15-859(D) Randomized Algorithms
Notes for 10/1/98

```
* approximating MAX SAT
* occupancy problems
```

MAX-SAT

Here is a classic problem, whose solution combines randomized rounding idea and the conditional expecation idea.

A CNF formula is an AND of clauses. A k-CNF is a CNF where each clause has each of size at most k. Let's define an "exactly-k CNF" to be a CNF where each clause is of size exactly k (a given variable is not allowed to appear several times in the same clause to inflate its size). Say we're given a formula and we want to satisfy as many clauses as we can.

Claim: For any "exactly k-CNF", there exists a solution that satisfies at least $1 - 1/2^k$ of the clauses.

Proof: consider a random assignment. Expected number of clauses satisfied is $(1 - 1/2^k)m$. (m =number of clauses). Therefore, an assignment with this property exists. ∎

What about MAX-SAT in general? If there are m_k clauses of size k, a random assignment will satisfy an expected $\sum_k m_k(1 - 1/2^k)$ of them.

How about a deterministic algorithm? Idea: given any partial assignment P, we can calculate the expected number of clauses satisfied given that those variables not specified in P are set randomly. (It is just the number of clauses already satisfied plus $\sum_k m_k(1-1/2^k)$, where m_k is the number of clauses of size k in the "formula to go".) So, we just use the conditional expectation method, going through the variables one at a time, setting the next x_i to 0 or 1 depending on which produces the greater expected value. Since the expected number of clauses satisfied by a random assignment is the *average* of the expected number given $x_i = 0$ and the expected number given $x_i = 1$, our "expectation to go" never decreases.

For the MAX-SAT problem in general, in the worst case this bound is 1/2. Can't hope for better, e.g., if the formula is just "x_1 and $\neg x_1$." How about an approximation algorithm that satisfies nearly as many clauses as the best possible solution? (note: MAX 2-SAT is NP-hard). It turns out there is a way to satisfy at least 3/4 of the maximum possible. Note: above will work so long as there are no singletons. We'll look next at a randomized rounding procedure that does well so long as all clauses are small. Then we'll combine them.

Here is the algorithm:

- Solve a fractional version of the problem. Instead of requiring variable $x_i \in \{0,1\}$, we allow $x_i \in [0,1]$. Define $\neg x_i$ to be $1 - x_i$. Allow clauses to be "partially satsified:" for clause $(x_1 \vee x_2 \vee \neg x_3)$, let "satisfiedness" be: $\min(1, x_1 + x_2 + (1 - x_3))$. I.e., if the sum is less than 1, then it represents how satisfied the clause is, and if greater than 1, then we say the clause is satisfied. Then we find the solution that maximizes total satisfaction.

  To set up as an LP: $x_i \in [0,1]$. Variable z_j for clause j: $z_j \leq 1$ and $z_j \leq$ sum-of-literals-in-clause-j. Then, maximize $\sum z_j$.

- Now, let's do randomized rounding: set variable i to 1 with probability x_i.

  Claim: if clause j has k literals, then $\mathbf{Pr}(\text{clause } j \text{ is satisfied}) \geq z_j(1 - (1 - 1/k)^k)$.

  [Note: for $k = 1$, this is z_j, for k=2, this is $3z/4$, for $k = 3$, this is $0.704z$]

  Proof: Say $z_j = 1$ and all variables in it are at $1/k$. Then, the probability the clause is not satisfied is exactly $(1 - 1/k)^k$. Say z_j may not be 1, but all variables are equal at z_j/k. Then the probability the clause is satisfied is $1 - (1 - z_j/k)^k$, which is $\geq z_j(1 - (1 - 1/k)^k)$. (Get equality at $z_j = 0$ and $z_j = 1$.) Then we just need to show that the "all equal" case is the worst case. This is the same as saying "given k quantities that sum to a given value $(k - z)$, the product is maximized when they are all equal."

- This strategy does well when the clauses are small, and the previous did well when the clauses are big. E.g., prob(clause j is satisfied) as a function of k is:

	strategy 1	strategy 2
k=1	1/2	z_j
k=2	3/4	3/4 z_j
k=3	7/8	0.704 z_j

- So, let's just flip a coin and with prob 1/2 use strategy 1, and with probability 1/2 use strategy 2. The probability clause j is satisfied is then the average of the two values from the above table. Just want this to be $\geq (3/4)z_j$, which, in fact, it is. (just have to do the calculation in general).

Notes:

Current best approxs: 0.931 for MAX 2-SAT [Feige-Goemans], 0.801 for MAX 3-SAT [Trevisan, Sorkin, Sudan, Williamson], (if formula is satisfiable then can get 7/8 [Karloff, Zwick]), 0.770 for MAX-SAT [Asano] building on [Goemans-Williamson].

Current best hardness results: 7/8 for MAX-3SAT [Hastad], 73/74=0.986 for MAX 2-SAT.

Occupancy problems

Next time: hashing (universal and perfect hashing). Now: set the stage by talking about the question: say we put m balls randomly into n boxes; what can we say about the distribution that comes up?

Let's look at some things we can say, and also how much independence we are using to say them.

Let B_i be the size of the ith bin. X_{ij} = indicator for ball j falling into ith bin.

- What is $\mathbf{E}[B_i]$? Answer: m/n.
- What is $Var[B_i]$? $Var[B_i] = \sum_j Var[X_{ij}] = m(1/n - 1/n^2) = m/n(1 - 1/n)$. Note: we only needed pairwise independence to calculate this.

What does a single bin look like, and the distribution of bins? (Let's say $m = n$ for concreteness).

- $\mathbf{Pr}$[get zero balls] $= (1 - 1/n)^n \approx 1/e$. So we expect a $1/e$ fraction of bins to be empty.
- $\mathbf{Pr}$[exactly 1 ball] $= n(1/n)(1 - 1/n)^{n-1} \approx 1/e$.
- $\mathbf{Pr}$[exactly 2 balls] $= \binom{n}{2}(1/n^2)(1 - 1/n)^{n-2} \approx 1/2e$.
- $\mathbf{Pr}$[get exactly i balls] $\binom{n}{i}(1/n)^i(1 - 1/n)^{n-i} \leq 1/(e \cdot i!)$.

What can we say about largest bin? $1/(e \cdot i!) = 1/n^c$ at $i = O(log(n)/loglog(n))$. So, with high probability, largest bin has size at most $O(log(n)/loglog(n))$. In fact, we expect some bin to be that full. (pick i so have $1/n$ chance that bin has $\geq i$ balls. Prob that bin i is at least this full given that previous bins are not this full is only larger. (formally, look at martingales)

Note: this argument uses full independence. What if we only have pairwise independence. What can we say about size of largest bin?

Chebyshev: σ is about 1, so $\mathbf{Pr}$[bin size $< t$] $< 1/t^2$. Expected maximum is at most $O(\sqrt{n})$.