Tutorial 12: Further Discussion on Set Cover and Hitting Set

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong
Set Cover

Let $U$ be a finite set called the universe.

We are given a family $S$ where

- each member of $S$ is a set $S \subseteq U$;
- $\bigcup_{S \in S} S = U$.

A sub-family $C \subseteq S$ is a universe cover if every element of $U$ appears in at least one set in $C$.

- Define the cost of $C$ as $|C|$.

The set cover problem:
Find a universe cover with the smallest cost.
Example: $U = \{1, 2, 3, 4, 5, 6, 7, 8\}$ and $S = \{S_1, S_2, \ldots, S_5\}$ where

$S_1 = \{1, 2, 3, 4\}$

$S_2 = \{2, 5, 7\}$

$S_3 = \{6, 7\}$

$S_4 = \{1, 8\}$

$S_5 = \{1, 2, 3, 8\}$.

An optimal solution is $C = \{S_1, S_2, S_3, S_4\}$. 

Our Approximation Algorithm

1. \( C = \emptyset \)
2. while \( U \) still has elements not covered by any set in \( C \)
3. \( F \leftarrow \) the set of elements in \( U \) not covered by any set in \( C \)
   /* for each set \( S \in S \), define its benefit to be \(|S \cap F|\) */
4. add to \( C \) a set in \( S \) with the largest benefit
5. return \( C \)

We proved in the lecture that the algorithm is \((1 + \ln |U|)\)-approximate.

Next, we will prove that the algorithm is also \( h \)-approximate, where \( h = \max_{S \in S} |S| \).
Example: \( S = \{ S_1, S_2, ..., S_5 \} \) where

\[
\begin{align*}
S_1 &= \{1, 2, 3, 4\} \\
S_2 &= \{2, 5, 7\} \\
S_3 &= \{6, 7\} \\
S_4 &= \{1, 8\} \\
S_5 &= \{1, 2, 3, 8\}.
\end{align*}
\]

Then, \( h = 4 \).
**Theorem:** The algorithm returns a universe cover with cost at most $h \cdot OPT_S$.

**Proof.** Suppose that our algorithm picks $t$ sets. Every time the algorithm picks a set, at least one new element is covered. For each $i \in [1, t]$, denote by $e_i$ an arbitrary element that is newly covered when the $i$-th set is picked.

Let $C^*$ be an optimal universe cover. Because each $e_i$ exists in at least one set of $C^*$, we have:

\[
    t = \sum_{i=1}^{t} 1 \leq \sum_{i=1}^{t} \# \text{ sets in } C^* \text{ containing } e_i \\
    \leq \sum_{e \in U} \# \text{ sets in } C^* \text{ containing } e \\
    = \sum_{S \in C^*} |S| \leq |C^*| \cdot h.
\]
**Corollary:** If \( h = O(1) \), then our algorithm achieves a constant approximation ratio.

**Remark:** With a more careful analysis, we can actually prove that our algorithm has an approximation ratio of \( 1 + \ln h \).

- Not required in this course.
Our set cover algorithm can be used to solve many problems with approximation guarantees. Next, we will see two examples.
Vertex Cover

\(G = (V, E)\) is an undirected graph. We want to find a small subset \(V' \subseteq V\) such that every edge of \(E\) is incident to at least one vertex in \(V'\). The optimization goal is to minimize \(|V'|\).

Convert the problem to set cover:

- For every \(v \in V\), define \(S_v\) = the set of edges incident on \(v\).
- Apply our algorithm on the set-cover instance: \(S = \{S_v \mid v \in V\}\).

This gives an \(O(\ln |V|)\)-approximate solution.

Remark: We have already learned how to ensure an approximation ratio of 2. But the point here is to demonstrate the usefulness of set cover, rather than improving the approximation ratio.
Red-Black Coverage

\( R = \) a set of \( n \) red points in 2D space
\( B = \) a set of \( n \) black points in 2D space
\( \epsilon = \) a positive integer.

A subset \( S \subseteq R \) is a \textbf{\( B \)-guarding set} if, for every black point \( b \in B \), there is at least one point \( r \in S \) with \( \text{dist}(r, b) \leq \epsilon \).

\( \text{OPT} = \) the smallest size of all \( B \)-guarding sets.
\textbf{Goal:} Return a \( B \)-guarding set with size \( \text{OPT} \cdot O(\log n) \) (assume that at least one \( B \)-guarding set exists).
Convert the problem to set cover:

- For every $r \in R$, define $S_r = \{ b \mid \text{dist}(r, b) \leq \epsilon \}$.
- Apply our algorithm on the set-cover instance: $S = \{ S_r \mid r \in R \}$.

This gives an $O(\log n)$-approximate solution.
Next, we will turn our attention to the hitting set problem, which is in fact equivalent to set cover.
Let $U$ be a finite set called the universe.

We are given a family $S$ where

- each member of $S$ is a set $S \subseteq U$;
- $\bigcup_{S \in S} S = U$.

A subset $H \subseteq U$ hits a set $S \in S$ if $H \cap S \neq \emptyset$.
A subset $H \subseteq U$ is a hitting set if it hits all the sets in $S$.

**The hitting set problem:**
Find a hitting set $H$ of the minimize size.
Example: \( U = \{1, 2, 3, 4, 5\} \) and \( S = \{S_1, S_2, \ldots, S_8\} \) where

\[
\begin{align*}
S_1 &= \{1, 4, 5\} \\
S_2 &= \{1, 2, 5\} \\
S_3 &= \{1, 5\} \\
S_4 &= \{1\} \\
S_5 &= \{2\} \\
S_6 &= \{3\} \\
S_7 &= \{2, 3\} \\
S_8 &= \{4, 5\}
\end{align*}
\]

An optimal solution is \( H = \{1, 2, 3, 4\} \).
Next, we will provide a matrix-view of set cover and hitting set, which hopefully will help you better understand their equivalence. We will achieve the purpose through a “bridging problem” defined on a matrix.
$M$ = an $n \times m$ matrix.

$M[i, j] = 0$ or $1$ for every $i \in [1, n]$ and $j \in [1, m]$.

**Constraint:** At least one $1$ at each row and at each column.

**Row Cover:** a set $R$ of rows s.t. every column has at least one $1$ at the rows of $R$.

$OPT_{row}$ : the minimum size of all row covers.

**Example**

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

An optimal row cover takes the first four rows.

Using our set-cover algorithm, we can find a row cover of size $OPT_{row} \cdot O(\log m)$.
Let us now relate the matrix problem to hitting set.

Consider the hitting set instance \( U = \{1, 2, 3, 4, 5\} \) and \( S = \{S_1, S_2, ..., S_8\} \) where \( S_1 = \{1, 4, 5\} \), \( S_2 = \{1, 2, 5\} \), \( S_3 = \{1, 5\} \), \( S_4 = \{1\} \), \( S_5 = \{2\} \), \( S_6 = \{3\} \), \( S_7 = \{2, 3\} \), and \( S_8 = \{4, 5\} \).

We can describe the instance with

\[
M = \begin{bmatrix}
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 
\end{bmatrix}
\]

where the \( i \)-th row corresponds to integer \( i \in U \) and the \( j \)-th column corresponds to \( S_j \). Now, the goal is to find an optimal row cover! We can find an \( O(\log m) \) approximation using our set-cover algorithm.
We have seen why hitting set can be converted to set cover. We will now discuss the opposite.

Consider the matrix row cover problem again.

**Example**

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

An optimal row cover takes the first four rows.

We can also interpret the problem as a **hitting set** problem!

See the previous slide.
Consider the set-cover instance $U = \{1, 2, \ldots, 8\}$ and $S = \{S_1, S_2, \ldots, S_5\}$ where $S_1 = \{1, 2, 3, 4\}$, $S_2 = \{2, 5, 7\}$, $S_3 = \{6, 7\}$, $S_4 = \{1, 8\}$, and $S_5 = \{1, 2, 3, 8\}$.

We can describe the instance with

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

where each row corresponds to a set, and each column corresponds to an integer in $U$. The goal is again to find an optimal row cover!

Hence, if we have a $\rho$-approximate algorithm for hitting set, we can achieve approximation ratio $\rho$ for set cover as well.