6.854 Advanced Algorithms

Lecture 14: 10/27/2003 Lecturer: David Karger

Scribe: Sachin Katti

Relaxation Techniques

14.1 Introduction

Given a problem P we solve a different relaxed version of the problem $P^{'}$ which is related to P. $P^{'}$ is chosen such that every solution to P is feasible for $P^{'}$. Hence the optimum for $P^{'}$ bounds the optimum for P.

But once $P^{'}$ has been solved, we have to find a way to convert that solution to that of P which will involve some conversion cost. This will bound the approximation algorithm. We consider some examples to illustrate the technique.

14.2 Travelling Salesman Problem

14.2.1 Problem formulation

Given a set of cities(vertices V) and distances(edge lengths E) between them the TSP is to find a hamiltonian cycle (i.e. a tour such that each vertex is visited exactly once) of minimum cost.

14.2.2 Metric TSP

The problem is NP-hard, hence there is no approximation algorithm for the problem. But if we restrict the problem to a TSP with the triangle inequality i.e. $d(i,j) + d(j,k) < d(i,k)$ for all $i, j, k \in V$, we can find approximation algorithms. This problem is called the Metric TSP. Consider the following algorithm

- Construct a Minimum Spanning Tree for the graph.
- Take an Euler tour around the MST.

The idea is to remove the constraint of takinbg a tour, instead visit everything in as cheap a manner as possible. Since every tour without the last edge is a spanning tree, Cost of MST < Cost of TSP. But in the Euler tour each edge is visited twice. Hence

$$\text{Cost of Euler Tour} \leq 2^*\text{Cost of MST} \leq 2^*\text{Cost of Opt TSP} \tag{14.1}$$

To get a tour, we just shortcut the Euler tour, i.e. when we are visitng a vertex for the second time while going back, instead take the shortest path from the previous vertex to the next vertes on the tour. Due to the triangle inequality above, the cost of the resulting tour will be less than that of the Euler tour. Thus we have a 2 approximation algorithm.

Christofedes' heuristic for Metric TSP

From the above algorithm we see that basically we need to backtrack from odd degree vertices. Hence if we can modify the graph such that all vertices have even degree, we can find a much better approximation. Hence the algorithm is as follows

- Compute the minumum spanning tree T of the graph $G = (V, E)$.
- Let O be the odd degree vertices in T. $|O|$ is even since the sum of the degress of vertices in T is even and if we consider only the odd degree vertices, their sum must add up to an even number which implies that $|O|$ is even.
- Compute a mim-cost perfect matching M on the graph induced by O.
- Add the edges in M to E. Now the degree of every vertex in G is even, Therefor G has an Eulerian tour. Trace the tour and take shortcuts when the same vertex is reached twice. This cannot increase the cost due to the triangle inequality.

We claim that the cost of M is no more than half the cost of $Opt(TSP)$. Consider the optimal tour visiting only the vertices in O. Clearly by the triangle inequality, this is of length no more than $Opt(TSP)$. There are an even number of vertices in the tour and hence even number of edges. The tour defines two disjoint matchings on the graph induced by O. Atleast one of these has cost $\leq \frac{1}{2}Opt(TSP)$, and hence the cost of M is no more than this.

Therefore the total cost of the resulting Eulerian tour is no more than $\frac{3}{2}Opt(TSP)$ and hence we get a $\frac{3}{2}$ approximation algorithm.

14.2.3 Directed TSP

The directed TSP can be relaxed to a problem of finding a min cost cycle cover. In min cost cycle cover we find a way to cover all the vertices with directed cycles. To go back to solving TSP, we replace each cycle by a vertex and recursively apply the above algorithm until we can find a TSP tour on the reduced graph. Once we have found a tour we use it to patch the cycles together into one cycle.

The patching cost is at most 2 times that of the cycle cover since we might have to go at most twice over each edge. Hence we get the following recursion

$$\text{Cost}(n) = 2 * \text{Cost}(\text{Cycle Cover}) + \text{Cost}(\frac{n}{2}) \leq 2 * \text{Cost}(\text{TSP}) + \text{Cost}(\frac{n}{2}) \leq 2 * \text{logn} * (\text{Cost}(\text{TSP})) \tag{14.2}$$

14.3 LP relaxation techniques

Almost any problem can be expressed as an integer LP. But integer LP can be relaxed very easily by just allowing non-integer solutuions. We then have to find a way of converting the non-integer solutions back to integer solutions. We use the vertex cover problem to illustrate the technique.

14.3.1 Minimum Cost Vertex Cover

Problem Formulation

A vertex cover U in a graph $G = (V, E)$ is a subset of the vertices such that every edge is incident to atleast one vertex in U. The vertex cover problem is defined as follows

Definition 1 ***Given a graph*** $G = (V, E)$ ***and costs*** $c(v)$ ***for each vertex*** $v \in V$***, find a vertex cover*** $U \in V$ ***which minimizes*** $c(u) = \sum_{v \in U} c(v)$.

LP formulation and relaxation

The corresponding integer linear program for the problem is as follows

$$C_{opt} = min \sum_{v \in V} c(v)x(v) \tag{14.3}$$

where

$$x(v) + x(w) \geq 1 \qquad \forall (v, w) \in E \tag{14.4}$$

$$x(v) \in \{0, 1\} \qquad \forall v \in V \tag{14.5}$$

This LP can be relaxed by allowing non integer solutiuons as follows

$$LB = min \sum_{v \in V} c(v)x(v) \tag{14.6}$$

where

$$x(v) + x(w) \geq 1 \qquad \forall (v, w) \in E \tag{14.7}$$

$$x(v) \geq 0 \qquad \forall v \in V \tag{14.8}$$

Rounding

Let x^* be the optimal solution of the LP relaxation. Let

$$U = \{v \in V : x^*(v) \geq \frac{1}{2}\} \tag{14.9}$$

We claim U is a 2-approximation of the minimum cost VC. Clearly U is a vertex cover because for $(u,v) \in E$ we have $x^*(u) + x^*(w) \geq 1$, which implies $x^*(u) \geq \frac{1}{2}$ or $x^*(v) \geq \frac{1}{2}$. Also

$$\sum_{v \in U} c(v) \leq \sum_{v \in V} c(v) 2x^*(v) = 2 * LB \tag{14.10}$$

since $2x^*(v) \geq 1$ for all $v \in U$.
Since $LB \leq C_{Opt}$ we have a 2-approximation algorithm for min-cost VC.

14.3.2 Max SAT

Problem Formulation

Given a collection of OR clauses C_j over some literals x_i, find an assignment of boolean values to the literals which maximises the number of true clauses.
The approximation technique is to use a random guess for each x_i. Since they are OR clauses

$$\text{Probability(given clause is satisfied)} \geq \frac{1}{2} \tag{14.11}$$

Hence

$$\text{E[No of satisfied clauses]} \geq \frac{\text{\# of clauses}}{2} \tag{14.12}$$

LP formulation and relaxation

Now consider the integer LP formulation for the problem

$$N_{opt} = max \sum z_j \tag{14.13}$$

$$\sum_{X_i \text{ positive in } C_j} y_i + \sum_{x_i \text{ negated in } C_j} (1 - y_i) \geq z_j \qquad 0 \leq y_i \leq 1 \qquad 0 \leq z_j \leq 1 \tag{14.14}$$

where

$$z_j = 1 \text{ if } C_j \text{ is satisfied else } 0 \tag{14.15}$$

$$y_i = 1 \text{ if } x_i \text{ is true, else } 0 \tag{14.16}$$

Now relaxing the integer LP, we allow non-integer solutions. To convert the solution back to the original problem we use the following rounding technique:

Set x_i true with probability y_i.
We claim that a k variable literal clause C_j is satisfied with probability $\geq \beta_k z_j$ where

$$\beta_k = 1 - (1 - \frac{1}{k})^k \tag{14.17}$$

To prove the above claim consider the following

$$\text{Probability}(C_j \text{ is satisfied}) = 1 - \prod_{x_i \in C_j} 1 - y_i \tag{14.18}$$

This probability is minimized when all of the y_is are equal, i.e. $y_i = \frac{z_j}{k}$. Hence a clause C_j is satisfied with probability $\geq (1 - \frac{1}{e})$.
Hence

$$\text{E[No. of satisfied clauses]} = N_{\text{expected}} \geq (1 - \frac{1}{e}) \sum z_j \tag{14.19}$$

But since $\sum z_j \geq N_{opt}$, the expected number of satisfied clauses, $N_{\text{expected}} \geq (1 - \frac{1}{e}) N_{opt}$. Hence we have a $(1 - \frac{1}{e})$ approximation algorithm.