Quick review

directed graph w/ $s, t \in V$

$c: E \to \mathbb{R}^{\geq 0}$ – capacities on edges

What are the inputs to the max flow problem?
How fast can max flow be solved?

Kyle + Tara

$O(VE)$ 1950s

$O(E^{1+o(1)})$ 2022

"almost linear"

What does it mean for a problem to be in NP? In P?

Taewen + Daniel

"yes" answer is verifiable in poly time

solvable in poly time

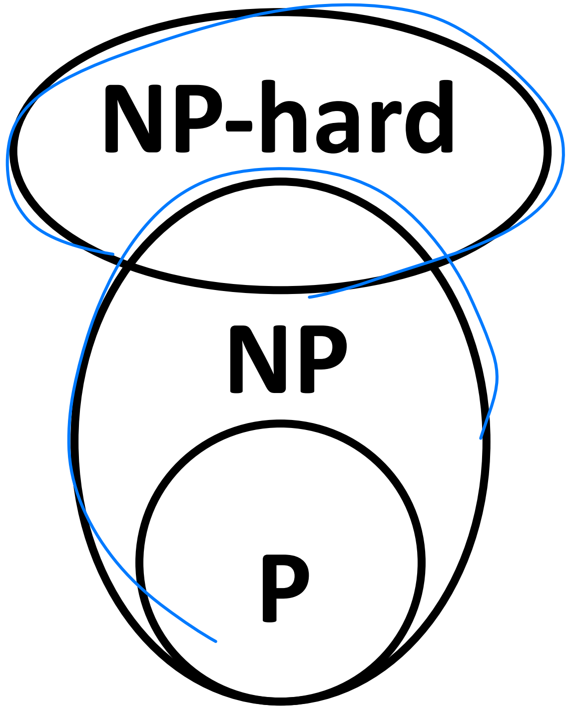
What does it mean for a problem to be NP-hard?

Kaleb + Weslee

If solvable in poly time, then $P = NP$

Handling NP-Hardness

Handling NP-Hardness



Your problem is NP-hard. What to do?

1. Brute Force ← input small
2. Heuristics
3. Approximation Algorithms

Approximation Algorithms

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

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Approximation Algorithms

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

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example: If my CheapestEggsInMissoula algorithm is a 1.25-approximation algorithm, the cost of the eggs it finds is at most 1.25 times the optimal cost.

Approximation Algorithms

$5 $\qquad\qquad$ $4.00

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

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example: If my CheapestEggsInMissoula algorithm is a 1.25-approximation algorithm, the cost of the eggs it finds is at most 1.25 times the optimal cost.

I.e. If cheapest eggs in Missoula are $4.00/dozen, CheapestEggsInMissoula will find eggs that is at most $ _____ /dozen.

Approximation Algorithms

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

Cost (size) of algorithm’s solution.

Approximation Ratio

Cost (size) of optimal solution.

Example: If my CheapestEggsInMissoula algorithm is a 1.25-approximation algorithm, the cost of the eggs it finds is at most 1.25 times the optimal cost.

I.e. If cheapest eggs in Missoula are \$4.00/dozen, CheapestEggsInMissoula will find eggs that is at most \$5.00/dozen.

Approximation Algorithms

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

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example:

Approximation Algorithms

$$ALG \leq \alpha\ OPT$$

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

$$ALG \leq \alpha \; OPT$$

ALG: Cost (size) of algorithm's solution.

α: Approximation Ratio

OPT: Cost (size) of optimal solution.

Example:

- Suppose I know my algorithm is a 1.12-approximation algorithm. α
- Suppose my algorithm returns a solution of cost (size) 746.125. ALG

What do I know about OPT?

$746.125 \leq 1.12 \; OPT \Rightarrow 666 \leq OPT$

Approximation Algorithms

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

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 (size) 746.125.

What do I know about OPT?

Approximation Algorithms

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

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 (size) 746.125.

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

Approximation Algorithms

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

Cost (size) of algorithm's solution.

Approximation Ratio

Cost (size) of optimal solution.

Example:

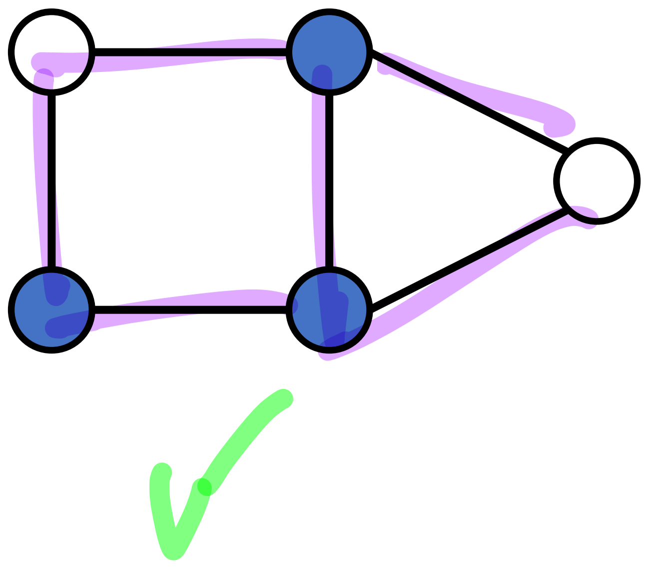
- Suppose I know my algorithm is a 1.12-approximation algorithm.
- Suppose my algorithm returns a solution of cost (size) 746.125.

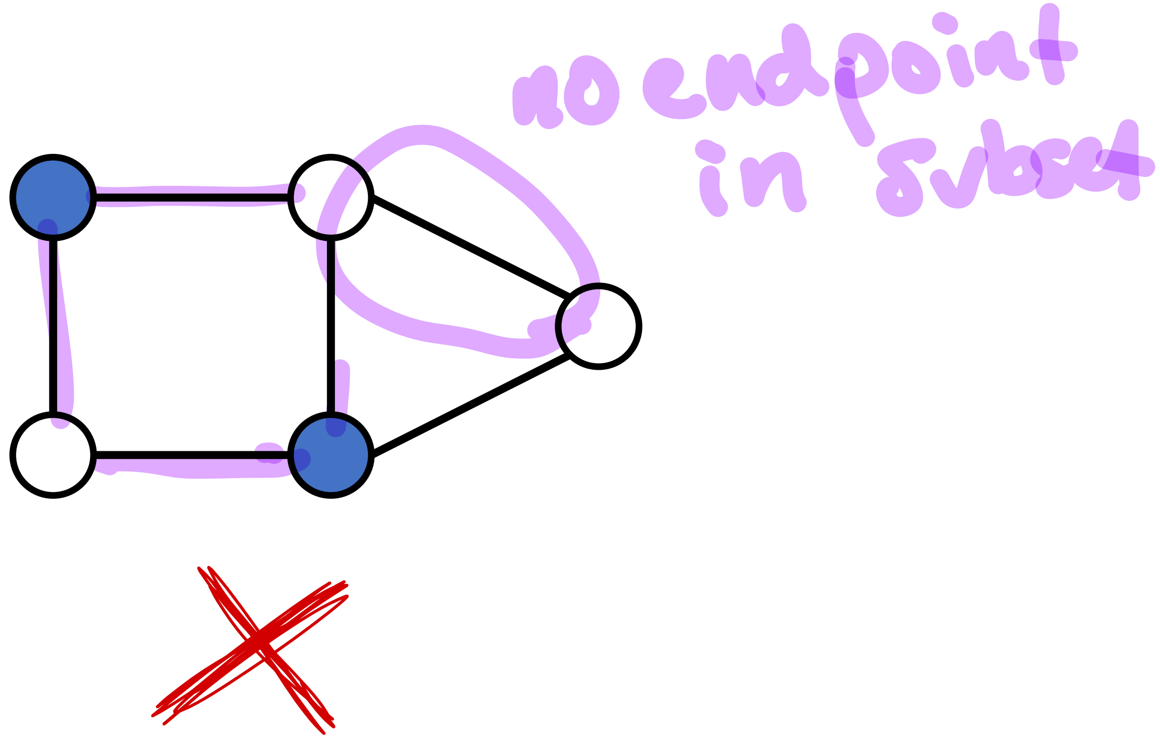
Then, I know that $746.125 \leq 1.12$ OPT

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

Vertex Cover – Problem

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. 3





Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

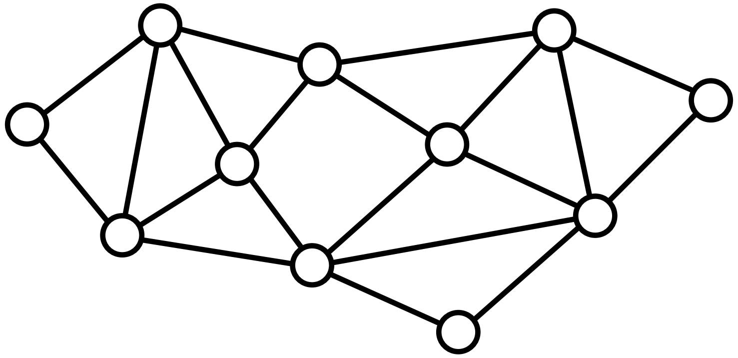
```
VC 2-approximation algorithm:
   while uncovered edge exists
      select both vertices from uncovered edge
```

Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
   while uncovered edge exists
      select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges.

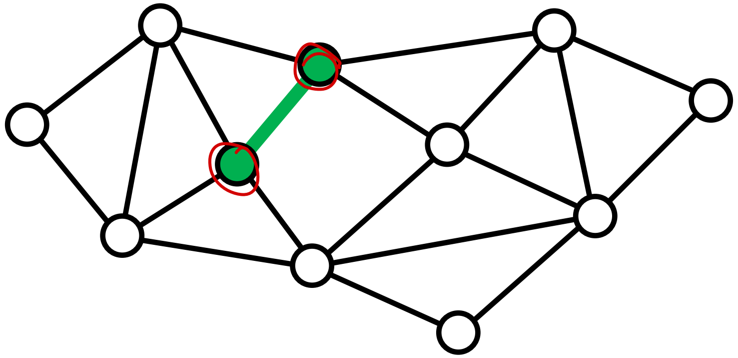


Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges.

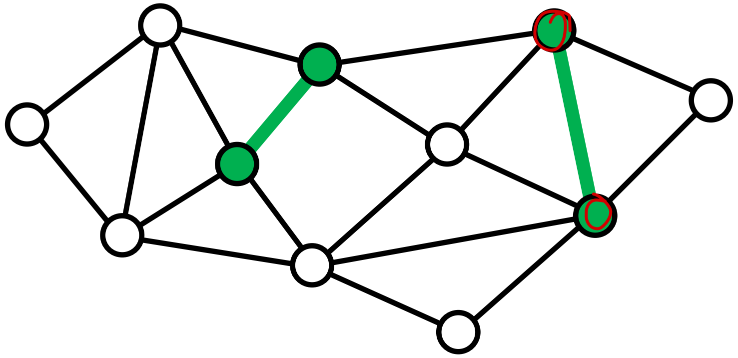


Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges.

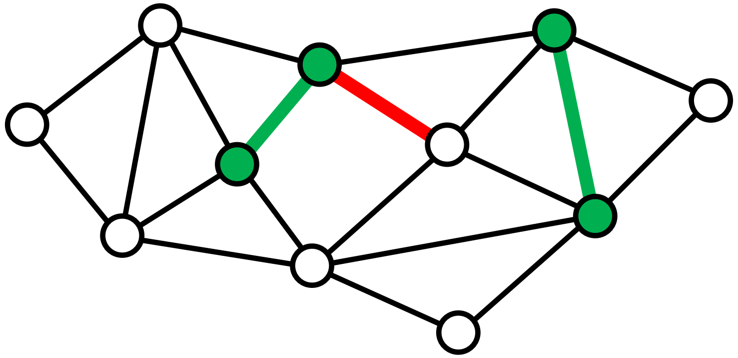


Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges.

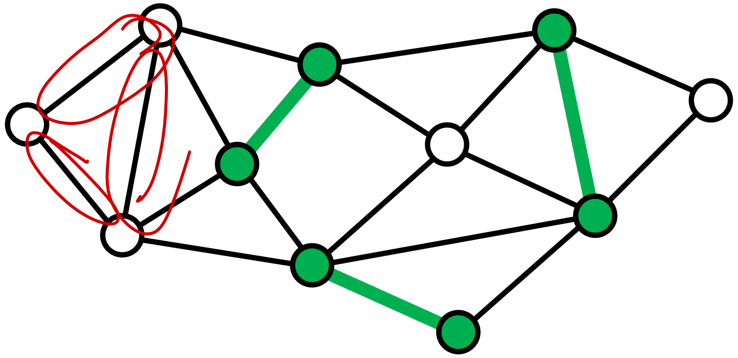


Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
   while uncovered edge exists
      select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges.



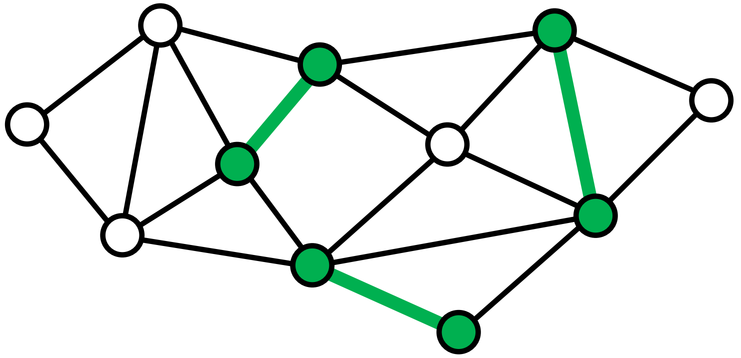
Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
   while uncovered edge exists
      select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges.

Which edge gets selected next?



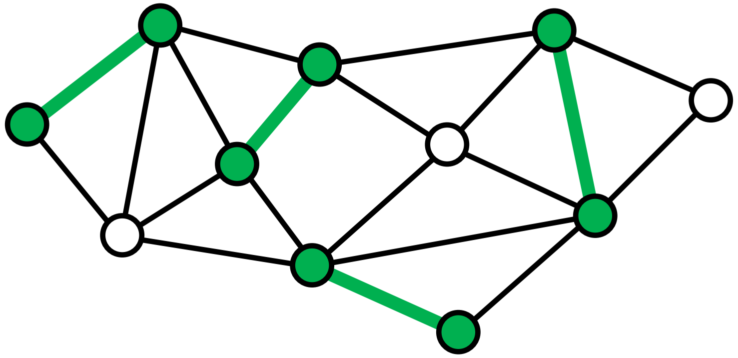
Vertex Cover

$\alpha = 2$

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges.

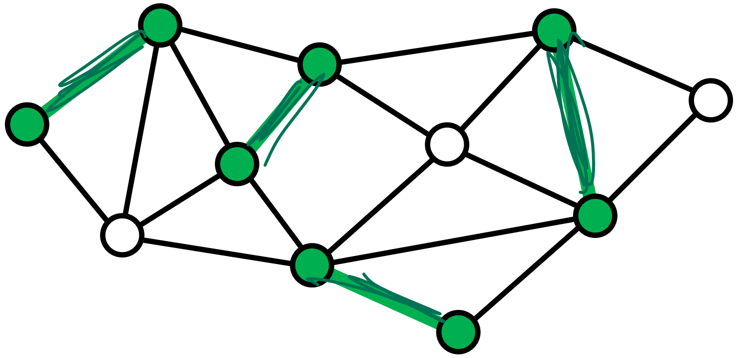


Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let ***E'*** be the edges selected by the algorithm.



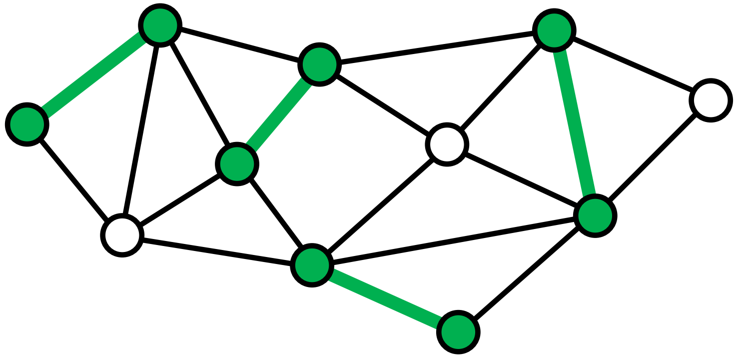
Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let ***E'*** be the # of edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = ??



Vertex Cover

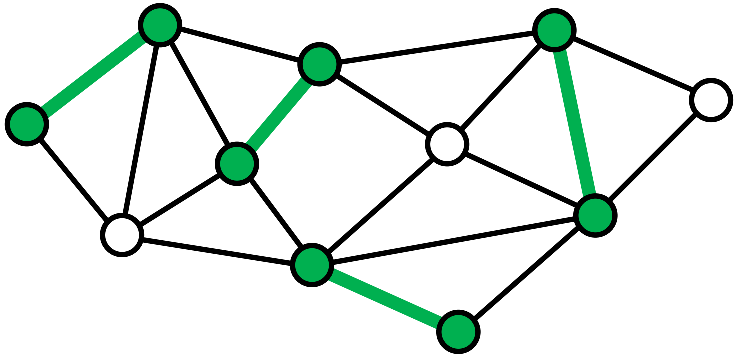
Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
   while uncovered edge exists
      select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let E' be the edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = ??

Discuss with a partner



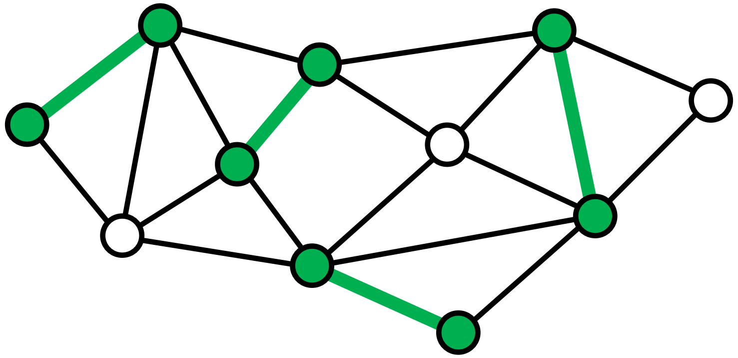
Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let E' be the edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = 2 $|E'|$



Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

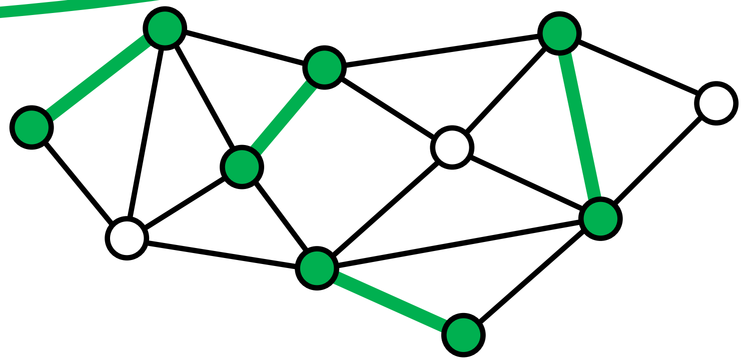
An edge is uncovered if it does not share vertices with any previously selected edges. Let ***E'*** be the edges selected by the algorithm.

⇒ # vertices selected by algorithm = ALG = 2 |***E'***|

A vertex from each edge in ***E'*** must be part of *every* vertex cover.

True or False?

Discuss with a ~~partner~~ table



Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

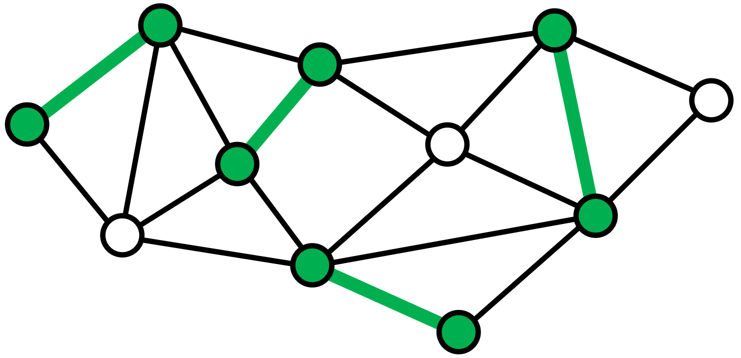
```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let ***E'*** be the edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = 2 |***E'***|

A vertex from each edge in ***E'*** must be part of *every* 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

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

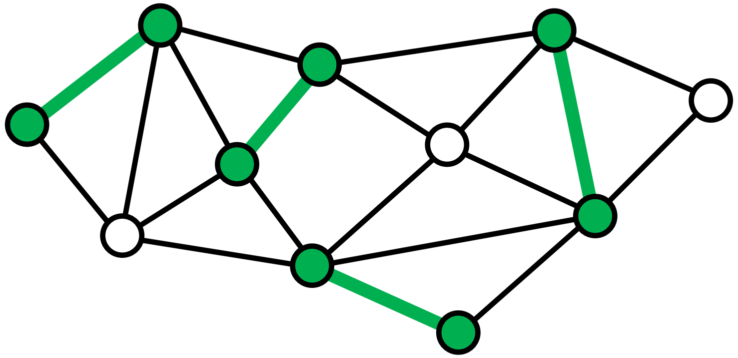
```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let ***E'*** be the edges selected by the algorithm.

⇒ # vertices selected by algorithm = ALG = 2 |***E'***|

A vertex from each edge in ***E'*** must be part of *every* vertex cover.

⇒ In relation to OPT??



Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

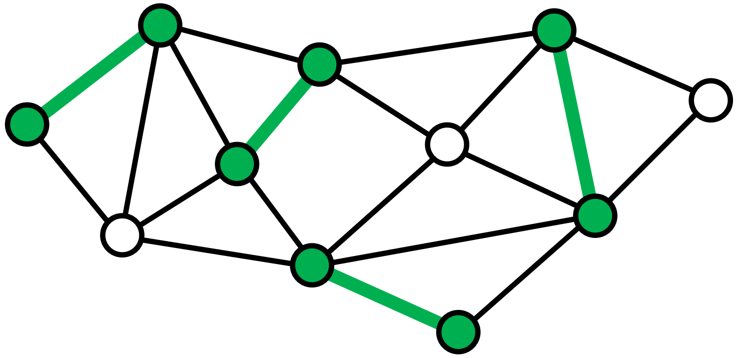
```
VC 2-approximation algorithm:
   while uncovered edge exists
      select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let E' be the edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = 2 $|E'|$

A vertex from each edge in E' must be part of *every* vertex cover.

$\Rightarrow |E'| \leq$ OPT



Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

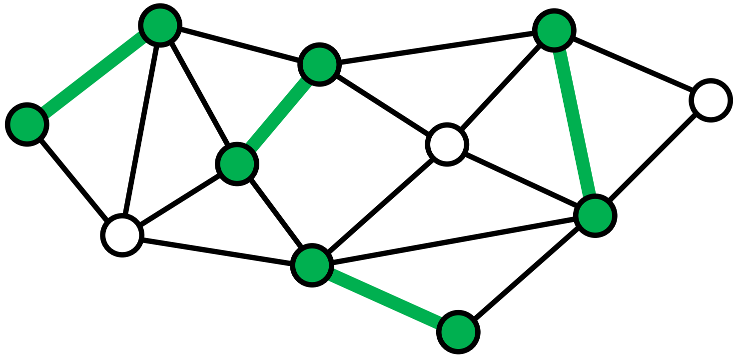
```
VC 2-approximation algorithm:
   while uncovered edge exists
      select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let E' be the edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = 2 $|E'|$

A vertex from each edge in E' must be part of *every* vertex cover.

$\Rightarrow |E'| \leq$ OPT



…ALG ≤ α OPT??

Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

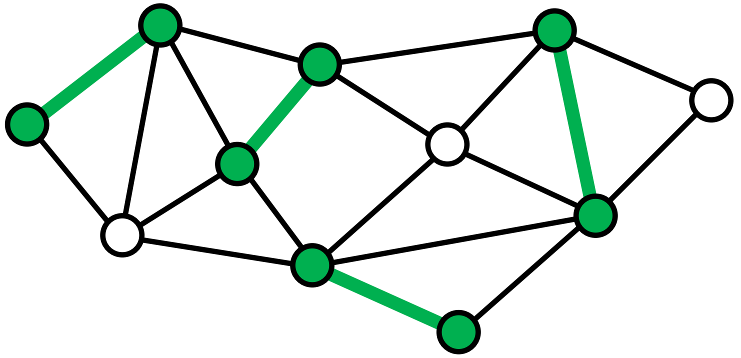
```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let E' be the edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = 2 $|E'|$

A vertex from each edge in E' must be part of <u>*every*</u> vertex cover.

$\Rightarrow |E'| \leq$ OPT



Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

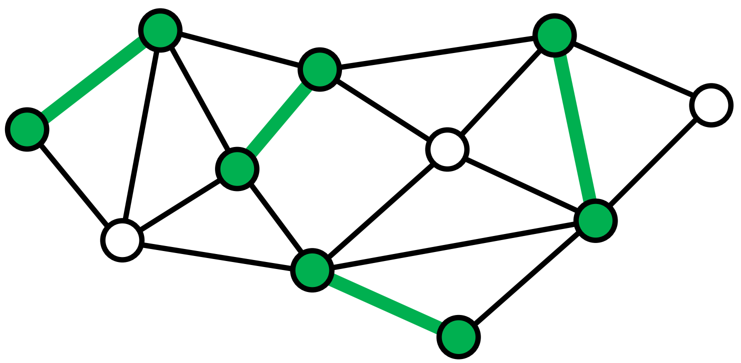
```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let E' be the edges selected by the algorithm.

$\Rightarrow$ # vertices selected by algorithm = ALG = 2 $|E'|$

A vertex from each edge in E' must be part of *every* vertex cover.

$$\Rightarrow |E'| \leq \text{OPT}$$
$$\Rightarrow 2|E'| \leq 2\text{OPT}$$



Vertex Cover

Vertex Cover: Given graph $G = (V, E)$, find smallest $V' \subseteq V$ such that each edge in E contains an end point in V'.

```
VC 2-approximation algorithm:
  while uncovered edge exists
    select both vertices from uncovered edge
```

An edge is uncovered if it does not share vertices with any previously selected edges. Let E' be the edges selected by the algorithm.

$$\Rightarrow \text{\# vertices selected by algorithm} = \text{ALG} = 2\,|E'|$$

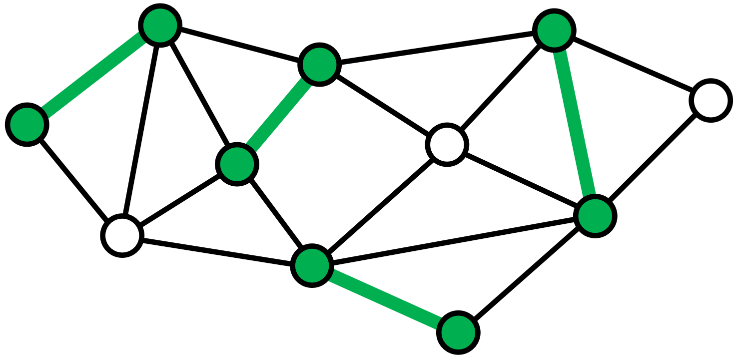
A vertex from each edge in E' must be part of *every* vertex cover.

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

$$\Rightarrow 2|E'| \leq 2\text{OPT}$$

$$\Rightarrow \text{ALG} \leq 2\text{OPT}$$

$\alpha = 2$



Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

$$\Longrightarrow \text{ALG} \leq 2\ \text{OPT}$$

Is this the best this algorithm can do?

Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

$$\Longrightarrow \text{ALG} \leq 2\ \text{OPT}$$

Is this the best this algorithm can do?

I.e. Can we guarantee this algorithm does better than 2 OPT?

Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

$$\Longrightarrow \text{ALG} \leq 2\ \text{OPT}$$

Is this the best this algorithm can do?

no

I.e. Can we guarantee this algorithm does better than 2 OPT?

Is there a graph where this algorithm does exactly 2 OPT?

Which of these would be easier to prove?

Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

$\Longrightarrow \text{ALG} \leq 2\ \text{OPT}$

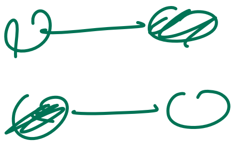
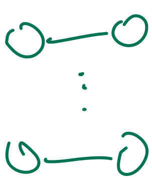
Is this the best this algorithm can do?

I.e. Can we guarantee this algorithm does better than 2 OPT?

Is there a graph where this algorithm does exactly 2 OPT?

Try to find a graph where ALG = 2 OPT for this algorithm

challenge: find a class of graphs for n=2, 4, 6, … where ALG = 2 OPT

Graph	ALG	OPT	α
	2	1	2
	4	2	2
$\vdots$			
			

Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

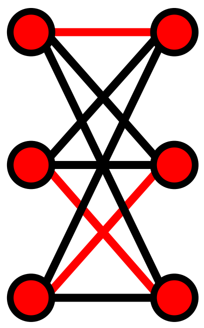
$$\Longrightarrow \text{ALG} \leq 2\ \text{OPT}$$

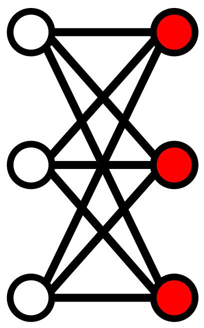
Is this the best this algorithm can do?

I.e. Can we guarantee this algorithm does better than 2 OPT?

Is there a graph where this algorithm does exactly 2 OPT?

Complete Bipartite Graph





Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

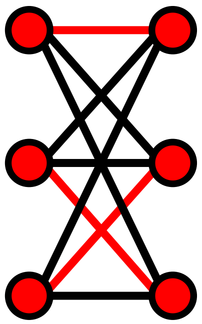
$$\Longrightarrow \text{ALG} \leq 2\ \text{OPT}$$

Is this the best this algorithm can do?

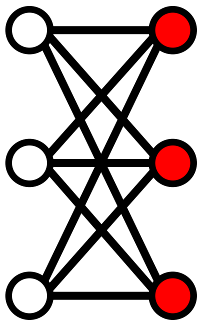
I.e. Can we guarantee this algorithm does better than 2 OPT?

Is there a graph where this algorithm does exactly 2 OPT?

Complete Bipartite Graph



ALG



OPT

$|\text{ALG}| = 2k$: $v \notin \text{ALG} \Rightarrow$ all neighbors are $\Longrightarrow k$ edges selected $\Longrightarrow$ all $2k$ nodes selected.

$|\text{OPT}| = k$: Fewer than k nodes selected $\Longrightarrow \exists$ unselected edge.

Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

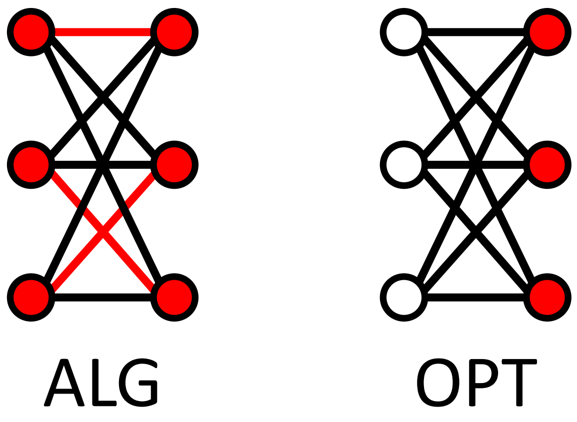
$\Longrightarrow$ ALG $\leq$ 2 OPT

Is this the best this algorithm can do?

I.e. Can we guarantee this algorithm does better than 2 OPT?

Is there a graph where this algorithm does exactly 2 OPT?

Complete
Bipartite Graph



∴ The best Vertex Cover can be approximated is within a factor of 2

Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

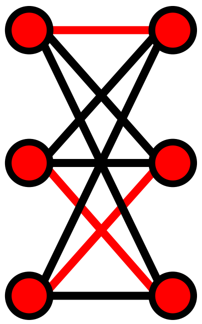
$$\implies \text{ALG} \le 2\ \text{OPT}$$

Is this the best this algorithm can do?

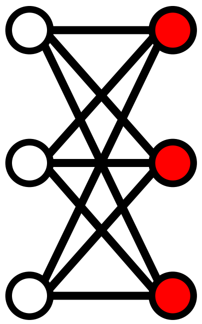
I.e. Can we guarantee this algorithm does better than 2 OPT?

Is there a graph where this algorithm does exactly 2 OPT?

Complete Bipartite Graph



ALG



OPT

The best Vertex Cover can be approximated is within a factor of 2

True or false?

Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

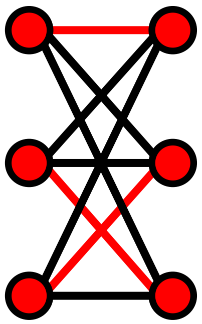
$$\Longrightarrow \text{ALG} \leq 2\ \text{OPT}$$

Is this the best this algorithm can do?

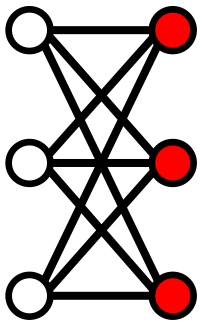
I.e. Can we guarantee this algorithm does better than 2 OPT?

Is there a graph where this algorithm does exactly 2 OPT?

Complete Bipartite Graph



ALG



OPT

~~∴ The best Vertex Cover can be approximated is within a factor of 2~~

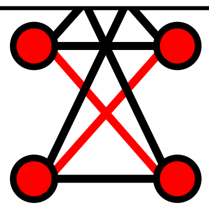
Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

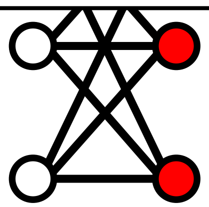
VC Inapproximability:

- Cannot be approximated within a factor of 1.3606 unless P=NP.

Complete Bipartite Graph



ALG



OPT

~~can be approximated is within a factor of 2~~

Vertex Cover – Improvement

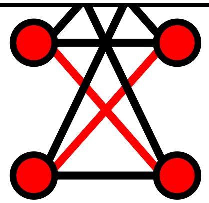
```
while uncovered edge exists
   select both vertices from uncovered edge
```

VC Inapproximability:

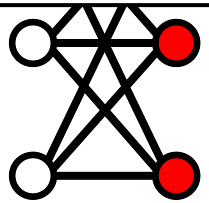
- Cannot be approximated within a factor of 1.3606 unless P=NP.

How do you think we prove this?

Complete
Bipartite Graph



ALG



OPT

~~can be approximated is within a factor of 2~~

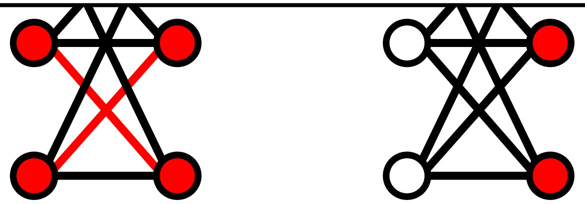
Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

VC Inapproximability:

- Cannot be approximated within a factor of 1.3606 unless P=NP.
- Cannot be approximated within any constant factor better than 2 unless the Unique Games Conjecture is false.

Complete Bipartite Graph



ALG OPT

~~can be approximated is within a factor of 2~~

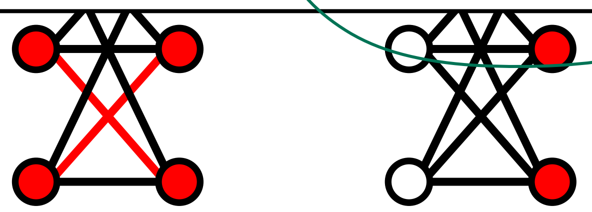
Vertex Cover – Improvement

```
while uncovered edge exists
   select both vertices from uncovered edge
```

VC Inapproximability:

- Cannot be approximated within a factor of 1.3606 unless P=NP.
- Cannot be approximated within any constant factor better than 2 unless the Unique Games Conjecture is false.
- Is approximable within $2 - \frac{\log \log |V|}{2 \log |V|}$.

Complete Bipartite Graph



ALG OPT

~~can be approximated is within a factor of 2~~