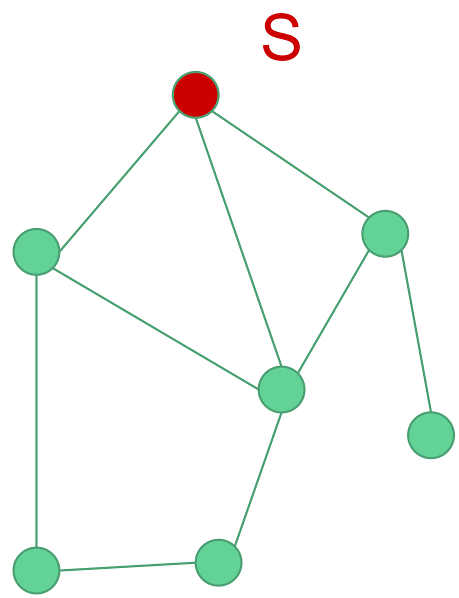
Parallel Maximal Independent Set

Comp 5703 - Edward Duong

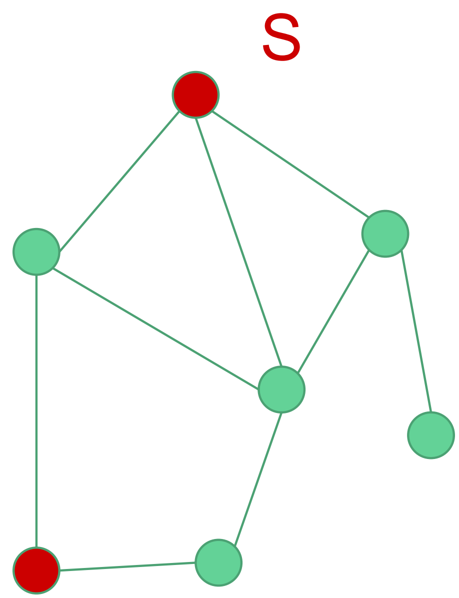
Independent Set

A subset S of vertices. No two vertices the are connected by an edge in G.



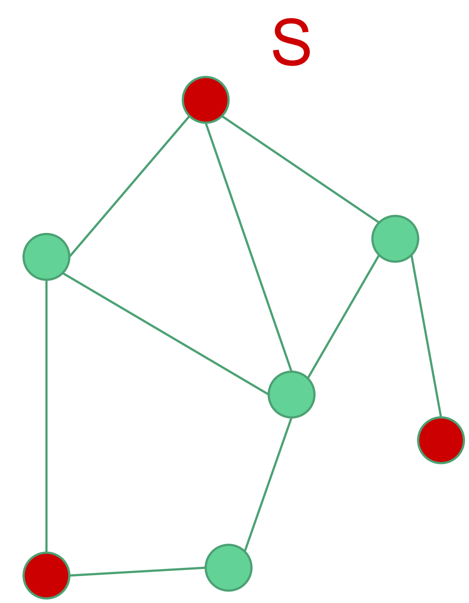
Independent Set

A subset S of vertices. No two vertices the are connected by an edge in G.



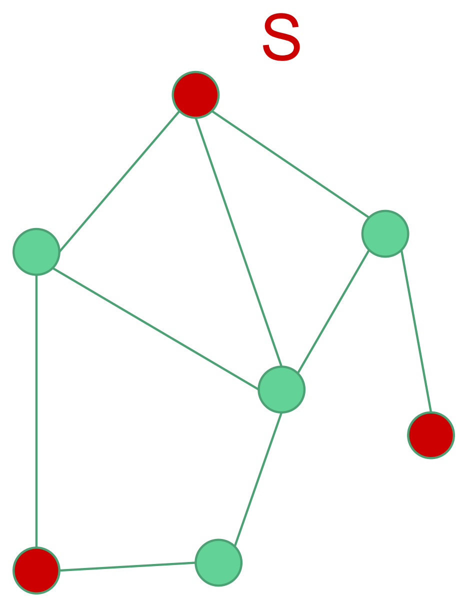
Independent Set

A subset S of vertices. No two vertices the are connected by an edge in G.



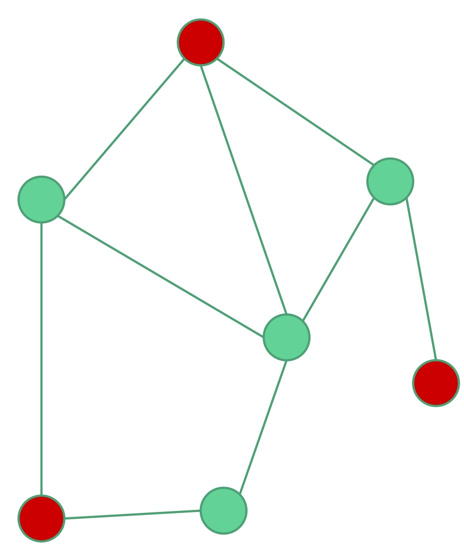
Maximal Independent Set (MIS)

- Not subset of any independent set



Maximal Independent Set (MIS)

- Not subset of any independent set
- Each **edge** has an endpoint not in S
- Each **vertex** is in S or has at least 1 neighbor in S

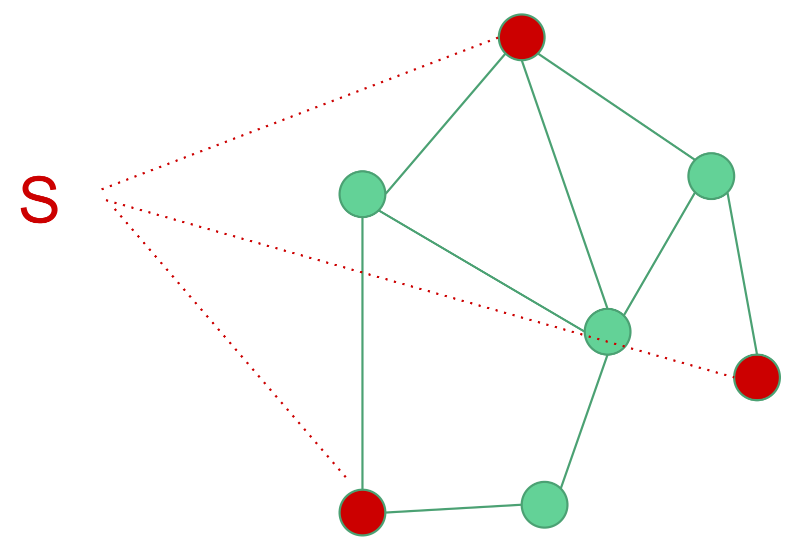


Maximal Independent Set (MIS)

Given $G = (V, E)$, a maximal independent set $S \subset V$ satisfies the conditions:

$$N(S) \cup S = V \text{ and } N(S) \cap S = \varnothing$$

$N(S)$ denotes the vertex neighbors of the set S

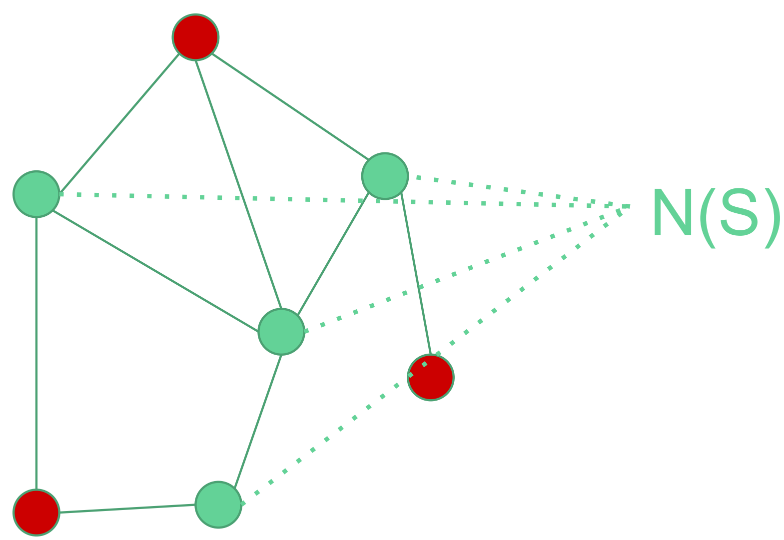


Maximal Independent Set (MIS)

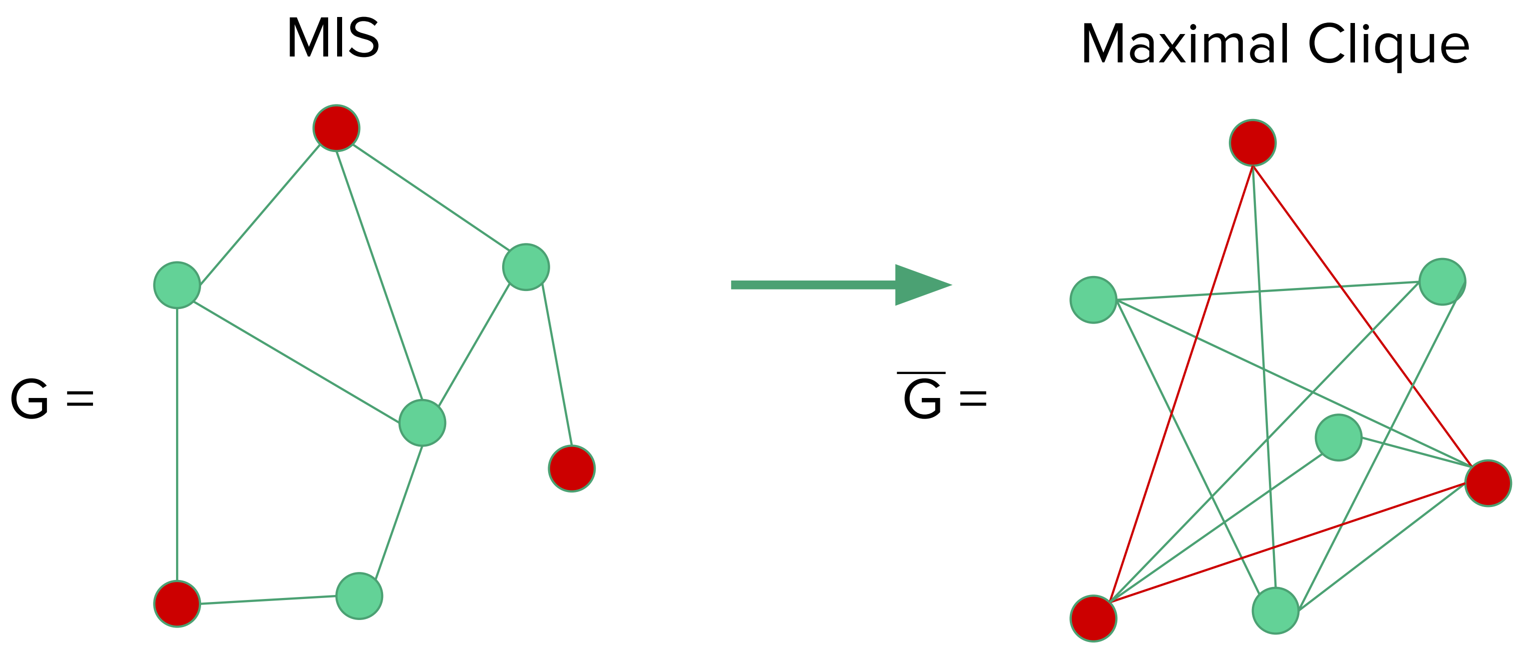
Given G = (V, E), a maximal independent set $S \subset V$ satisfies the conditions:

$$N(S) \cup S = V \text{ and } N(S) \cap S = \emptyset$$

N(S) denotes the vertex neighbors of the set S



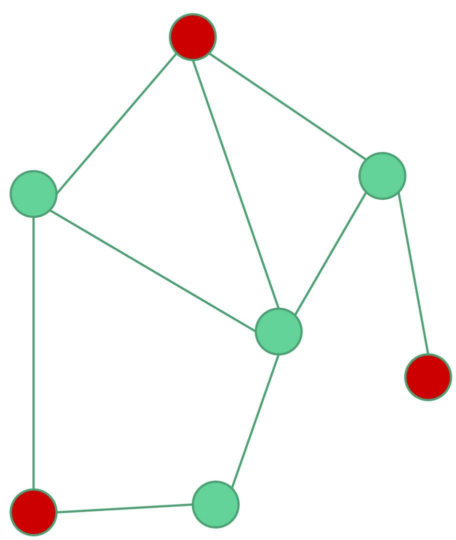
Relationship with Similar Problems



Relationship with Similar Problems

MIS

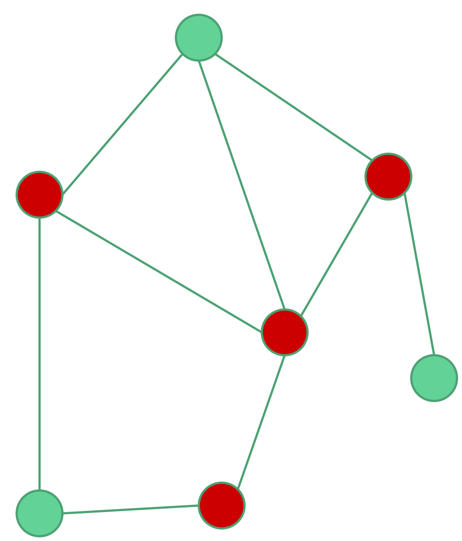
(S, E) =



[Gallai 1959]

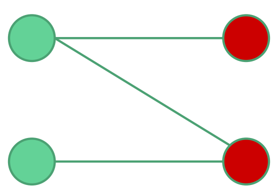
Minimal Vertex Cover

(V / S, E) =



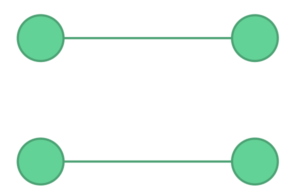
Relationship with Similar Problems

Max IS for
Bipartite Graph



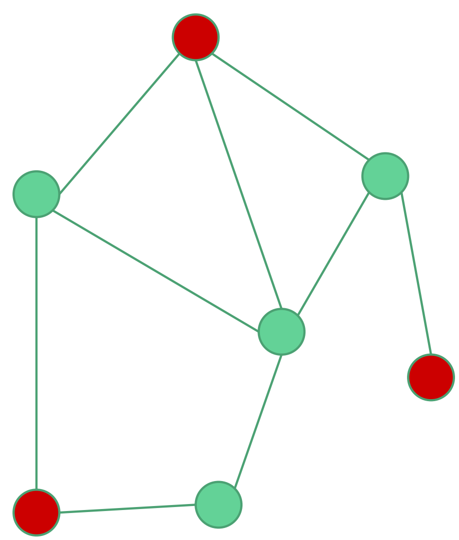
to vertex cover + to
maximum matching
[König 1931]

Maximum Matching
for Bipartite Graph



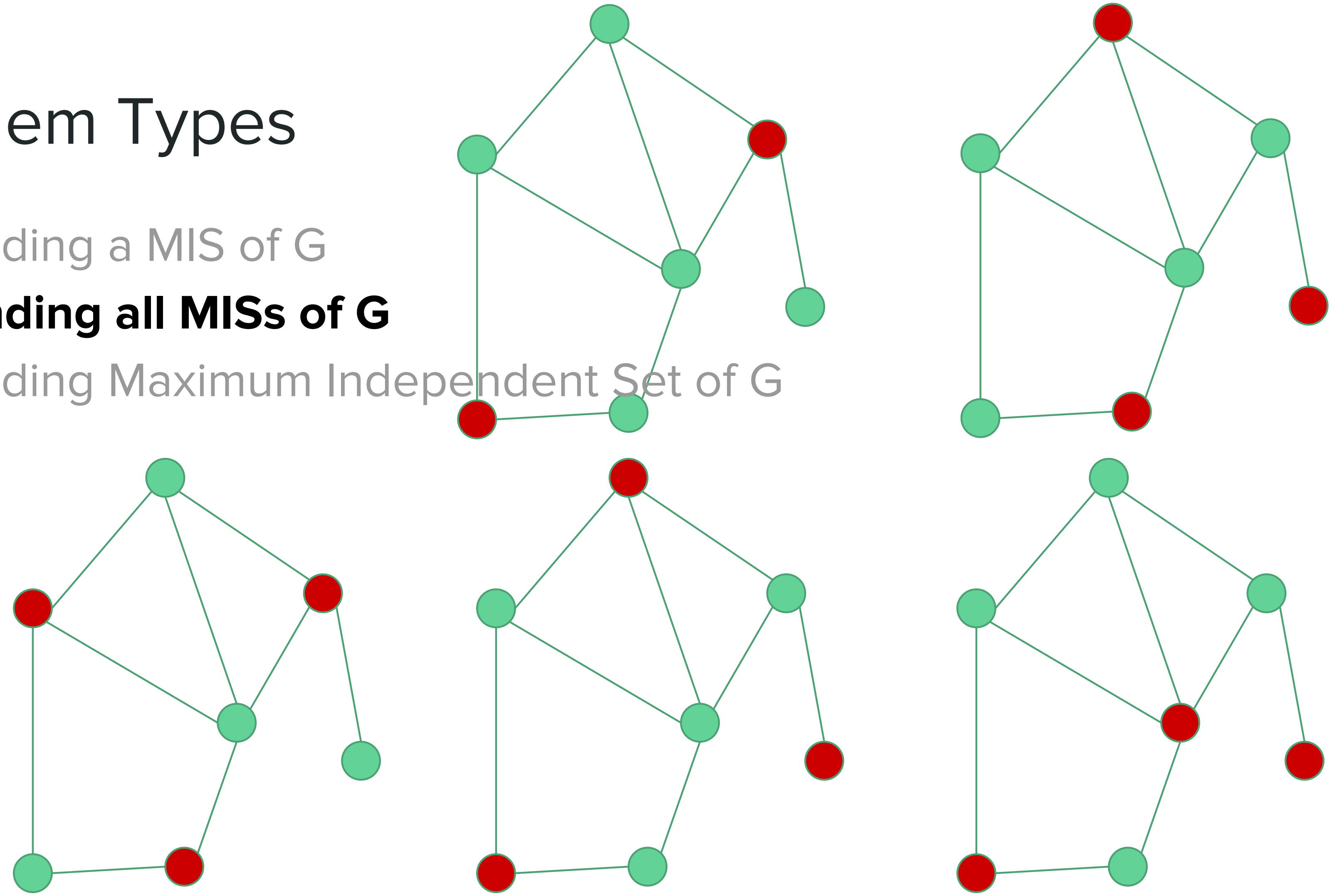
Problem Types

- **Finding a MIS of G**
- Finding all MISs of G
- Finding Maximum Independent Set of G



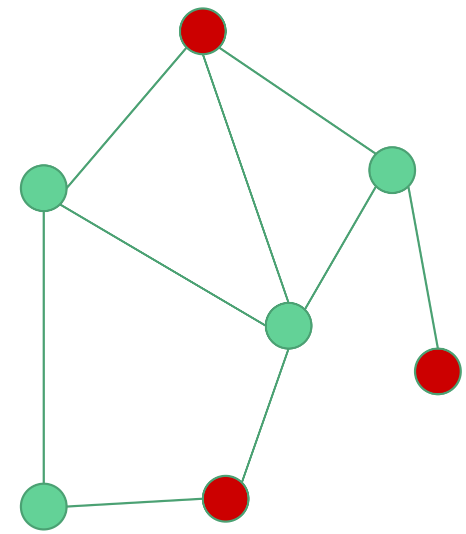
Problem Types

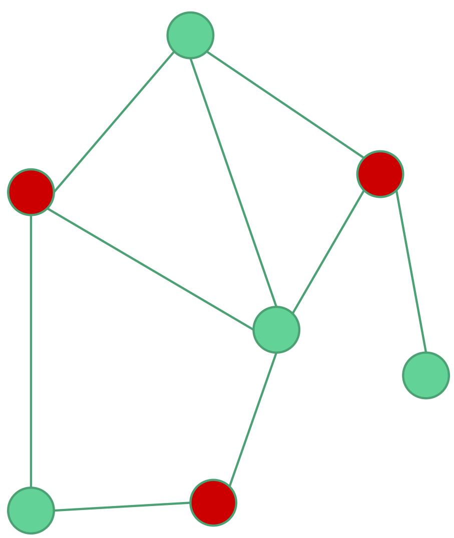
- Finding a MIS of G
- **Finding all MISs of G**
- Finding Maximum Independent Set of G

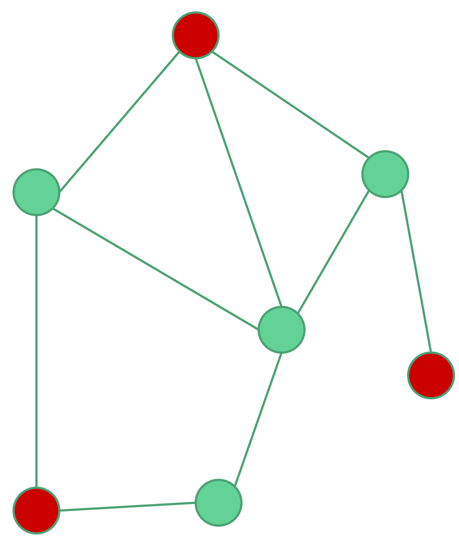


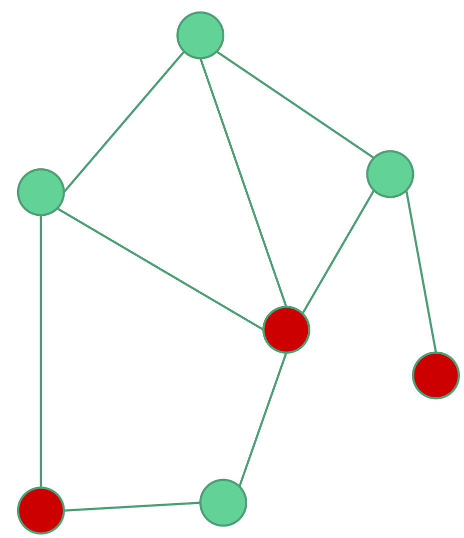
Problem Types

- Finding a MIS of G
- Finding all MISs of G
- **Finding Maximum Independent Set of G**









Finding a MIS of G

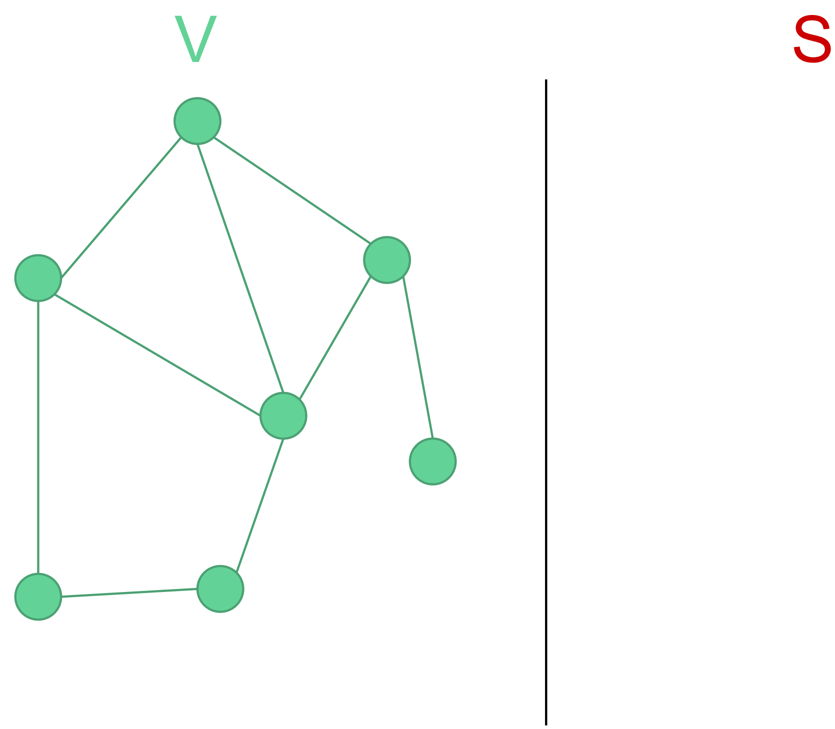
SequentialMIS(G = (V, E))

```
Let S = ∅
While(V ≠ ∅)
        Arbitrarily select v ∈ V
        S = S ∪ v
        V = V \ (v ∪ N(v))
Output S
```

Finding a MIS of G

SequentialMIS(G = (V, E))

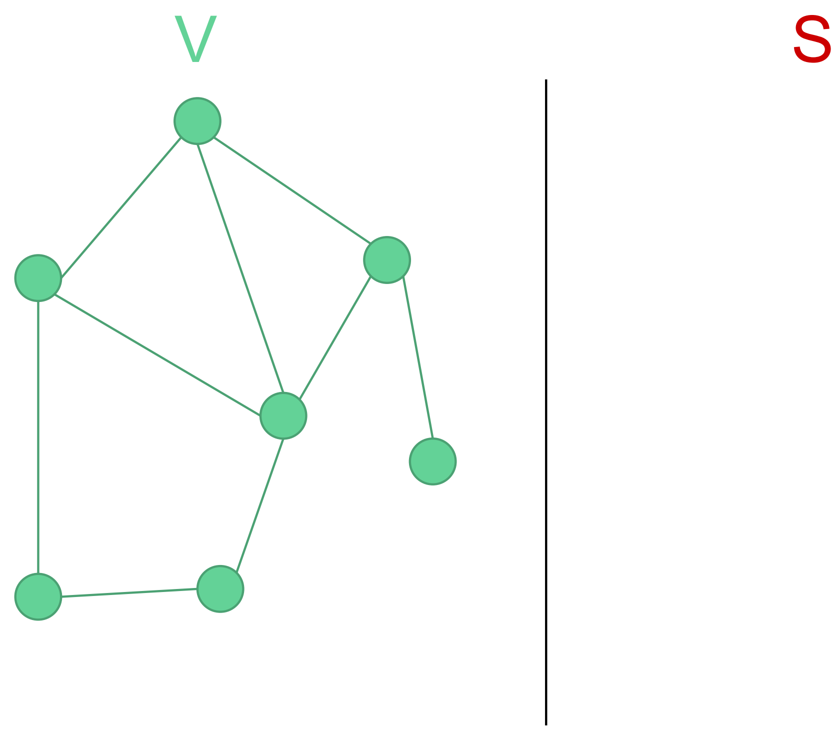
1. **Let S** = ∅
2. While(V ≠ ∅)
3. Arbitrarily select v ∈ V
4. S = S ∪ v
5. V = V \ (v ∪ N(v))
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

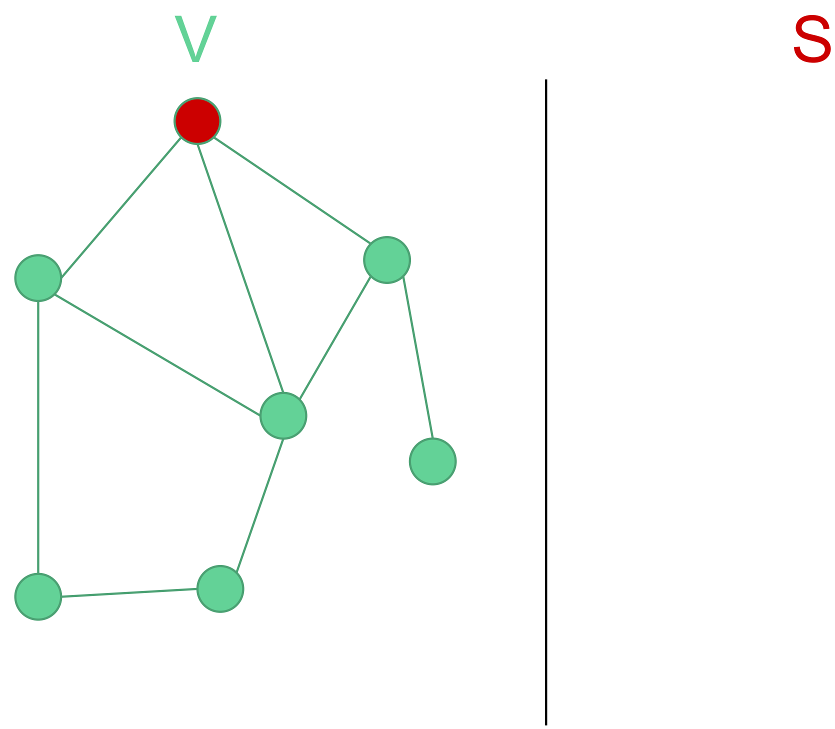
1. Let $S = \emptyset$
2. **While($V \neq \emptyset$)**
3. Arbitrarily select $v \in V$
4. $S = S \cup v$
5. $V = V \setminus (v \cup N(v))$
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

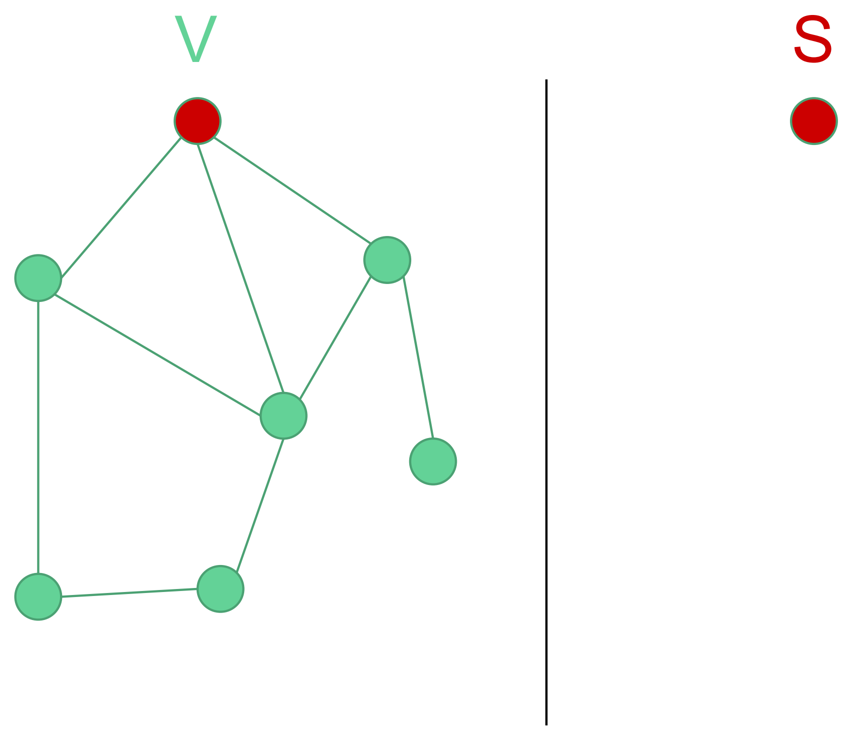
1. Let S = ∅
2. While(V ≠ ∅)
3. **Arbitrarily select v ∈ V**
4. S = S ∪ v
5. V = V \ (v ∪ N(v))
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

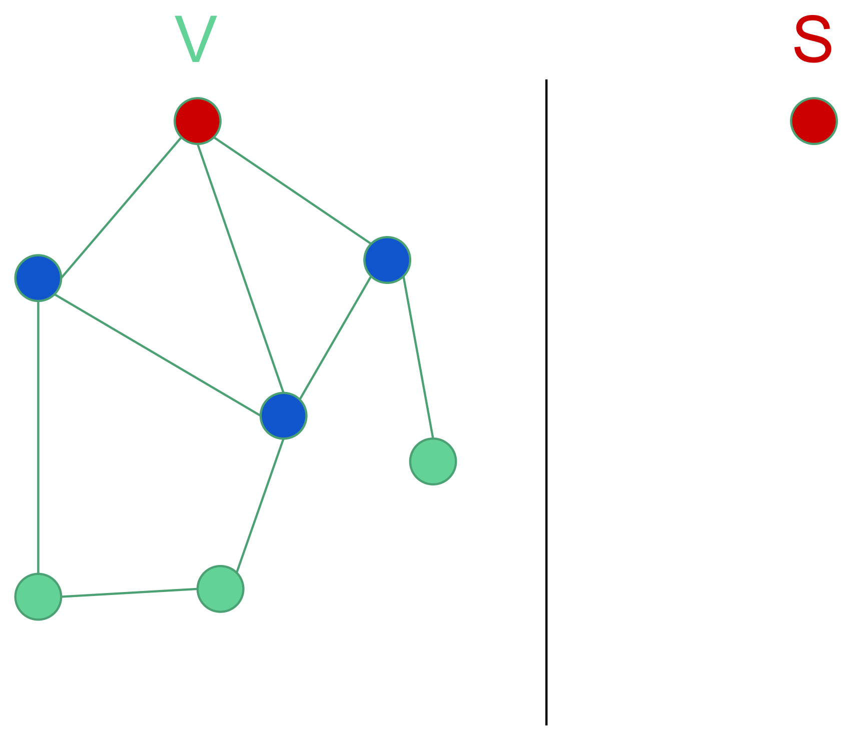
1. Let S = ∅
2. While(V ≠ ∅)
3. Arbitrarily select v ∈ V
4. **S = S** ∪ **v**
5. V = V \ (v ∪ N(v))
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

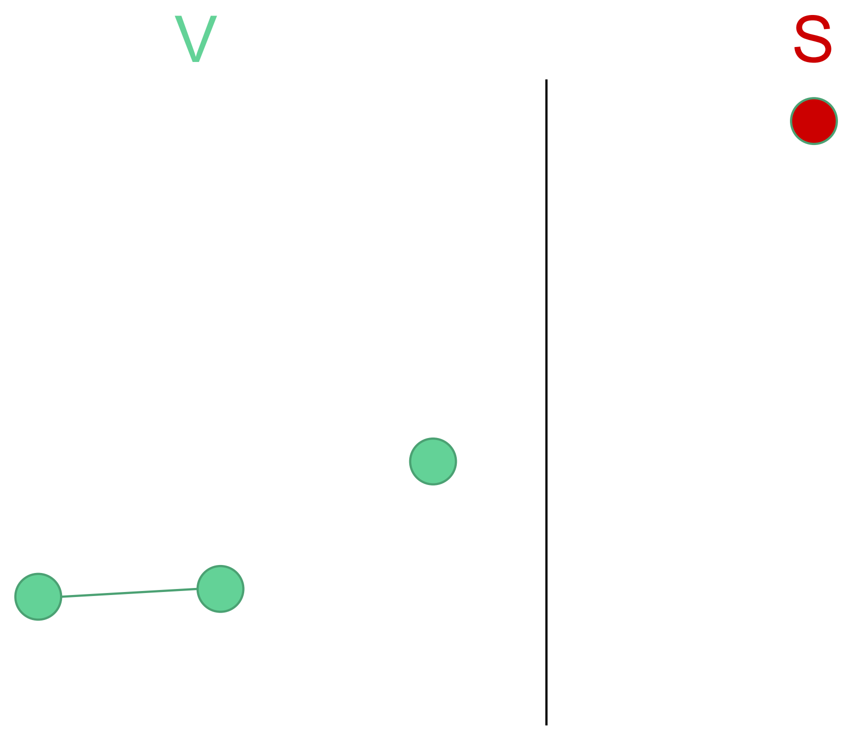
1. Let S = ∅
2. While(V ≠ ∅)
3. Arbitrarily select v ∈ V
4. S = S ∪ v
5. **V = V \ (v ∪ N(v))**
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

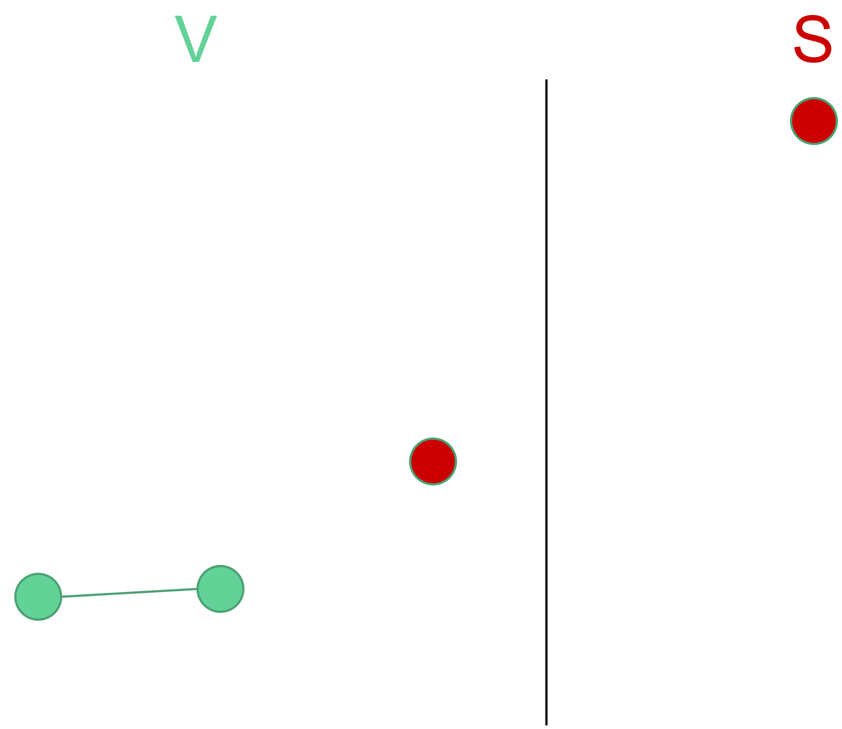
1. Let S = ∅
2. While(V ≠ ∅)
3. Arbitrarily select v ∈ V
4. S = S ∪ v
5. **V = V \ (v ∪ N(v))**
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

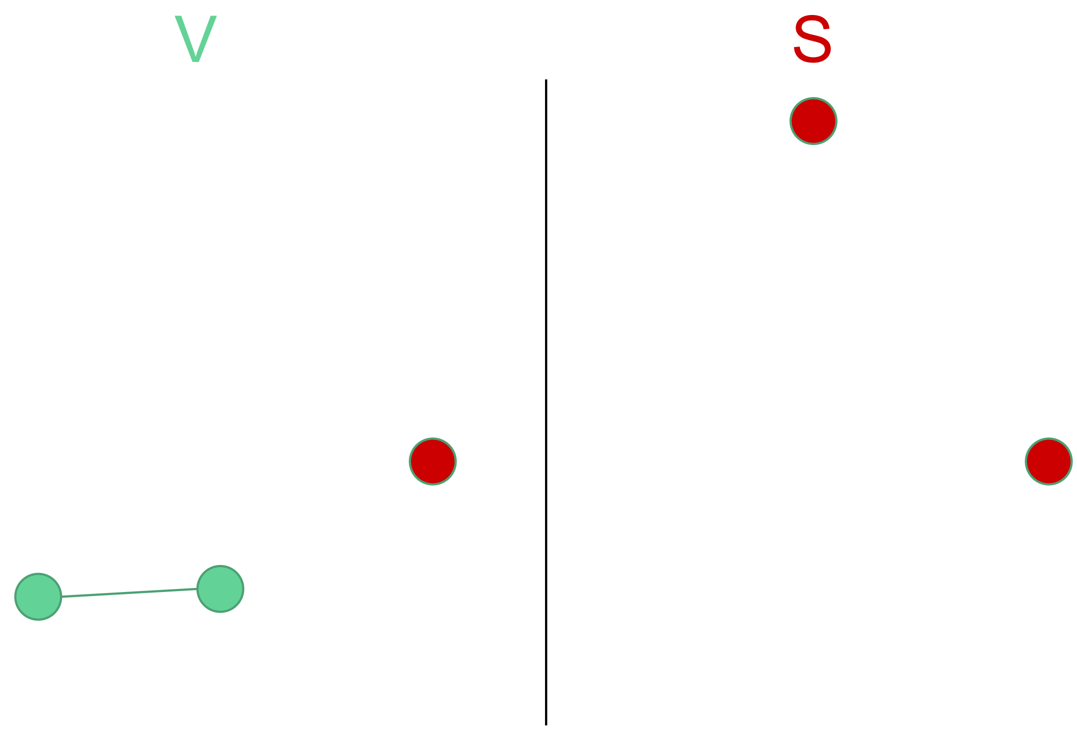
1. Let S = ∅
2. While(V ≠ ∅)
3.   **Arbitrarily select v ∈ V**
4.   S = S ∪ v
5.   V = V \ (v ∪ N(v))
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

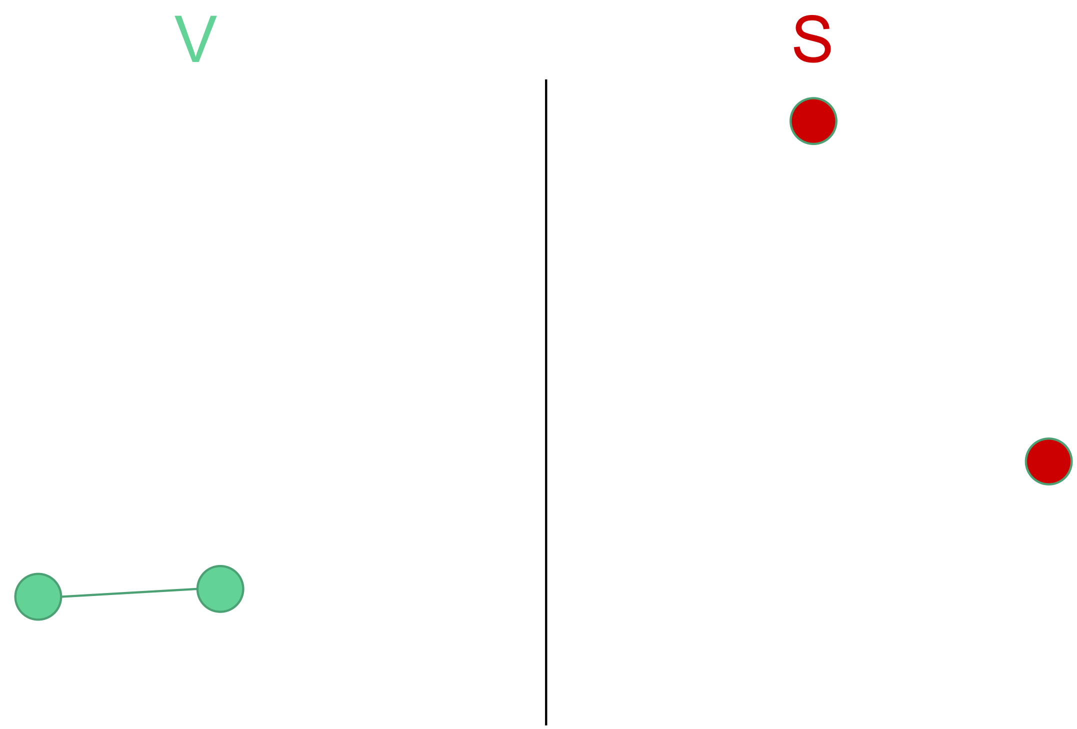
1. Let S = ∅
2. While(V ≠ ∅)
3. Arbitrarily select v ∈ V
4. **S = S** ∪ **v**
5. V = V \ (v ∪ N(v))
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

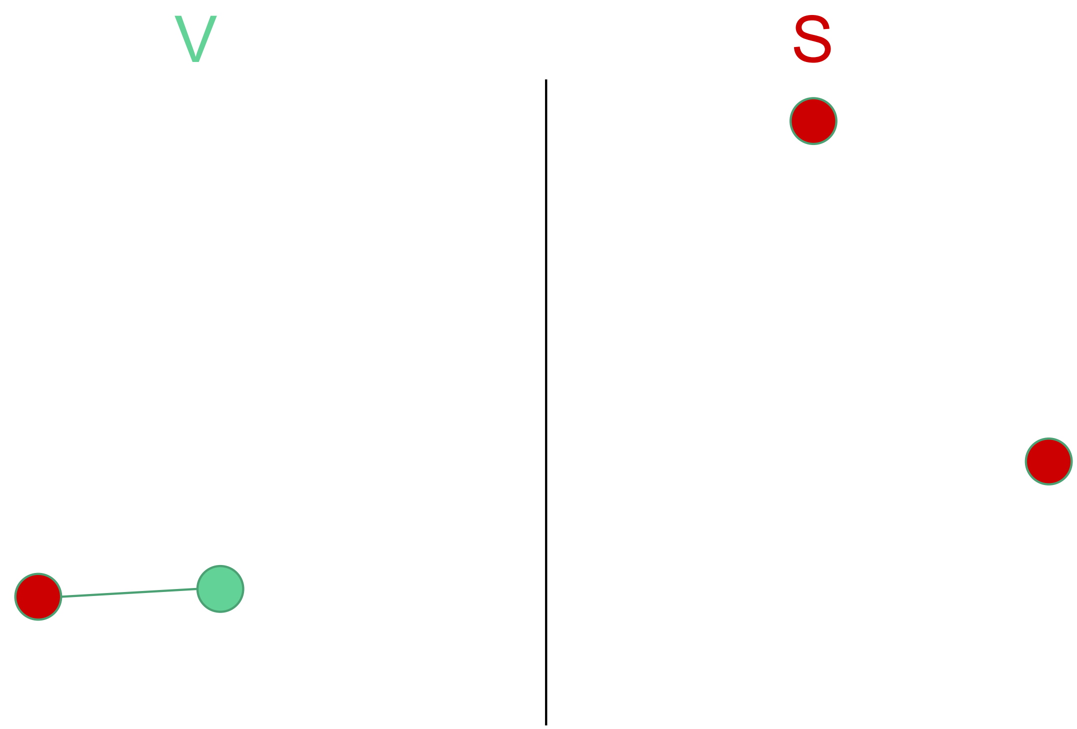
1. Let $S = \emptyset$
2. While($V \neq \emptyset$)
3. Arbitrarily select $v \in V$
4. $S = S \cup v$
5. **$V = V \setminus (v \cup N(v))$**
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

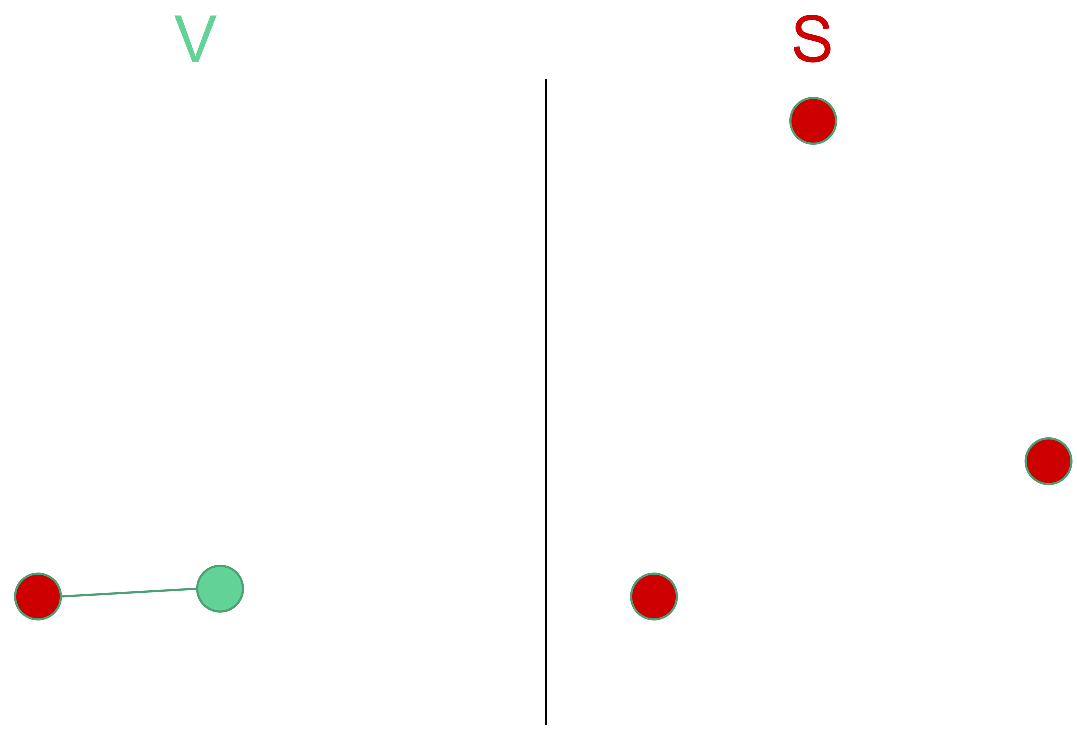
1. Let $S = \emptyset$
2. While($V \neq \emptyset$)
3.   **Arbitrarily select $v \in V$**
4. $S = S \cup v$
5. $V = V \setminus (v \cup N(v))$
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

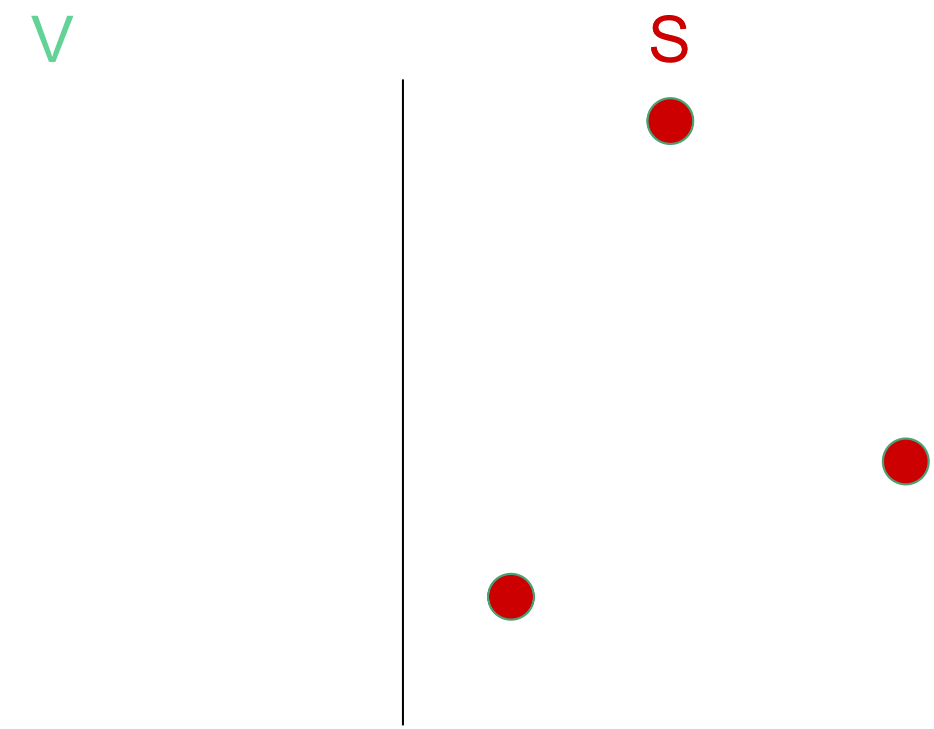
1. Let S = ∅
2. While(V ≠ ∅)
3. Arbitrarily select v ∈ V
4. **S = S** ∪ **v**
5. V = V \ (v ∪ N(v))
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

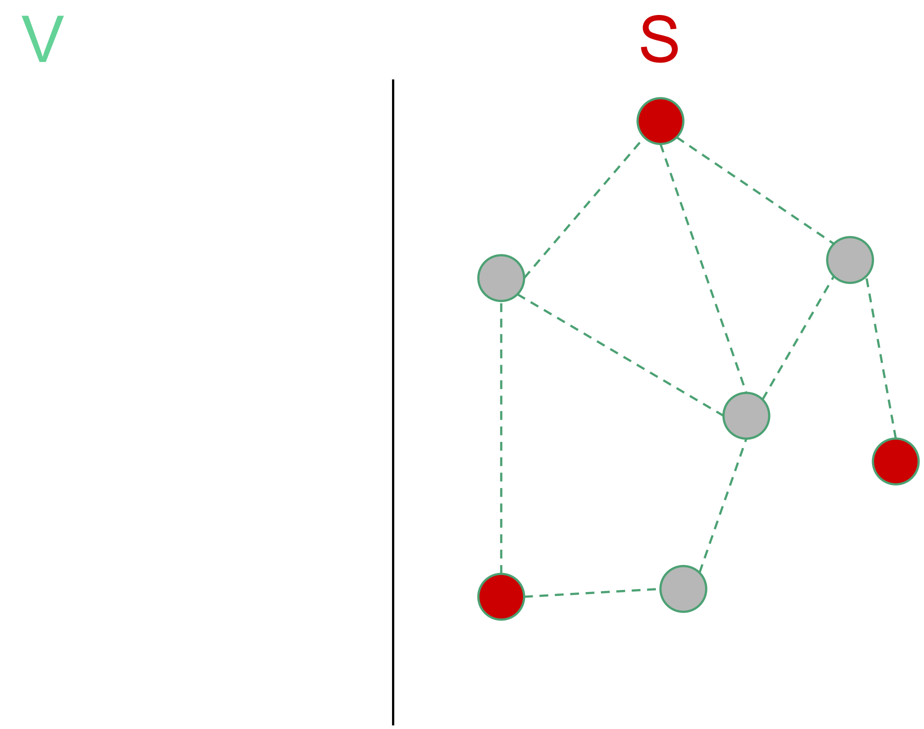
1. Let S = ∅
2. While(V ≠ ∅)
3. Arbitrarily select v ∈ V
4. S = S ∪ v
5. **V = V \ (v ∪ N(v))**
6. Output S



Finding a MIS of G

SequentialMIS(G = (V, E))

1. Let $S = \varnothing$
2. While($V \neq \varnothing$)
3. Arbitrarily select $v \in V$
4. $S = S \cup v$
5. $V = V \setminus (v \cup N(v))$
6. **Output S**



Finding a MIS of G

SequentialMIS(G = (V, E))

```
1.  Let S = ∅
2.  While(V ≠ ∅)
3.          Arbitrarily select v ∈ V
4.          S = S ∪ v
5.          V = V \ (v ∪ N(v))
6.  Output S
```

Runtime:

O(n)

Memory:

O(n)

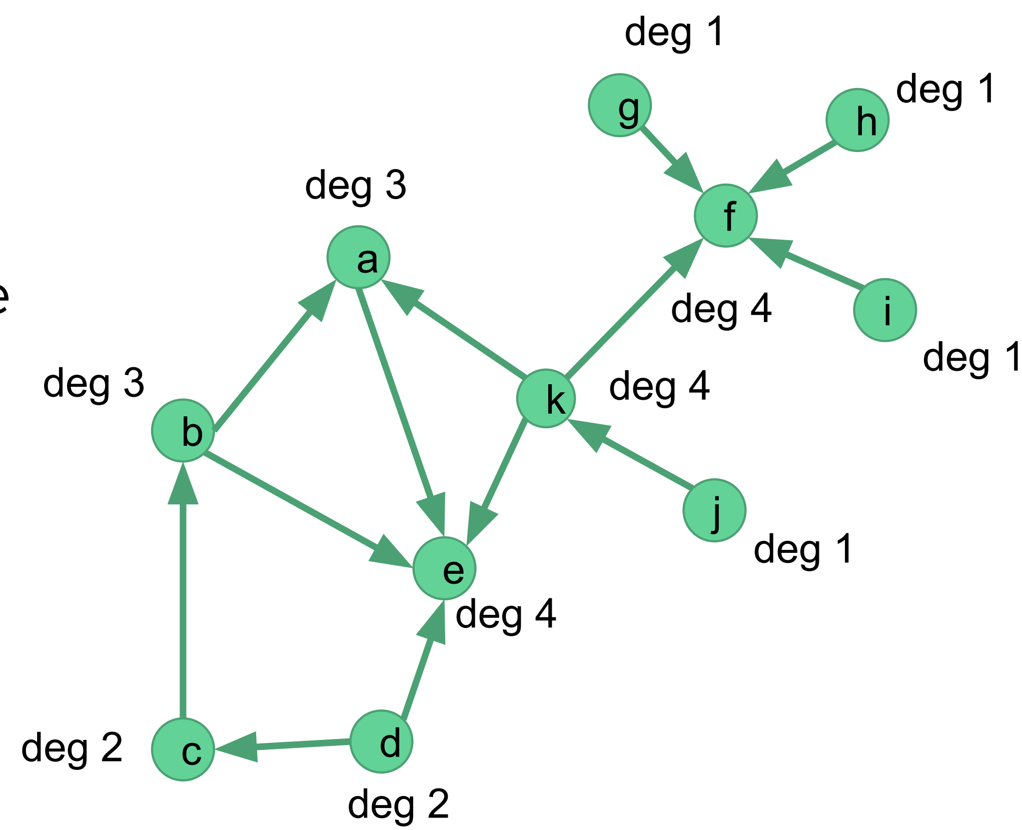
Luby 1986

ParallelMIS(G = (V, E))

```
1.  Let S = ∅
2.  While(V ≠ ∅)
3.      For each v ∈ V add it to S with probability 1/2d(v)
4.      For each e ∈ E, if both endpoints are in S, then remove the
        endpoint with lower degree (break ties lexicographically, etc.)
6.      V = V \ (S ∪ N(S))
7.  Output S
```

Luby 1986

Each edge becomes a directed edge towards the vertex with the higher degree



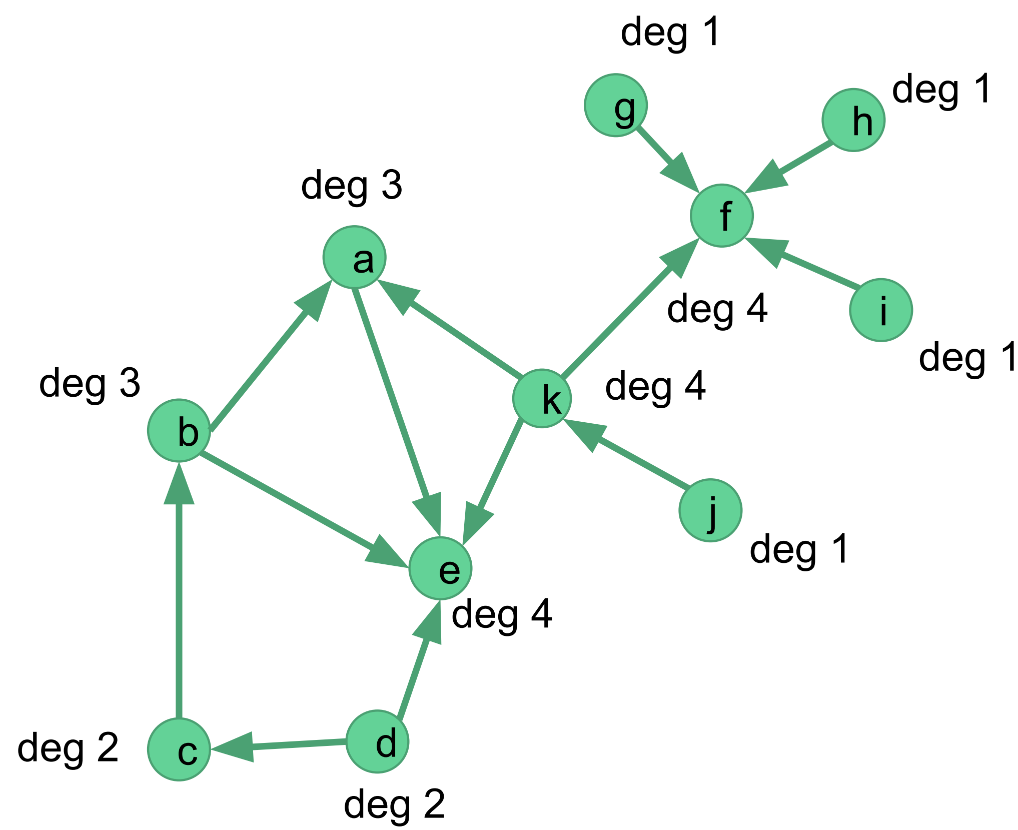
Luby 1986

A vertex v is bad if:

>2/3 of N(v) is higher

An edge is bad if:

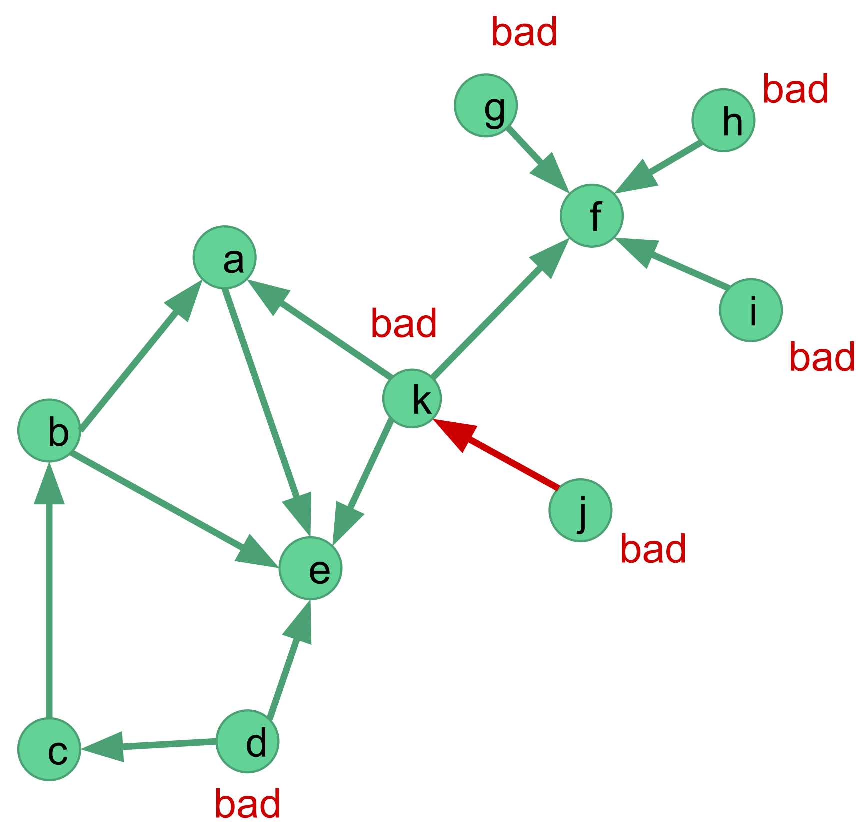
both endpoints are bad



Luby 1986

A vertex v is bad if:
>2/3 of N(v) is higher

An edge is bad if:
both endpoints are bad



Luby 1986 - Outline

- Show that at least 1/2 of edges are always ‘good’
- A good node has a constant prob. of being added to S
- So, every good edge has a constant prob. of being removed
- Half the edges are good, therefore $|E|$ drops by a constant factor each iteration

Métivier et al. 2010

ParallelMIS2(G = (V, E))

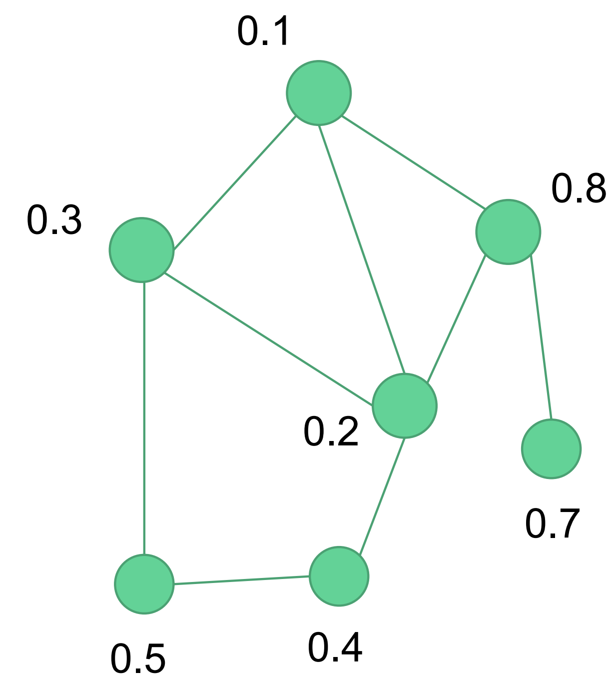
1. Let $S = \emptyset$
2. While($V \neq \emptyset$)
3. For each $v \in V$ select a random number $r(v)$ in $[0, 1]$ and send it to its neighbors, $u \in N(v)$
4. If $r(v) < r(u)$, add v to S and inform its neighbors
5. If v is added, remove v and $N(v)$ from V
6. Output S

Métivier et al. 2010 - Properties

- If v is added to S, then $u \in N(v)$ is prevented from being added to S
- The vertex with the smallest value(s) during each iteration joins S, thus removing vertices from V until it becomes $\varnothing$
- In each iteration, half the edges are removed, so the expected runtime is $O(\log n)$

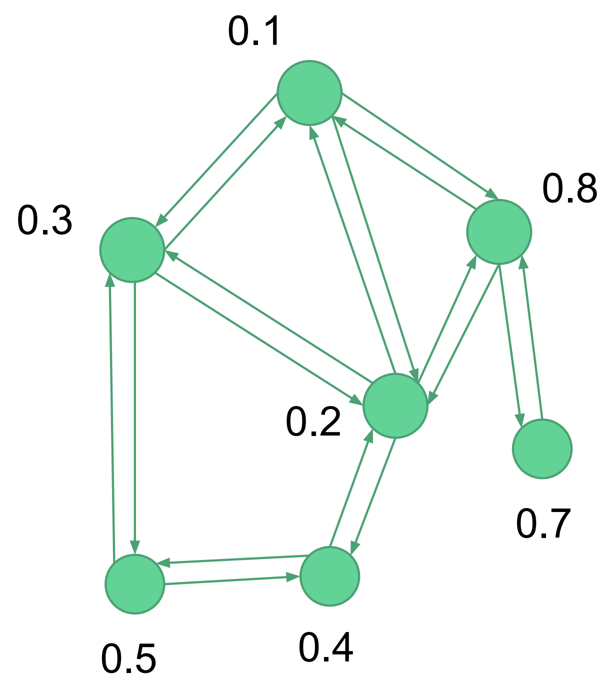
Métivier et al. 2010 - Outline

- Split each edge (v, w) $\in$ E into 2 directed edges: (v, w) and (w, v)
- Define an event when a vertex is removed
- Using linear expectation, $E[\sum_i X_i] = \sum_i E[X_i]$, show that in expectation that a constant fraction of edges are removed



Métivier et al. 2010 - Proof

- Split each edge (v, w) ∈ E into 2 directed edges: (v, w) and (w, v)



Métivier et al. 2010 - Proof

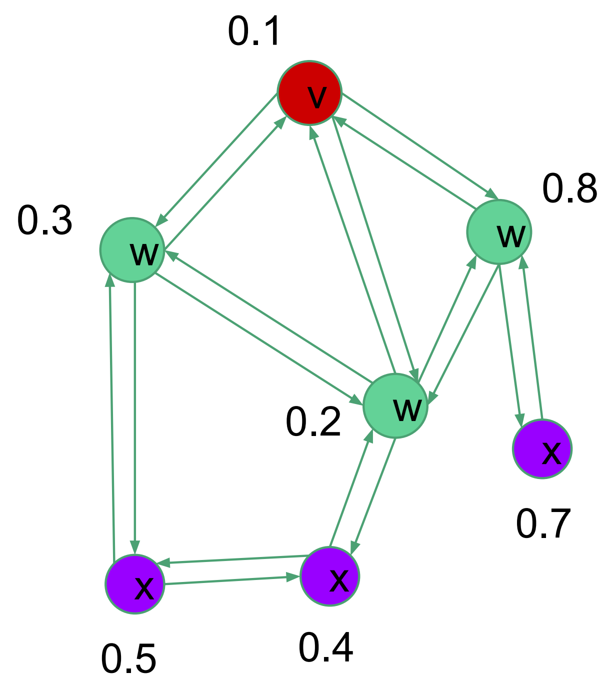
- Define an event when a vertex is removed

Given an ordered vertex pair (v, w), define the event (v ➡ w):

$r(v) < r(w), w \in N(v)$

and

$r(v) < r(x), x \in N(w)$



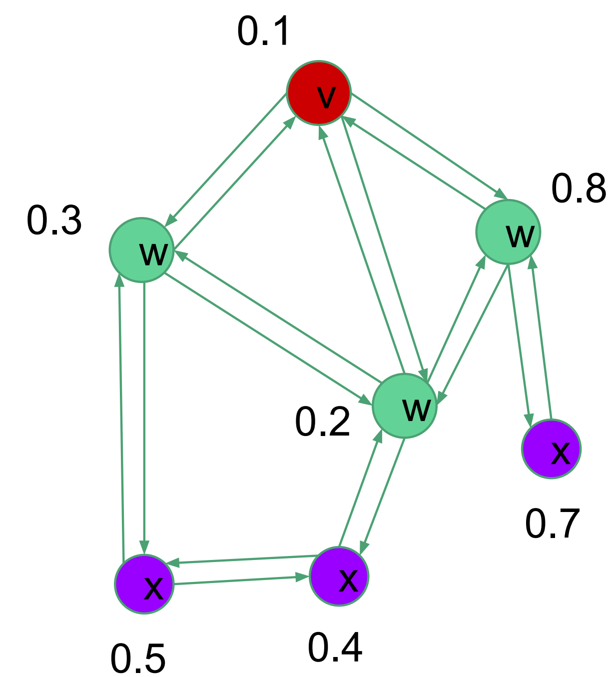
Métivier et al. 2010 - Proof

$r(v) < r(w), w \in N(v)$

and

$r(v) < r(x), x \in N(w)$

occurs with prob. at least $1 / (d(v) + d(w))$



Métivier et al. 2010 - Proof

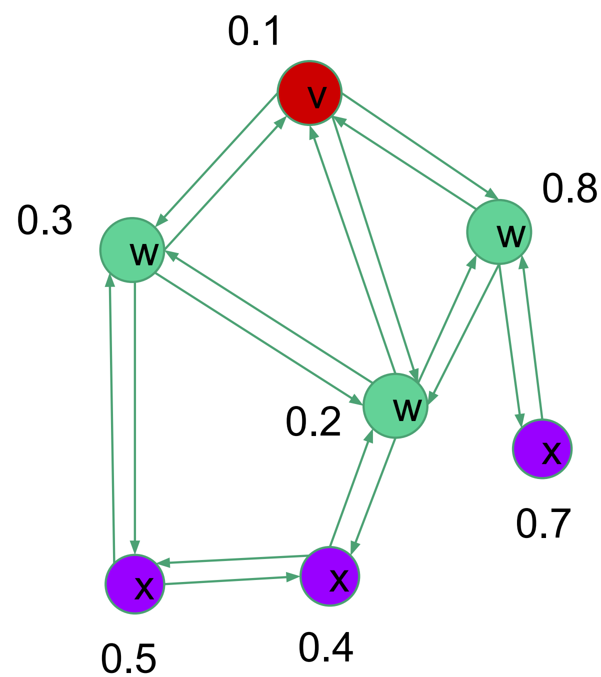
$r(v) < r(w), w \in N(v)$

and

$r(v) < r(x), x \in N(w)$

occurs with prob. at least 1 / (d(v) + d(w))

Pr(Event (v ➔ w)) = 1 / (d(v) + d(w))



Métivier et al. 2010 - Proof

$r(v) < r(w), w \in N(v)$

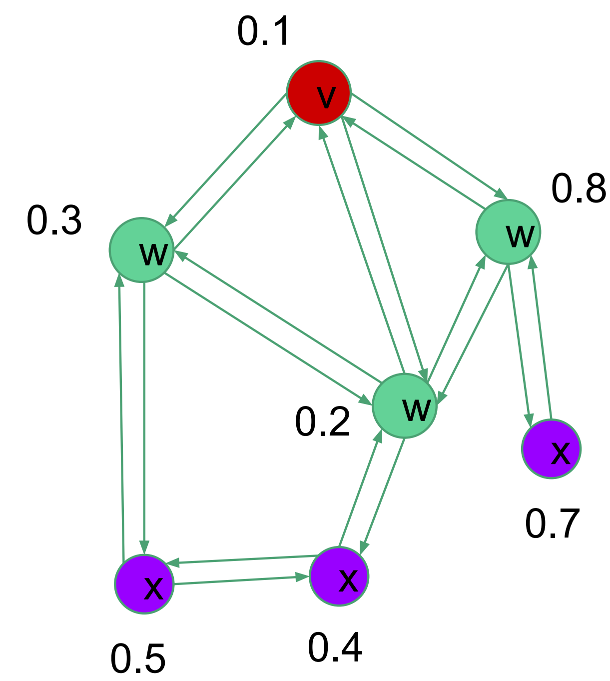
and

$r(v) < r(x), x \in N(w)$

occurs with prob. at least $1 / (d(v) + d(w))$

$Pr(Event\ (v \rightarrow w)) = 1 / (d(v) + d(w))$

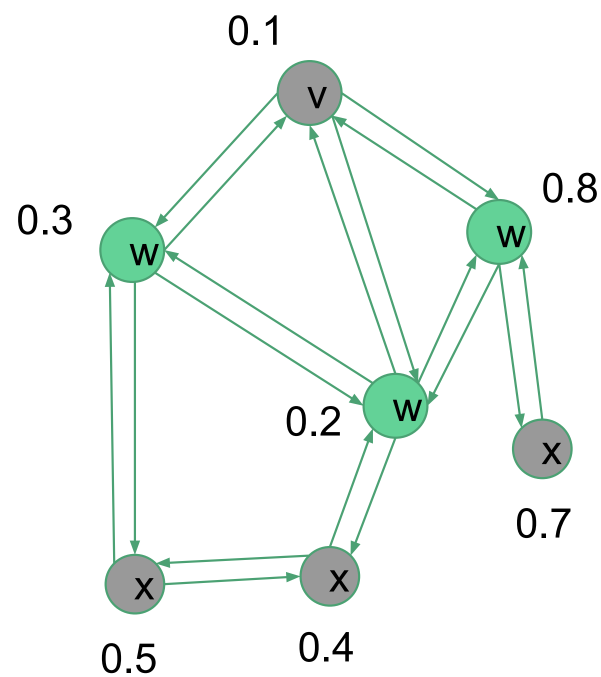
$Pr(Event\ (w \rightarrow v)) = 1 / (d(w) + d(v))$



Métivier et al. 2010 - Proof

How many directed edges are removed when the event (v ➡ w) occurs? d(w) edges

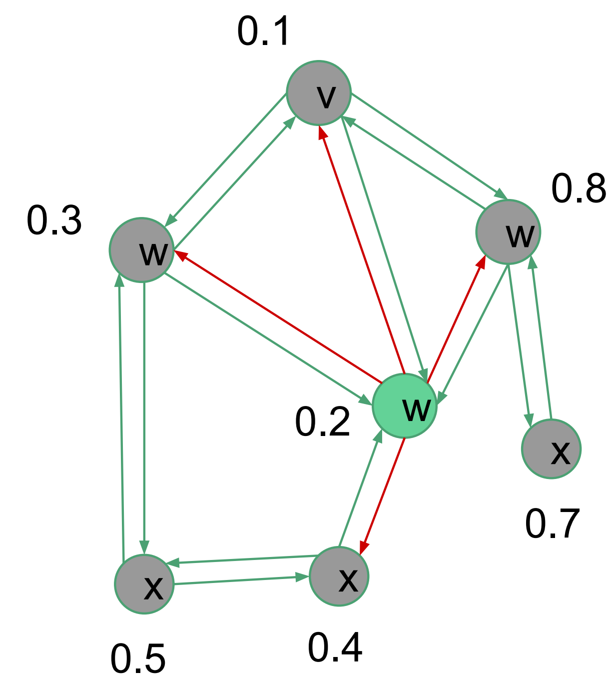
And for (w ➡ v), d(v) edges



Métivier et al. 2010 - Proof

How many directed edges are removed when the event (v ➔ w) occurs? at least d(w) edges

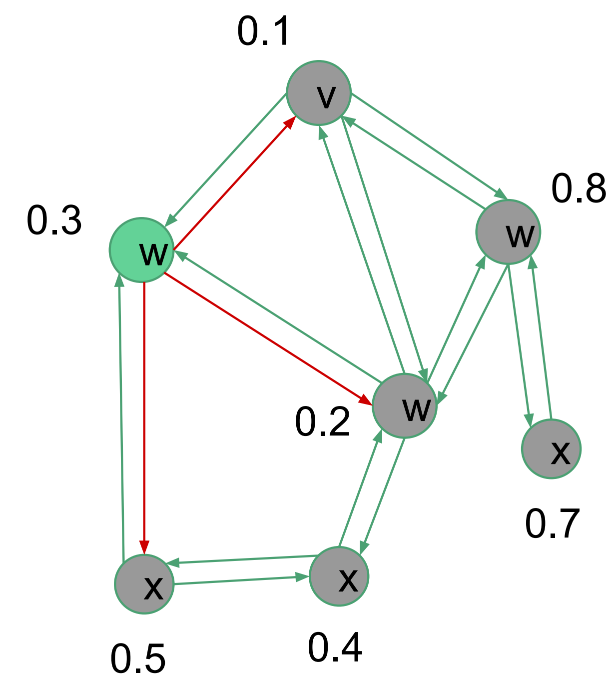
And for (w ➔ v), d(v) edges



Métivier et al. 2010 - Proof

How many directed edges are removed when the event (v ➔ w) occurs? at least d(w) edges

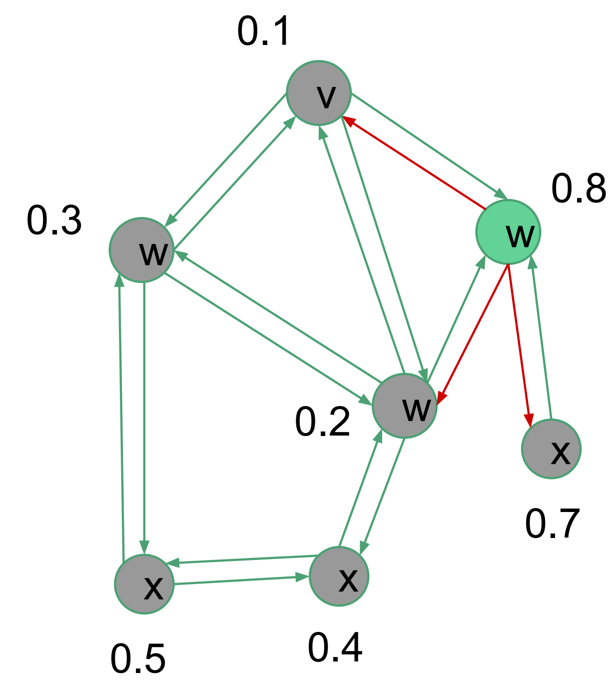
And for (w ➔ v), d(v) edges



Métivier et al. 2010 - Proof

How many directed edges are removed when the event (v ➡ w) occurs? at least d(w) edges

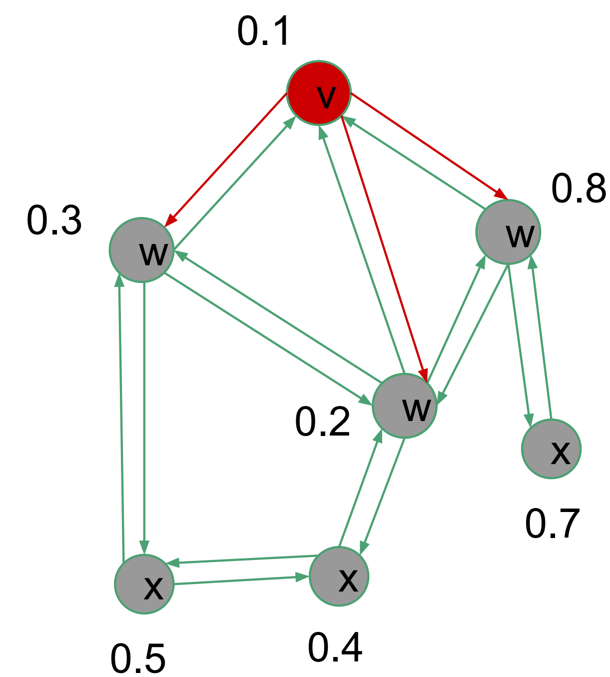
And for (w ➡ v), d(v) edges



Métivier et al. 2010 - Proof

How many directed edges are removed when the event (v → w) occurs? at least d(w) edges

And for (w → v), d(v) edges



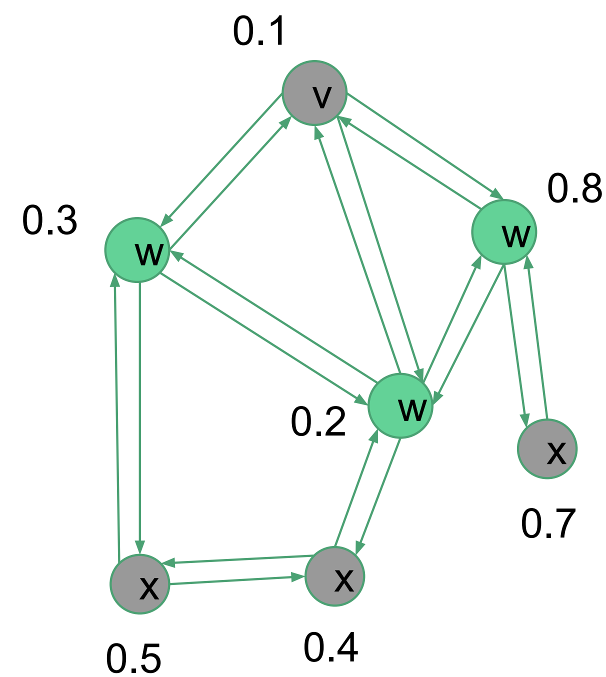
Métivier et al. 2010 - Proof

How many directed edges are removed when the event (v → w) occurs? at least d(w) edges

And for (w → v), d(v) edges

So if event (v → w) occurs then $X_{(v \to w)}$: d(w)

If event (w → v) occurs then $X_{(w \to v)}$: d(v)



Métivier et al. 2010 - Proof

$$E[X] = \sum_{\{v,\, w\} \in E} E[X_{(v\text{->}w)}] + E[X_{(w\text{->}v)}]$$

$$= \sum_{\{v,\, w\} \in E} d(w) \square \Pr[\text{Event } (v\text{->}w)] + d(v) \square \Pr[\text{Event } (w\text{->}v)]$$

Métivier et al. 2010 - Proof

$$E[X] = \Sigma_{\{v,\ w\} \in E}\ E[X_{(v\text{->}w)}] + E[X_{(w\text{->}v)}]$$

$$= \Sigma_{\{v,\ w\} \in E}\ d(w) \square \Pr[\text{Event } (v\text{->}w)] + d(v) \square \Pr[\text{Event } (w\text{->}v)]$$

$$= \Sigma_{\{v,\ w\} \in E}\ d(w) / (d(v) + d(w)) + d(v) / (d(v) + d(w))$$

$$= \Sigma_{\{v,\ w\} \in E}\ d(v) + d(w) / (d(v) + d(w))$$

$$= \Sigma_{\{v,\ w\} \in E} 1$$

Métivier et al. 2010 - Proof

$E[X] = \sum_{\{v, w\} \in E} 1 = |E|$

But we counted twice as many edges in the directed graph as in the undirected graph.

Therefore at least |E| / 2 edges are removed each iteration.

Remarks

Usable in distributed computing or in multicore environments

Open question as to can we do better

References

M. Luby, A simple parallel algorithm for the maximal independent set problem, Proc, 17th ACM Symposium on Theory of Computing, 1985, pp. 1-10

Métivier, Y.; Robson, J. M.; Saheb-Djahromi, N.; Zemmari, A. (2010). "An optimal bit complexity randomized distributed MIS algorithm". Distributed Computing 23 (5–6): 331.

https://en.wikipedia.org/wiki/Maximal_independent_set

"Principles of Distributed Computing" ETH Zurich. Retrieved 21 February 2015

Gallai, T. "Über extreme Punkt- und Kantenmengen." *Ann. Univ. Sci. Budapest, Eőtvős Sect. Math.* **2**, 133-138, 1959

Kőnig, Dénes (1931), "Gráfok és mátrixok", *Matematikai és Fizikai Lapok* **38**: 116–119