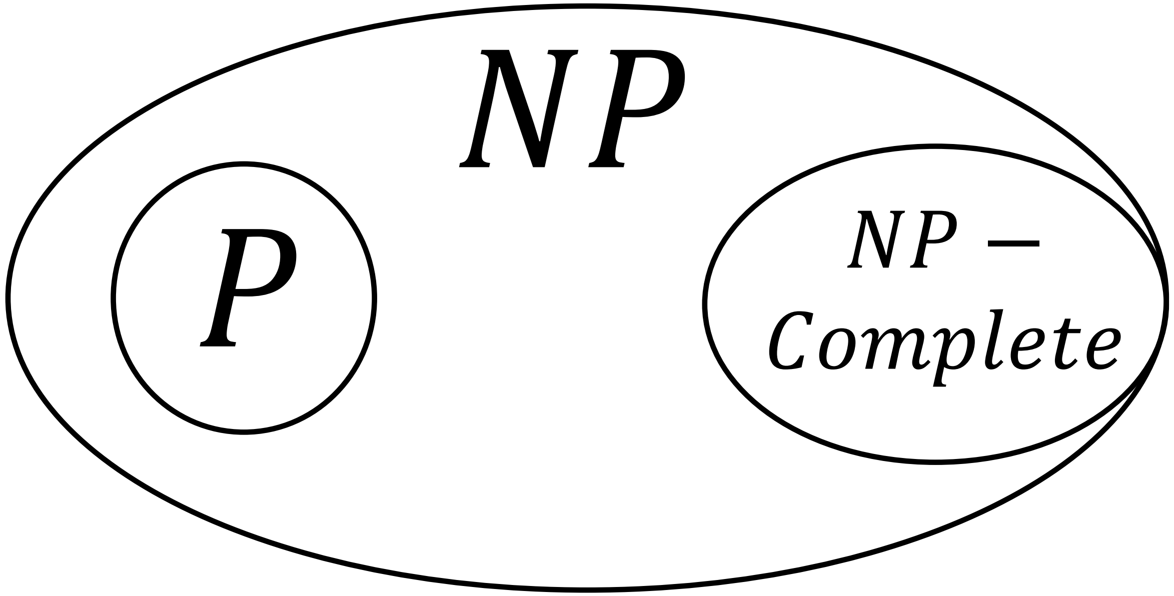
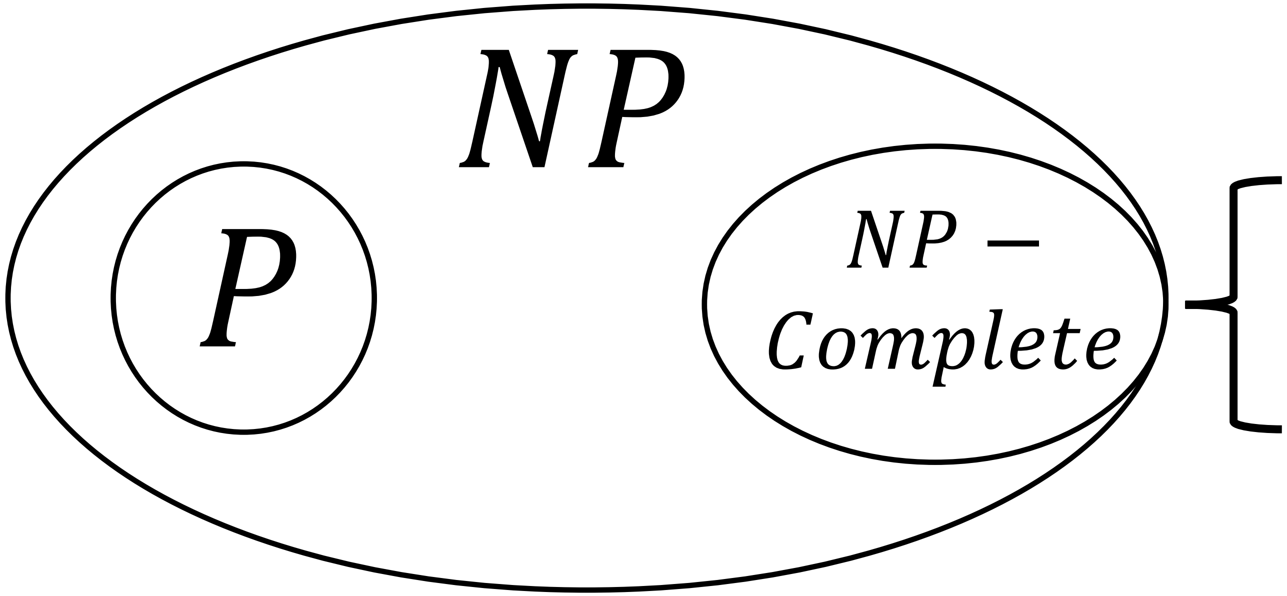
Approximation Algorithms
CSCI 532

NP Complete



Problems in NP whose algorithms can solve any other problem in NP with polynomial extra time.

NP Complete



Problems in NP whose algorithms can solve any other problem in NP with polynomial extra time.

Techniques to handle NP-Complete problems:

1. Brute Force (i.e. Exponential Time).
2. Heuristics.
3. Approximation Algorithms.
4. Fixed-parameter Tractable Algorithms.

Approximation Algorithms

$$\text{ALG} \leq \alpha \ \text{OPT}$$

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Approximation Algorithms

$$\text{ALG} \leq \alpha \text{ OPT}$$

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

if problem is a maximization problem, $\text{ALG} \geq \frac{1}{\alpha} \text{OPT}$

Approximation Algorithms

$$\text{ALG} \leq \alpha \,\text{OPT}$$

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example:

Approximation Algorithms

1.12

$$\mathrm{ALG} \leq \alpha\ \mathrm{OPT}$$

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example:

- Suppose I know my algorithm is a 1.12-approximation algorithm.

Approximation Algorithms

$$\text{ALG} \leq \overset{1.12}{\alpha}\ \text{OPT}$$

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example:

- Suppose I know my algorithm is a 1.12-approximation algorithm.
- Suppose my algorithm returns a solution of cost 746.125.

Approximation Algorithms

1.12

$$\text{ALG} \leq \alpha \text{ OPT}$$

Cost (size) of algorithm’s solution.

Approximation Ratio

Cost (size) of optimal solution.

Example:

- Suppose I know my algorithm is a 1.12-approximation algorithm.
- Suppose my algorithm returns a solution of cost 746.125.

Then, I know that $746.125 \leq 1.12$ OPT

Approximation Algorithms

1.12

$$\text{ALG} \le \alpha \text{ OPT}$$

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example:

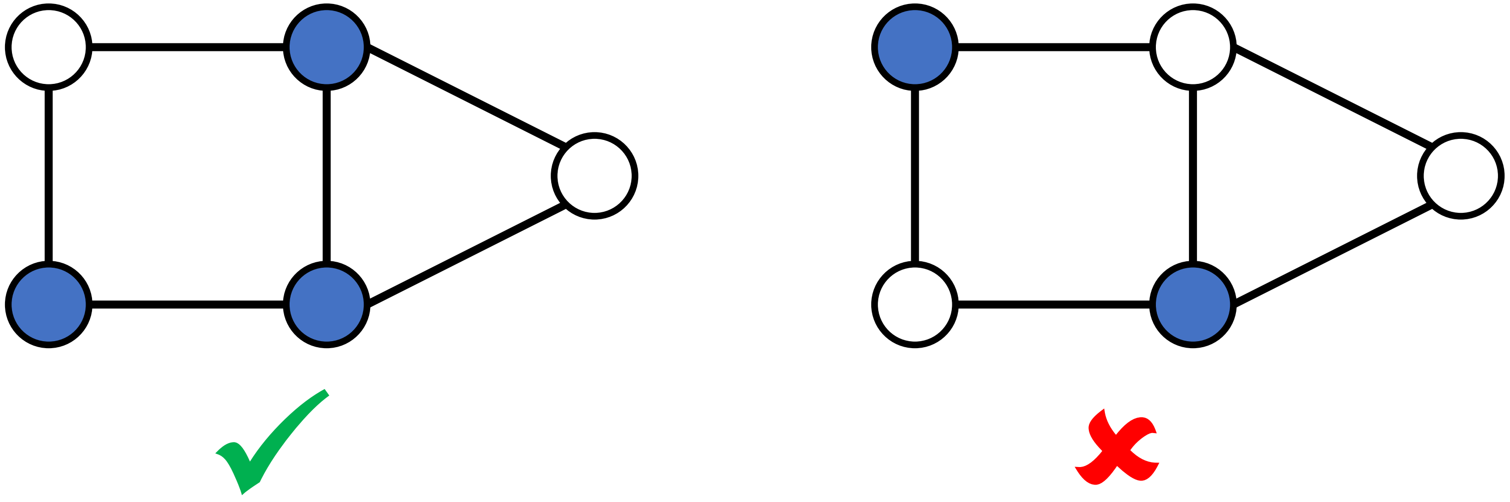
- Suppose I know my algorithm is a 1.12-approximation algorithm.
- Suppose my algorithm returns a solution of cost 746.125.

Then, I know that $746.125 \le 1.12$ OPT

$$\Rightarrow \frac{746.125}{1.12} = 666.183 \le \text{OPT} \le 746.125$$

Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.



Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.

Algorithm:

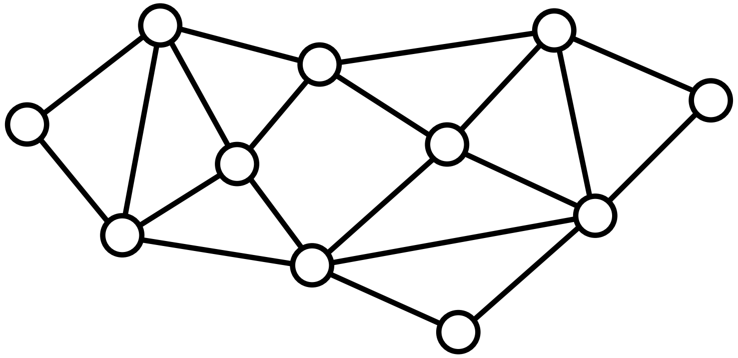
```
while uncovered edge exists
  select both of its vertices
```

Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.

Algorithm:

```
while uncovered edge exists
  select both of its vertices
```

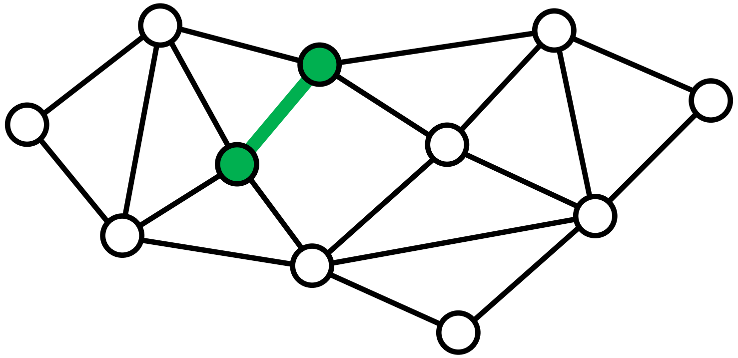


Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.

Algorithm:

```
while uncovered edge exists
  select both of its vertices
```

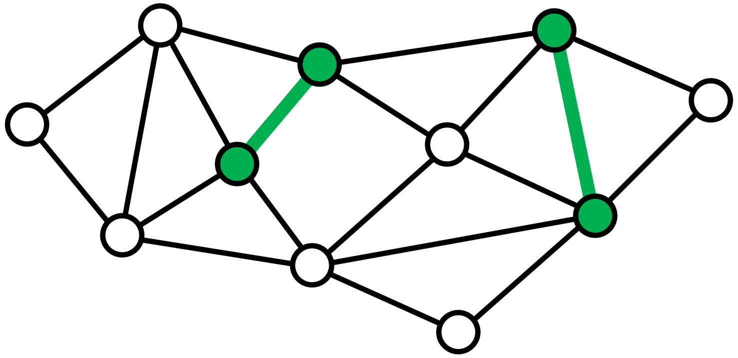


Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.

Algorithm:

```
while uncovered edge exists
  select both of its vertices
```

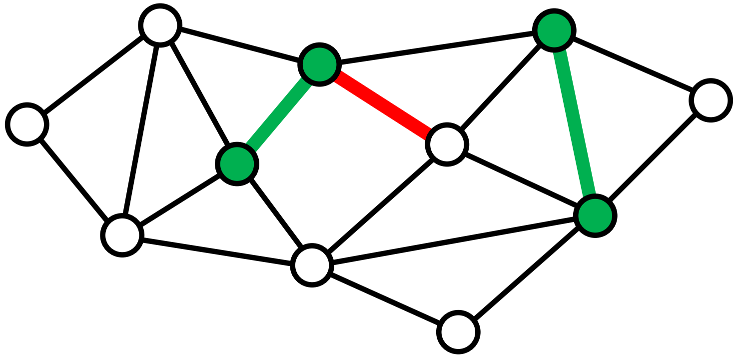


Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.

Algorithm:

```
while uncovered edge exists
  select both of its vertices
```

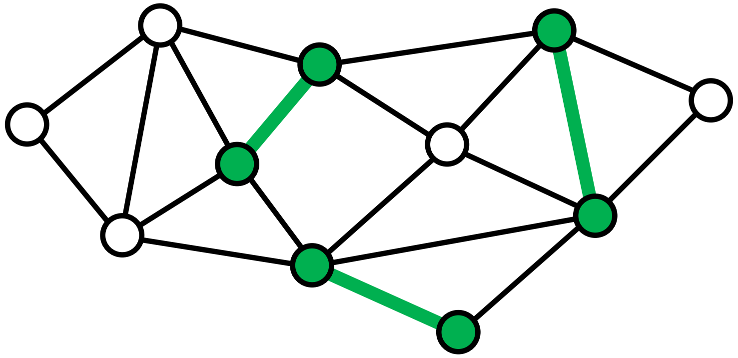


Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.

Algorithm:

```
while uncovered edge exists
  select both of its vertices
```

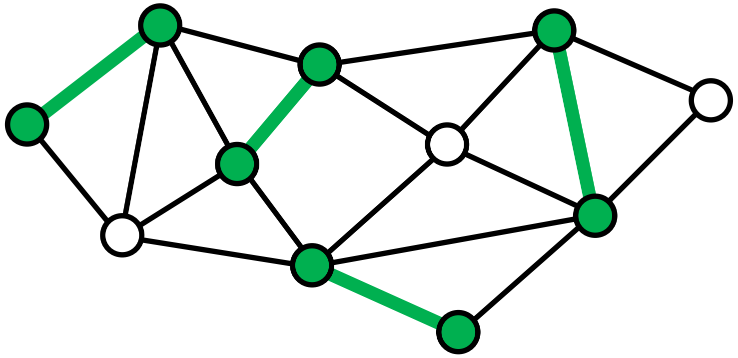


Vertex Cover

Vertex Cover: Given graph, find the smallest subset of vertices such that every edge in the graph has at least one vertex in the subset.

Algorithm:

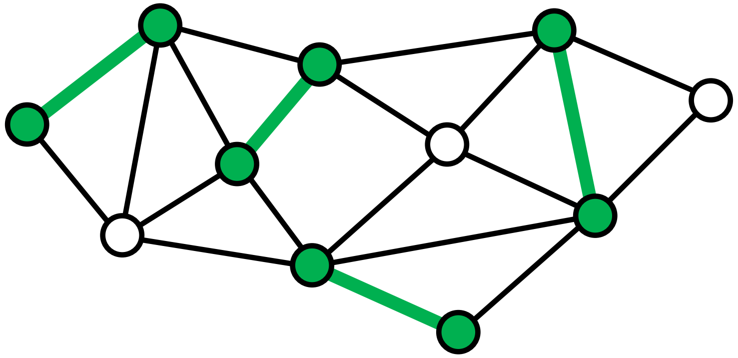
```
while uncovered edge exists
  select both of its vertices
```



Vertex Cover

```
while uncovered edge exists
   select both of its vertices
```

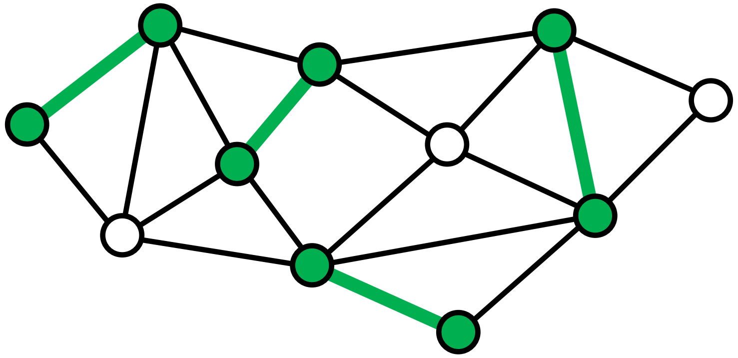
Consider a set of edges, $E' \subset E$, that do not share vertices.



Vertex Cover

```
while uncovered edge exists
   select both of its vertices
```

Consider a set of edges, $E' \subset E$, that do not share vertices. Is there a relationship between the minimum vertex cover and $|E'|$?



Vertex Cover

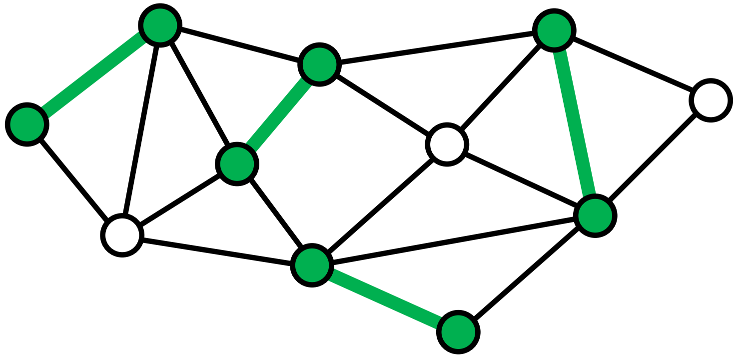
```
while uncovered edge exists
   select both of its vertices
```

Consider a set of edges, $E' \subset E$, that do not share vertices. Is there a relationship between the minimum vertex cover and $|E'|$?

$$|E'| \leq \text{OPT}$$

Size of actual smallest vertex cover.

If we selected fewer than one vertex per edge, we would not have a vertex cover, because that edge would not be covered!



Vertex Cover

```
while uncovered edge exists
   select both of its vertices
```

Consider a set of edges, $E' \subset E$, that do not share vertices. Is there a relationship between the minimum vertex cover and $|E'|$?

$$|E'| \leq \text{OPT}$$

Does the size of the algorithm's output relate to a set of edges that do not share vertices?

Vertex Cover

```
while uncovered edge exists
   select both of its vertices
```

Consider a set of edges, $E' \subset E$, that do not share vertices. Is there a relationship between the minimum vertex cover and $|E'|$?

$$|E'| \leq \text{OPT}$$

Does the size of the algorithm's output relate to a set of edges that do not share vertices?

$$\text{ALG} = 2\,|E'|$$

Vertex Cover

```
while uncovered edge exists
   select both of its vertices
```

Consider a set of edges, $E' \subset E$, that do not share vertices. Is there a relationship between the minimum vertex cover and $|E'|$?

$$|E'| \leq \text{OPT}$$

Does the size of the algorithm's output relate to a set of edges that do not share vertices?

$$\text{ALG} = 2\,|E'|$$

$$\Longrightarrow \text{ALG} = 2\,|E'| \leq 2\ \text{OPT} \Longrightarrow \text{ALG} \leq 2\ \text{OPT}$$

Vertex Cover

```
while uncovered edge exists
   select both of its vertices
```

Consid

a relati

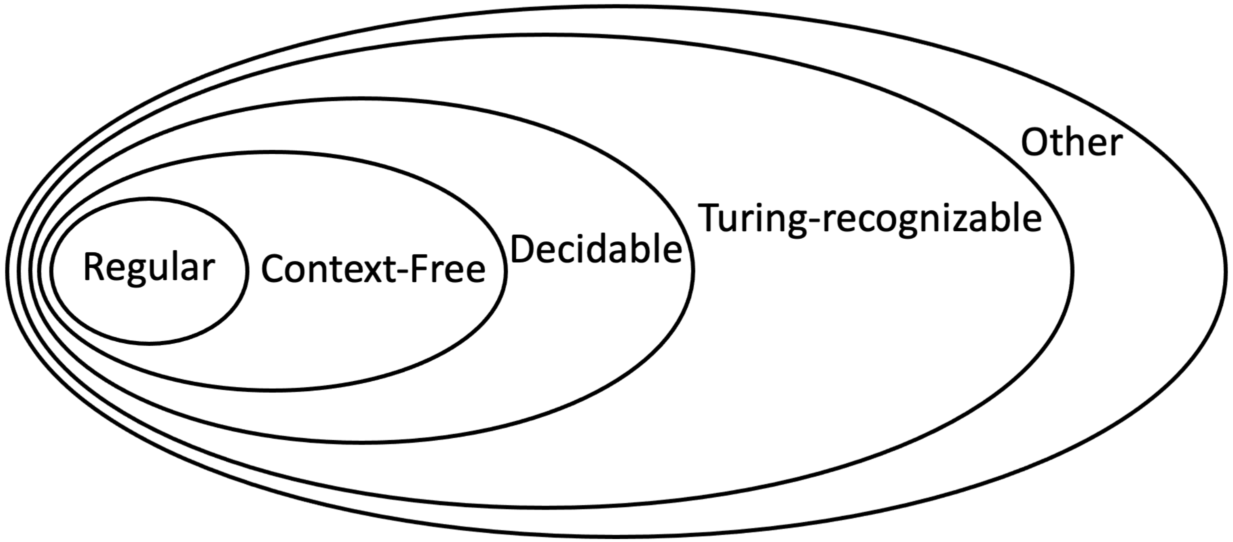
We cannot find optimal vertex covers in poly time unless $P = NP$, but this algorithm is at worst 2-times optimal.

Does t

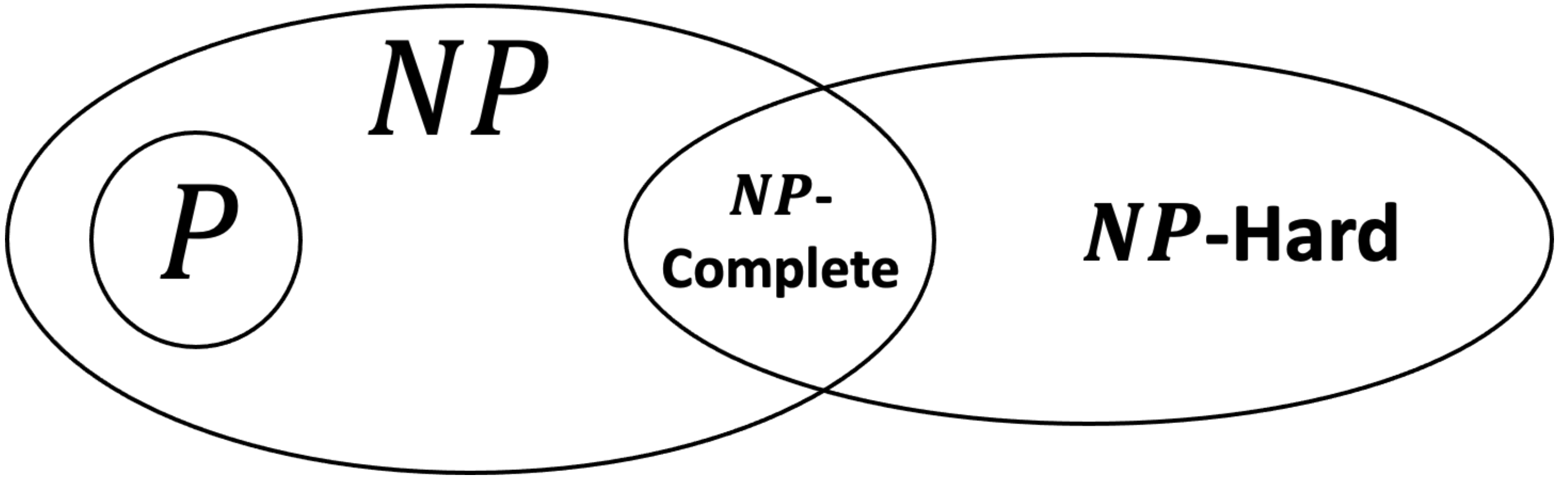
do not share vertices?

ALG = 2 $|E'|$

$\Longrightarrow$ ALG = 2 $|E'| \leq$ 2 OPT $\Longrightarrow$ ALG $\leq$ 2 OPT

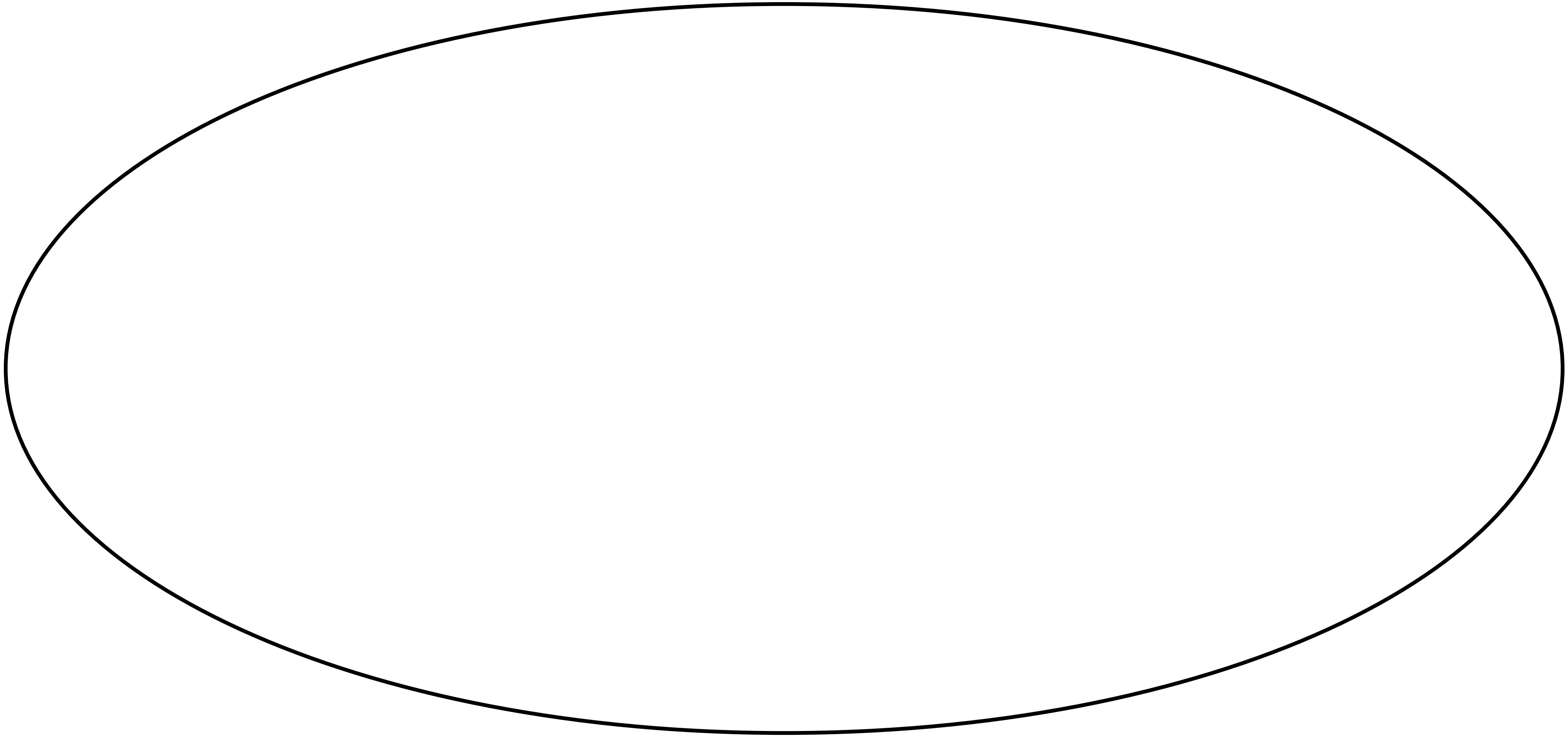


Computability Hierarchy



Complexity Hierarchy

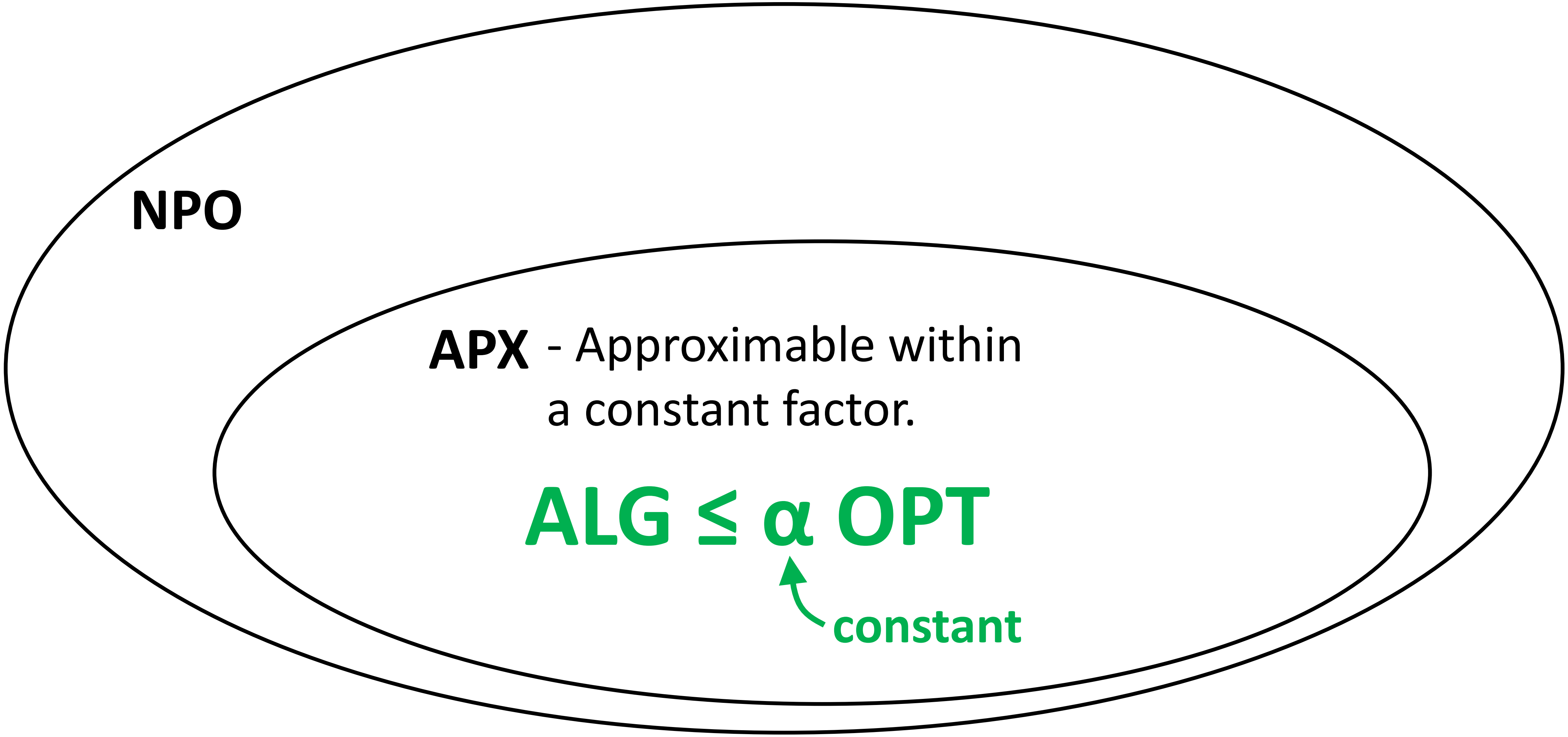
Approximability Hierarchy



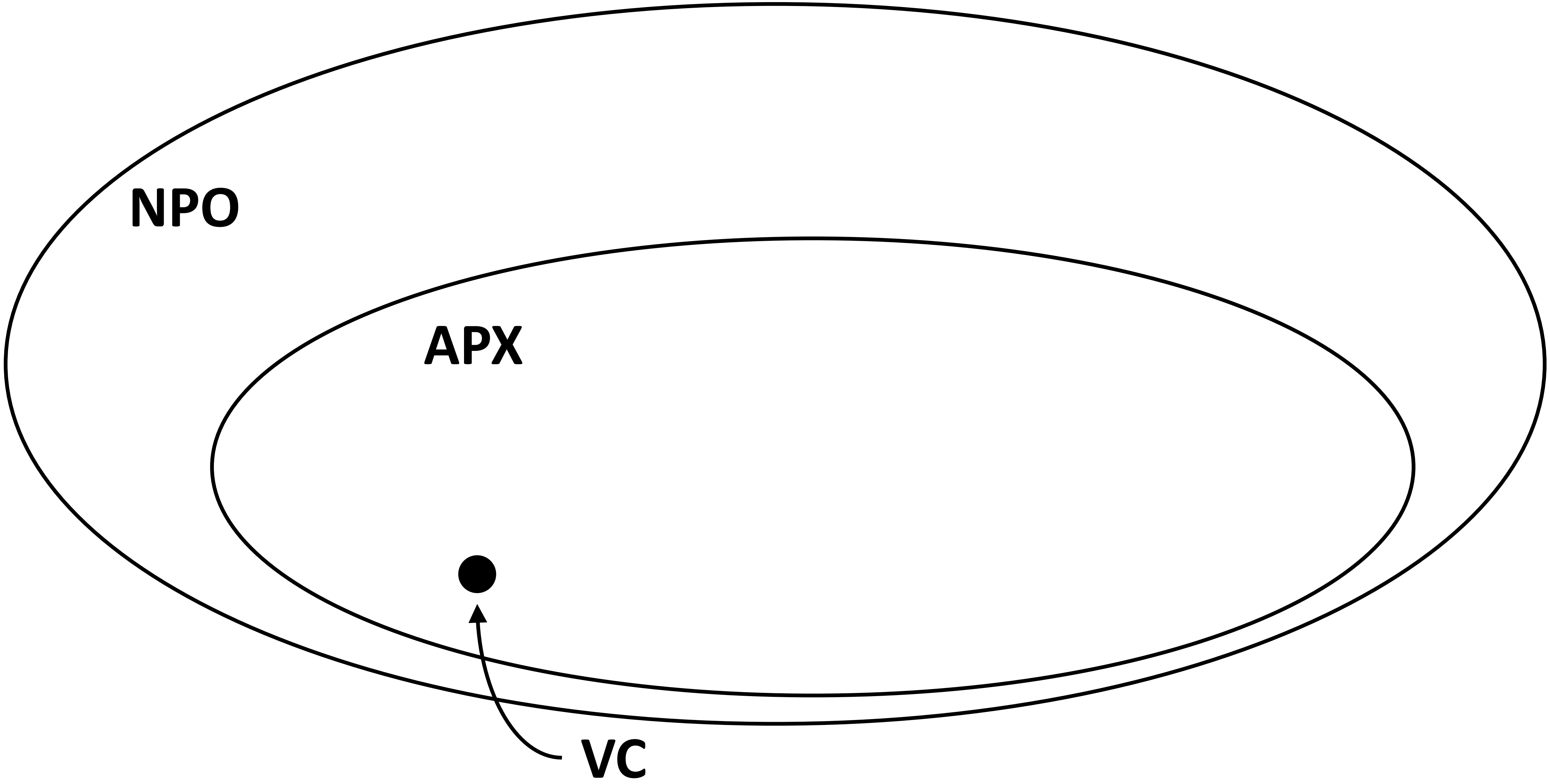
Approximability Hierarchy

NPO - Optimization versions of problems in NP.

Approximability Hierarchy



Approximability Hierarchy



Set Cover

Set Cover: Given a set of elements (the universe), and sets containing those elements, find the smallest number of sets so that every element of the universe is included.

Example:

Set Cover

Set Cover: Given a set of elements (the universe), and sets containing those elements, find the smallest number of sets so that every element of the universe is included.

Example:

$$U = \{1, 4, 7, 8, 10\}$$

$$S = \{\{1, 7, 8\}, \{1, 4, 7\}, \{7, 8\}, \{4, 8, 10\}\}$$

Set Cover

Set Cover: Given a set of elements (the universe), and sets containing those elements, find the smallest number of sets so that every element of the universe is included.

Example:

$$U = \{1, 4, 7, 8, 10\}$$

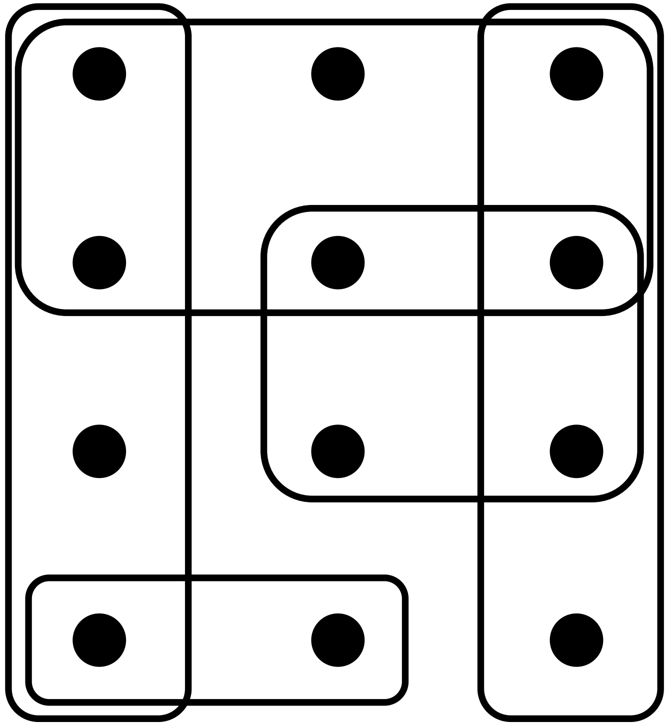
$$S = \{\{1, 7, 8\}, \{1, 4, 7\}, \{7, 8\}, \{4, 8, 10\}\}$$

$\{\{1, 7, 8\}, \{4, 8, 10\}\}$ ✓ $\quad$ $\{\{1, 4, 7\}, \{7, 8\}\}$ ✗

Set Cover

Set Cover: Given a set of elements (the universe), and sets containing those elements, find the smallest number of sets so that every element of the universe is included.

Example:



Set Cover

Set Cover: Given a set of elements (the universe), and sets containing those elements, find the smallest number of sets so that every element of the universe is included.

Algorithm:

?

Set Cover

Set Cover: Given a set of elements (the universe), and sets containing those elements, find the smallest number of sets so that every element of the universe is included.

Greedy Algorithm:

```
while element of universe not included
  select S_i with largest number of excluded elements.
```

Set Cover

Suppose the universe contains n elements.

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

$$ALG \leq \alpha\, OPT$$

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

$$ALG \leq \alpha\, OPT$$

Game Plan:

Bound the maximum number of sets in ALG by…

Bounding the maximum number of iterations of the algorithm by...

Bounding the size of each set added by the algorithm.

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

What can we say about the first set selected?

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

What can we say about the first set selected?

It's the biggest!

At each iteration, we cover the largest number of uncovered elements, and all the elements are uncovered in the first iteration.

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

What can we say about the first set selected?

It’s the biggest!

$$\textbf{?} \quad \leq |\textbf{Biggest Set}| \leq \textbf{?}$$

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

What can we say about the first set selected?

It's the biggest!

? $\leq$ |Biggest Set| $\leq$?

Which do we care about?

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

What can we say about the first set selected?

It's the biggest!

$$? \leq |\textbf{Biggest Set}| \leq ?$$

Guarantee we do at least this good.

Guarantee we don't do better than this.

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

What can we say about the first set selected?

It's the biggest!

$$\frac{\boldsymbol{n}}{\boldsymbol{OPT}} \leq |\textbf{Biggest Set}|$$

Why?

Set Cover

ALG = # sets selected by the algorithm to cover all n elements.

OPT = # sets in an optimal solution to cover all n elements.

Suppose the universe contains n elements.

What can we say about the first set selected?

It's the biggest!

$$\frac{\boldsymbol{n}}{\boldsymbol{OPT}} \leq |\textbf{Biggest Set}|$$

What if each set had fewer than $\frac{n}{OPT}$ elements in it?

Then OPT of those sets would cover fewer than n elements: