CSC 373 Lecture 30

Announcements:

As posted, weekly TA office hour Fridays 1-2 in Pratt 378.

Course evaluations today. Now orat end of class?

Next assignment due Friday, Dec 2; term test 3 on Monday, Dec 5. last class Wed, Dec 7.

Today

Randomized rounding

- O(log m) approximation for Set cover

Randomized algorithm for 2SAT

Set cover IP/LP randomized rounding

There is a very natural and efficient greedy algorithm for solving the weighted set cover problem with approximation h_d where $d = max\_i\ |S\_i|$. But we want to use this problem to give a final example of IP and randomized rounding.

Note that in the randomized Max Sat algorithms, we never had to worry about whether or not a solution was feasible since every truth assignment is feasible. The only issue was the approximation ratio.

The following randomized algorithm will with high probability produce a cover that is within a factor $O(H\_d) = O(log\ m)$ of the optimum where m is the size of the universe. This is also an opportunity to (re)introduce a little more probability.

The IP/LP randomized rounding

- The IP is to $\min \sum_i w_i x_i$

  subj to $\sum_{\{i:\, u_j \in S_i\}} x_i >= 1$

  $x_i \in \{0,1\}$ for IP; $x_i >= 0$ for LP

- We solve this LP

  and find an optimal solution $\{x^*_1, \ldots, x^*_n\}$.

  We know that $x^*_i <= 1$ since in an optimal solution, each x^*_i is at most 1.

We treat the x^*_i values as probabilities and choose S_i (to be in our set cover) with probability x^*_i. This is a covering problem and the chosen sets will most likely not be a cover. So we will have to repeat this process enough times to have a good probability that all elements are covered.

The analysis

- It is easy to calculate the expected cost of the "partial cover" *C'* of sets selected by the LP optimum. Namely,

$\mathbf{E}$[cost(C')] = sum w_i Prob[S_i is chosen]
= sum w_i x*_i = OPT-LP

- Now we need to calculate the probability that a given *u_j = u* is not covered. Lets say that *u* occurs in sets *S_1, ..., S_k*. The LP solution must satisfy the constraint : *sum_{i: u in S_i} x*_i >=1*.

Analysis continued

- Under this constraint, we can minimize the probability that u is covered by $x^*_i = 1/k$ for $1 \le i \le k$. It follows that the probability that u is not covered is at most $(1-1/k)^k \le 1/e$.
- Suppose now that we run the same randomized rounding algorithm c ln m times ($m = |U|$) for some constant c, each time adding sets (given by the rounded LP) to the set cover. While we may be adding the same set many times (and hence overcounting), the cost of the "cover" is now at most $(c \ln m)$ *OPT-LP*
- The probability that u is not covered is now $\le (1/e)^{c \ln m} = (1/m)^c$.

Finishing the analysis

- Let $R_1, \ldots, R_m$ be a set of random events with $Prob[R_j] <= p_j$. Then $Prob[\text{at least one } R_j \text{ occurs}] <= p_1 + \ldots + p_m$. (This is called the union bound.)
- Let R_j be the event that element u_j is not covered. Then by the union bound, the probability that some u in U is not covered is $<= |U|\ (1/m)^c = (1/m)^{c-1}$.
- Using the Markov inequality we can also say that the expected cost is within *O(log m) OPT-LP* with good probability so that we get both a cover and cost *O(log m) OPT-LP* with good probability which certainly shows that with good probability we get a cover with cost *O(log m) OPT* since *OPT-LP <= OPT*.

Random walk algorithm for 2-SAT

- It is not difficult to show that 2-SAT (determining if a 2CNF formula is satisfiable) is efficiently computable whereas we know that 3SAT is NP complete. We will provide a conceptually simple randomized algorithm to show that 2SAT is computationally easy. The same basic approach can be used to derive a randomized (which in turn leads to a deterministic) algorithm for 3SAT that runs in time $(1.324)^n$. It is a big open question if one can get time $2^{o(n)}$ algorithm for 3-SAT.

Random walk on an undirected graph

- We consider a uniform random walk on a connected undirected graph $G = (V,E)$ with n nodes and m edges. That is, starting at some initial node, we uniformly at random chose a neighbour and move to that node.
- Defs: h_ij (expected) hitting time (note h_ij may not be equal to h_ji)

  c_ij (expected) commute time

  $C\_u(G)$ (expected) cover time starting at u

  $C(G) = max\_u\ C\_u(G)$ (expected) cover time

Stationary distribution and cover times

- Fact: If G is not bipartite, a stationary distribution exists for a uniform random walk on graph and that distribution is the vector

  $pi = <(d\_1)/2m, ...,(d\_n)/2m)>$ where d_i = degree of v_i and $m = |E|$.
- Can deal with bipartite graphs by looking at two step walks.
- Theorem: (Aleliunas, Karp, Lipton, Lovasz, Rackoff)

  The cover time $C(G) <= m^* (2n-1)$

The proof uses the previous concepts of hitting and commute times and the stationary distribution.

Application to 2SAT

RWALK (randomized algorithm to test if 2CNF *F* is satisfiable)

Choose a random (or arbitrary) initial truth assignment tau

For *i = 1 .. c *n^2* (for *c* sufficiently large)

If tau satisfies F, report that a satisfying assignment has been found

Else find an unsatisfied clause and choose one of its two literals ℓ_i^j at random. Change tau by flipping ℓ_i^j

End For

Claim: Let *tau** be a truth assignment satisfying *F*. Then we can view RWALK as a uniform random walk on a line graph (with nodes *0,1,…,n*) that is trying to reach node *0* where node *i* indicates that tau mismatches tau* in *i* coordinates.