BELLAIRS WORKSHOP ON APPROXIMATION ALGORITHMS
OPEN PROBLEM SESSION #2

(1) **Kunal Talwar**. 1) Given a database containing 'private' information (i.e. a census bureau) we are interested in revealing accurate statistics about a population while preserving the privacy of individuals. Differential privacy is such a rigorous notion of privacy which yields strong privacy guaranties even if an adversary has arbitrary auxiliary information. Privacy is in the algorithm, not in the output. The mechanism

$$M : D^N \to R$$

has ϵ-*differential privacy* (ϵ-DP) if for each $A, B \in D^N$ differing by at most one element and any $S \subseteq R$,

$$Pr[M(A) \in S] \leq (1 + \epsilon)Pr[M(B) \in S].$$

It turns out to be more convenient to deal with

$$Pr[M(A) \in S] \leq \exp(\epsilon)Pr[M(B) \in S].$$

[VM. Can you give a specific example?] [KT: For instance, I want to count the number of people in California. Answer: count + $Lap(1/\epsilon)$.]

Consider constructing an optimal rooted Steiner tree. A Steiner tree can be thought of as defining a path to the root for each terminal. The cost of the Steiner tree is the cost of the union of all the paths. So non-privately, we known how to solve it. But now suppose for each vertex, whether or not it is a terminal is private information. We will allow ourselves to define a vertex-to-root path for every in the graph. The cost of a solution (i.e. a path to root for every vertex) is the cost of the union of the paths corresponding to actual terminals. Think of it as: we tell everyone which path to pick if they need to be connected; the terminals are the ones who actually take their path, and we pay only for those edges that are actually used. So the goal is: given the set of terminals T, output a path-to-root for every vertex in the graph such that the distribution over outputs satisfies ϵ-DP, and the cost of the resulting Steiner tree is small. Also note that a universal Steiner tree algorithm is a 0-differentially private algorithm, since we did not even look at T. So we can get $O(\log^2 n)$ and in fact get $O(\log n)$ if the adversary choosing T is oblivious. Thus for the differentially private Steiner tree problem, we can get an $O(\log n)$ approximation for $\epsilon = 0$. The question is: *can we get an $O(1)$ approximation for a constant ϵ?*

Various other combinatorial optimization problems have been studied in the Differential Privacy model (SODA'10 paper available on arXiv: Differentially Private Combinatorial Optimiztion. Anupam Gupta, Katrian Ligett, Frank McSherry, Aaron Roth, Kunal Talwar). One example is set cover which currently has an $O(\log n)$-approximation algorithm.

2) Consider the following problem: we have a database $x \in \mathbb{Z}^n$ where x_i counts the number of people of "type" i. We want to output Fx, where F is a $d \times n$ 0-1 matrix. Thus $Fx \in \mathbb{Z}^d$ is a vector of answers to a bunch of count-type queries. The goal is to come up with a ϵ-DP mechanism M such that $\max_x E[||M(x) - Fx||_2]$ is as small as possible, where the expectation is taken over M's internal randomness. Given any F, there is an optimal M_F which has the least expected error. Is it hard to compute M_F? Are there good approximation algorithms? Hardt and Talwar (STOC '10) give a mechanism which is within $O(\log^{1.5} d)$ of the optimal, assuming a conjecture from convex geometry called the Slicing conjecture. Question a): *Is there a good unconditional approximation algorithm?* Question b): Usually, we would get the rows of F one at a time, and would have to provide an approximate $(Fx)_i$ before seeing the next row of F . This is essentially an online algorithms problem. *Is there a good online algorithm? Can the* $O(\log^{1.5} d)$ *algorithm be made online?*

(2) **Howard Karloff.** This is a problem coming from database people at AT&T. The general goal is to design extremely efficient algorithms as runtimes of $O(n^2)$ are considered as too slow by these people. (To them $O(n^2)$ is like exponential time.) One such problem is as follows. We are given an interval I, say $I = [0, \ell] := \{0, \ldots, \ell\}$ and n subintervals of I. The goal is to cover at least a given fraction of I (say, 80%), using as few as possible of the n subintervals. What can you do in $n \operatorname{polylog}(n)$? Dynamic programming gives $O(n^2)$, but this is too slow. The truncated greedy algorithm is $O(n)$ and approximates the problem with a ratio of 7 (or is it 9?). *Is a ratio of* $1 + \epsilon$ *doable in* $n \operatorname{polylog}(n)$ *time?*

[HK: A nice research direction: find super-fast approximation algorithms for problems that can be solved in polytime.]

(3) **Moses Charikar**. 1) Suppose we want to sort n elements by comparisons. This can be done in $O(n \log n)$ comparisons. Now, disallow some comparisons. So we are given an explicit list of comparisons that are allowed, and all other comparisons are forbidden. The oracle guarantees that if we perform all the allowed comparisons, we will obtain a total order. *Can you still sort after performing* $O(n \log n)$ *comparisons? Or can you force* $\omega(n \log n)$ *comparisons?.* If the set of allowed comparisons is suffiently dense, we can use the regularity lemma to sort in $O_c(n \log n)$ number of comparisons. This implies that, for all $c > 0$, it is possible to sort in at most cn^2 comparisons (this is an unpublished result).

[MS: where does this problem come from?] [MC: this comes from a class of problems studied in: Moses Charikar, Ronald Fagin, Venkatesan Guruswami, Jon M. Kleinberg, Prabhakar Raghavan (JCSS, 2002)]

2) This problem concerns the Polymath 5 project that is ongoing on Tim Gower's weblog. It is a problem where mathematical programming techniques could potentially be useful. Pick a collection of subsets $S_1, \ldots, S_m$ of $[n] :=$

$\{1, 2, \ldots, n\}$. Now write down a sequence x of $+1$'s and -1's of size n, say

$$x = (+1, -1, -1, +1, +1, -1, +1).$$

Pick a set S_j in the collection and compute $|\sum_{i \in S_j} x_i|$. That is the discrepancy of the set. The question is: can we find $x \in \{-1, +1\}^n$ such that every set has a small discrepancy?

A particular case of this is the Erdös discrepancy problem, where the sets S_j are *homogeneous* arithmetic progressions, that is, sets of the form $\{d, 2d, \ldots, kd\}$ where k and d are positive integers such that $kd \leqslant n$. In case *all* arithmetic progressions, that is, sets of the form $\{a, a + d, \ldots, a + kd\}$, are allowed, the discrepancy is $\Omega(n^{1/4})$. [MS: are the results in this area constructive?] [MC: Joel Spencer proved that in case $m = n$, the discrepancy is at $O(\sqrt{n})$. The case $m = n$ is in some sense the most interesting. Randomized rounding gives $O(\sqrt{n \log n})$. Recently, Nikhil Bansal found a way to make Spencer's $O(\sqrt{n})$ result constructive (see his very recent paper on arXiv).]

Polymath 5 studies the following question: *what is the discrepancy of homogenous arithmetic progressions?*. There is an upper bound of $\log_3 n$. For the lower bound, nothing better than constant is known.

Actually, there is a natural semidefinite relaxation for this. Find vectors v_1, v_2, ..., v_n in $S^{n+1} = \{v \in \mathbb{R}^n : ||v|| = 1\}$ such that $||v_d + v_{2d} + \cdots + v_{dk}||^2$ is small for all valid choices of k and d. Actually, the $\log_3 n$ upper bound also applies to the SDP relaxation.

Can you find a dual solution whose value is slowly increasing?

What is the dual of this SDP? We seek coefficients $c_{d,k}$ and b_i such that:

$$\sum_{d,k} c_{d,k}(x_d + x_{2d} + \cdots + x_{kd})^2 - \sum_i b_i x_i^2 \geq 0.$$

for all $x \in \mathbb{R}$. The corresponding lower bound is

$$\frac{\sum_i b_i}{\sum_{d,k} c_{d,k}}.$$

We can even find an LP relaxation. The dual of this LP is as follows. For two homogeneous arithmetic progressions P, Q (we identify arithmetic progressions with their characteristic vector) define the tensor product of P, Q as $P \otimes Q := P \cdot Q^T$. We have to construct a convex combination of the tensor products $P \otimes Q$ and $-P \otimes Q$, where P and Q are homogeneous arithmetic progressions, in such a way that

$$\sum_i \lambda_i(\pm P_i \otimes Q_i) = \mathrm{diag}(\lambda_1, \ldots, \lambda_n).$$

The corresponding lower bound is $\sum_j \lambda_j$, that is, the trace of the diagonal matrix in the right hand side.