COMP 670J – APPROXIMATION ALGORITHMS

BACKGROUND

- Linear Algebra
  - matrices
  - vectors
  - linear transformations
  - etc.
- Probability Theory
  - Probability Spaces
  - Expectation, Variance
  - Basic Distributions (Binomial, Poisson, Exponential,...)
  - etc.
- COMP271, especially
  - $O()$ notation,
  - data structures
  - graph algorithms, e.g. Breadth and Depth first search, minimal spanning trees, topological sort.
  - P, NP, and NP completeness.

References

Most of the material in this lecture can be found in

- Cormen, Leiserson, Rivest. Chapter 37.
- Vazirani. Chapter 1

General facts about graphs and graph algorithms can be found in Cormen, Leiserson, Rivest. Background on NP-Complete and NP-hard problems is in Garey and Johnson.

Vertex Cover

A *vertex cover* of an undirected graph $G = (V, E)$ is a subset of vertices $V' \subseteq V$ such that
if $(u, v) \in E$ *then either* $u \in V'$, $v \in V'$ *or both.*

```
Approximate Vertex Cover
C ← ∅
E' ← E[G]
while E' is not empty
        let (u, v) be any edge from E'
        C' ← C ∪ {u, v}
        remove from E' every edge touching u or v.
return(C)
```

For any graph G define

$$\begin{aligned} AVC(G) &= \text{size of (\# of vertices in) } C,\ |C| \\ OPT(G) &= \text{size of smallest vertex cover of } G. \end{aligned}$$

Theorem: C is a vertex cover and for all G

$$1 \leq \frac{AVC(G)}{OPT(G)} \leq 2.$$

"Approximate Vertex Cover" is an *approximation algorithm* for the Vertex Cover problem. It does not find an optimal (minimal size) vertex cover but instead finds an approximately optimal one.

AVC is a *factor two approximation algorithm* for the minimal vertex cover problem. This is sometimes also called a 2-*approximation algorithm*

For any given optimization (minimization) problem and approximation algorithm $\mathcal{A}$ to solve it:

- Let Π = Set of all *instances* of the problem.
- For all instances $I \in \Pi$ define

$$\begin{aligned} OPT(I) &= \text{cost of optimal solution for } I \\ A(I) &= \text{cost of solution produced by } A \text{ on } I. \end{aligned}$$

Then if

$$\forall I \in \Pi, \quad \frac{A(I)}{OPT(I)} \leq \rho$$

A is a *factor* ρ *approximation algorithm (*ρ*-approximation algorithm).*

WHY IS THIS INTERESTING?

It is because finding optimal vertex covers is an NP-Hard problem, which means that it has no polynomial time solution except if $P = NP$ Many (most?) interesting combinatorial problems are NP hard yet we NEED to be able to solve them.

Part of the theory of approximation algorithms is devoted to finding good *polynomial time* approximation algorithms for NP-hard problems. That is the motivation for the first part of this course.

The approach of this course is theoretical, devoted not to getting the best results in practice, but to getting the best provable results.

Note: Approximate Vertex Cover runs in time $O(|V| + |E|)$ given the appropriate representation of graphs as adjacency lists.

NP-Complete vs. NP-Hard

An optimization problem: find a best something

Given graph G find a minimal size vertex cover.

A decision problem: has a yes/no answer:

Given graph G and integer k, does G have a vertex cover of size $\leq k$.

NP-complete problems are, by definition, decision problems. If they can be solved in polynomial time then $P = NP$.

Problems that have the property that if they can be solved in polynomial time then $P = NP$ but not necessarily vice-versa are called NP-Hard. The optimization versions of NP-complete decision problems are NP-Hard.

Three Graph Problems

- Hamiltonian Cycle (HC): NP-Complete
- Travelling Salesman Problem (TSP): NP-Hard
- Eulerian Tour (ET): Polynomial time solvable

Let $G = (V, E)$ be a graph. A *Hamiltonian Cycle* is a tour (cycle) in G passing through every vertex exactly once.

HC: Does $G = (V, E)$ contain a Hamiltonian Cycle?

Now let $G = (V, E)$ be a *complete* graph such that every edge $e \in E$ has a *cost,* $c(e)$. The cost of a set of edges $E' \subseteq E$ is $cost(E') = \Sigma_{e \in E'} c(e)$. The cost of a Hamiltonian cycle T is then $c(T)$, the sum of the costs of its edges.

TSP: Find a Hamiltonian cycle, T, with minimal cost among all cycles.

Now let $G = (V, E)$ be any graph again. This time we are even allowed to have repeated edges. An *Eulerian Tour* is path in the graph that uses each edge in the graph exactly once (a repeated edge is used exactly as many times as it appears). Such a tour exists if and only if the degree of each vertex is even. In this case a tour can be found in polynomial time, $O(|V| + |E|)$.

Metric TSP

A graph G with cost $c()$, has the *triangle inequality* if for all vertices $u, v, w \in V$,

$$c(u, w) \leq c(u, v) + c(v, w).$$

A graph with the triangle inequality is called a *Metric Graph.*

<u>Metric TSP</u> 1. Find a Minimum Spanning Tree T' of G. 2. Double every edge in T' to get new graph G'. 3. Find an Eulerian Tour $\mathcal{T}'$ of G' 4. Output the vertices of G in the order in which they first appear in $\mathcal{T}'$. Let T be the Hamiltonian cycle thus created.

<u>Lemma:</u> Metric TSP is a 2-approximation algorithm for the TSP problem on Metric Graphs. Furthermore, it runs in polynomial time in the graph size.
<u>Proof:</u>

- $c(T') \leq OPT(G)$ since erasing an edge from *any* Hamiltonian cycle gives a tree and *every* tree in G has cost $\geq c(T')$.
- $c(\mathcal{T}') = 2c(T')$.
- $c(T) \leq c(\mathcal{T}')$ from the triangle inequality.

Combining these three facts gives

$$c(T) \leq 2c(T') \leq 2 \cdot OPT(G).$$

Lemma: Let G be metric graph and T, T' be paths such that

$$\begin{aligned} T &= v_1 v_2 \ldots v_{i-1} v_i v_{i+1} \ldots v_m \\ T' &= v_1 v_2 \ldots v_{i-1} v_{i+1} \ldots v_m \end{aligned}$$

Then $c(T') \leq C(T)$.
Proof:

$$\begin{aligned} c(T') &= c(v_m, v_1) + \sum_{j<m} c(v_j, v_{j+1}) \\ &\quad + c(v_{i-1}, v_{i+1}) - c(v_{i-1}, v_i) - c(v_i, v_{i+1}) \\ &= c(T) + c(v_{i-1}, v_{i+1}) - c(v_{i-1}, v_i) - c(v_i, v_{i+1}) \\ &\leq c(T) \end{aligned}$$

where the last inequality follows from the triangle inequality.

Corollary: Let $T = v_1 v_2 v_3 \ldots v_m$ be any path in G. Let T' be the tour that results by writing the vertices of G in the order that they first appear in T. Then $c(T') \leq c(T)$.
Proof: Repeated applications of the previous lemma.

$\boxed{G \text{ has a Hamiltonian cycle if and only if } A(G') \leq \rho |V|.}$

A polynomial time algorithm for solving Hamiltonian cycle follows since, given any G we can simply (i) construct G' and $c()$ and (ii) run A on G' and check if $A(G') \leq \rho |V|$ or not.

A Negative Result

Lemma: If, for *any* $\rho \geq 1$, there exists a polynomial time ρ-approximation algorithm for TSP, then there exists a polynomial time algorithm for solving Hamiltonian cycle.

Corollary: If, for *any* $\rho \geq 1$, there exists a polynomial time ρ-approximation algorithm for TSP, then $P = NP$.

Proof of Lemma: Suppose there exists a ρ-approximation algorithm A.. Let $G = (V, E)$ be any instance of Hamiltonian cycle. Let G' be the complete graph with the same vertex set as G and with

$$c(u, v) = \begin{cases} 1 & (u, v) \in E \\ \rho |V| + 1 & \text{otherwise} \end{cases}$$

G' and $c()$ can be constructed in time polynomial in the size of G.

If G has a Hamiltonian cycle T then all edges $e \in T$ have $c(e) = 1$ so T is a min cost tour in G' and $OPT(G') = |V|$. Thus $A(G') \leq \rho \cdot OPT(G') = \rho |V|$.

If G does not have a Hamiltonian Cycle then *every* Hamiltonian cycle T, in G' contains at least one edge $e \notin E$ so $c(e) = \rho |V| + 1$ and

$$c(T) \geq c(e) + |V| - 1 = \rho |V| + 1 + |V| - 1 > \rho |V|.$$

Thus $A(G') \geq OPT(G') > \rho |V|$.