The Design of Approximation Algorithms

RtB*

Last edited in: March 13, 2025

*School of Artificial Intelligence, Southeast University

Contents

1 The Set Cover Problem **1**
1.1 Problem Description 1
1.2 Linear Program 1
1.3 A Deterministic Algorithm 1
1.4 A Dual Solution 2
1.4.1 Rounding the Dual Solution 3
1.5 A Greedy Algorithm 4
1.6 A Randomized Rounding Algorithm 6

1 The Set Cover Problem

1.1 Problem Description

Given a ground set of elements $E = \{e_1, \dots, e_n\}$, some subsets of those elements $S_1, \dots, S_m$ where each $S_j \subseteq E$, and a nonnegative weight $w_j \geq 0$ for each subset S_j. The goal is to find a minimum-weight collection of subsets $I \subseteq \{1, \dots, m\}$ that minimizes $\sum_{j \in I} w_j$ subject to $\bigcup_{j \in I} S_j = E$.

The problem can be description as a **integer program**:

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

1.2 Linear Program

By extending the domain of x_j to the field of real numbers, the problem can be transformed into a **linear programming problem**.

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

1.3 A Deterministic Algorithm

Given the LP solution x^*, let $f_i = |\{j : e_i \in S_j\}|$, $i = 1, \dots, n$ and $f = \max_{i, \dots, n} f_i$, the subset S_j with $x_j^* \geq 1/f$ will be take into the solution. Let I indexes the set cover.

Lemma 1. *The collection of subsets S_j, $j \in I$ is a set cover.*

Proof. Consider the element e_i, there are only less than f subsets that contain e_i, and we have $\sum_{j : e_i \in S_j} x_j^* \geq 1$. Thus, there must be at least one subset S_j with $x_j^* \geq 1/f$ that contains e_i. Therefore, the collection of subsets S_j, $j \in I$ is a set cover. □

Theorem 1. *The rounding algorithm is an f-approximation algorithm for the set cover problem.*

Proof. Let Z^*_{LP} be the value of the optimal linear program solution, and OPT be the value of the optimal integer program solution. We have

$$\begin{aligned}
\sum_{j\in I} w_j &\leq \sum_{j=1}^{m} w_j \cdot (f \cdot x^*_j) \\
&= f \cdot \sum_{j=1}^{m} w_j x^*_j \\
&= f \cdot Z^*_{LP} \\
&\leq f \cdot \text{OPT}
\end{aligned}$$

□

The key of the proof is to find the connection between the rounding solution and the LP solution. A trick used in this solution is to introducing variables through constants by the rounding algorithm condition $1 \leq f \cdot x^*_j$ for each $j \in I$. And because the feasible solution of integer program is a subset of the feasible solution of the linear program, we have $Z^*_{LP} \leq \text{OPT}$.

1.4 A Dual Solution

The dual program of the set cover linear programming relaxation is

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n} y_i \\
\text{subject to} \quad & \sum_{i:e_i\in S_j} y_i \leq w_j, \quad j = 1, \ldots, m \\
& y_i \geq 0, \qquad i = 1, \ldots, n
\end{aligned}$$

The **dual problem** can be derived through the following steps. First, we can write the Lagrangian function of the linear program:

$$\begin{aligned}
L(x, y, \lambda) &= \sum_{j=1}^{m} w_j x_j + \sum_{i=1}^{n} y_i \left(1 - \sum_{j:e_i\in S_j} x_j\right) - \sum_{j=1}^{m} \lambda_j x_j \\
&= \sum_{j=1}^{m} \left(w_j - \sum_{i:e_i\in S_j} y_i - \lambda_j\right) x_j + \sum_{i=1}^{n} y_i
\end{aligned}$$

And the Lagrange dual function is:

$$g(y,\lambda) = \inf_{x_1,\ldots,x_m} L(x,y,\lambda)$$

Notice that if $w_j - \sum_{i:e_i\in S_j} y_i - \lambda_j \geq 0$, then $\left(w_j - \sum_{i:e_i\in S_j} y_i - \lambda_j\right) x_j$ must be 0, else $g(y,\lambda) = -\infty$. In order to prevent the function from diverging, we need to add the constraint $w_j - \sum_{i:e_i\in S_j} y_i - \lambda_j \geq 0$. Therefore, the dual optimization problem can be formulated as

$$\begin{aligned}
\text{maximize} \quad & \sum_{i=1}^{n} y_i \\
\text{subject to} \quad & \sum_{i:e_i\in S_j} y_i + \lambda_j \leq w_j, \quad j = 1,\ldots,m \\
& y_i \geq 0, \qquad i = 1,\ldots,n \\
& \lambda_j \geq 0, \qquad j = 1,\ldots,m
\end{aligned}$$

λ_j can be directly eliminated through simplification.

1.4.1 Rounding the Dual Solution

An simple idea is to choose the subset that satisfy the condition $\sum_{i:e_i\in S_j} y_i = w_j$. Let I' denote the indices of the subsets in the solution, we will show that this is a set cover.

Lemma 2. *The collection of subsets S_j, $j \in I'$ is a set cover.*

Proof. Suppose there exist e_i that is not covered, then for any subset S_j that contains e_i, we must have $\sum_{i:e_i\in S_j} y_i < w_j$. We can increase the value of y_i to get the larger $\sum_{i=1}^{n} y_i$ until at least subset that contain e_i is chosen. □

Theorem 2. *The dual rounding algorithm described above is an f-approximation algorithm for the set cover problem.*

Proof.

$$\begin{aligned}
\sum_{j\in I'} w_j &= \sum_{j\in I'} \sum_{i:e_i\in S_j} y_i^* \\
&= \sum_{i=1}^{n} |\{j \in I' : e_i \in S_j\}| \cdot y_i^* \\
&\leq f \cdot \sum_{i=1}^{n} y_i^* \\
&\leq f \cdot \text{OPT}
\end{aligned}$$

□

The key of the proof lies in swapping the order of the summation to articulate the optimization objective.

1.5 A Greedy Algorithm

Let n_k denote the number of elements that remain uncoverd at the start of the kth iteration, let I_k denote the indices of the sets chosen in iterations 1 through $k-1$, and for each $j = 1, \ldots, m$, let $\hat{S}_j$ denote the set of uncovered elements in S_j at the start of this iteration. Then we have a greedy algorithm. Let $H_k = 1 + \frac{1}{2} + \cdots + \frac{1}{k}$, we have the following theorem.

```
I ← ∅;
Ŝ_j ← S_j   ∀j;
while I is not a set cover do
    I ← arg min_{j:Ŝ_j≠∅} w_j/|Ŝ_j|;
    I ← I ∪ l;
    Ŝ_j ← Ŝ_j − S_l   ∀j;
end
```

Algorithm 1: A greedy algorithm for the set cover problem.

Theorem 3. *The greedy algorithm is an H_n-approximation algorithm for the set cover problem.*

Proof. First, we will show that

$$w_j \leq \frac{n_k - n_{k+1}}{n_k} \cdot \mathrm{OPT}$$

The inequation can be derived from the fact:

$$\min_{i=1,\ldots,k} \frac{a_i}{b_i} \leq \frac{\sum_{i=1}^{k} a_i}{\sum_{i=1}^{k} b_i} \leq \max_{i=1,\ldots,k} \frac{a_i}{b_i}$$

Using the fact, let O contains the indices of the sets in an optimal solution, we can get:

$$\min_{j:\hat{S}_j \neq \emptyset} \frac{w_j}{|\hat{S}_j|} \leq \min_{j \in O} \frac{w_j}{|\hat{S}_j|} \leq \frac{\sum_{j \in O} w_j}{\sum_{j \in O} |\hat{S}_j|} = \frac{\mathrm{OPT}}{n_k}$$

where the first inequality follows the definition of $\hat{S}_j$, if $j \in O$ but $\hat{S}_j = 0$, we must have $\frac{w_j}{|\hat{S}_j|} = \infty$. else it can be considered by $\min_{j:\hat{S}_j \neq \emptyset}$.

Then due to the algorithm choose the j which minimize the ratio $\frac{w_j}{|\hat{S}_j|}$, we have

$$w_j \leq \frac{|\hat{S}_j| \cdot \mathrm{OPT}}{n_k} = \frac{n_k - n_{k+1}}{n_k} \cdot \mathrm{OPT}$$

Notice that the elements in $\hat{S}_j$ are covered in the $k+1$ iteration, so $|\hat{S}_j| = n_k - n_{k+1}$.

Let I contain the indices of the sets in the algorithm solution, we have

$$\begin{aligned}
\sum_{j \in I} w_j &= \sum_{k=1}^{l} \frac{n_k - n_{k+1}}{n_k} \cdot \mathrm{OPT} \\
&\leq \mathrm{OPT} \cdot \sum_{k=1}^{l} \left(\frac{1}{n_k} + \frac{1}{n_k - 1} + \cdots + \frac{1}{n_{k+1} + 1} \right) \\
&= \mathrm{OPT} \cdot \sum_{i=1}^{n} \frac{1}{i} \\
&= H_n \cdot \mathrm{OPT}
\end{aligned}$$

□

Theorem 4. *The solution of the greedy algorithm satisfiy* $\sum_{j \in I} w_j \leq H_g \cdot Z^*_{LP}$, *where* g *is the maximum size of any subset* S_j.

Proof. To prove the theorem, we can consider the conclution in dual program: for the feasible solution of the dual program, we have $\sum_{i=1}^{n} y'_i \leq Z^*_{LP}$. We want to construct the solution so that we have $\sum_{j \in I} w_j = H_g \sum_{i=1}^{n} y'_i \leq H_g \cdot Z^*_{LP}$ First, we need to construct the solution of the dual program. suppose we choose to add subset S_j to our solution in iteration k. Then for each $e_i \in \hat{S}_j$, we set $y_i = \frac{w_j}{|\hat{S}_j|}$. Since each e_i is chosen only once, so we have $\sum_{j \in I} w_j = \sum_{i=1}^{n} = y_i$. That is not a feasible solution for the dual program because we only consider the subsets in I. Then we need to show that $y' = \frac{1}{H_g} y$ is feasible.

For an arbitrary subset S_j, let a_k denote the number of elements in this subset that are still uncovered at the beginning of the kth iteration. Let A_k be the uncoverd elements of S_j, so $|A_k| = a_k - a_{k+1}$. If S_j is chosen in the kth iteration, then for each element $e_i \in A_k$ covered in the kth iteration,

$$y'_i = \frac{w_p}{H_g |\hat{S}_p|} \leq \frac{w_j}{H_g a_k}$$

Thus, we have

$$
\begin{aligned}
\sum_{i:e_i\in S_j} y'_i &= \sum_{k=1}^{l} \sum_{i:e_i\in A_k} y'_i \\
&\le \sum_{k=1}^{l} (a_k - a_{k+1}) \frac{w_j}{H_g a_k} \\
&\le \frac{w_j}{H_g} \sum_{i=1}^{|S_j|} \frac{1}{i} \\
&= \frac{w_j}{H_g} H_{|S_j|} \\
&\le w_j
\end{aligned}
$$

Therefore, the solution y' is feasible for the dual program. $\square$

If we calculate $\sum_{i:e_i\in S_j} y_i$ straightforwardly, we can get $\sum_{i:e_i\in S_j} y_i \le w_j \cdot H_{|S_j|}$, then we may consider to find a factor bigger than $H_{|S_j|}$ to make it feasible.

1.6 A Randomized Rounding Algorithm

Let X^* be an optimal LP solution to the LP relaxation. The idea of the randomized algorithm is is that we interpret the fractional value x_j^* as the probability that $\hat{x}_j$ should be set to 1.

Let X_j be a random variable that is 1 if subset S_j is included in the solution, and 0 otherwise. Then the expected value of the solution is

$$
\mathbb{E}\left[\sum_{j=1}^{m} w_j X_j\right] = \sum_{j=1}^{m} w_j \Pr\left[X_j = 1\right] = \sum_{j=1}^{m} w_j x_j^* = Z_{LP}^*
$$

But the problem lies in the fact that the solution may be not a set cover. And we can get the probability of this situation.

$$
\begin{aligned}
\Pr\left[e_i \text{ is not covered}\right] &= \prod_{j:e_i\in S_j} (1 - x_j^*) \\
&\le \prod_{j:e_i\in S_j} \mathrm{e}^{-x_j^*} \\
&= \mathrm{e}^{-\sum_{j:e_i\in S_j} x_j^*} \\
&\le \mathrm{e}^{-1}
\end{aligned}
$$

The first inequality is due to $\mathrm{e}^{-x} \geq 1 - x$, the second inequality is due to $\sum_{j:e_i \in S_j} x_j^* \geq 1$ in the LP program.

Consider to impose a guarantee in keeping with our focus on polynomial-time algorithms, for any constant c, we could devise a polynomial-time algorithm whose chance of failure is at most an inverse polynomial n^{-c} , then we say that we have an algorithm that works **with high probability**

For example, we consider the following algorithm. For each Subset S_j, we choose $c \ln n$ times with equal probability x_j^*, if it is chosen at least once, we include S_j in the solution. Then we have

$$\begin{aligned} \Pr[e_i \text{ is not covered}] &= \sum_{j:e_i \in S_j} (1 - x_j^*)^{c \ln n} \\ &\leq \mathrm{e}^{-c \ln n \sum_{j:e_i \in S_j} x_j^*} \\ &\leq \frac{1}{n^c} \end{aligned}$$

Theorem 5. *The algorithm is a randomized $O(\ln n)$-approximation algorithm that produces a set cover with high probability*

Proof. Let $p_j(x_j^*)$ be the probability that S_j is included in the solution, we know that $p_j(x_j^*) = 1 - (1 - x_j^*)^{(c \ln n)}$. If $x_j^* \in [0, 1]$ and $c \ln n \geq 1$, we have

$$p_j'(x_j^*) = c \ln n (1 - x_j^*)^{c \ln n} \leq c \ln n$$

where p_j' is the derivative of p_j.

Then we have

$$\begin{aligned} \mathbb{E}\left[\sum_{j=1}^{m} w_j X_j\right] &= \sum_{j=1}^{m} w_j \Pr[X_j = 1] \\ &\leq c \ln n \sum_{j=1}^{m} w_j x_j^* \\ &= (c \ln n) Z_{LP}^* \end{aligned}$$

□

However, we want to consider the situation that our solution can give a set cover. Let F be the event that the solution obtained by the procedure is a feasible set cover, let $\bar{F}$ be the complement of this event, we have

$$\begin{aligned}\Pr[\bar{F}] &= \Pr[\text{there exists an element uncoverd}] \\ &\leq \sum_{i=1}^{n} \Pr\left[e_i \text{ is not covered}\right] \\ &= \frac{1}{n^{c-1}}\end{aligned}$$

The inequality can be proved by inclusion-exclusion principle.
So $\Pr[F] \geq 1 - \frac{1}{n^{c-1}}$, and we have

$$\mathbb{E}\left[\sum_{j=1}^{m} w_j X_j\right] = \mathbb{E}\left[\sum_{j=1}^{m} w_j X_j \middle| F\right] \Pr[F] + \mathbb{E}\left[\sum_{j=1}^{m} w_j X_j \middle| \bar{F}\right] \Pr[\bar{F}]$$

Since $w_j \geq 0$ for all j,

$$\mathbb{E}\left[\sum_{j=1}^{m} w_j X_j \middle| \bar{F}\right] \geq 0$$

Thus

$$\begin{aligned}\mathbb{E}\left[\sum_{j=1}^{m} w_j X_j \middle| F\right] &= \frac{1}{\Pr[F]}\left(\mathbb{E}\left[\sum_{j=1}^{m} w_j X_j\right] - \mathbb{E}\left[\sum_{j=1}^{m} w_j X_j \middle| \bar{F}\right]\right) \\ &\leq \frac{(c \ln n) Z_{LP}^*}{1 - \frac{1}{n^{c-1}}}\end{aligned}$$

If $n \geq 2$ and $c \geq 2$, we have

$$\mathbb{E}\left[\sum_{j=1}^{m} w_j X_j \middle| F\right] \leq 2c(\ln n) Z_{LP}^*$$

References

[1] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.