Set Cover in Sub-linear Time

Piotr Indyk
MIT

Sepideh Mahabadi
Columbia University

Ronitt Rubinfeld
MIT/TAU

Ali Vakilian
MIT

Anak Yodpinyanee
MIT
Set Cover Problem

Input: Collection $\mathcal{F}$ of sets $S_1, \ldots, S_m$ subset of $\mathcal{U} = \{1, \ldots, n\}$
Set Cover Problem

Input: Collection $\mathcal{F}$ of sets $S_1, \ldots, S_m$ subset of $\mathcal{U} = \{1, \ldots, n\}$

Output: a subset $\mathcal{C}$ of $\mathcal{F}$ such that:

- $\mathcal{C}$ covers $\mathcal{U}$
- $|\mathcal{C}|$ is minimized
Set Cover Problem

Input: Collection $\mathcal{F}$ of sets $S_1, \ldots, S_m$ subset of $\mathcal{U} = \{1, \ldots, n\}$

Output: a subset $\mathcal{C}$ of $\mathcal{F}$ such that:
- $\mathcal{C}$ covers $\mathcal{U}$
- $|\mathcal{C}|$ is minimized

Complexity:
Set Cover Problem

Input: Collection $\mathcal{F}$ of sets $S_1, ..., S_m$ subset of $\mathcal{U} = \{1, ..., n\}$

Output: a subset $\mathcal{C}$ of $\mathcal{F}$ such that:

- $\mathcal{C}$ covers $\mathcal{U}$
- $|\mathcal{C}|$ is minimized

Complexity:

- NP-hard
Set Cover Problem

Input: Collection $\mathcal{F}$ of sets $S_1, \ldots, S_m$ subset of $\mathcal{U} = \{1, \ldots, n\}$

Output: a subset $\mathcal{C}$ of $\mathcal{F}$ such that:
- $\mathcal{C}$ covers $\mathcal{U}$
- $|\mathcal{C}|$ is minimized

Complexity:
- NP-hard
- Greedy $(\ln n)$-approximation algorithm
Set Cover Problem

Input: Collection $\mathcal{F}$ of sets $S_1, \ldots, S_m$ subset of $\mathcal{U} = \{1, \ldots, n\}$

Output: a subset $\mathcal{C}$ of $\mathcal{F}$ such that:
- $\mathcal{C}$ covers $\mathcal{U}$
- $|\mathcal{C}|$ is minimized

Complexity:
- NP-hard
- Greedy $(\ln n)$-approximation algorithm
- Can’t do better unless $P=NP$

[LY91][RS97][Fei98][AMS06][DS14]
Set Cover for Massive Data

• Previously studied in the massive data models
  • Application in “Big Data”: Clustering, Data Mining, ...
Set Cover for Massive Data

• Previously studied in the massive data models
  • Application in “Big Data”: Clustering, Data Mining, ...
  • Introduced by [Saha and Getoor 09] in the streaming model, has been further studied: [ER’14, DIMV’14, CW’16, HIMV’16, AKL’16, A’17, MV’17, IMRUVVY’17]
Set Cover for Massive Data

• Previously studied in the massive data models
  • Application in “Big Data”: Clustering, Data Mining, ...
  • Introduced by [Saha and Getoor 09] in the streaming model, has been further studied: [ER’14, DIMV’14, CW’16, HIMV’16, AKL’16, A’17, MV’17, IMRUVY’17]

• In this Talk: Sublinear time algorithm for Set Cover
Sub-linear Time Set Cover

Data Access Model?
Sub-linear Time Set Cover

**Data Access Model** [NO’08, YYI’12]

- \( \text{EltOf}(S, i) \): \( i \)th element in \( S \)
- \( \text{SetOf}(e, j) \): \( j \)th set containing \( e \)
Sub-linear Time Set Cover

Data Access Model [NO’08, YYI’12]

• No assumption on the order

EltOf(S, i): i-th element in S
SetOf(e, j): j-th set containing e
Sub-linear Time Set Cover

Data Access Model [NO’08, YYI’12]
• No assumption on the order
• Incidence list in (sub-linear) algorithms for graphs

EltOf(S, i): i-th element in S
SetOf(e, j): j-th set containing e
Sub-linear Time Set Cover

**Data Access Model** [NO’08, YYI’12]
- No assumption on the order
- Incidence list in (sub-linear) algorithms for graphs
- Sublinear in $mn$

$\text{EltOf}(S, i)$: $i$th element in $S$
$\text{SetOf}(e, j)$: $j$th set containing $e$
Sub-linear Time Set Cover

**Data Access Model** [NO’08, YYI’12]

- No assumption on the order
- Incidence list in (sub-linear) algorithms for graphs
- Sublinear in $mn$

**Prior Results**

\[
\text{EltOf}(S, i): \text{$i$th element in $S$} \\
\text{SetOf}(e, j): \text{$j$th set containing $e$}
\]
Sub-linear Time Set Cover

Data Access Model [NO’08, YYI’12]

- No assumption on the order
- Incidence list in (sub-linear) algorithms for graphs
- Sublinear in $mn$

Prior Results

- [Nguyen, Onak’08][Yoshida, Yamamoto, Ito’12]
  - Constant queries, if degree is constant
Sub-linear Time Set Cover

Data Access Model \[\text{NO’08, YYI’12}\]
- No assumption on the order
- Incidence list in (sub-linear) algorithms for graphs
- Sublinear in $mn$

Prior Results
- [Nguyen, Onak’08][Yoshida, Yamamoto, Ito’12]
  - Constant queries, if degree is constant

- Output the size, not cover
- Mixed additive/multiplicative approximation
Sub-linear Time Set Cover

Data Access Model [NO’08, YYI’12]
- No assumption on the order
- Incidence list in (sub-linear) algorithms for graphs
- Sublinear in $mn$

Prior Results
- [Nguyen, Onak’08][Yoshida, Yamamoto, Ito’12]
  - Constant queries, if degree is constant
- [Koufogiannakis, Young’14][Grigoriadis, Kachiyan’95]:
  - Find $(1 + \epsilon)$-approximate fractional solution, then perform randomized rounding to achieve $O(\log n)$-approximation

\begin{align*}
\text{EltOf}(S, i) & : i\text{th element in } S \\
\text{SetOf}(e, j) & : j\text{th set containing } e
\end{align*}
Sub-linear Time Set Cover

**Data Access Model** \[\text{NO’08, YYI’12} \]
- No assumption on the order
- Incidence list in (sub-linear) algorithms for graphs
- Sublinear in \(mn\)

**Prior Results**
- \[\text{Nguyen, Onak’08, Yoshida, Yamamoto, Ito’12}\]
  - Constant queries, if degree is constant
- \[\text{Koufogiannakis, Young’14, Grigoriadis, Kachiyan’95}\]:
  - Find \((1 + \varepsilon)\)-approximate *fractional solution*, then perform *randomized rounding* to achieve \(O(\log n)\)-approximation
  - \(O(mk^2 + nk^2)\) (can be improved to \(O(m + nk)\))

\(n = \text{number of elements} \quad m = \text{number of sets} \quad k = \text{size of the optimal solution}\)
## Our Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>$-$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}}$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
<td>$\alpha \leq 1.01$ \n $k = O(n/\log m)$</td>
</tr>
<tr>
<td><strong>Cover Verification</strong></td>
<td>$-$</td>
<td>$\tilde{\Omega}(nk)$</td>
<td>$k \leq n/2$</td>
</tr>
</tbody>
</table>

$\rho = \text{approximation factor for offline Set Cover}$

$n = \text{number of elements} \quad m = \text{number of sets} \quad k = \text{size of the optimal solution}$
# Our Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}}$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
<td>$\alpha \leq 1.01$ $k = O(n/\log m)$</td>
</tr>
<tr>
<td><strong>Cover Verification</strong></td>
<td>–</td>
<td>$\tilde{\Omega}(nk)$</td>
<td>$k \leq n/2$</td>
</tr>
</tbody>
</table>

$\rho$ = approximation factor for offline **Set Cover**  

$n$ = number of elements  

$m$ = number of sets  

$k$ = size of the optimal solution
Our Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Cover</td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}}$</td>
</tr>
<tr>
<td>Cover Verification</td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
<td>$\alpha \leq 1.01$ \newline $k = \Theta(n/\log m)$</td>
</tr>
<tr>
<td></td>
<td>$-$</td>
<td>$\tilde{\Omega}(nk)$</td>
<td>$k \leq n/2$</td>
</tr>
</tbody>
</table>

**Cover Verification:** given **k sets**, verify whether their **union** covers the universe.

$\rho = \text{approximation factor for offline Set Cover}$

$n = \text{number of elements} \quad m = \text{number of sets} \quad k = \text{size of the optimal solution}$
Our Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Cover</td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha - 1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha + 1}}$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
<td>$\alpha \leq 1.01$ $k = O(n/\log m)$</td>
</tr>
<tr>
<td>Cover Verification</td>
<td>$-$</td>
<td>$\tilde{\Omega}(nk)$</td>
<td>$k \leq n/2$</td>
</tr>
</tbody>
</table>

**Cover Verification**: given $k$ sets, verify whether their union covers the universe.

$\rho$ = approximation factor for offline **Set Cover**

$n$ = number of elements    $m$ = number of sets    $k$ = size of the optimal solution
## Our Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>(\alpha \rho + 1)</td>
<td>(\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha - 1}} + nk \right))</td>
<td>(\alpha \geq 2)</td>
</tr>
<tr>
<td></td>
<td>(\rho + 1)</td>
<td>(\tilde{O} \left( \frac{mn}{k} \right))</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>(\alpha)</td>
<td>(\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right))</td>
<td>(k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha + 1}})</td>
</tr>
<tr>
<td>Cover Verification</td>
<td>–</td>
<td>(\tilde{\Omega}(nk))</td>
<td>(k \leq n/2)</td>
</tr>
</tbody>
</table>

**Cover Verification**: given \(k\) sets, verify whether their union covers the universe.

\(\rho\) = approximation factor for offline **Set Cover**

\(n = \text{number of elements}\)  \(m = \text{number of sets}\)  \(k = \text{size of the optimal solution}\)
## Our Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>( \alpha \rho + 1 )</td>
<td>( \tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right) )</td>
<td>( \alpha \geq 2 )</td>
</tr>
<tr>
<td></td>
<td>( \rho + 1 )</td>
<td>( \tilde{O} \left( \frac{mn}{k} \right) )</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>( \alpha )</td>
<td>( \tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right) )</td>
<td>( k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}} )</td>
</tr>
<tr>
<td><strong>Cover Verification</strong></td>
<td>( \alpha )</td>
<td>( \tilde{\Omega} \left( \frac{mn}{k} \right) )</td>
<td>( \alpha \leq 1.01 ), ( k = O(n/\log m) )</td>
</tr>
<tr>
<td></td>
<td>–</td>
<td>( \tilde{\Omega}(nk) )</td>
<td>( k \leq n/2 )</td>
</tr>
</tbody>
</table>

**Cover Verification**: given \( k \) sets, verify whether their union covers the universe.

- \( \rho \) = approximation factor for offline **Set Cover**
- \( n \) = number of elements
- \( m \) = number of sets
- \( k \) = size of the optimal solution
## Our Results

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>( \alpha \rho + 1 )</td>
<td>( \tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right) )</td>
<td>( \alpha \geq 2 )</td>
</tr>
<tr>
<td></td>
<td>( \rho + 1 )</td>
<td>( \tilde{O} \left( \frac{mn}{k} \right) )</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>( \alpha )</td>
<td>( \tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right) )</td>
<td>( k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}} )</td>
</tr>
<tr>
<td><strong>Cover Verification</strong></td>
<td>–</td>
<td>( \tilde{\Omega}(nk) )</td>
<td>( k \leq n/2 )</td>
</tr>
</tbody>
</table>

**Cover Verification**: given \( k \) sets, verify whether their union covers the universe.

\( \rho \) = approximation factor for offline **Set Cover**

\( n \) = number of elements  \( m \) = number of sets  \( k \) = size of the optimal solution
This Talk

**Theorem:** Any randomized algorithm that with probability at least 2/3 distinguishes whether the minimum Set Cover size is 2 or at least 3 requires $\tilde{\Omega}(mn)$ number of queries.
High Level Plan

1. Construct a **basic instance** $I^*$
   - Minimum set cover size is at least 3
High Level Plan

1. Construct a **basic instance** $I^*$
   - Minimum set cover size is at least 3
2. **Randomized Procedure** on $I^*$ to get a **modified instance** $I$
   - Minimum set cover size is 2
High Level Plan

1. Construct a **basic instance** $I^*$
   - Minimum set cover size is at least 3
2. **Randomized Procedure** on $I^*$ to get a **modified instance** $I$
   - Minimum set cover size is 2
   - $I^*$ and $I$ only **differ in a few positions**
   - The differences are **distributed almost uniformly** at random
High Level Plan

1. Construct a basic instance $I^*$
   • Minimum set cover size is at least 3
2. Randomized Procedure on $I^*$ to get a modified instance $I$
   • Minimum set cover size is 2
   • $I^*$ and $I$ only differ in a few positions
   • The differences are distributed almost uniformly at random

Distance of two instances

is proportional to

Number of positions SetOf and EltOf oracles differ in
High Level Plan

1. Construct a **basic instance** $I^*$
   - Minimum set cover size is at least 3
2. **Randomized Procedure** on $I^*$ to get a **modified instance** $I$
   - Minimum set cover size is 2
   - $I^*$ and $I$ only **differ in a few positions**
   - The differences are **distributed almost uniformly** at random

**Applying Yao’s Principle**

**Input distribution**: Flip an unbiased coin,
1. Head $\rightarrow$ input is $I^*$ (i.e. w.p. 1/2)
2. Tail $\rightarrow$ input is a random modified instance $I$ generated from $I^*$
High Level Plan

1. Construct a **basic instance** $I^*$
   - Minimum set cover size is at least 3
2. **Randomized Procedure** on $I^*$ to get a **modified instance** $I$
   - Minimum set cover size is 2
   - $I^*$ and $I$ only **differ in a few positions**
   - The differences are **distributed almost uniformly** at random

**Input distribution**: Flip an unbiased coin,
1. Head $\rightarrow$ input is $I^*$ (i.e. w.p. 1/2)
2. Tail $\rightarrow$ input is a random modified instance $I$ generated from $I^*$

Any deterministic algorithm that returns the correct output with success probability $\geq 2/3$ requires $\Omega\left(\frac{mn}{\log m}\right)$ queries
The Basic Instance

**Construction:** For every $S, e$ the set $S$ contains $e$ with probability $1 - p_0$ where

$$p_0 = \sqrt{\frac{9 \log m}{n}}$$
The Basic Instance

**Construction:** For every $S, e$ the set $S$ contains $e$ with probability $1 - p_0$ where

$$p_0 = \sqrt{\frac{9 \log m}{n}}$$

Most of such instances have the following properties:

1. No two sets cover all elements
2. For any two sets, the number of uncovered elements is $O(\log m)$
3. The intersection size is at least $\Omega(n)$
4. For each element, the number of sets not covering it is at most $6m \sqrt{\frac{\log m}{n}}$
5. For any pair of elements, the number of sets containing only the first element is at least $\frac{m \sqrt{9 \log m}}{4 \sqrt{n}}$
6. For any three sets, the number of elements in the first two but not in the third one is at least $6 \sqrt{n \log m}$
The Basic Instance

**Construction:** For every $S, e$ the set $S$ contains $e$ with probability $1 - p_0$ where

$$p_0 = \sqrt{\frac{9 \log m}{n}}$$

Most of such instances have the following properties:

1. No two sets cover all elements
2. For any two sets, the number of uncovered elements is $O(\log m)$
3. The intersection size is at least $\Omega(n)$
4. For each element, the number of sets not covering it is at most $6m \sqrt{\frac{\log m}{n}}$
5. For any pair of elements, the number of sets containing only the first element is at least $\frac{m \sqrt{9 \log m}}{4 \sqrt{n}}$
6. For any three sets, the number of elements in the first two but not in the third one is at least $6 \sqrt{n \log m}$
The Basic Instance

Sets

Elements

\[ e \in S \]
\[ e \notin S \]
Generating a Modified Instance
Generating a Modified Instance

\[ U = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.

\[ S_1 = \{e_2, e_3, \ldots\} \]

\[ S_2 = \{e_2, e_4, \ldots\} \]
Generating a Modified Instance

\[ \mathcal{U} = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in \mathcal{U} \setminus (S_1 \cup S_2) \),
  \[
  S_1 = \{e_2, e_3, \ldots\}
  
  S_2 = \{e_2, e_4, \ldots\}
  
\]
Generating a Modified Instance

\[ u = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in u \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)

\[ S_1 = \{e_2, e_3, \ldots\} \]
\[ S_2 = \{e_2, e_4, \ldots\} \]
Generating a Modified Instance

\[ U = \{ e_1, e_2, e_3, e_4, \ldots \} \]

Pick \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in U \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)

\[ S_1 = \{ e_2, e_3, \ldots \} \]

\[ S_2 = \{ e_2, e_4, \ldots \} \]
Generating a Modified Instance

\[ U = \{ e_1, e_2, e_3, e_4, \ldots \} \]

Pick \( S_1 \) and \( S_2 \) in the \textbf{basic instance} uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in U \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  - Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)

\[ S_1 = \{ e_2, e_3, \ldots \} \]
\[ S_2 = \{ e_2, e_4, \ldots \} \]
\[ S_3 = \{ e_1, e_4, \ldots \} \]
Generating a Modified Instance

\[ U = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in U \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  - Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)
  - \( S_2 \) and \( S_3 \) swap \( e_1 \) and \( e_2 \)
Generating a Modified Instance

Pick $S_1$ and $S_2$ in the **basic instance** uniformly at random and turn them into a set cover.

- For an uncovered $e_1 \in \mathcal{U} \setminus (S_1 \cup S_2)$,
  - Add $e_1$ to $S_2$
  - Remove a random $e_2 \in S_2 \cap S_1$ from $S_2$
  - Pick a random set $S_3$ that contains $e_1$ but not $e_2$
  - $S_2$ and $S_3$ swap $e_1$ and $e_2$

\[
\mathcal{U} = \{e_1, e_2, e_3, e_4, \ldots\}
\]

$S_1 = \{e_2, e_3, \ldots\}$

$S_2 = \{e_2, e_4, \ldots\}$

$S_3 = \{e_1, e_4, \ldots\}$
Generating a Modified Instance

\[ \mathcal{U} = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in \mathcal{U} \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  - Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)
  - \( S_2 \) and \( S_3 \) swap \( e_1 \) and \( e_2 \)

\[ S_1 = \{e_2, e_3, \ldots\} \]
\[ S_2 = \{e_1, e_4, \ldots\} \]
\[ S_3 = \{e_2, e_4, \ldots\} \]
Generating a Modified Instance

\[ U = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in U \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  - Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)
  - \( S_2 \) and \( S_3 \) swap \( e_1 \) and \( e_2 \)

\( S_1 = \{e_2, e_3, \ldots\} \)

\( S_2 = \{e_2, e_4, \ldots\} \)

\( S_3 = \{e_1, e_4, \ldots\} \)
Generating a Modified Instance

\[ \mathcal{U} = \{e_1, e_2, e_3, e_4, \ldots\} \]

- \( S_1 \) and \( S_2 \) in the **basic instance** uniformly at random and turn them into a set cover.
- For an uncovered \( e_1 \in \mathcal{U} \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  - Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)
  - \( S_2 \) and \( S_3 \) swap \( e_1 \) and \( e_2 \)

\( S_1 = \{e_2, e_3, \ldots\} \)
\( S_2 = \{e_2, e_4, \ldots\} \)
\( S_3 = \{e_1, e_4, \ldots\} \)

“swap” operation
Generating a Modified Instance

\[ U = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the basic instance uniformly at random and turn them into a set cover.

• For an uncovered \( e_1 \in U \setminus (S_1 \cup S_2) \),
  • Add \( e_1 \) to \( S_2 \)
  • Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  • Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)
  • \( S_2 \) and \( S_3 \) swap \( e_1 \) and \( e_2 \)

In a swap operation: only four positions change in the query access model.
Generating a Modified Instance

Pick $S_1$ and $S_2$ in the basic instance uniformly at random and turn them into a set cover.

- For an uncovered $e_1 \in U \setminus (S_1 \cup S_2)$,
  - Add $e_1$ to $S_2$
  - Remove a random $e_2 \in S_2 \cap S_1$ from $S_2$
  - Pick a random set $S_3$ that contains $e_1$ but not $e_2$
  - $S_2$ and $S_3$ swap $e_1$ and $e_2$

$U = \{e_1, e_2, e_3, e_4, \ldots\}$

$S_1 = \{e_2, e_3, \ldots\}$

$S_2 = \{e_2, e_4, \ldots\}$

$S_3 = \{e_1, e_4, \ldots\}$

In a swap operation: only four positions change in the query access model.
Generating a Modified Instance

\[ U = \{ e_1, e_2, e_3, e_4, \ldots \} \]

\[ S_1 = \{ e_2, e_3, \ldots \} \]

\[ S_2 = \{ e_2, e_4, \ldots \} \]

\[ S_3 = \{ e_1, e_4, \ldots \} \]

Pick \( S_1 \) and \( S_2 \) in the basic instance uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in U \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  - Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)
  - \( S_2 \) and \( S_3 \) swap \( e_1 \) and \( e_2 \)

- Performing the swap operation for all \( U \setminus (S_1 \cup S_2) \), \( S_1 \cup S_2 \) becomes \( U \)
Generating a Modified Instance

\[ \mathcal{U} = \{e_1, e_2, e_3, e_4, \ldots\} \]

Pick \( S_1 \) and \( S_2 \) in the basic instance uniformly at random and turn them into a set cover.

- For an uncovered \( e_1 \in \mathcal{U} \setminus (S_1 \cup S_2) \),
  - Add \( e_1 \) to \( S_2 \)
  - Remove a random \( e_2 \in S_2 \cap S_1 \) from \( S_2 \)
  - Pick a random set \( S_3 \) that contains \( e_1 \) but not \( e_2 \)
  - \( S_2 \) and \( S_3 \) swap \( e_1 \) and \( e_2 \)

- Performing the swap operation for all \( \mathcal{U} \setminus (S_1 \cup S_2) \), \( S_1 \cup S_2 \) becomes \( \mathcal{U} \)
- The resulting and the initial instances only differ in \( O(\log m) \) positions
Overall Argument

Lemma: For any set $S$ and any element $e$, the probability that $(e,S)$ participates in a swap is almost uniform, i.e., $O\left(\frac{\log m}{mn}\right)$.

- Using the properties of basic instances and the randomized procedure for generating modified instances

Input:
- W.p. $\frac{1}{2}$ the input is the basic instance $I^*$
- W.p. $\frac{1}{2}$ the input is a randomly generated modified instance $I$ from $I^*$
**Overall Argument**

**Lemma:** For any set $S$ and any element $e$, the probability that $(e,S)$ participates in a swap is almost uniform, i.e., $O\left(\frac{\log m}{mn}\right)$.

- Using the properties of basic instances and the randomized procedure for generating modified instances

**Input:**
- W.p. $\frac{1}{2}$ the input is the basic instance $I^*$
- W.p. $\frac{1}{2}$ the input is a randomly generated modified instance $I$ from $I^*$

**Theorem:** Any randomized algorithm that with probability at least $2/3$ distinguishes whether the minimum set cover size is 2 or at least 3 requires $\tilde{\Omega}(mn)$ number of queries.
**Summary**

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha - 1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
<td></td>
</tr>
<tr>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>$-$</td>
<td></td>
</tr>
<tr>
<td><strong>Set Cover</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha + 1}}$</td>
<td></td>
</tr>
<tr>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
<td>$\alpha \leq 1.01$ $k = O(n/\log m)$</td>
<td></td>
</tr>
<tr>
<td><strong>Cover Verification</strong></td>
<td>$-$</td>
<td>$\tilde{\Omega}(nk)$</td>
<td>$k \leq n/2$</td>
</tr>
</tbody>
</table>

**Remark:**
- For larger values of $\alpha$, change the construction of basic instance and modified instances slightly
## Summary

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>αρ + 1</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>ρ + 1</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>α</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}}$</td>
</tr>
</tbody>
</table>
| **Cover Verification**| –             | $\tilde{\Omega} (nk)$ | $\alpha \leq 1.01 
 k = O(n/\log m)$ |
|                       |               |                  | $k \leq n/2$               |

**Remark:**
- For larger values of $\alpha$, change the construction of basic instance and modified instances slightly
- The lower bound works for the corresponding estimation problem
Open Problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Cover</td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left(m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\Omega \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}}$</td>
</tr>
</tbody>
</table>
|                               | $\alpha$       | $\Omega \left( \frac{mn}{k} \right)$              | $\alpha \leq 1.01$  
$\quad$                     | $\quad$                         | $k = O(n/\log m)$     |
| Cover Verification             | –              | $\Omega(nk)$      | $k \leq n/2$                          |

- Improve the bounds for the corresponding estimation problem
Open Problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Cover</td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{1 \over \alpha-1} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>$-$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{1 \over 2\alpha} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{1 \over 4\alpha+1}$</td>
</tr>
<tr>
<td>Cover Verification</td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
<td>$\alpha \leq 1.01$; $k = O(n/\log m)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\tilde{\Omega}(nk)$</td>
<td>$k \leq n/2$</td>
</tr>
</tbody>
</table>

- Improve the bounds for the corresponding estimation problem

Best upper bound: $\tilde{O}(mn/k)$
Best lower bound: $\tilde{\Omega}(mn/k^2)$
### Open Problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha-1}} + nk \right)$</td>
<td>$\alpha \geq 2$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
<td>$k \leq \left( \frac{n}{\log m} \right)^{\frac{1}{4\alpha+1}}$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
<td>$\alpha \leq 1.01$ (k = O(n/\log m))</td>
</tr>
<tr>
<td><strong>Cover Verification</strong></td>
<td>–</td>
<td>$\tilde{\Omega}(nk)$</td>
<td>$k \leq n/2$</td>
</tr>
</tbody>
</table>

- Improve the bounds for the corresponding estimation problem
- Prove a lower bound of $\Omega(nk)$ for the set cover problem as well
### Open Problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>Approximation</th>
<th>Query Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Set Cover</strong></td>
<td>$\alpha \rho + 1$</td>
<td>$\tilde{O} \left( m \left( \frac{n}{k} \right)^{\frac{1}{\alpha - 1}} + nk \right)$</td>
</tr>
<tr>
<td></td>
<td>$\rho + 1$</td>
<td>$\tilde{O} \left( \frac{mn}{k} \right)$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( m \left( \frac{n}{k} \right)^{\frac{1}{2\alpha}} \right)$</td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>$\tilde{\Omega} \left( \frac{mn}{k} \right)$</td>
</tr>
<tr>
<td><strong>Cover Verification</strong></td>
<td>$-$</td>
<td>$\tilde{\Omega}(nk)$</td>
</tr>
</tbody>
</table>

- Improve the bounds for the corresponding estimation problem
- Prove a lower bound of $\Omega(nk)$ for the set cover problem as well