CSC2420: Algorithm Design, Analysis and Theory
Fall 2023
An introductory (i.e. foundational) level graduate course.

Allan Borodin

October 24, 2023

Week 7

Announcements

I posted three questions for Assignment 2. One more to follow.

Todays agenda

We will discuss the following topics:

- Johnson's algorithm.
- The "canonical" randomization of a modified Johnson's algorithm.
- Evidence againt an online determintsic priority algorithmn for Max-Sat
- Yannakakis IP/LP and randomized rounding for Max-Sat
- The deterministic and randomized two sided greedy algorithm for unconstrained non monotone submodular maximization. Application to Max-Sat.
- Another randomized online max-sat algorithm and its de-randomizations in extended online algorithms
- Online bipartite matching

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 $\sum_{C_i \in P} w_i' \geq \sum_{C_j \in N} w_j'$
    $x := \mathit{true}; L := L \setminus P$
    For all $C_r \in N$, $w_r' := 2w_r'$ **End For**
  Else
    $x := \mathit{false}; L := L \setminus N$
    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])

Johnson's algorithm is the derandomization of the naive randomized algorithm

- Twenty years after Johnson's algorithm, Yannakakis [1994] presented the naive randomized 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. Otherwise use $F = (x \vee \bar{y}) \wedge (\bar{x} \vee y) \wedge y$.
- Chen, Friesen, Zheng [1999] showed that Johnson's algorithm achieves approximation ratio $\frac{2}{3}$ for arbitrary weighted Max-Sat.
- 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.

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 by using a random ordering of the propositional variables (i.e. the input items). This is another example of the random order model (ROM), a randomized variant of online algorithms.
- To precisely model the Max-Sat problem within an online or priority framework, we need to specify the input model.
- In increasing order of providing more information (and possibly better approximation ratios), we can consider (on the next slide) four input models.

The MaxSat online and priority input models

M0 Each propositional variable x is represented by the names of the positive and negative clauses in which it appears.

M1 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.

M2 In addition, for each C_i and C_j, a list of the other variables in that clause is specified.

M3 The variable x is represented by a complete specification of each clause it which it appears.

The naive randomized algorithm can be implemented in a "model 0" where we don't even specify the lenths of the clauses and Johnson's algorithm can be implemented using input model 1.

Improving on Johnson's algorithm

- 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 Johnson's algorithm in the ROM model is at most $2\sqrt{15}–7 \approx .746 < 3/4$, noting that $\frac{3}{4}$ is the ratio first obtained by Yannakakis' IP/LP approximation that we will soon present.
- Poloczek and Schnitger first consider a "canonical randomization" of Johnson's algorithm; namely, the canonical randomization sets a variable $x_i = \mathit{true}$ 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.

A few comments on the Poloczek and Schnitger algorithm

- The Poloczek and Schnitger algorithm is called Slack and has approximation ratio $= 3/4$.
- The Slack algorithm is a randomized online algorithm (i.e. adversary chooses the ordering) where the variables are 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 input model 0.
- Finally (in this regard), Poloczek [2011] shows that no deterministic priority algorithm can achieve a $3/4$ approximation within input Model 2. This provides a sense in which to claim the that Poloczek and Schnitger Slack algorithm "cannot be derandomized".
- The best deterministic priority algorithm in the third (most powerful) Model 3 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 deramdomized 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 an algorithm that generates $2n$ parallel streams where each stream is an online algorithn.
- The Buchbinder et al USM algorithm is the basis for a randomized 3/4 approximation online Max-Sat (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 $z_j \geq 0$.

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 derandomized algorithm will still be solving LPs.)

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 (assuming an exponental number of value oracle calls), 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 the FOCS [2012] conference, 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, Ψ)

$X_0 \leftarrow \emptyset$, $Y_0 \leftarrow \mathcal{N} \times \{0, 1\}$.
for i = 1 *to* n **do**
  $a_{i,0} \leftarrow g(X_{i-1} \cup \{u_i, 0\}) - g(X_{i-1})$.
  $a_{i,1} \leftarrow g(X_{i-1} \cup \{u_i, 1\}) - g(X_{i-1})$.
  $b_{i,0} \leftarrow g(Y_{i-1} \setminus \{u_i, 0\}) - g(Y_{i-1})$.
  $b_{i,1} \leftarrow g(Y_{i-1} \setminus \{u_i, 1\}) - g(Y_{i-1})$.
  $s_{i,0} \leftarrow \max\{a_{i,0} + b_{i,1}, 0\}$.
  $s_{i,1} \leftarrow \max\{a_{i,1} + b_{i,0}, 0\}$.
  with probability $s_{i,0}/(s_{i,0} + s_{i,1})^*$ **do**:
  $X_i \leftarrow X_{i-1} \cup \{u_i, 0\}$, $Y_i \leftarrow Y_{i-1} \setminus \{u_i, 1\}$.
  else (with the compliment probability $s_{i,1}/(s_{i,0} + s_{i,1})$) **do**:
  $X_i \leftarrow X_{i-1} \cup \{u_i, 1\}$, $Y_i \leftarrow Y_{i-1} \setminus \{u_i, 0\}$.
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.
- 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 that are no longer satisfiable.

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., clauses no longer satisfiable) 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 that are no longer satisfiable.

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., clauses no longer satisfiable) 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 that are no longer satisfiable.

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., clauses no longer satisfiable) 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 $x_i = \text{true}$
  Else if $t_i \leq 0$,
    then set $x_i = \text{false}$
  Else set x_i true with probability $\frac{t_i}{t_i + f_i}$.
End For

The randomized max-sat approximation algorithm continued

For $i = 1 \ldots n$
  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

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 remiaing $n - i$ variables are set as in $\mathbf{x}^*$. (Note: x^* is not calculated.)

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

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

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 another 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.

Randomized online bipartite matching and the adwords problem.

- We return to online algorithms and algorithms in the random order model (ROM). We have already seen evidence of the power of randomization in the context of the USM and MaxSat problems.
- 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. (The FOCS 2012 conference had a session of three papers related to this topic.)
- 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 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 randomized ranking 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).
- Aside: 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.

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.
- In the ROM model, Karande, Mehta, Tripathi [2011] show that Ranking achieves approximation at least .653 (beating $1 - 1/e$) and no better than .727. This ratio was improved to .696 by Mahdian and Yan [2011]].

And some more recent progress

- Karande et al show that any ROM approximation result implies the same result for the unknown 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.
- There is a large landscape (and continuing research) of weighted versions of online bipartite matching such as the *adwords* problem and the *display ads* problem that are motivated by applications to online advertising.
- Although out of data, the survey by Mehta [2013] is a good starting reference. Note: The table in the survey identifies the ROM and unknown i.i.d. model. Recently, Correa et al [Math of OR 2022] show that there is a provable gap between the known and unknown i.i.d. ratios when there is one offline node.