

Disclaimer: *These are rough notes, with some exercises.*

Question: What would be a linear program for the vertex cover problem?

$$\min \sum_i x_i.$$

$$\forall e = (i, j) x_i + x_j \geq 1.$$

$$x_i \in \{0, 1\}$$

If this is integral solution. Then we are done. Is it? No. Consider a triangle. A valid LP solution is $1/2$ on all nodes for a total value of $3/2$. Any valid integral solution has value 2.

Question: Given a linear programming solution, can we get a close by integer solution?

Sure. Include any $x_i \geq 1/2$ in S . $|S|$ will be at most twice the cost of the linear program solution and S is a valid solution since at least one of the two endpoints of any edge must have $x_i \geq 1/2$. That is, this is a factor of two approximation since the LP value provides a lower bound.

Moreover, it may be better in practice since the LP may give a better guide than the greedy maximal matching algorithm from last time. Note, that even works for a weighted version of the linear program.

Question: Set Cover linear program?

Let's make it more interesting, and have it be weighted.

Exercise 1: Give an algorithm for weighted set cover with an $O(\log n)$ approximation factor.

$$\min \sum_s x_s C_s.$$

$$\forall \text{element } e \sum_{e \in S} x_s \geq 1.$$

$$x_s \in \{0, 1\}.$$

Let's illustrate random rounding for this problem. Round variable x_s to 1 with probability x_s . (None are bigger than 1 so it's all good here.)

Question: What is the expected cost?

The expected cost here is exactly the cost of the linear programming solution.

Question: What is the probability that an element is uncovered?

At most

$$\pi_s \text{ contains } (1 - x_s),$$

which is at most $1/e$ since $\sum_s x_s = 1$. Use calculus to observe that the worst is when all the x_s are equal and we get the expression $(1 - 1/k)^k$ once again.

Question: What to do?

Repeating the process $\ln n$ times gives a reasonable probability that everyone is covered.

Question: Other problems?

TSP (perhaps later). MAX-3SAT. LP has trivial solution (all literals get $1/2$.)

20.1 Facility location.

Question: Have you heard of facility location?

Given a set of clients C and possible facility locations F with opening costs f_i . Open a set of facilities and assign clients to them to minimize total cost of opening facilities and assigning clients (it costs c_{ij} to assign client i to facility j .)

Question: How does c_{ij} arise?

Euclidean point, costs, distances, many ways. For today, we consider distances. That is, they obey triangle inequality $c_{ij} + c_{i'j} + c_{i'k} \geq c_{i,k}$. Or one can go from i to a facility j' by going to a facility j going to its customer i' and then going to i' from j' .

Question: How to solve?

Well, let's consider that each client providing bids to open facility as follows. Each client begins growing a ball around himself, say for client i the diameter is v_i around himself looking for an open facility and say he is willing to pay facility j what he would save if j is open. That is, he is willing to pay $\beta_{ij} = v_i - c_{ij}$ to j since that is cost he is saving.

If a facility has enough bids to be opened, it is "fully paid." That is, when $f_j = \sum_i \beta_{ij}$, we say facility j is fully paid and say all clients that contain j in their ball are connected.

If a client is connected (has a fully paid facility in his ball, he stops growing or bidding anymore.) We continue growing unsatisfied clients.

Question: What then?

Should we open all "fully paid" facilities. No, each client pays to more than one facility. That would be a bad deal.

Question: What then?

We open first fully paid facility, j . And for any i that pays j that also "pays" for another fully paid facility j' . We delete that facility and for any client i' served by that facility we assign them to j . The cost of serving client i' from j is at most $v_{i'} + 2 * v_i$ by the triangle inequality. Since j was fully paid first we have

that the cost of serving i' at most $3v_{i'}$.

Question: What is the total cost?

The opened facility cost is at most

$$\sum_{i \text{ directly connected to } j} v_i - c_{ij}.$$

The total serving cost is

$$\sum_{i \text{ indirectly connected}} c_{ij} + \sum_{j \text{ not directly connected}} 3v_i.$$

Summing these together gets an upper bound of

$$\sum_i 3v_i,$$

on the total cost.

Question: So?

Well, perhaps, the $\sum_i v_i$ is a lower bound on the optimal solution?

Question: What is the linear program for facility location?

$$\min \sum_i x_{ij} c_{ij} + \sum_j f_j y_j.$$

$$\forall i \sum_j x_{ij} = 1.$$

$$\forall i, j x_{ij} \leq y_j.$$

All variables are in range $\{0, 1\}$. (Don't need upper bound on 1.)

Question: What is the dual?

$$\max \sum_i v_i.$$

$$\forall j \in F, i \in C, v_i - \beta_{ij} \leq c_{ij}.$$

$$\forall j \sum_i \beta_{ij} \leq f_j.$$

Hey, the algorithm above produced a feasible dual solution, v_i and β_{ij} . Thus, the $\sum_i v_i$ is a lower bound in the optimal. Moreover, it produced a facility location solution that is within a factor of three of the lower bound. Factor of 3 approximation!

Exercise 2: How do you recover the β_{ij} values from just the v_i 's (in either the primal-dual algorithm above or in the dual linear program.) This is easy enough by the way.

Question: How about starting with an LP solution?

OK, let's assume we have primal and dual LP solutions. Here is an algorithm for "rounding". Select remaining client with smallest v_j . Make a cluster of all facilities in v_j ball, and all clients that use those facilities. Remove and continue. Open cheapest facility, assign all clients to facility.

Question: Service cost?

At most $3v_i$ for each client i in cluster. So overall 3 times optimal.

Question: Facility cost?

The cheapest facility has cost cheaper than all the facilities used by j . And none of these are opened, but the y_j 's sum to 1. So, this cost is less than the cost of fractionally opening the facilities in this cluster in the lp optimal.

So, it is less than optimal.

Question: Can we do better?

OK, in a clustered center. Open one at random..next time...