Lecture 25: May 14, 2002

Today: Approximation algorithms
Coping with NP-hardness

- “Efficient” exponential time algorithms for small inputs (e.g., $(100/99)^n$ time is not bad for $n \leq 1000$)
- Polynomial time algorithms for some (e.g., random) inputs
- Polynomial time algorithms for all inputs, but incorrect (popular approach, but not advised)
- Polynomial time algorithms for all inputs, but approximate
Approximation algorithms

Consider *minimization* problem $\Pi$. An algorithm $A$ for $\Pi$ is *$\rho$-approximate*, if for any input, the cost $C_A$ of the solution it produces is at most $\rho$ times the cost $C^*$ of the optimal solution.

E.g., an algorithm which produces TSP tour at most 50\% longer than the optimal one, is 1.5-approximate.

Similar definition for *maximization* problems.

We are only interested in polynomial time approximation algorithms.
Examples of approximation algorithms

Are there any approximation algorithms?

Yes, almost every NP-hard problem has a corresponding approximation algorithm. In particular:

- TSP: has a 2-approximation algorithm (can be improved to 1.5).
- Set Cover: has $O(\log n)$-approximation algorithm.
- Vertex Cover: has 2-approximation algorithm.
TSP

- Input: Undirected complete graph with lengths $c(u, v)$ on edges.
- Output: Smallest tour that visits every vertex exactly once.
- Problem is NP-hard:
  - Hamiltonian cycle (HC): given an unweighted graph, is there any tour which visits any vertex exactly once?
  - HC is NP-hard (CLRS)
  - HC is poly time reducible to TSP:
    * if $\{u, v\}$ in the graph $\Rightarrow c(u, v) = 1$
    * if $\{u, v\}$ not in the graph $\Rightarrow c(u, v) = W$
  - HC exists iff there is a TSP tour with cost $= n$
  - No HC $\Rightarrow$ any TSP tour has cost $\geq W + (n - 1)$
Approximating TSP

- But this means there is no poly time \( W/n \)-approximation algorithm (otherwise we have a poly time algorithm for HC)
- \( W \) can be very large \( \Rightarrow \) panic?
- Suppose weights satisfy **triangle inequality:**

\[
c(u, w) \leq c(u, v) + c(v, w)
\]

for all \( u, v, w \in V \)

\[
a + b \leq c
\]

- TSP is still NP-complete (take \( W = 2 \))
- However, it *can* be solved approximately
TSP Approximation Algorithm

APPROX-TSP-TOUR

1. Build a minimum spanning tree $T$ for $G$
2. Let $L$ be the list of vertices visited in preorder tree walk of $T$
3. return a tour $H$ that visits vertices in the order $L$
Example

• Find shortest tour for:

• Find MST.
Example

- Short-cutting walk of MST

- Yielding tour:

- Total distance $\approx 24.00$ units.
Optimal Solution

- Total distance $\approx 20.44$ units.
Approximation factor

**Theorem:** APPROX-TSP-TOUR with triangle inequality has ratio bound 2.

**Proof:**

- $H^*$ = optimal tour for $G$.
- $T$ is a MST for $G \Rightarrow c(T) \leq c(H^*)$.
- $W = \text{full walk of } T \Rightarrow c(W) = 2c(T)$ (each MST edge is taken twice)
- $H$ is the tour obtained by shortcutting $T$. By triangle inequality, $c(H) \leq c(W)$
- $c(H) \leq c(W) \leq 2c(T) \leq 2c(H^*)$
Set cover

Set cover (SC):

- **Input:** a family \( S \) of sets \( S_1 \ldots S_n \subset U \), \( |U| = m \), such that \( \bigcup_{i} S_i = U \).

- **Output:** A set \( I \subset \{1 \ldots n\} \) of smallest size, such that

\[
\bigcup_{i \in I} S_i = U
\]

Very natural problem, e.g.:

- **\( U \)-** the set of tasks

- **\( S_i \) -** the set of tasks which can be performed by person \( i \)

- Want to find the minimal team which can perform all tasks

Unfortunately, problem is NP-complete (previous lecture).
Greedy algorithm

Repeat until all elements are covered:

- Choose a new set $S_i$ which contains largest number of yet uncovered elements
- Add $i$ to $I$
- Mark all elements from $S_i$ as covered

Natural algorithm, but can be fooled
(e.g., consider sets $\{1, 2\}$, $\{3, 4\}$, $\{5, 6\}$, $\{1, 3, 5\}$).

Nevertheless, it is an ln $m$-approximation algorithm
Proof

Let $k$ be the size of the minimal cover.

**Fact:** At any iteration of the algorithm, there is at least one set $S_i$ which covers $\geq 1/k$ fraction of yet uncovered elements.

Note 1: Such $S_i$ cannot be in the current cover (i.e., $i \notin I$), since by definition no $S_i$ included in the current cover contains any uncovered elements (i.e., all such elements are marked as covered).

Note 2: The above note is not relevant for the correctness of the algorithm. It was included since a few students asked about it. If it confuses you, just ignore it.

**Proof:** If each set covered $< 1/k$ fraction of (yet uncovered) elements, then there would be no way to cover $U$ using only $k$ sets.
\textbf{$\ln m$-approximation}

Let $u_i$ be the number of uncovered elements after the $i$th step. We know that

$$u_{i+1} \leq (1 - 1/k)u_i$$

Initially, $u_0 = m$, thus

$$u_i \leq (1 - 1/k)^i m$$

Therefore, for $i = k \ln m$ we have

$$u_i \leq (1 - 1/k)^k \ln m m \leq e^{-\ln m} m = 1$$

and thus after $k \ln m + 1$ steps the algorithm covers all elements and stops.

Thus, greedy is $(\ln m + 1/k)$-approximate (we skip the $1/k$ term for simplicity).
Set cover summary

- Greedy gives $\ln m$-approximation
- Cool fact: Feige’95 showed that no $(1 - \epsilon)\ln m$-approximation algorithm exists, unless $P=NP$
- Not-so-cool (but useful) fact: greedy is in fact

$$(\ln m - \ln \ln m + 0.78) - \text{approximate}$$
**Vertex cover**

- **Input:** Undirected graph \((V, E)\).
- **Output:** A set \(C \subseteq V\) of minimum size such that every edge in \(E\) has at least one endpoint in \(C\) ?
- **Special case of set cover,** when every element is contained in only two sets
Approximation Algorithm

**APPROX-VERTEX-COVER**($G$)

1. $C \leftarrow \emptyset$
2. $E' \leftarrow E[G]$
3. while $E' \neq \emptyset$
   
4. do let $\{u, v\}$ be an arbitrary edge of $E'$
5. $C \leftarrow C \cup \{u, v\}$
6. remove from $E'$ every edge incident to either $u$ or $v$
7. 
8. return $C$
Vertex Cover Approximation Example
Vertex Cover Approximation Example Cont.
Analysis of Vertex Cover Approximation

- **Correctness**
  - Only remove "covered" edges from $E'$.
  - **Approx-Vertex-Cover** returns a vertex cover.

- **Running Time** is $O(V + E)$. 

Further Analysis

• Theorem Approx-Vertex-Cover has ratio bound 2.

• Proof

  – \( A = \{ \text{edges picked in line 4} \} \) (\( A \) is a set).
  – No two edges in \( A \) share an endpoint.
  – \( |C| = 2 |A| \).
  – Optimal cover, \( C^* \) must include at least one end-point for each edge in \( A \).
  – \( |A| \leq |C^*| \).
  – Conclude that \( |C| \leq 2 |C^*| \);
    that is, size of approximate cover is at worst twice size of optimal cover!