Outline

Approximation Algorithms

Set-Cover

Polynomial Time Reductions
Approximation Ratios

Definition

The \textit{performance ratio} of an algorithm is

\[ \max_{x : |x| = n} \frac{C_{\text{alg}}(x)}{C_{\text{opt}}(x)} \quad \text{for a minimization problem} \]

\[ \max_{x : |x| = n} \frac{C_{\text{opt}}(x)}{C_{\text{alg}}(x)} \quad \text{for a maximization problem} \]

where $C_{\text{alg}}(x)$ is the value of the algorithm solution on input $x$ and $C_{\text{opt}}(x)$ is the value of the optimal solution on input $x$. 

Outline

Approximation Algorithms

Set-Cover

Polynomial Time Reductions
Set-Cover

Problem:
- Input: A collection $C = \{S_1, S_2, \ldots, S_m\}$ of subsets of $\{1, 2, \ldots, n\}$
- Output: Find $C' \subset C$ such that

$$\bigcup_{S \in C'} S = \{1, 2, \ldots, n\}$$

that minimizes $|C'|$. 
Approximation Algorithm for Set Cover

Algorithm

1. $C' = \emptyset$.
2. Repeat until all elements are covered:
   2.1 Let $S$ be the set that covers the most new elements: $C' \leftarrow C' \cup \{S\}$.

Theorem

The algorithm is a $\ln n$-approximation and runs in polynomial time.
Approximation Algorithm for Set Cover: Analysis

Suppose it is possible to cover all elements with $k$. Whenever you haven’t covered all the elements, there’s a set that covers at least $1/k$ fraction of the uncovered elements.

After $t$ sets have been chosen the number of uncovered elements is

$$n(1 - 1/k)^t < ne^{-t/k}$$

For $t = k \ln n$ this is less than 1, i.e., all elements have been covered.
Outline

Approximation Algorithms

Set-Cover

Polynomial Time Reductions
Problem 1: Clique

Definition
A clique of size $k$ in a graph $G$ is a completely connected subgraph of $G$ with $k$ vertices.

- **Input:** Given graph $G = (V, E)$ and integer $k$.
- **Question:** Does $G$ contain a clique of size $k$?
Problem 2: 3-SAT

- **Input:** A boolean formula $\phi(x_1, \ldots, x_n)$ in *conjunctive normal form* with $m$ clauses and 3 literals per clause, e.g.,

$$(x_1 \lor \overline{x}_2 \lor x_3) \land (\overline{x}_1 \lor x_2 \lor \overline{x}_3)$$

where $\overline{x}_i$ is “not $x_i$”, $\land$ is “and”, $\lor$ is “or.” We call $x_i$ and $\overline{x}_i$ *literals*.

- **Question:** Is there a setting of each $x_i$ to TRUE or FALSE such that the formula is satisfied.
A Polynomial Time Reduction for 3-SAT to Clique

We’ll show that if you have a polynomial time algorithm for Clique, then you also have a polynomial time algorithm for 3-SAT.

Given formula 3-SAT

\[ \phi = (l_{1,1} \lor l_{1,2} \lor l_{1,3}) \land (l_{2,1} \lor l_{2,2} \lor l_{2,3}) \land \ldots \land (l_{m,1} \lor l_{m,2} \lor l_{m,3}) \]

in poly-time, we can construct \( G_\phi = (V_\phi, E_\phi) \):

\[ V_\phi = \{l_{i,j} : i \in [m], j \in [3]\} \]
\[ E_\phi = \{(l_{i,j}, l_{k,l}) : i, k \in [m], j \in [3], i \neq k, l_{i,j} \neq \overline{l}_{k,l}\} \]

We’ll show \( \phi \) is satisfiable iff \( G_\phi \) has a clique of size \( m \)
φ is satisfiable iff \( G_\phi \) has a clique of size \( m \)

Suppose \( \phi \) is satisfiable:

1. In a satisfying assignment, at least one literal is true in each clause
2. Pick one true literal per clause: let \( Y \) be set of corresponding nodes
3. \( G_\phi[Y] \) is a clique because \( x_k \) and \( \bar{x}_k \) can’t both be in \( Y \) for any \( k \)

Suppose \( G_\phi \) has a clique of size \( m \):

1. Let \( Y \) be the clique of size \( m \)
2. For each clause:
   - Exactly one node \( l \) from \( i \)-th clause is in \( Y \)
   - Set \( x_k = \text{TRUE} \) if \( l = x_k \) and set \( x_k = \text{FALSE} \) if \( l = \bar{x}_k \)
3. We can’t set \( x_k \) to be true and false because literals \( x_k \) and \( \bar{x}_k \) can’t both be in \( Y \)