Lec 22

Sunday, November 9, 2025 12:24 PM

* HW6 out, due Monday
* Exam 3 on Nov 20 (Thurs)
* lecture on Nov 18 virtual and optional (not part of Exams)

Algorithmic Reductions

Idea: Reduce (convert) solving problem A efficiently to a different problem B, i.e., a 'good' soln to pb B $\Rightarrow$ a 'good' soln to pb A.

Benefits:

1) Algorithm: If we can solve B efficiently then A is efficient.

E.g., reduce max flow to solving an LP

2) Hardness: If solving A is Hard then solving B is also hard, as otherwise we can use efficient alg for B to solve A.

E.g., if A is NP-complete & we reduce A to pb B $\Rightarrow$ B is also NP complete.

Example 1: Convex Optimization

Problem A: $\min_{x \in K} f(x)$ where f is convex with $\|\nabla f\| \leq G$ and $K \ni 0$ with $\text{diam}(K) = D$

Problem B: $\min_{x \in K} g(x)$ where g is α-strongly convex with $\|\nabla f\| \leq G$ in time $\frac{G^2}{\alpha T}$ and $K \ni 0$

Thm: Reduce A to B such that time T_G

αT

Thm: Reduce A to B such that time $\frac{DG}{\sqrt{T}}$

Pf: Suppose $g(x) = f(x) + S \cdot \|x\|^2$ for $S = \frac{G}{D\sqrt{T}}$

Note g is S-strongly convex ← Exercise

Hence, after T iterations, we have

$$g(Alg) - g(x^*) \leq \frac{G^2}{ST} \quad \text{where } x^* = \operatorname*{argmin}_{x \in K} f(x)$$

$$\Rightarrow \left[f(Alg) + S\|Alg\|^2\right] - \left[f(x^*) + S\|x^*\|^2\right] \leq \frac{G^2}{ST}$$

$$\Rightarrow f(Alg) - f(x^*) \leq \frac{G^2}{ST} + S\|x^*\|^2 \leq \frac{G^2}{ST} + SD^2 \leq \frac{2DG}{\sqrt{T}}.$$

∎

Example 2: Set Cover via Max-Coverage

- Universe of elements $[n] = \{1, 2, \ldots, n\}$
- m subsets $\mathcal{S} = \{S_1, \ldots, S_m\}$ where $S_i \subseteq [n]$ and $\bigcup_{i \in [m]} S_i = [n]$

Problem A (Set Cover): min # subsets s.t. their union is $[n]$

Problem B (Max-Coverage): Given k, find at most k subsets to maximize size of union

Thm: We can reduce A to solving B such that $\frac{1}{2}$ approx

Thm: We can reduce A to solving B such that $\frac{1}{2}$ approx for B in polytime implies $\log_2 n$ - approx for A in polytime.

Remark: (1) This about approx ratios
(2) We will call B multiple times.

Pf: Given an instance for A (set cover), start by guessing the size of optimal soln, say k ← only m choices

Alg: (1) Starting with sets $S_1, \ldots S_m$, solve B with size k.

(2) Add these k sets into Alg and update each $S_i \leftarrow S_i \setminus \left\{ \bigcup_{j \in Alg} S_j \right\}$. ← Remove covered elements

Repeat till all sets are empty.

To prove the theorem, we just need to show $\log n$ iterations.

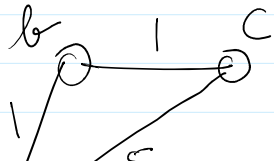
This is true because each subproblem B has a soln that covers all the remaining elems.

Since we get a 2-approx in each iteration

$\Rightarrow$ # remaining elems becomes half

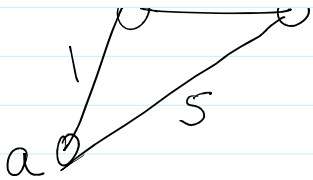
$\Rightarrow \leq \log_2 n$ iterations. ∎

Example 3: Hardness for non-metric TSP



Non-Metric TSP:

Non-Metric TSP:



Given an undirected graph $G = (V, E)$

weights $w : E \to \mathbb{R}_{\geq 0}$

Goal: Find a tour π (cycle with vertex repeats NOT allowed) that visits all vertices and has shortest length.

Thm: Assuming $P \neq NP$, there is no α-approx alg for non-metric TSP (for any $\alpha \geq 1$).

Hamiltonian Cycle: Given a graph $G = (V, E)$, does there exist a tour that visits every vertex exactly once (and returns back).

— this pb is NP-complete.

Proof: Problem A = Hamiltonian Cycle on $G = (V, E)$

Problem B = Non-metric TSP on complete graph with $w_e = \begin{cases} 1 & \text{if } e \in E \text{ and} \\ \alpha n & \text{for } e \notin E \end{cases}$

We can use α-approx for B to solve A exactly.

This is ∵ if G has Hamiltonian cycle then $opt(B) = n$

and otherwise $opt(B) > \alpha n$

Hence, α-approx for B tells which case we are in. ∎