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

Allan Borodin

October 10, 2023

Week 5

Announcements:

- Assignment 1 is due Wednesday 11 AM. I changed the last two parts of question 2. I apologize for not making this clearer.

Todays agenda

We will discuss the following topics:

- Proof of non-oblivious local search algorithm for exact max-2-sat.
- Some experimental results
- k-set packing and $k+1$ claw free graphs. Oblivious and non-oblivious local search.
- The metric facility location and k-median problems
- Submodular functions
- Local search for monotone submodular functions subject to a constraint
- Randomization
- Exact Max Sat revisited; the naive algorithm and its de-randomization; Johnson's algorithm.

The non-oblivious local search

- We consider the idea that satisfied clauses in S_2 are more valuable than satisfied clauses in S_1 (because they are able to withstand any single variable change).

- The idea then is to weight S_2 clauses more heavily.

- Specifically, in each iteration we attempt to find a $\tau' \in N_1(\tau)$ that improves the potential function

$$\frac{3}{2}W(S_1) + 2W(S_2)$$

instead of the oblivious $W(S_1) + W(S_2)$.

- More generally, for all k, there is a setting of scaling coefficients $c_1, \ldots, c_k$, such that the non-oblivious local search using the potential function $c_1 W(S_1) + c_2 W(S_2 + \ldots + c_k W(S_k)$ results in approximation ratio $\frac{2^k-1}{2^k}$ for exact Max-k-Sat.

Sketch of $\frac{3}{4}$ totality bound for the non oblivious local search for Exact Max-2-Sat

- Renaming variables, we can assume that τ is the all true assignment.
- Let $P_{i,j}$ be the weight of all clauses in S_i containing x_j.
- Let $N_{i,j}$ be the weight of all clauses in S_i containing $\bar{x}_j$.
- Here is the key observation for a local optimum τ wrt the stated potential:

  $-\frac{1}{2}P_{2,j} - \frac{3}{2}P_{1,j} + \frac{1}{2}N_{1,j} + \frac{3}{2}N_{0,j} \leq 0$

- Summing over variables $P_1 = N_1 = W(S_1)$, $P_2 = 2W(S_2)$ and $N_0 = 2W(S_0)$ and using the above inequality we obtain $3W(S_0) \leq W(S_1) + W(S_2)$

Some experimental results concerning Max-Sat

- Of course, one wonders whether or not a worst case approximation will actually have a benefit in "practice".
- "In practice", local search becomes more of a "heuristic" where one uses various approaches to escape (in a principled way) local optima and then continuing the local search procedure. Perhaps the two most commonly used versions are Tabu Search and Simulated Annealing.
- Later, we will also discuss methods based on online algorithms and "random walks" and other randomized methods (and their derandomizations). .
- We view these algorithmic ideas as starting points.
- But for what it is worth, here are some 2010 experimental results both for artifically constructed instances and well as for one of the many benchmark test sets for Max-Sat.
- Experimental results by Poloczek and Willamson show that various ways to use greedy and local search algorithms can compete (wrt. various test sets) with "state of the art" simulated annealing algorithms and walk-sat algorithms while using much less time.

Experiment for unweighted Max-3-Sat

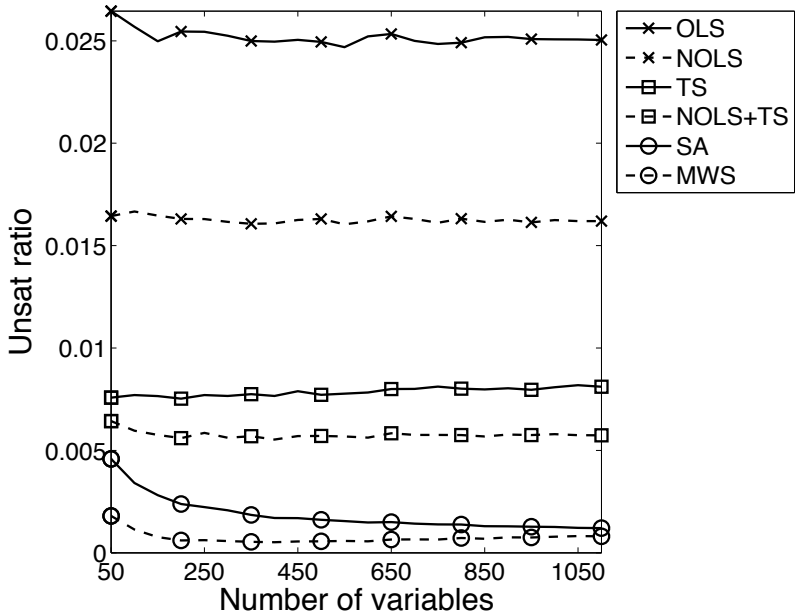

Fig. 1. Average performance when executing on random instances of exact MAX-3-SAT.

[From Pankratov and Borodin, 2010]

Experiments for benchmark Max-Sat Instances

Table 2. The Performance of Local Search Methods

	NOLS+TS		2Pass+NOLS		SA		WalkSat	
	% sat	∅ time	% sat	∅ time	% sat	∅ time	% sat	∅ time
SC-APP	90.53	93.59s	99.54	45.14s	99.77	104.88s	96.50	2.16s
MS-APP	83.60	120.14s	98.24	82.68s	99.39	120.36s	89.90	0.48s
SC-CRAFTED	92.56	61.07s	99.07	22.65s	99.72	70.07s	98.37	0.66s
MS-CRAFTED	84.18	0.65s	83.47	0.01s	85.12	0.47s	82.56	0.06s
SC-RANDOM	97.68	41.51s	99.25	40.68s	99.81	52.14s	98.77	0.94s
MS-RANDOM	88.24	0.49s	88.18	0.00s	88.96	0.02s	87.35	0.06s

Figure: Table from Poloczek and Williamson ACM Experimental Results 2017

Note: *2Pass* is a deterministic "2-pass online algorithm" that is derived from a randomized online algorithm that we will discuss "soon".

Oblivious and non-oblivious local search for $k+1$ claw free graphs

- Consider the k set packing problem and its generalization to $(k+1)$ claw free graphs. (The intersection graph of a k-set instance is a $k+1$ claw free graph where each vertex represents a k-set and there is an edge whenever two sets intersect.)
- We consider the weighted max (independent) vertex set problem (WMIS) in a $k+1$ claw free graph. (The greedy and oblivious local search approximations for the weighted k set packing problem generalize to $k+1$ claw free graphs.)
- The standard greedy algorithm and the 1-swap oblivious local search both achieve a k approximation for the WMIS in $k+1$ claw free graphs. Here we define an "ℓ-swap" oblivous local search by using neighbrourhoods defined by bringing in a set S of up to ℓ vertices and removing all vertices adjacent to S.
  NOTE: I keep trying to use fractional approximation ratios that are absolute constants and ratios greater than 1 for maximization problems when the ratios are parameterized.

Berman's [2000] non-oblivious local search

- For the unweighted MIS, Halldórsson shows that a a 2-swap oblivious local search will yield a $\frac{1}{2}k + 1$ approximation.
- For the weighted MIS, the "ℓ-swap" oblivous local search results in a $k + \epsilon$ approximation ratio for any constant ℓ where ϵ depends on ℓ.
- Chandra and Halldórsson [1999] show that by first using a standard greedy algorithm to initialize a solution and then using a "greedy" oblivious local search, the approximation ratio improves to $\frac{2}{3}(k + 1)$
- Can we use non-oblivious local search to improve the locality gap? Once again given two solutions V_1 and V_2 having the same weight, when is one better than the other?

Berman's [2000] non-oblivious local search

- For the unweighted MIS, Halldórsson shows that a a 2-swap oblivious local search will yield a $\frac{1}{2}k + 1$ approximation.
- For the weighted MIS, the "ℓ-swap" oblivous local search results in a $k + \epsilon$ approximation ratio for any constant ℓ where ϵ depends on ℓ.
- Chandra and Halldórsson [1999] show that by first using a standard greedy algorithm to initialize a solution and then using a "greedy" oblivious local search, the approximation ratio improves to $\frac{2}{3}(k + 1)$
- Can we use non-oblivious local search to improve the locality gap? Once again given two solutions V_1 and V_2 having the same weight, when is one better than the other?
- Intuitively, if one vertex set V_1 is small but vertices in V_1 have large weights that is better than a solution with many small weight vertices.

Berman's [2000] non-oblivious local search

- For the unweighted MIS, Halldórsson shows that a a 2-swap oblivious local search will yield a $\frac{1}{2}k + 1$ approximation.
- For the weighted MIS, the "ℓ-swap" oblivous local search results in a $k + \epsilon$ approximation ratio for any constant ℓ where ϵ depends on ℓ.
- Chandra and Halldórsson [1999] show that by first using a standard greedy algorithm to initialize a solution and then using a "greedy" oblivious local search, the approximation ratio improves to $\frac{2}{3}(k + 1)$
- Can we use non-oblivious local search to improve the locality gap? Once again given two solutions V_1 and V_2 having the same weight, when is one better than the other?
- Intuitively, if one vertex set V_1 is small but vertices in V_1 have large weights that is better than a solution with many small weight vertices.
- Berman chooses the potential function $g(S) = \sum_{v \in S} w(v)^2$. Ignoring some small ϵ's, his k-swap non-oblivious local search achieves an approximation ratio of $\frac{1}{2}(k + 1)$ for WMIS on $k + 1$ claw-free graphs.

A little more detail on the Chandra and Halldóssron greedy + local search algorithm

For the oblivious local search results when we didn't care about the choice of the initial solution, it didn't matter which local improvement we make.

Chandra and Halldóssron show that for their $\frac{2}{3}$ ratio result the choice local improvement does matter. They use the best local improvement (i.e., swapping in the best vertex and extending to a maximal independent set).

They also consider an algorithm ANY_α where the improvement must improve by at least an α factor and show that for a suitable choice of α, the algorithm achieves a ratio $\frac{4(k+1)}{5}$.

They show that this $\frac{4}{5}$ ratio is asymptiocally the best ratio for any α.

NOTE: I am not aware of other theoretical results showing how an initial greedy (or random) solution can impove the local search guarantee.

The (metric) facility location and k-median problems

- Two extensively studied problems in operations research and CS algorithm design are the related uncapacitated facility location problem (UFL) and the k-median problem. In what follows we restrict attention to the (usual) metric case of these problems defined as follows:

Definition of UFL

Input: (F, C, d, f) where F is a set of faciltites, C is a set of clients or cities, d is a metric distance function over $F \cup C$, and f is an opening cost function for facilities.
Output: A subset of facilities F' minimizing $\sum_{i \in F'} f_i + \sum_{j \in C} d(j, F')$ where f_i is the opening cost of facility i and $d(j, F') = min_{i \in F'} d(j, i)$.

- In the capacitated version, facilities have capacities and cities can have demands (rather than unit demand). The constraint is that a facility can not have more assigned demand than its capacity so it is not possible to always assign a city to its closest facility.

UFL and k-median problems continued

Deifnition of k-median problem

Input: (F, C, d, k) where F, C, d are as in UFL and k is the number of facilities that can be opened.
Output: A subset of facilities F' with $|F'| = k$ minimizing $\sum_{j \in C} d(j, F')$

- These problems are clearly well motivated. In particular, the k-median problem (and the k-means problem) are well studied as clustering problems. Moreover, they have been the impetus for the development of many new algorithmic ideas.
- There are many variants of these problems and in many papers the problems are defined so that $F = C$; that is, any city can be a facility. If a solution can be found when F and C are disjoint then there is a solution for the case of $F = C$.
- I want to just mention these problems because of their importance and the techincal depth that appears in many of the results.

UFL and k-median problems continued

- It is known (Guha and Khuller) that UFL is hard to approximate to within a factor better than 1.463 assuming NP is not a subset of $DTIME(n^{\log \log n})$ and the k-median problem is hard to approximate to within a factor better than $1 + 2/e \approx 1.736$ (Jain, Mahdian, Saberi).
- The UFL problem is better understood than k-median. After a long sequence of improved approximation results the current best polynomial time approximation is 1.488 (Li, 2011).
- For k-median, for about 12 years, the best approximation was by an oblivious local search algorithm. Using a p-flip (of facilities) neighbourhood, Arya et al (2001) obtain a $3 + 2/p$ approximation which yields a $3 + \epsilon$ approximation running in time $O(n^{2/\epsilon})$.
- Li and Svensson (STOC 2013, SICOMP, 2016) have obtained a $(1 + \sqrt{3} + \epsilon) \approx 2.73$ approximation running in time $O(n^{1/\epsilon^2})$. Surprisingly, they show that an α approximate "pseudo solution" using $k + c$ facilities can be converted to an $\alpha + \epsilon$ approximate solution running in $n^{O(c/\epsilon)}$ times the complexity of the pseudo solution.

A little history of the k-median problem

For an arbitrary metric space, the first constant $(\frac{20}{3})$ approximation was achieved by Charikar et al [1999].
A PTAS $1 + 1/c$ approximation (in time $O(n^{O(c+1)})$ was given by Arora et al [STOC 1998] for Euclidean 2-space.
For arbitrary Euclidean spaces, Bhattacharya et al [Approx 2021] provide a hardness of approximation (based on the *unique games conjecture* showing that there is no PTAS for the problem; that is, under this conjecture, there exists a constant $\epsilon > 0$ such that no polynomial time algorithm can achieve a $(1 + \epsilon)$ approximation.

Building on the method of Li and Svensson, Byrka et al [2015] achieve a 2.675 approximation. These methods use a "natural" LP and *dependent rounding* to achieve the pseudo solution and then use a black-box method to convert to a solution. The Byrka et al bound remains the current best approximation.

A non-oblivious local search algorithm for the k-median problem

Recently, Cohen-Addad et al [Arxiv 2021] have applied non-oblivious local search to the k-median problem to achieve a $(2.836 + \epsilon)$ approximation. This does not match or beat the best known approximation by Bryka et al but it provides an approach that could lead to an improved approxiamtion. And, it now is the best approxiation via a local search algorithm.

The running time will be (as in other algorithms where ϵ appears in the approximation) $O(n^{\frac{1}{\epsilon}})$.

The approach will be to apply non-oblivious local search (swapping in some number of new medians) to derive a pseudo solution and then convert to a true solution with k median.

So what can we use for a potential function?

A non-oblivious local search algorithm for the k-median problem

Recently, Cohen-Addad et al [Arxiv 2021] have applied non-oblivious local search to the k-median problem to achieve a $(2.836 + \epsilon)$ approximation. This does not match or beat the best known approximation by Bryka et al but it provides an approach that could lead to an improved approxiamtion. And, it now is the best approxiation via a local search algorithm.

The running time will be (as in other algorithms where ϵ appears in the approximation) $O(n^{\frac{1}{\epsilon}})$.

The approach will be to apply non-oblivious local search (swapping in some number of new medians) to derive a pseudo solution and then convert to a true solution with k median.

So what can we use for a potential function?

The basic idea is quite natural but the devil is in the "details". Namely, instead of just considering the closest facility to each point, we also consider the second closest facility.

The Cohen-Addad potential function

Letting Let F be a set of facitilites. Let $d_1((c, F)$ be the distance of a client c to the closest facility in F and let Let $d_2((c, F)$ be the distance of a client c to the second closest facility in F. Then the potential function is:

$$\Phi(F) = \sum_{c \in C} [d_1(c, F) + \beta \min\{d_2(c, F), \alpha d_1(c, F)\}]$$

They choose $\alpha = 3$ and $\beta = 1/5$. That is, a client whose second closest facility is much further away will pay more in with respect to this potiential and the net effect is to allow for a broader space of possible solutions.

Theorem

Let α, β be as above, and let $p(\epsilon)$ be the swap size, and let $r(\epsilon)$ be the number of additional facilities in the pseudo solution, then for
$|F| = k + r(\epsilon)$
$ALG(F) \leq (2.836 + \epsilon) \cdot OPT(F^*)$ *where F^* is a solution with k facilities.*

Another possible non-oblivious approach for k-median

In class another approach was suggested for a possible non-oblivious local search aalgorithm. Namely, we should give some weight to the *dispersion* of the set of facilities.

Indeed this approach is realized in the metric k-center problem where the objective is to select a set S of k facilities so to minimize the maximum distance of every client a facility $F \in S$. That is, thinking of each faciltiy as a center, the goal is to minimize the maximize radius of the neighbourhoods centred at the facilities in S.

The algorithm simply iteratively adds a new facility to maximize the distance to each of the current set of facilities.

Some additional comments on local search

- An interesting (but seemingly difficult) open problem is to consider the second and third closest. Can non-oblivious local search be beneficial for either the UFL or k-means problems?
- Suffice it to say for now that local search is the basis for many practical algorithms, especially when the idea is extended by allowing some well motivated ways to escape local optima (e.g. simulated annealing, tabu search) and when combined with other paradigms.
- Although local search with all its variants is viewed as a great "practical" approach for many problems, local search is not often theoretically analyzed. It is not surprising then that there hasn't been much interest in formalizing the method and establishing limits.
- LP is itself often solved by some variant of the simplex method, which can also be thought of as a local search algorithm, moving fron one vertex of the LP polytope to an adjacent vertex.
  - No simples method is known to run in polynomial time in the worst case.

Submodular functions

Let U be a universe. In what follows, we will only be interested in set functions that satisfy $f(S) \geq 0$ for all $S \subseteq U$. We will also assume functions are *normalized* in that $F(\varnothing) = 0$, These assumptions are not that essental but they are standard and without these assumptions statements and proofs become somewwhat more complex.

- A *sublinear set function* satisfies the property that $f(S \cup T) \leq f(S) + f(T)$ for all subsetes S, T of U.
- When $f(S \cup T) + f(S \cap T) = f(S) + f(T)$, the function is a linear (also called modular) function.
- A *submodular set function* $f : U \to \mathbb{R}$ satisfies the following property: $f(S \cup T) + f(S \cap T) \leq f(S) + f(T)$
- It follows that modular set functions are submodular and submodular functions are sublinear.
- Submodular functions can be monotone or non-monotone. A monotone submodular function also satisifes the property that $f(S) \leq f(T)$ whenever $S \subseteq T$.

An alternative characterization and examples of submodular functions

Submodular functions satisfy and can also be defined as those satisfying a *decreasing marginal gains* property. Namely,
For $S \subset T, f(T \cup \{x\}) - f(T) \leq f(S \cup \{x\}) - f(S)$. That is, adding additional elements has decreasing (more precisely, non increasing) marginal gain for larger sets.

Most applications of submodular functions are for monotone submodular functions. For example, in practice, when we are obtaining results from a search engine, as we obtain more and more results, we tend to obtain less additional value.

Modular functions are monotone.

The rank function of a matroid is a monotone submodular function.

The two most common examples of non-monotone submodular functions are max-cut and max-di-cut (i.e., max directed cut)

Monotone submodular function maximization

- The monotone problem is only interesting when the submodular maximization is subject to some constraint.
- Probably the simplest and most widely used constraint is a cardinality constraint; namely, to maximize $f(S)$ subject to $|S| \leq k$ for some k and since f is monotone this is the same as the constraint $f(S) = k$.
- Following Cornuéjols, Fisher and Nemhauser [1977] (who study a specific submodular function), Nemhauser, Wolsey and Fisher [1978] show that the standard greedy algorithm achieves a $1 - \frac{1}{e}$ approximation for the cardinality constrained monotone problem. More precisely, for all k, the standard greedy is a $1 - (1 - \frac{1}{k})^k$ approximation for a cardinality k constraint.

Standard greedy for submodular functions wrt cardinality constraint

$S := \varnothing$
While $|S| < k$
  Let u maximize $f(S \cup \{u\}) - f(S)$
  $S := S \cup \{u\}$
End While

Proof: greedy approx for monotone submodular maximization subject to cardinality constraint

We want to prove the $1-(1-\frac{1}{k})^k$ approximation bound.
Let S_i be the set after i iterations of the standard greedy algorithm and let $S^* = \{x_1, \ldots, x_k\}$ be an optimal seti so that $OPT = f(S^*)$. For any set S and element x, let $f_S(x) = f(S \cup \{x\}) - f(S)$ be the marginal gain by adding x to S. The proof uses the following sequence of inequalities:

$$
\begin{aligned}
f(S^*) &\leq f(S_i \cup S^*) \text{ by monotonicity} \\
&\leq f(S_i) + (f(S_i \cup \{x_1\}) - f(S_i)) + (f(S_i) \cup \{x_1, x_2\} - f(S_i \cup \{x_1\})) + \ldots \\
&\qquad \text{(by submodularity)} \\
&\leq f(S_i) + f_{S_i}(x_1) + f_{S_i}(x_2) + \ldots f_{S_i}(x_k) \\
&\qquad \text{(again by submodularity)} \\
&\leq f(S_i) + k \cdot (f(S_{i+1} - f(S_i)) \text{ by the greedy assumption}
\end{aligned}
$$

Equivalently, $f(S_{i+1}) \geq f(S_i) + \frac{1}{k}(f(OPT) - f(S_i)$

The proof is completed by showing $f(S_i) \geq (1-(1-\frac{1}{k})^i) \cdot OPT$ by induction on i.

Generalizing to a matroid constraint

- Nemhauser and Wolsey [1978] showed that the $1 - \frac{1}{e}$ approximation is optimal in the sense that an exponential number of value oracle queries would be needed to beat the bound for the cardinalily constraint.
- Furthermore, Feige [1998] shows it is NP hard to beat this bound even for the explicitly represented maximum k-coverage problem.
- Following their first paper, Fisher, Nemhauser and Wolsey [1978] extended the cardinality constraint to a matroid constaint.
- Fisher, Nemhauser and Wolsey show that both the standard greedy algorithm and a 1-exchange local search algorithm (that will follow) achieve a $\frac{1}{2}$ approximation for maximzing a monotone submodular function subject to an arbitrary matroid constraint.
- They also showed that this bound was tight for the greedy and 1-exchange local search algorithms.