

## Lecture 19 Approximation Algorithms : 4.1.08

*Lecturer: Satish Rao**Scribe: Rao, for now***Disclaimer:** *These are rough notes, with some exercises.***Question: Have you heard of NP-completeness?**

Of course. In any case, there is reading posted on the web. Chapter 8 of Dasgupta, Papadimitriou, and Vazirani.

But, essentially there are (many) unsolvable (well, at the least provably “universal” to solve) problems.

**Question: So, what to do?**

Kind of solve them.

**Question: Huh?**

We will give algorithms that provably get within some factor of optimal. Let’s begin.

**19.1 Set Cover.**

In the set cover problem, we are given subsets  $\{S_1, \dots\}$  of  $n$  elements, and wish to find the smallest set of subsets which contain all of the elements. For example, ice cream makes two of my three kids happy, while cookies makes a different two happy, broccoli makes no one happy.

**Question: What should I buy?**

Well, broccoli, of course. Unless I want my kids to be happy. Then, I suppose it would be ice cream and cookies. Hmmm.

**Question: How should I solve this problem?**

Let’s begin by trying the obvious and choosing the largest set. Removing all of its elements and continuing.

**Question: Is this optimal?**

No. Consider the following example. Given  $n$  elements. We include in our set of subsets, the first half of the elements and the second half. That is, there is a solution consisting of 2 subsets.

We also include in our subsets a subset that contains slightly more than half of each half. Then we include a subset that contains slightly more than half of what remains. And so on. This adds  $O(\log n)$  sets.

The greedy algorithm chooses the latter cover.

One can make this worse. As bad as  $\ln n - o(\ln n)$  in fact.

**Question: Is that as bad as it gets?**

Yes.

**Question: Why?**

Ok, consider an optimal solution and that its size is  $k$ . Say  $n_i$  elements are left uncovered at step  $i$ .

**Question: At least how many do you cover at this step?**

At least  $n_i/k$ .

**Question: How many are left after this step?**

At most  $n_i - n_i/k$  or at most  $n_i(1 - 1/k)$ .

**Question: When are you done?**

The value of  $t$  where the inequality

$$(1 - 1/k)^t < 1/n$$

first holds is an upper bound. This is at most  $k \ln n$  using the fact that  $(1 - x)^{1/x} \approx e - 1$ .

**Question: Can we do better?**

No. To do better would imply that any NP problem is solvable in something like  $n^{O(\log^k n)}$  time (or quasipolynomial time.)

## 19.2 Vertex Cover.

In a graph find a minimum sized set  $S$  of nodes where each edge has at least one endpoint in  $S$ . This is NP-complete.

**Question: An approximation algorithm?**

Sure this is a set cover problem, where each edge is an element and each node is a set.  $O(\log n)$  approximation!

**Question: Can we do better?**

Sure. Choose an uncovered edge. Place *both* endpoints in the set  $S$  and repeat.

**Question: Why is this better?**

Consider the set of edges that we chose. Call it  $M$ . It is a matching. That is, no two edges in  $M$  share an endpoint.

Any vertex cover must cover  $M$ , and to do so, must have size  $|M|$ .

**Question: OK. So?**

Well, the set has size  $2|M|$ . So, we are within a factor of 2 of the lower bound on the optimal (and of course within a factor of 2 of the optimal.)

**Question: Slightly counterintuitive, but..**

We construct a lower bound on the optimal in this algorithm. That's an important concept. In the set cover analysis, we argued against an unknown optimal (or in some sense in each step we construct a lower bound on the optimal.)

**Question: Can we do better?**

There is a lower bound of  $4/3 - \epsilon$  based on NP not being too easy. There is a lower bound of  $2 - \epsilon$  based on a controversial conjecture about something called unique games.

## 19.3 Travelling Salesman Problem.

Given a set of cities and a set of “distances” between them, can you find a tour (a cycle that visits every city once) of minimum cost.

**Question: Approximation?**

Not in general. The Hamiltonian path problem which is to find a cycle that visits every city once (without repeats) is NP-complete. And one can reduce this to TSP, where any approximation for the TSP problem gives a solution to the Hamiltonian path problem.

**Question: How?**

In reading. But, transform the hamiltonian path input or graph  $G$  into a TSP input by giving distance one to the edges in the  $G$  and distance “a lot” to the non-edges. Any solution that doesn’t contain an edge that weighs “a lot” corresponds to a hamiltonian path in  $G$ . By adjusting “a lot” one can befuddle any claims of approximation.

**Question: That’s just cheating. Yes?**

Yes. Let’s consider that you can (re)visit a city (generally don’t make horrible enemies.) Then the distances must obey the triangle inequality. That is,  $d(i, j) + d(j, k) \leq d(i, k)$ , since to get from  $i$  to  $k$ , you could always go through  $j$ .

**Question: How do we approximate in such a case?**

Take a minimum spanning tree. It has cost at most the cost of the TSP. Then walk along the tree (short-cutting if possible.)

**Question: Approximation factor?**

Two. You walk along each edge twice.

**Question: Can we do better?**

Well. We double the cost of the tree. Is there a better structure.

**Question: Consider a connected even degree graph. Is there a tour that only hits each edge once?**

Sure. Take a walk, if you enter a node you can always leave (no “Hotel California”) until you get back to the beginning. This is a cycle. Begin again, at a point with available edges. Inductively, can always start/end previous walk at such a point.

**Question: Convert tree to even degree graph?**

Perhaps add degree to odd degree vertices. At minimum cost.

**Question: Ok, so we want to add a minimum cost set of edges so that each odd degree vertex gets one more adjacency. How?**

Minimum cost bipartite matching!!

**Question: How much does this cost, compared to TSP?**

At most half the cost of the TSP. The TSP is a cycle of the cities. One can split the cycle into two matchings of the odd nodes along the cycle, one of which costs at most half.

**Question: Can we do better?**

It is open. The next threshold that researchers are trying to address is  $4/3$ . On the other hand, one cannot do better than  $1.01$  under reasonable assumptions about NP.

Interesting project to examine and summarize results around this problem.

## 19.4 Max Sat

Given a CNF formula. Find an assignment that satisfies as many clauses as possible. For now, let's just consider 3SAT.

**Question: How?**

Randomly.

**Question: How do we do?**

Well. We get on expectation  $7/8$  of them. Since each clause fails to be satisfied with probability  $(1/2)^3$ .

**Question: Can we do this well?**

Perhaps. But, one can do so deterministically using something called the method of expectations.

**Question: How does this work?**

Well. The expectation of the random algorithm is easy to compute for any set of clauses. Now, for a variable we can simply check which set of clauses has better value when setting it to true or false.

In the end, we must do as well as the initial expectation. (Note, this is essentially some sort of greedy algorithm.)

**Question: Can we do better?**

Probably not. Unless NP is easier than we think  $7/8 - \epsilon$  is the best possible.

## 19.5 Linear Programming Approaches.

**Question: Can we set them up the previous problems as linear programs?**

Yes. We'll see.

**Question: Are we going to get integral solutions?**

No. Otherwise NP would be easy since linear programs are easy.

**Question: Linear program for set cover?**

Next time.

**Exercise 1: Give a randomized approximation algorithm that gives you a factor of  $1/2$  approximation algorithm for MAX 2-SAT. (Satisfy the maximum number of a set of 2-literal clauses.)**