Analysis of Algorithms, I
CSOR W4231.002

Eleni Drinea
Computer Science Department

Columbia University

Thursday, April 26, 2016

Outline

1. Recap: Integer Programming
2. The LP relaxation for `Set Cover`
   - Rounding the LP solution
   - An f-approximation algorithm for `Set Cover`
3. Approximation algorithms for `Set Cover` and `Vertex Cover`
   - An f-approximation algorithm from the dual solution
   - A greedy $\log n$-approximation algorithm

Today

1. Recap: Integer Programming

2. The LP relaxation for `Set Cover`
   - Rounding the LP solution
   - An f-approximation algorithm for `Set Cover`

3. Approximation algorithms for `Set Cover` and `Vertex Cover`
   - An f-approximation algorithm from the dual solution
   - A greedy $\log n$-approximation algorithm

Integer Programming

`Integer programming (IP(D))`: Given a system of linear inequalities in n variables and m constraints with integer coefficients, and an integer target value k, does it have an integer solution of value k?

- Applications: production planning, scheduling, etc.
- The feasible region of `IP` is no longer a convex set: feasible solutions are points in the n-dimensional space

Claim 1.

$\texttt{VC(D)} \leq_P \texttt{IP(D)}$

Reduction for instance $(G = (V, E), k)$ of `VC(D)`:

$$\sum_{i=1}^{n} x_i \leq k$$

$$\begin{aligned} \text{subject to} \quad & x_i + x_j \geq 1, \quad \text{for every } (i, j) \in E \\ & x_i \in \{0, 1\}, \quad \text{for every } i \in V \end{aligned}$$

Minimum-weight Set Cover

Input

- a set $E = \{e_1, e_2, \ldots, e_n\}$ of n elements
- a collection of subsets of these elements $S_1, S_2, \ldots, S_m$, where each $S_j \subseteq E$
- a non-negative weight w_j for every subset S_j

Output

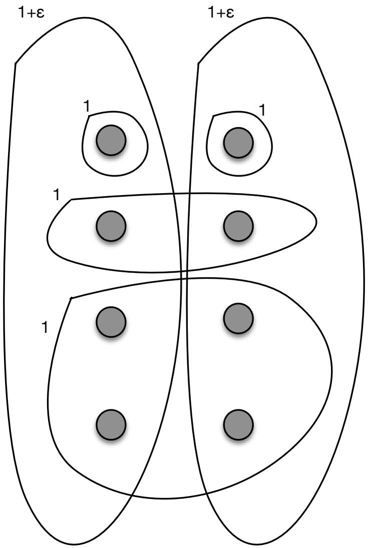
A minimum-weight collection of subsets that cover all of E.

In symbols: find an $I \subseteq \{1, \ldots, m\}$ such that

1. $\cup_{j \in I} S_j = E$
2. $\sum_{j \in I} w_j$ is minimum.

Fact: `Set Cover(D)` is $\mathcal{NP}$-complete.
(Proof: `VC(D)` $\leq_P$ `Set Cover(D)`.)

Example instance of Set Cover



constant 0< ε < 1

n=8 ground elements

m=6 subsets

4 subsets have weight 1

2 subsets have weight 1+ε

Designing the integer program for `Set Cover`

Variables: we introduce one variable per set S_j; intuitively,

$$x_j = \begin{cases} 1, & \text{if } S_j \text{ is included in the solution} \\ 0, & \text{otherwise} \end{cases}$$

Constraints: ensure that every element is *covered*:

for every element e_i, at least one of the sets S_j containing e_i appears in the final solution

Objective function: minimize the sum of the weights of the sets included in the solution

An integer programming formulation of `Set Cover`

Integer program for `Set Cover`:

$$\begin{aligned} \min \quad & \sum_{j=1}^{m} w_j x_j \\ \text{subject to} \quad & \sum_{j: e_i \in S_j} x_j \geq 1, \quad \text{for every } 1 \leq i \leq n \\ & x_j \in \{0, 1\}, \quad \text{for every } 1 \leq j \leq m \end{aligned} \tag{1}$$

Let Z^*_{IP} be the optimum value of this integer program;
OPT be the value of the optimum solution to `Set Cover`.

$$Z^*_{IP} = OPT.$$

△ We cannot solve this integer program efficiently *(why?)*.

Today

1 Recap: Integer Programming

2 The LP relaxation for `Set Cover`
- Rounding the LP solution
- An f-approximation algorithm for `Set Cover`

3 Approximation algorithms for `Set Cover` and `Vertex Cover`
- An f-approximation algorithm from the dual solution
- A greedy $\log n$-approximation algorithm

LP relaxation: a bound for the value of the IP

LP relaxation for `Set Cover`:

$$\begin{aligned} \min_{\mathbf{x}\geq \mathbf{0}} \quad & \sum_{j=1}^{m} w_j x_j \\ \text{subject to} \quad & \sum_{j: e_i \in S_j} x_j \geq 1, \quad \text{for every } 1 \leq i \leq n \end{aligned}$$

LP relaxation: a bound for the value of the IP

LP relaxation for `Set Cover`:

$$\begin{aligned} \min_{\mathbf{x} \geq \mathbf{0}} \quad & \sum_{j=1}^{m} w_j x_j \\ \text{subject to} \quad & \sum_{j : e_i \in S_j} x_j \geq 1, \quad \text{for every } 1 \leq i \leq n \end{aligned}$$

- Every feasible solution to the original IP is a feasible solution to the LP relaxation.
- The value of any feasible solution to the original IP is the same in the LP (the objectives are the same).
- Let Z^*_{LP} be the optimum value of the LP relaxation.

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

Rounding the solution to the LP

LP relaxation for `Set Cover`:

$$\begin{aligned} \min_{\mathbf{x}\geq\mathbf{0}} \quad & \sum_{j=1}^{m} w_j x_j \\ \text{subject to} \quad & \sum_{j:e_i\in S_j} x_j \geq 1, \quad \text{for every } 1\leq i\leq n \end{aligned}$$

- Let x^* be an optimal solution to the LP relaxation.
- Let $f_i = \#$ subsets S_j where element e_i appears.
- Let $f = \max_{1\leq i\leq n} f_i$.
- Set

$$\hat{x}_j = \begin{cases} 1, & \text{if } x_j^* \geq 1/f \\ 0, & \text{if } x_j^* < 1/f \end{cases}$$

Rounding yields a feasible solution to the original IP

The collection of sets S_j with $\hat{x}_j = 1$ is a set cover.

- Consider the optimal solution x^* for the LP relaxation.
- Fix any element e_i; recall that e_i appears in f_i subsets.
- For simplicity, relabel these subsets as $S_1, S_2, \ldots, S_{f_i}$. Then the optimal solution satisfies the constraint

  $$x_1^* + x_2^* + \ldots + x_{f_i}^* \geq 1$$

  Let x_m^* be the maximum of $x_1^*, x_2^*, \ldots, x_{f_i}^*$. Then

  $$x_m^* \geq \frac{1}{f_i} \geq \frac{1}{f}$$

$\Rightarrow$ Our rounding procedure guarantees that, for every element e_i, at least one set S_j that *covers* e_i is chosen.

An f-approximation algorithm for `Set Cover`

How far is the solution obtained by the rounding procedure above from to the optimal solution to `Set Cover`*?*

- We do **not** know OPT!
- **But** we have a bound for it: the value Z^*_{LP} of the LP relaxation!

Recall that we set $\hat{x}_j = 1$ if and only if $x^*_j \geq 1/f$. Then

$$\begin{aligned} \sum_j w_j \hat{x}_j &\leq \sum_j w_j (f x^*_j) = f \sum_j w_j x^*_j \\ &= f \cdot Z^*_{LP} \leq f \cdot OPT \end{aligned}$$

Today

1. Recap: Integer Programming
2. The LP relaxation for `Set Cover`
   - Rounding the LP solution
   - An f-approximation algorithm for `Set Cover`
3. Approximation algorithms for `Set Cover` and `Vertex Cover`
   - An f-approximation algorithm from the dual solution
   - A greedy $\log n$-approximation algorithm

What is an α-approximation algorithm

Definition 1.

An α-approximation algorithm for an optimization problem is a polynomial-time algorithm that, for all instances of the problem, produces a solution whose value is within a factor of α of the value of the optimal solution.

Remark 1.

- α *is the* approximation ratio *or* approximation factor
- *For* minimization *problems*, $\alpha > 1$.
- *For* maximization *problems*, $\alpha < 1$.

Examples

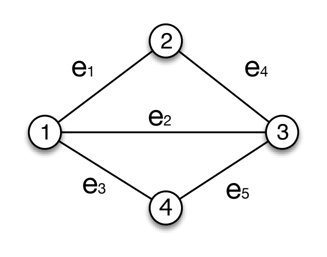
Example 1: the rounding procedure described on slide 11 gives an f-approximation algorithm for `Set Cover`:

- it can be completed in polynomial-time
- it always returns a solution whose value is at most f times the value of the optimal solution.

Remark: if an element appears in too many sets (e.g., $f = \Omega(n)$), this algorithm does not provide a good approximation guarantee.

Example 2: a 2-approximation algorithm for `VC` is a polynomial-time algorithm that always returns a solution whose value is at most twice the value of the optimal solution.

A 2-approximation algorithm for VC



- Let $E = \{e_1, \dots, e_m\}$ be the set of edges in the graph.
- Let S_j be the set of edges (ground elements) that are covered by vertex j.
- For every edge e_i, $f_i = 2$: e_i appears in exactly two subsets *(why?)*.
- Hence $f = \max_{1 \le i \le m} f_i = 2$.

An integer programming formulation of `Set Cover`

Recall the integer program for `Set Cover` (1):

$$\begin{aligned} \min \quad & \sum_{j=1}^{m} w_j x_j \\ \text{subject to} \quad & \sum_{j: e_i \in S_j} x_j \geq 1, \quad \text{for every } 1 \leq i \leq n \\ & x_j \in \{0, 1\} \end{aligned}$$

Remark 2.

1. *x_j indicates whether S_j is included the final solution*
2. *the constraints ensure that for every element e_i, at least one of the sets S_j containing e_i appears in the final solution*

The incidence vector of a set

Incidence (or characteristic) vector $\mathbf{a}^S$ of a set S:

$$a_i^S = \begin{cases} 1 & \text{, if element } e_i \in S \\ 0 & \text{, otherwise} \end{cases} \quad 1$$

Consider the $n \times m$ matrix A whose columns are the incidence vectors $\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^m$ of the sets $S_1, S_2, \ldots, S_m$ respectively.

$$A = \begin{bmatrix} \mathbf{a}^1 & \mathbf{a}^2 & \ldots & \mathbf{a}^m \end{bmatrix}$$

- The i-th row of A indicates the sets where e_i appears (e.g., the first row indicates the sets where element e_1 appears).

Example matrix of incidence vectors

Let $n = 5$, $m = 4$, $E = \{1, 2, 3, 4, 5\}$ and

- $S_1 = \{1, 2, 3\}, S_2 = \{1, 4\}, S_3 = \{2, 4, 5\}, S_4 = \{3, 5\}$.

The incidence vectors of the above sets are

- $\mathbf{a}^1 = [1\ 1\ 1\ 0\ 0]^T$
- $\mathbf{a}^2 = [1\ 0\ 0\ 1\ 0]^T$
- $\mathbf{a}^3 = [0\ 1\ 0\ 1\ 1]^T$
- $\mathbf{a}^4 = [0\ 0\ 1\ 0\ 1]^T$

The i-th row of A below indicates the sets where e_i appears.

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Incidence matrix of a graph

Recall that, in a vertex cover instance, edges correspond to ground elements and vertices to sets of incident edges.

Example: for the graph on slide 18, vertices $\{1, 2, 3, 4\}$ correspond to the sets S_1, S_2, S_3, S_4 on slide 21. Thus the matrix of constraints of the IP for Vertex Cover is matrix A of slide 21.

Incidence matrix of an **undirected** graph: the $n \times m$ matrix B whose rows are the incidence vectors of the vertices (so $B = A^T$); that is, entry $B_{ij} = 1$ if and only if edge j is incident to vertex i.

For **directed graphs**, we define

$$B_{ij} = \begin{cases} 1 & \text{, if edge } j \text{ leaves vertex } i \\ -1 & \text{, if edge } j \text{ enters vertex } i \\ 0 & \text{, otherwise} \end{cases}$$

Did we see an LP where B appeared in the constraint matrix?

A matrix-vector form for the IP for `Set Cover`

Hence we may re-write the IP (1) for `Set Cover` as follows:

$$\begin{aligned} \min \quad & \mathbf{w}^T\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \geq \mathbf{1} \\ & \mathbf{x} \in \{0,1\}^m \end{aligned}$$

where

- $\mathbf{x}$ is a the $m \times 1$ vector indicating which sets are included in the final solution;
- $\mathbf{w}$ is the $m \times 1$ vector of set weights;
- A is the $n \times m$ matrix whose columns are the incidence vectors of sets $S_1, S_2, \ldots, S_m$;
- $\mathbf{1}$ is the $n \times 1$ vector of all ones.

The LP relaxation for `Set Cover` and its dual

$$\begin{aligned} \min_{\mathbf{x} \geq \mathbf{0}} \quad & \mathbf{w}^T \mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} \geq \mathbf{1} \end{aligned}$$

It is now straightforward to take the dual of the LP relaxation:

Goal: lower bound the primal objective; to this end

1. left-multiply each constraint by a non-negative multiplier y_i;
2. add the resulting inequalities;
3. require the right-hand side to be a lower bound for $\mathbf{w}^T\mathbf{x}$.

In symbols,

1. $A\mathbf{x} \geq \mathbf{1} \Rightarrow \mathbf{y}^T A\mathbf{x} \geq \mathbf{y}^T\mathbf{1}$, since $\mathbf{y} \geq \mathbf{0}$.
2. Constrain $\mathbf{y}$ to satisfy $\mathbf{w}^T \geq \mathbf{y}^T A$.
   Since $\mathbf{x} \geq \mathbf{0}$, we have $\mathbf{w}^T\mathbf{x} \geq \mathbf{y}^T A\mathbf{x} \geq \mathbf{y}^T\mathbf{1}$ (from 1.).

The dual of the LP relaxation

$$\begin{aligned} \max_{\mathbf{y} \geq \mathbf{0}} \quad & \mathbf{1}^T \mathbf{y} \\ \text{subject to} \quad & A^T \mathbf{y} \leq \mathbf{w} \end{aligned}$$

Note that A^T is the $m \times n$ matrix whose rows are the incidence vectors $\mathbf{a}^1, \ldots, \mathbf{a}^m$ of the sets $S_1, \ldots, S_m$. We'll now rewrite the dual LP in a more intuitive way.

$$\begin{aligned} \max_{\mathbf{y} \geq \mathbf{0}} \quad & \sum_{i=1}^{n} y_i \\ \text{subject to} \quad & \sum_{i: e_i \in S_j}^{n} y_i \leq w_j, \quad \text{for every } 1 \leq j \leq m \end{aligned}$$

The dual has one variable per element e_i and one constraint per set S_j. *How can we interpret the dual variables/constraints?*

Interpreting the dual LP

- *Dual variables:* each element e_i is *charged* a price $y_i \geq 0$
- *Intuition:* low prices will be assigned to elements covered with low-weight subsets, high prices to those covered by high-weight subsets
- *Dual constraints:* the sum of the prices paid to cover all the elements in any subset S_j cannot exceed the weight w_j of that subset *(why?)*. Thus, for every subset S_j, we have

$$\sum_{i:e_i \in S_j}^{n} y_i \leq w_j$$

- *Strong duality:* if $\mathbf{y}^*$ is the optimal dual solution, then

$$\sum_{i=1}^{n} y_i^* = \sum_{j=1}^{m} w_j x_j^*$$

Rounding the dual solution to obtain a set cover

If the dual constraint for subset S_j meets with equality, that is

$$\sum_{i:e_i \in S_j}^{n} y_i^* = w_j,$$

then add subset S_j to the solution, that is, add j to I'.

Claim 2.

The collection of subsets S_j, $j \in I'$, is a set cover.

To prove the claim, we will show that, if some element e_ℓ is not covered by the sets in I', then $\mathbf{y}^*$ is not optimal.

Proof of Claim 2

- Suppose that e_ℓ is not covered by the sets in S^*.
- Then, **for every** set S_j containing e_ℓ, the constraint for S_j is **not** met with equality, hence

$$\sum_{i:e_i \in S_j}^{n} y_i^* < w_j \Rightarrow \overbrace{w_j - \sum_{i:e_i \in S_j}^{n} y_i^*}^{\delta_j} > 0$$

- Let $\delta = \min_{\text{j: } e_\ell \text{ appears in } j\text{-th constraint}} \delta_j$ and consider a new dual solution $\mathbf{y}'$, which differs from $\mathbf{y}^*$ only in the ℓ-th component:

$$y'_i = \begin{cases} y^*_\ell + \delta, & \text{if } i = \ell \\ y^*_i, & \text{if } i \neq \ell \end{cases}$$

- Then $\mathbf{y}'$ is feasible *(why?)* and $\sum_{i=1}^{n} y'_i > \sum_{i=1}^{n} y^*_i$, contradicting optimality of $\mathbf{y}^*$.

Dual rounding yields an f-approximation algorithm

Intuition: for every set in I', we *pay* y_i^* for each element it contains. Since every element appears at most f times, we pay at most $f \cdot \sum_{i=1}^{n} y_i^*$ in total.

Formally,

$$\begin{aligned}
\sum_{j \in I'} w_j &= \sum_{j \in I'} \sum_{i: e_i \in S_j} y_i^* \\
&= \sum_{i=1}^{n} y_i^* \cdot (\#\text{subsets } S_j \text{ in } I' \text{ that contain } e_i) \\
&\leq \sum_{i=1}^{n} f_i y_i^* \leq f \sum_{i=1}^{n} y_i^* \\
&\leq f \cdot OPT
\end{aligned}$$

The last inequality follows from weak duality.

A greedy algorithm for `Set Cover`

Intuition: let R be the set of yet uncovered elements; at every step, add to the final solution the subset S_k that minimizes the *covering cost per new element*: if we include subset S_j, then we pay a price w_j to cover $|S_j \cap R|$ elements.

$I = \emptyset$
$R = E$
while $R \neq \emptyset$ **do**
  Select subset S_k that minimizes $\frac{w_k}{|S_k \cap R|}$
  Add k to I
  Delete S_k from R
end while

Running time?

Greedy offers a $\log n$-approximation guarantee

Recall that $H_n \approx \ln n$ is the n-th harmonic number.

Theorem 2.

Let g be the size of the largest subset S_j. Then the greedy algorithm is an H_g-approximation algorithm for `Set Cover`*.*

(Proof: e.g., see your textbook, pp. 1117-1122)

Theorem 3.

There exists some constant $c > 0$ such that, if there exists a $c \ln n$-approximation algorithm for the unweighted set cover problem, then $\mathcal{P} = \mathcal{NP}$.

Theorem 4.

If there exists a α-approximation algorithm for the vertex cover problem with $\alpha < 10\sqrt{5} - 21 \approx 1.36$, then $\mathcal{P} = \mathcal{NP}$.

Optimization problems with PTAS

Definition 5.

A polynomial-time approximation scheme (PTAS) is a family of algorithms $\mathcal{A}_\epsilon$, where there is an algorithm for **every** constant $\epsilon > 0$, such that $\mathcal{A}_\epsilon$ is an $(1+\epsilon)$-approximation algorithm for minimization problems ($(1-\epsilon)$-approximation algorithm for maximization problems).

Example: `Euclidean TSP` has a PTAS

Harder optimization problems (the class MAX-SNP)

Theorem 6.

Problems in MAX-SNP do not have polynomial-time approximation schemes, unless $\mathcal{P} = \mathcal{NP}$.

Examples: `MAX-SAT`, `MAX-CUT`, `VC` are in MAX-SNP

Even *harder* problems!

Let $G = (V, E)$ be an undirected graph, $|V| = n$.

Definition 7.

A clique of size k is a subset of k vertices such that all $\binom{k}{2}$ possible edges appear in E.

`MAX CLIQUE`:
Input: an undirected graph $G = (V, E)$
Output: a clique of maximum size

Theorem 8.

Let $\epsilon > 0$ be any positive constant. There exists no $O(n^{\epsilon-1})$-approximation algorithm for `MAX CLIQUE` *unless $\mathcal{P} = \mathcal{NP}$.*