CSE515 Advanced Algorithms
Lecture 29: Set Cover and Deterministic Rounding

Antoine Vigneron

antoine@unist.ac.kr

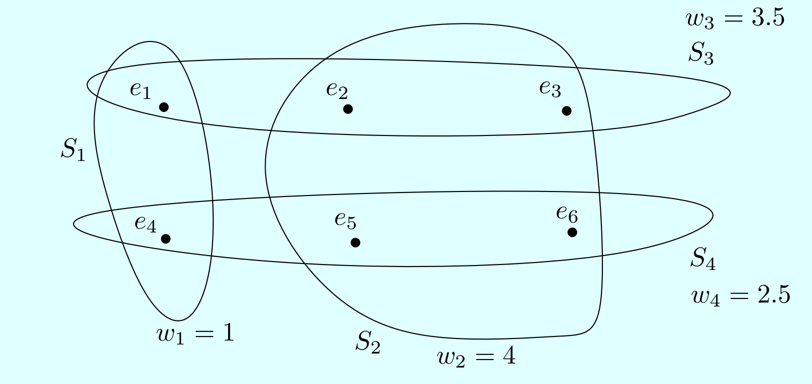
Ulsan National Institute of Science and Technology

July 27, 2021

Introduction

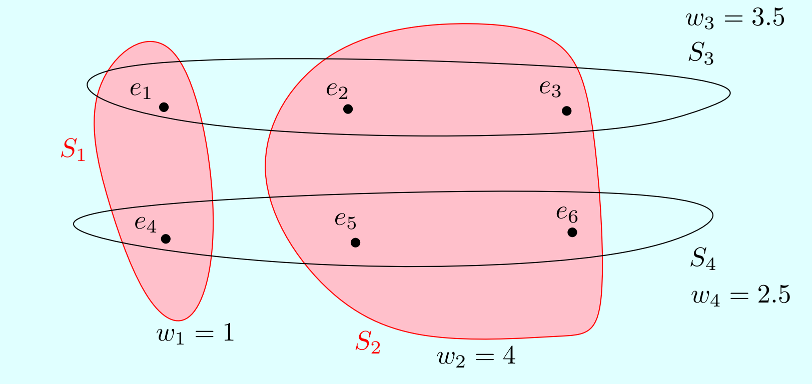
- Today's lecture: Chapter 1 in The Design of Approximation Algorithms by Shmoys and Williamson. (Available online.)
- See also Lecture 17 where we presented a greedy approximation algorithm for set cover.

Set Cover: Problem Statement



- A ground set $E = \{e_1, \ldots, e_n\}$.
- Some subsets $S_1, \ldots, S_m$ of E.
- Each subset S_j has a weight $w_j \geqslant 0$.
- Goal: find a minimum-weight collection of subsets covering E.

Set Cover: Problem Statement



- A ground set $E = \{e_1, \dots, e_n\}$.
- Some subsets $S_1, \dots, S_m$ of E.
- Each subset S_j has a weight $w_j \geqslant 0$.
- Goal: find a minimum-weight collection of subsets covering E.

Set Cover: Problem Statement

Input

- A ground set $E = \{e_1, \ldots, e_n\}$.
- Some subsets $S_1, \ldots, S_m$ of E.
- Each subset S_j has a weight $w_j \geqslant 0$.

Output

A subset $I \subseteq \{1, \ldots, m\}$ such that:

- For all i, we have $e_i \in S_j$ for some $j \in I$.
- $\sum_{j \in I} w_j$ is minimized.

Approximation Algorithms

- Set cover is **NP**-hard.
- So we cannot solve it in polynomial time, unless **P**=**NP**.
- Let $OPT = \sum_{j \in I} w_j$ be the optimal value, i.e. the weight of an optimal solution.
- Our goal is to find an *α-approximation algorithm* for some $\alpha > 1$:
  - Return in polynomial time a solution whose value is $\leqslant \alpha OPT$.

Integer Programming Formulation

- Variables $x_1, \ldots, x_m$ are defined as follows:
  - $x_j = 0$ if $j \notin I$,
  - $x_j = 1$ if $j \in I$.
- Our problem can be reformulated as:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{m} x_j w_j \\
\text{subject to} \quad & \sum_{j : e_i \in S_j} x_j \geqslant 1, \qquad i = 1, \ldots, n \\
& x_j \in \{0, 1\}, \qquad j = 1, \ldots, m.
\end{aligned}
$$

- This is an *Integer Program*.
- Difficulty: Integer programming is **NP**-hard.

Linear Programming Relaxation

- We now allow x_j to take any rational, non-negative value.
- We obtain a linear program, called *linear programming relaxation*:

$$\begin{aligned}
\text{minimize} \quad & \sum_{j=1}^{m} x_j w_j \\
\text{subject to} \quad & \sum_{j:e_i \in S_j} x_j \geqslant 1, && i = 1, \ldots, n \\
& x_j \geqslant 0, && j = 1, \ldots, m
\end{aligned}$$

- Abbreviation: *LP-relaxation*.
- Linear programming is in **P**.
- But the LP-relaxation is not equivalent to the original problem.

Comparison

- OPT: optimal value $\sum_{j \in I} w_j$ of set cover.
- Z^*_{IP}: optimal value $\sum_{1 \leqslant j \leqslant m} x_j w_j$.
- Z^*_{LP}: optimal value of the LP-relaxation.
- We have

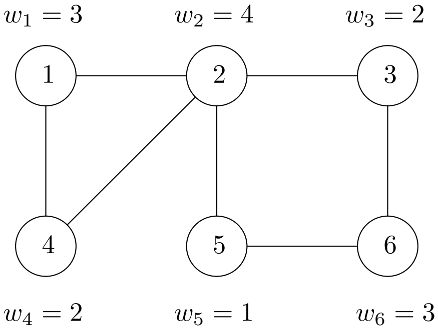
$$Z^*_{LP} \leqslant Z^*_{IP} = OPT.$$

- Strategy:
  - Find an optimal solution to the LP-relaxation.
  - Find an approximation of this solution with value at most αZ^*_{LP} and $\forall j, x_j \in \{0, 1\}$.
  - It gives an α-approximation of OPT

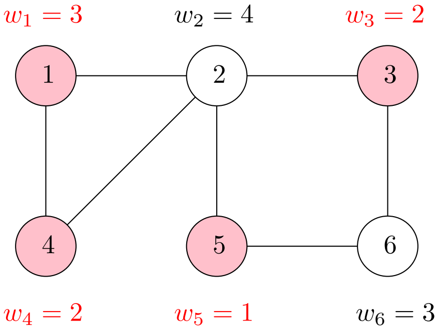
A Deterministic Rounding Algorithm

- Let f_i denote the number of sets S_j that contain e_i.
- Let $f = \max_i(f_i)$, i.e. f is the maximum number of sets in which any element appears.
- Let $x^* = (x_1^*, \ldots, x_m^*)$ be an optimal solution to LP-relaxation.
- We round x^* to $\hat{x}$ as follows:
  - $\hat{x}_j = 0$ iff $x_j^* < 1/f$.
  - $\hat{x}_j = 1$ iff $x_j^* \geqslant 1/f$.
- We denote by I the subset of indexes corresponding to $\hat{x}$.
- We will prove that it is an f-approximation algorithm. We need to prove:
  - This algorithm runs in polynomial time.
  - We obtain a set cover.
  - Its value is at most $f.OPT$.
- (Done in class, see textbook p. 19)

Vertex Cover



Vertex Cover



Vertex Cover

Problem (Vertex cover)

Given a graph $G = (V, E)$ with a weight $w_i \geqslant 0$ for each vertex $i \in V$, find a minimum-weight subset of vertices $C \subseteq V$ such that for each edge $(i, j) \in E$, either $i \in C$ or $j \in C$.

- This problem is **NP**-hard (even the unweighted version).
- No 1.36-approximation algorithm exists unless **P**=**NP**.
- Connection with set cover?
- What is the approximation factor of our algorithm if we apply it to vertex cover?

Conclusion

- We obtained an *approximation algorithm* for a combinatorial optimization problem by rounding a solution of its LP-relaxation.
- This is a standard approach for designing approximation algorithms.
- This approach can also be used as a *heuristic*, i.e. a practical algorithm that gives a reasonably good solution, without a proof that the approximation factor is good in the worst case.
- The textbook also presents a randomized rounding approach: The value x_j^* is interpreted as a probability to include S_j in the solution.