder et al Submodular Max-Sat

Now starting with $X_0 = X \times \varnothing$ and $Y_0 = Y \times \{0, 1\}$, each variable is considered and set to either 0 or to 1 (i.e. a standard assignment of precisely one truth value) depending on the marginals as in USM problem.

```
Algorithm 3: RandomizedSSAT($f, \Psi$)
1  $X_0 \leftarrow \emptyset$, $Y_0 \leftarrow \mathcal{N} \times \{0, 1\}$.
2  for $i$ = 1 to $n$ do
3      $a_{i,0} \leftarrow g(X_{i-1} \cup \{u_i, 0\}) - g(X_{i-1})$.
4      $a_{i,1} \leftarrow g(X_{i-1} \cup \{u_i, 1\}) - g(X_{i-1})$.
5      $b_{i,0} \leftarrow g(Y_{i-1} \setminus \{u_i, 0\}) - g(Y_{i-1})$.
6      $b_{i,1} \leftarrow g(Y_{i-1} \setminus \{u_i, 1\}) - g(Y_{i-1})$.
7      $s_{i,0} \leftarrow \max\{a_{i,0} + b_{i,1}, 0\}$.
8      $s_{i,1} \leftarrow \max\{a_{i,1} + b_{i,0}, 0\}$.
9      with probability $s_{i,0}/(s_{i,0} + s_{i,1})^*$ do:
       $X_i \leftarrow X_{i-1} \cup \{u_i, 0\}$, $Y_i \leftarrow Y_{i-1} \setminus \{u_i, 1\}$.
10     else (with the compliment probability
       $s_{i,1}/(s_{i,0} + s_{i,1})$) do:
11     $X_i \leftarrow X_{i-1} \cup \{u_i, 1\}$, $Y_i \leftarrow Y_{i-1} \setminus \{u_i, 0\}$.
12 return $X_n$ (or equivalently $Y_n$).
```

* If $s_{i,0} = s_{i,1} = 0$, we assume $s_{i,0}/(s_{i,0} + s_{i,1}) = 1$.

Further discussion of the Unconstrained Submodular Maximization and Submodular Max-Sat algorithms

- The Buchbinder et al [2012] online randomized $1/2$ approximation algorithm for Unconstrained Submodular Maximization (USM) cannot be derandomized into a "similar" deterministic online or priority style algorithm by a result of Huang and Borodin [2014]. Like the Poloczek result, we claimed that this was "in some sense" evidence that this algorithm cannot be derandomized.
- Their algorithm is shown to have a $\frac{3}{4}$ approximation ratio for Monotone Submodular Max-Sat.
- For the standard, weighted Max-Sat problem, Poloczek et al [2017] show that the Buchbinder et al algorithm turns out to be equivalent to a previous Max-Sat algorithm by van Zuylen.

The randomized (weighted) max-sat $\frac{3}{4}$ approximation algorithm

The idea of the algorithm is that in setting the variables, we want to balance the weight of clauses satisfied with that of the weight of clauses not yet unsatisfied.
Let S_i be the assignment to the first i variables and let SAT_i (resp. $UNSAT_i$) be the weight of satisfied clauses (resp., unsatsifed clauses) with respect to S_i. Let $B_i = \frac{1}{2}(SAT_i + W - UNSAT_i)$ where W is the total weight of all clauses.

The randomized (weighted) max-sat $\frac{3}{4}$ approximation algorithm

The idea of the algorithm is that in setting the variables, we want to balance the weight of clauses satisfied with that of the weight of clauses not yet unsatisfied.
Let S_i be the assignment to the first i variables and let SAT_i (resp. $UNSAT_i$) be the weight of satisfied clauses (resp., unsatsifed clauses) with respect to S_i. Let $B_i = \frac{1}{2}(SAT_i + W - UNSAT_i)$ where W is the total weight of all clauses.

The algorithm's plan is to randomly set variable x_i so as to increase $\mathbb{E}[B_i - B_{i-1}]$.

The randomized (weighted) max-sat $\frac{3}{4}$ approximation algorithm

The idea of the algorithm is that in setting the variables, we want to balance the weight of clauses satisfied with that of the weight of clauses not yet unsatisfied.
Let S_i be the assignment to the first i variables and let SAT_i (resp. $UNSAT_i$) be the weight of satisfied clauses (resp., unsatsifed clauses) with respect to S_i. Let $B_i = \frac{1}{2}(SAT_i + W - UNSAT_i)$ where W is the total weight of all clauses.

The algorithm's plan is to randomly set variable x_i so as to increase $\mathbb{E}[B_i - B_{i-1}]$.

To that end, let t_i (resp. f_i) be the value of $w(B_i) - w(B_{i-1})$ when x_i is set to true (resp. false).

The randomized max-sat approximation algorithm continued

```
For $i = 1 \ldots n$
   If $f_i \leq 0$, then set $x_i = $ true
   Else if $t_i \leq 0$,
     then set $x_i = $ false
   Else set $x_i$ true with probability $\frac{t_i}{t_i + f_i}$.
End For
```

The randomized max-sat approximation algorithm continued

```
For $i = 1 \dots n$
    If $f_i \leq 0$, then set $x_i =$ true
    Else if $t_i \leq 0$,
      then set $x_i =$ false
    Else set $x_i$ true with probability $\frac{t_i}{t_i + f_i}$.
End For
```

Consider an optimal solution (even an LP optimal) $\mathbf{x}^*$ and let OPT_i be the assignment in which the first i variables are as in S_i and the remaining $n - i$ variables are set as in $\mathbf{x}^*$. (Note: x^* is not calculated.)

The analysis follows as in Poloczek and Schnitger, Poloczek, and in Buchbinder et al. One shows the following:

- $t_i + f_i \geq 0$
- $\mathbb{E}[w(OPT_{i-1}) - w(OPT_i)] \leq \mathbb{E}[w(B_i) - w(B_{i-1})]$

De-randomizing the Max-Sat algorithm by a two pass online algorithm

The idea of the two pass de-randomization is that the first pass is setting the relevant probabilities t_i and f_i but not setting the truth assignment of any propositional variable.

Now the goal is to set the variables in the second pass using the method of conditional expectation. Sounds good?

De-randomizing the Max-Sat algorithm by a two pass online algorithm

The idea of the two pass de-randomization is that the first pass is setting the relevant probabilities t_i and f_i but not setting the truth assignment of any propositional variable.

Now the goal is to set the variables in the second pass using the method of conditional expectation. Sounds good?

But unlike the naive randomized algorithm, we do not have a probability for each level of the randomized tree as the probability $p_i(b_{i-1})$ for setting $x_i - 1$ depends on the partial assignment b_{i-1} to the first $i-1$ variables $x_1, \ldots, x_{i-1}$.

The additional substantial and interesting idea in Poloczek et al is to contract all the nodes at level $i-1$ into a super node where each node at that level is represented by the probability of the node being the one reached by the two pass algorithm.

The resulting 2-pass algorithm is deterministic and "online" in the sense

The Buchbinder and Feldman derandomization of the USM algorithm

- Contrary to the Poloczek, (resp. Huang and B.) priority inapproximations for Max-Sat (resp. USM), there is a sense in which these algorithms can be derandomized.
- In fact the derandomization becomes an “online algorithm” in the sense that an adversary is choosing the order of the input variables. However rather than creating a single solution, the algorithm is creating a tree of solutions and then takng the best of these.
- The idea is as follows. The analysis of the randomized USM approximation bound shows that a certain linear inequality holds at each iteration of the algorithm. Namely,

  $$E[f(OPT_{i-1} - f(OPT_i)] \leq \frac{1}{2}E[f(X_i) - f(X_{i-1}) + f(Y_i) - f(Y_{i-1}]$$

  That is, the expected change in restricting OPT in an iteration (by setting the i^{th} variable) is bounded by the average change in the two values being maintained by the algorithm.

Continuing the Buchbinder and Feldman derandomization idea

- These inequalities induce two additional inequalties per iteration on the distributions of solutions that can exist at each iteration.
- This then gets used to describe an LP corresponding to these $2i$ constraints we have for the distributions that hold at each iteration of the algorithm.
- But then using LP theory again (i.e. the number of non-zero variables in a basic solution). It follows that we only need distributions with support $2i$ at each iteration rather than the naive 2^i that would follow from just considering the randomized tree.
- Finally, since there must be at least one distribution (amongst the final $2n$ distributions) for which the corresponding solution is at least as good as the expected value. Thus if suffices to take the max over a "small" number of solutions.

How much online paralellism is needed to maintain $\frac{3}{4}$ approximation? for Ma-Sat

In the recent (so far unpublihsed) journal version of the Buchbinder and Feldman paper, they show that width $O(\frac{1}{\epsilon})$ is sufficient to obtain a $(1-\epsilon)\frac{1}{2}$ approximation for their USM de-randomization.

It is likely that a similar result holds for the de-randomization of Max-Sat.

Pena and Boroodin show that for Max-Sat input model 2, exponential width would be required to obtain an approximation better than $\frac{3}{4}$. It is an open problem (just as in the case of one pass priority algorithms) if one can obtain the same inapproximation for the max-sat input model 3.

Randomized online bipartite matching

- We have seen evidence of the power of randomization for online algorithms in the context of the USM and MaxSat problems. We now consider the same issues for the online bipartite matching problem.
- Another nice sequence of results begins with a randomized online algorithm for bipartite matching due to Karp, Vazirani and Vazirani [1990]. We quickly overview some results in this area as it represents a topic of continuing interest.
- In the online bipartite matching problem, we have a bipartite graph G with nodes $U \cup V$. Nodes in U enter online revealing all their edges. A deterministic greedy matching produces a maximal matching and hence a $\frac{1}{2}$ approximation.
- It is easy to see that any deterministic online (i.e., adversarial order) algorithm cannot be better than a $\frac{1}{2}$ approximation even when the degree of every $u \in U$ is at most (equal) 2

The Ranking $(1 - \frac{1}{e})$ approximation algorithm

- The algorithm chooses a random permutation of the nodes in V and then when a node $u \in U$ appears, it matches u to the highest ranked unmatched $v \in V$ such that (u, v) is an edge (if such a v exists).
- **Note:** Making a random choice for each u is still only a $\frac{1}{2}$ approx.
- Equivalently, this algorithm can be viewed as a deterministic greedy (i.e. always matching when possible and breaking ties consistently) algorithm in the ROM model.
- That is, let $\{v_1, \ldots, v_n\}$ be any fixed ordering of the vertices and let the nodes in U enter randomly, then match each u to the first unmatched $v \in V$ according to the fixed order.
- To argue this, consider fixed orderings of U and V; the claim is that the matching will be the same whether U or V is entering online. **Note:** This is not a claim that there is any equivalence between deterministic ROM algorithms and deterministic online algorithms. It is just the special symmetrc nature of this algorithm that makes it possible to view the KVV randomized algorithm as a deterministic ROM algorithm.

Can KVV be derandomized using parallel online streams or multi pass algorithms?

As previously noted, there is an immediate correspondence between online algorithms using t bits of advice (in the advice tape model) and algorithms comprised of 2^t online streams (and then taking the best outcome).

There are two interesting and perhaps surprising results concerning the relation of randomized online algorithms and deterministic online algorithms with advice.

1. Böckenhauer et al show that a randomized online algorithm achieving an approximation ratio c on inputs of length n implies that there is a deterministic online algorithm using $O(\log n)$ advice achieving ratio $(1+\epsilon) \cdot c$ algorithm
2. Mikklesen show an equivalence between randomized online algorithms with ratio $c+\epsilon$) and deterministic algorithms with $o(n)$ advice and ratio $c+\epsilon$). This result requires a technical definition as to the nature of the online problem but suffice it to say the results hold for many or most online problems including the online bipartite matching problem.

The KVV result and recent progress

KVV Theorem

Ranking provides a $(1 - 1/e)$ approximation.

- Original analysis is not rigorous. There is an alternative proof (and extension) by Goel and Mehta [2008], and then another proof in Birnbaum and Mathieu [2008]. Other alternative proofs have followed.
- Recall that this positive result can be stated either as the bound for a particular deterministic algorithm in the stochastic ROM model, or as the randomized Ranking algorithm in the (adversarial) online model.
- KVV show that the $(1 - 1/e)$ bound is essentially tight for any randomized online (i.e. adversarial input) algorithm. In the ROM model, Goel and Mehta state inapproximation bounds of $\frac{3}{4}$ (for deterministic) and $\frac{5}{6}$ (for randomized) algorithms.

Some more progress with regard to bipartite matching in the ROM model

- In the ROM model, Karande, Mehta, Tripathi [2011] show that randomized Ranking achieves approximation at least .653 (beating $1 - 1/e$) and no better than .727. The approximation ratio was improved to .696 by Mahdian and Yan [2011]].
- Karande et al show that any ROM approximation result implies the same result for the unknown i.i.d. model and hence the known i.i.d. model.
- Manshadi et al give a .823 inapproximation for biparitie matching in the known i.i.d. distribution model. This implies the same inapproximation in the unknown i.i.d. and ROM models improving the $\frac{5}{6}$ inapproximation of Goel and Mehta.
- Dürr et al show that a deterministic 2 pass online algorithm can achieve ratio $\frac{3}{5}$. This can be extended to obtain a $\frac{F_{2k-1}}{F_{2k}} \approx .618$ approximation by a k pass algorithm. Here F_k is the k^{th} Fibonacci number.

Weighted extensions of online bipartite matching

- There is a large landscape (and continuing research) of weighted online bipartite matching problems such as the *adwords* and *display ads* problems motivated by applications to online advertising.
- Although slightly out of data, the survey by Mehta [2013] is an excellent reference. Note: The table in the survey identifies the ROM and unknown i.i.d. model and although there are no provable separations, there are problems where better results are known for the unknown i.i.d model.

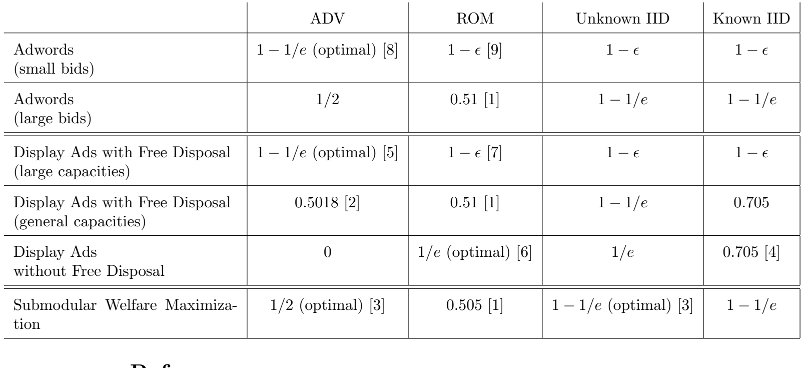
  The following problems can all be studied in the adversarial, ROM and i.i.d online input models. These weighted problems will be defined in the following slides.

  1. Vertex and edge weighted online matching
  2. Adwords with small and large (compared to the budget) bids.
  3. The Display Ads problem with and without free disposal and with small and large capacities. .
  4. The adwords problem with small bids can be reduced to the display ads problem with lage capacities. Both of these problems are generalized by the submodular welfare maximization problem.

The Mehta survey table

	Adversarial Order	Random Order/ unknown IID	Known IID
Bipartite matching	$1-\frac{1}{e}$ (optimal)	0.696 (?)	0.702 (0.823)
Vertex-weighted bipartite matching	$1-\frac{1}{e}$ (optimal)	$1-\frac{1}{e}$ (?)	$1-\frac{1}{e}$ (?)
Adwords (small bids)	$1-\frac{1}{e}$ (optimal)	$1-\epsilon$ (optimal)	$1-\epsilon$ (optimal)
Adwords (general bids)	$\frac{1}{2}$ (?)	$1-\frac{1}{e}$ (?)	$1-\frac{1}{e}$ (?)
Display Ads with free-disposal (large capacities)	$1-\frac{1}{e}$ (?)	$1-\frac{1}{e}$ (IID) (?)	$1-\frac{1}{e}$ (?)
Display Ads with free-disposal (general capacities)	$\frac{1}{2}$ (?)	$1-\frac{1}{e}$ (IID) (?)	$1-\frac{1}{e}$ (?)
Display Ads no free-disposal (general capacities)	0 (0)	$\frac{1}{e}$ (optimal)	? (?)
Submodular welfare	$\frac{1}{2}$ (optimal)	$1-\frac{1}{e}$ (IID) (?)	$1-\frac{1}{e}$ (?)

Updating the Mehta table



	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]	$1-1/e$

Figure : An updated table due to Chris Karavasilis. See end of slides for refereces to papers in the table.

Vertex weighted bipartite matching

- Aggarwal et al [2011] consider a vertex weighted version of the online bipartite matching problem. Namely, the vertices $v \in V$ all have a known weight w_v and the goal is now to maximize the weighted sum of matched vertices in V when again vertices in U arrive online.
- This problem can be shown to subsume the adwords problem when all bids $b_{q,i} = b_i$ from an advertiser are the same.
- It is easy to see that Ranking can be arbitrarily bad when there are arbitrary differences in the weight. Greedy (taking the maximum weight match) can be good in such cases. Can two such algorithms be somehow combined? Surprisingly, Aggarwal et al are able to achieve the same 1-1/e bound for this class of vertex weighted bipartite matching.

The vertex weighted online algorithm

The perturbed greedy algorithm

For each $v \in V$, pick r_v randomly in $[0, 1]$
Let $f(x) = 1 - e^{-(1-x)}$
When $u \in U$ arrives, match u to the unmatched v (if any) having the highest value of $w_v * f(x_v)$. Break ties consistently.

In the unweighted case when all w_v are identical this is the Ranking algorithm.

The classic edge weighted bipartite matching problem

In the offline setting, edge weighted bipartite matchinga (called the assignment problem) can be solved by the "Hungarian algorithm" which is based on a primal dual approach.

In the online adversarial setting it is easy to see that no constant approximation ratio is possible even for randomized algorithms.

In the ROM model, the problem can be viewed as an extension of the classical *secretary problem*. The secretary problem is the case of one offline node when studied in the ROM model. It is where the ROM model was first studied. It is known that $\frac{1}{e}$ is the optimal appoximation ratio for the secretary problem. Kesselheim et al [2013] show that the same optimal $\frac{1}{e}$ bound is possible for edge weighted bipartite matching in the ROM model.

I am now aware of any results for the adversaial and ROM versions of online biparitite matching when the *online* vertices are weighted.

The adwords problem: an extension of bipartite matching motivated by online auctions

- In the (single slot) adwords problem, the nodes in U are queries and the nodes in V are advertisers. For each query q and advertiser i, there is a bid $b_{q,i}$ representing the value of this query to the advertiser.
- Each advertiser also usually has a hard budget B_i which cannot be exceeded. The goal is to match the nodes in U to V so as to maximize the sum of the accepted bids without exceeding any budgets. Without budgets and when each advertiser will pay for at most one query, the problem then is edge weighted bipartite matching.
- In the online case, when a query arrives, all the relevant bids are revealed.

Some results for the adwords problem

- Here we are just considering the combinatorial problem and ignoring game theoretic aspects of the problem.
- The problem has been studied for the special (but well motivated case) that all bids are small relative to the budgets. As such this problem is incomparable to the matching problem where all bids are in $\{0,1\}$ and all budgets are 1.
- For this small bid case, Mehta et al [2005) provide a deterministic online algorithm achieving the $1 - 1/e$ bound and show that this is optimal for all randomized online algorithms (i.e. adversarial input).

The submodular welfare maximization problem

A generaliztion of both the Adwords problem and the Display Ads problem with free disposal is the online submodular welfare maximization problem. Submodular welfare maximization is well studied in auctions.

In the online version, the online vertices correspond to sellers (selling a unique item) and the offline vertices crrespnd to buyers where each buyers valuation function is a monotone submodular function of the items purchased (ie. assigned to the buyer) at the edge weight (i.e. price of the item for that buyer).

As one can see in the Mehta and Karavasilis tables, there are provably better results for display ads with free disposal for the adversarial, ROM and known i.i.d models.

Greedy for a class of adwords problems

- Goel and Mehta [2008] define a class of adwords problems which include the case of small budgets, bipartite matching and b-matching (i.e. when all budgets are equal to some b and all bids are equal to 1).
- For this class of problems, they show that a deterministic greedy algorithm achieves the familiar $1 - 1/e$ bound in the ROM model. Namely, the algorithm assigns each query (.e. node in U) to the advertiser who values it most (truncating bids to keep them within budget and consistently breaking ties). Recall that Ranking can be viewed as greedy (with consistent tie breaking) in the ROM model.

Some concluding remarks on max-sat and bipartite matching

- The ROM model subsumes the stochastic model where inputs are chosen i.i.d. from an unknown distribution (which in turn subsumes i.i.d. inputs from a known distribution). Why? Hence a positive result in the ROM model implies a positive result in the i.i.d. unknown distribution model.
- A research problem of current interest (work by Nicolas Pena) is to see to what extent some form of an extended online framework can yield a deterministic online bipartite matching algorithm with approximation ratio better than $1/2$.
- As mentioned before, Pena can show that a $3/4$ approximation can be obtained by a ideterministic "poly width" online algorithm.
- One can formulate the Buchbinder and Feldman method in the framework of the priority BT model of Alekhnovich et al. Can we show that a bounded width online (or priority) BT algorithm cannot obtain a $3/4$ ratio?

Online and priority width bounds for max-sat and bipartite matching

We have the following width inapproximation results.

- To improve upon the $\frac{3}{4}$ approximation (using online width $2n$) result, we need exponential width. More precisely,
  For any $\epsilon > 0$ there exists $\delta > 0$ such that, for $k < e^{\delta n}$, no online width-cut-k algorithm can achieve an asymptotic approximation ratio of $3/4 + \epsilon$ for unweighted exact max-2-sat with input model 2.
- For any $\epsilon > 0$ there exists $\delta > 0$ such that, for $k < e^{\delta n}$, no PBR width-cut-k algorithm can achieve an asymptotic approximation ratio of $21/22 + \epsilon$ for unweighted max-2-sat with input model 3.
- For any $\epsilon > 0$, no bounded width online algorithm can achieve a $\frac{1}{2} + \epsilon$ approximation for bipartite matching.
- For any $\epsilon > 0$, no priority algorithm can achieve a $\frac{1}{2} + \epsilon$ approximation for bipartite matching.

References for updated table of weighted bipartite matching problems

References

[1] Nitish Korula, Vahab Mirrokni, Morteza Zadimoghaddam. *Online Submodular Welfare Maximization: Greedy Beats 1/2 in Random Order*. STOC '15.

[2] Morteza Zadimoghaddam. *Online Weighted Matching: Beating the 1/2 Barrier.*(2017)
`https://arxiv.org/pdf/1704.05384.pdf`

[3] Michael Kapralov, Ian Post, Jan Vondrak. *Online Submodular Welfare Maximization: Greedy is optimal*. SODA '13.

[4] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, Pan Xu. *New Algorithms, Better Bounds, and a novel mOdel for Online Stochastic Matching*. ESA '16.

[5] Jon Feldman, Nitish Korula, Vahab Mirrokni, S. Muthukrishnan, Martin Pal. *Online Ad Assignment with Free Disposal*. WINE '09.

[6] Thomas Kesselheim, Klaus Radke, Andreas Tonnis, Berthold Vocking. *An Optimal Online ALgorithm for Weighted Bipartite Matching and Extensions to Combinatorial Auctions*. ESA '13.

[7] Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S Mirrokni, Cliff Stein. *Online stochastic packing applied to display ad allocation*. ESA '10.

[8] A. Mehta, A. Saberi, U. V. Vazirani, and V. V. Vazirani. *Adwords and generalized online matching*. Journal of ACM, vol. 54, no. 5, 2007.

[9] N. R. Devanur and T. P. Hayes. *OThe adwords problem: online keyword matching with budgeted bidders under random permutations*. ACM Conference on Electronic Commerce, pp. 71–78, 2009.