

Network Optimization by Randomization  
(Distributed Computing):

# Maximal Independent Set

# What is a MIS?

---

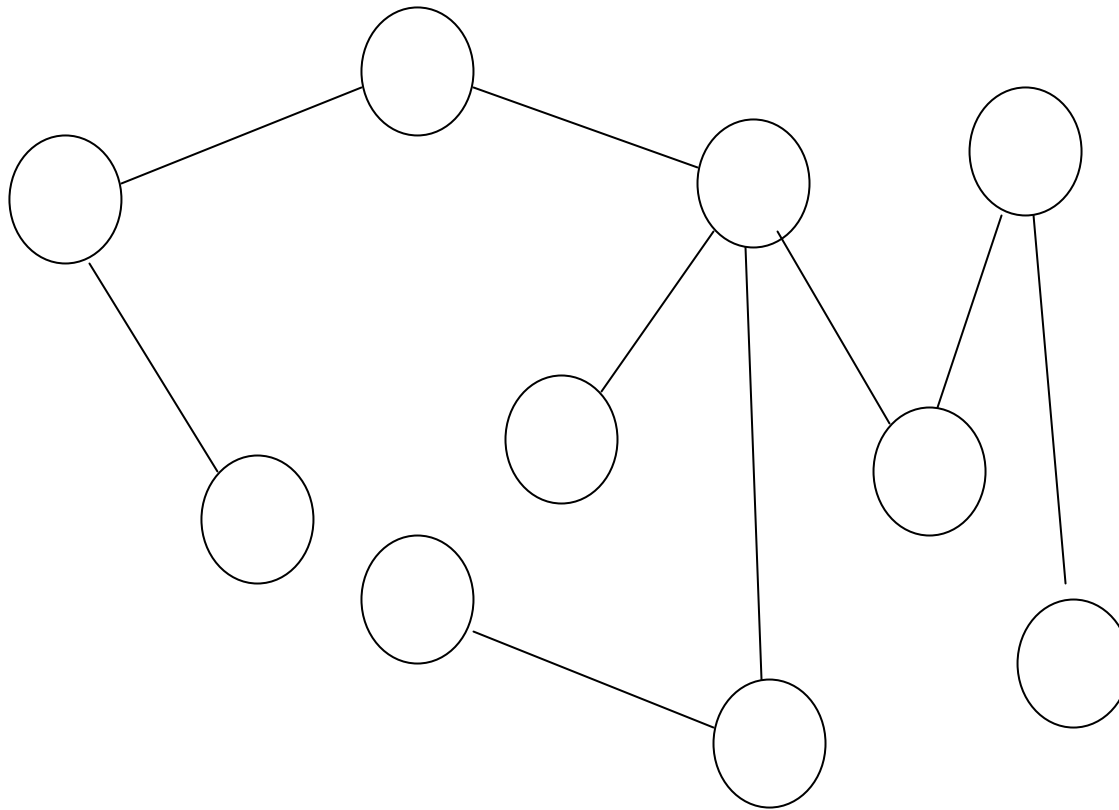
## MIS

An independent set (IS) of an undirected graph is a subset  $U$  of nodes such that no two nodes in  $U$  are adjacent. An IS is maximal if no node can be added to  $U$  without violating IS (called **MIS**). A maximum IS (called **MaxIS**) is one of maximum cardinality.

... known from „classic TCS“:  
applications?  
backbone, parallelism, ...  
complexities?

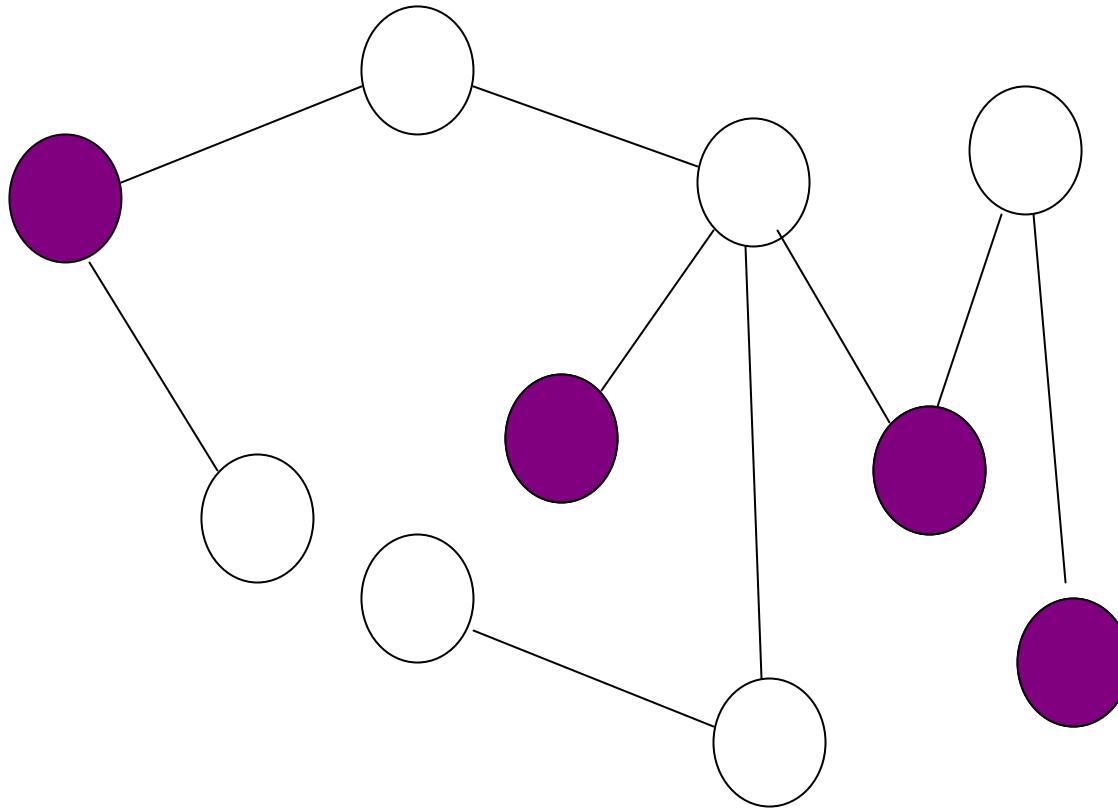
# MIS and MaxIS?

---



# Nothing, IS, MIS, MaxIS?

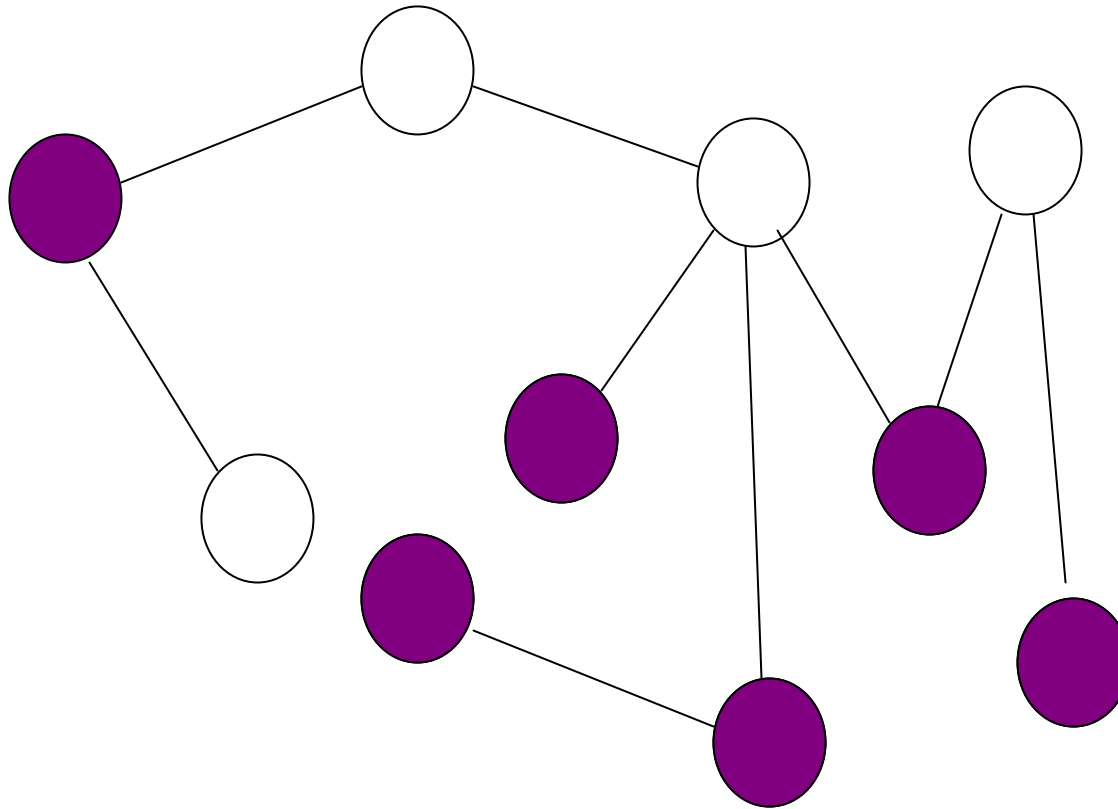
---



IS but not MIS.

# Nothing, IS, MIS, MaxIS?

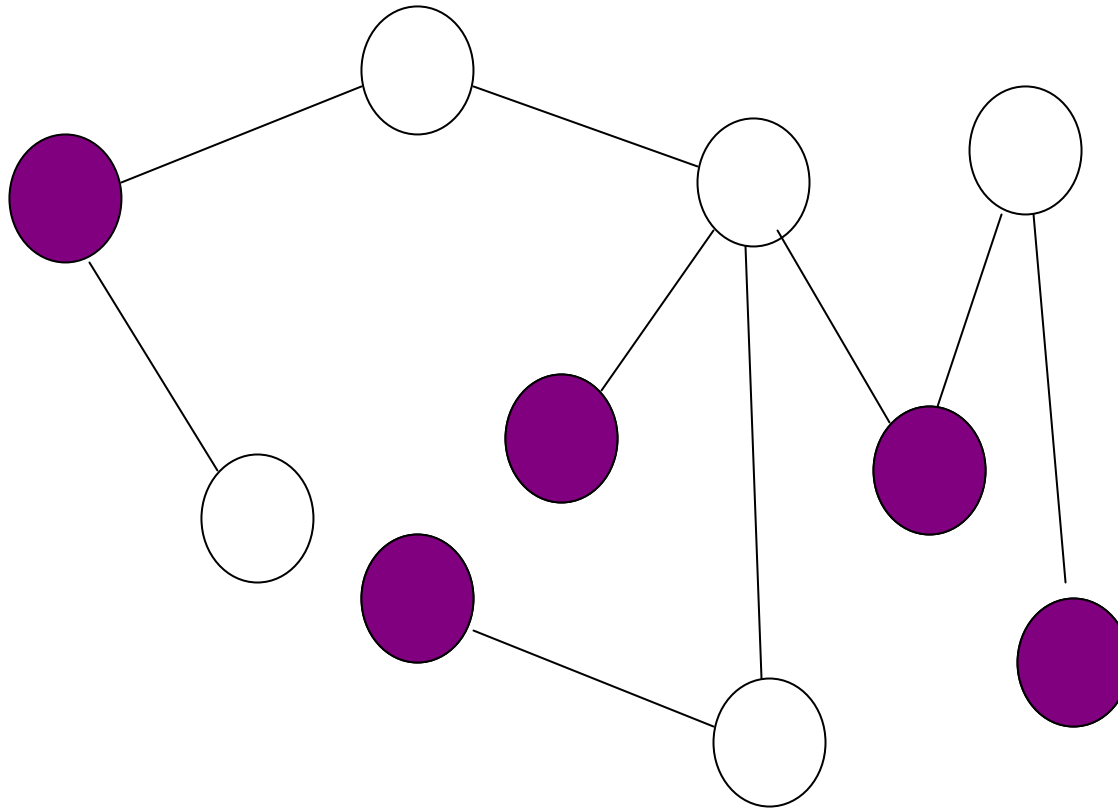
---



Nothing.

# Nothing, IS, MIS, MaxIS?

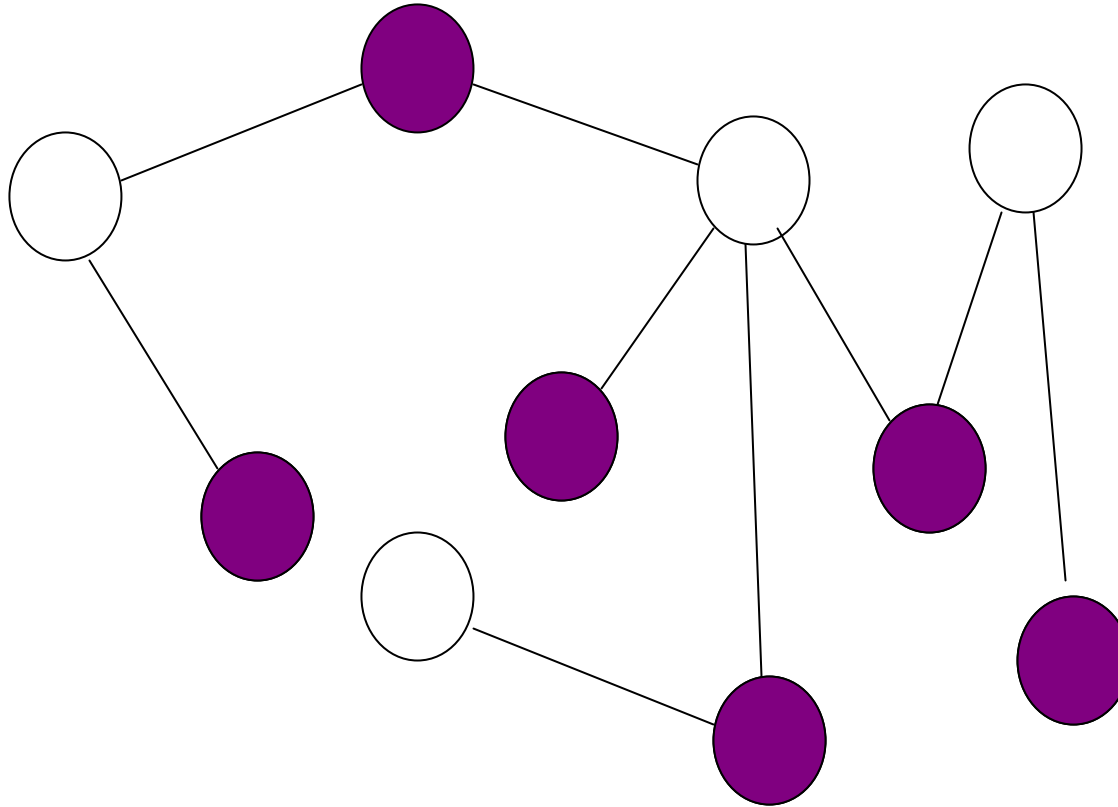
---



MIS.

# Nothing, IS, MIS, MaxIS?

---



MaxIS.

# Complexities?

---

**MaxIS is NP-hard!**

So let's concentrate on MIS...

**How much worse can MIS be than MaxIS?**

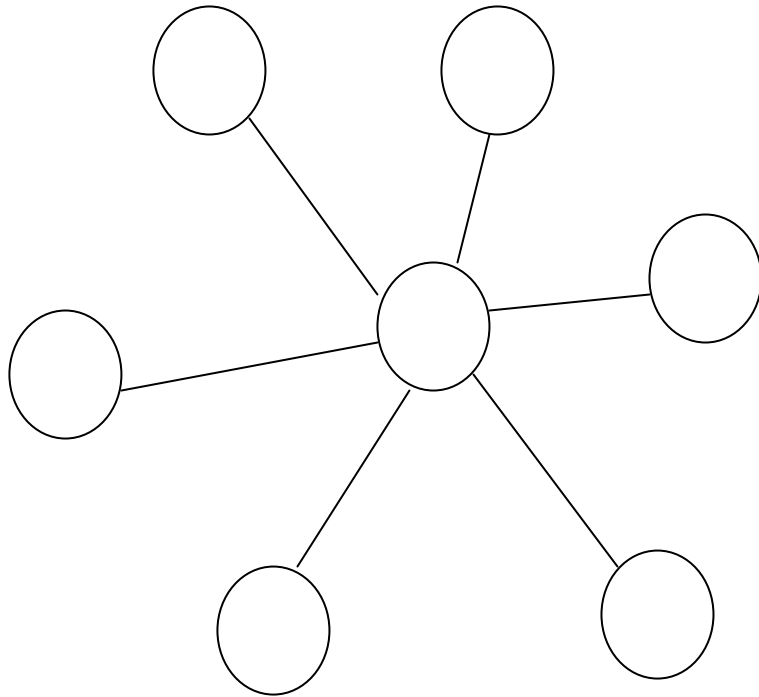


# MIS vs MaxIS

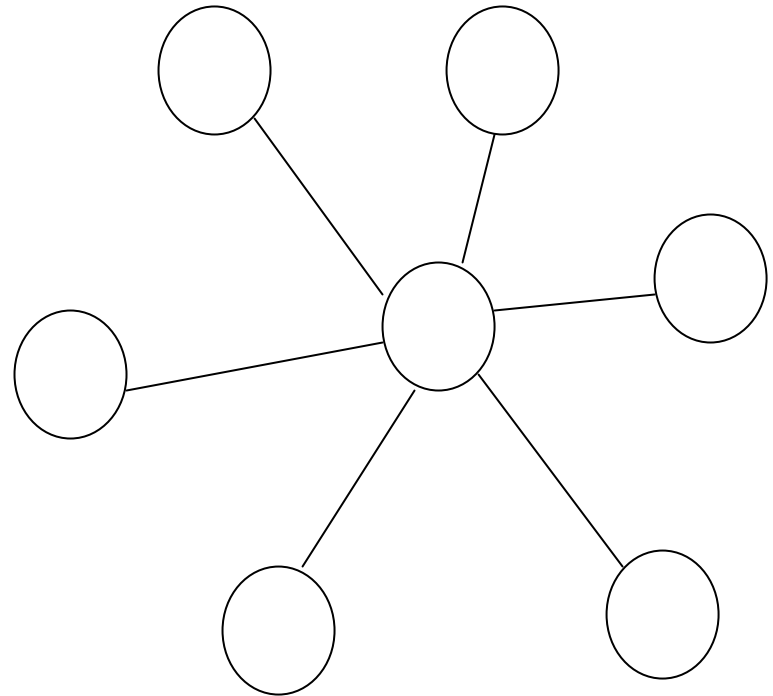
---

## How much worse can MIS be than MaxIS?

minimal MIS?



maxIS?

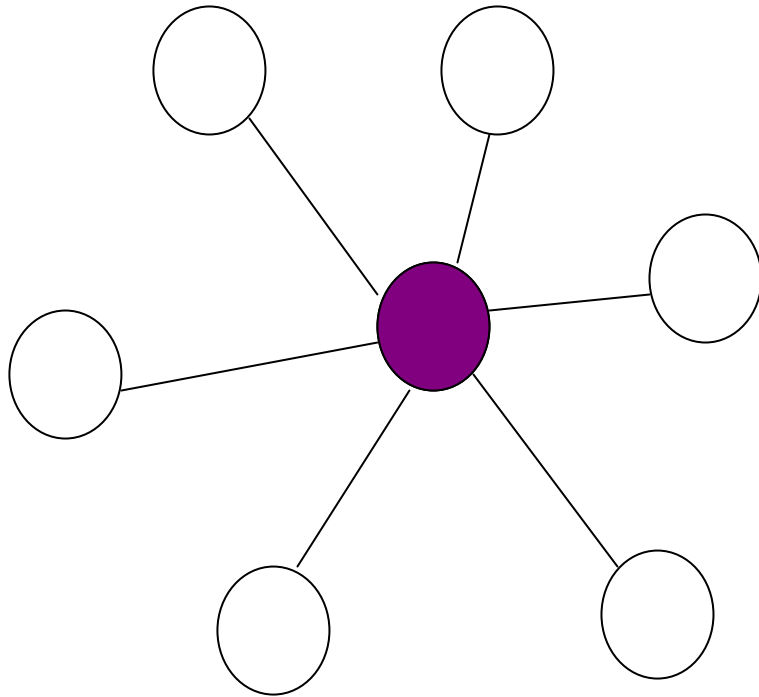


# MIS vs MaxIS

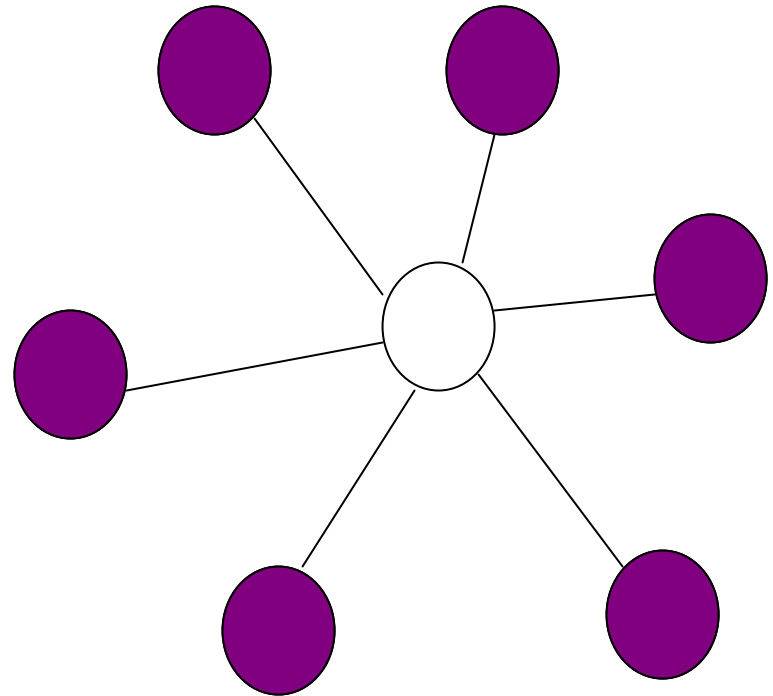
---

## How much worse can MIS be than Max-IS?

minimal MIS?



maxIS?

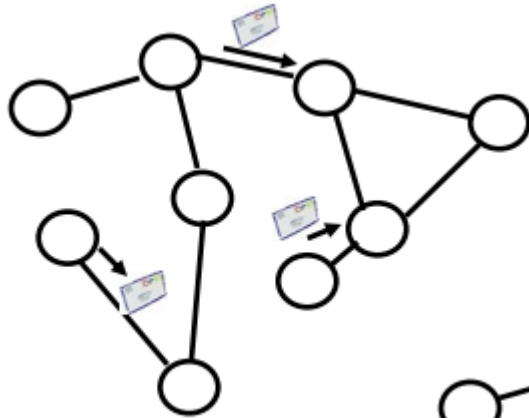


**How to compute a MIS in a distributed manner?!**

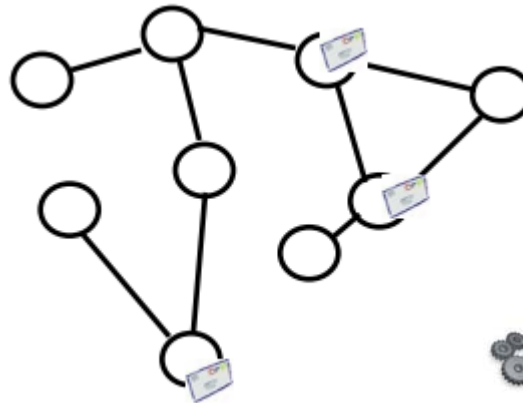
# Recall: Local Algorithm

---

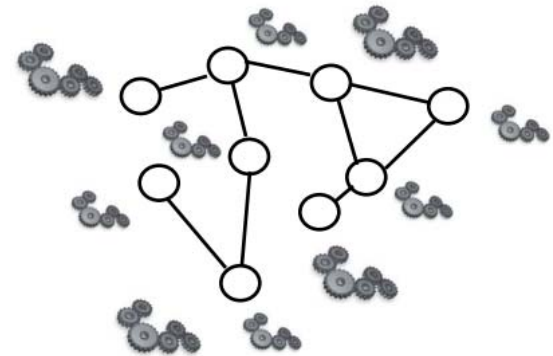
Send...



... receive...



... compute.



## Slow MIS

assume node IDs

Each node  $v$ :

1. If all neighbors with larger IDs have decided not to join MIS then:  
     $v$  decides to join MIS

**Analysis?**

## Time Complexity?

Not faster than sequential algorithm!  
Worst-case example?  
E.g., sorted line:  $O(n)$  time.

## Local Computations?

Fast! 😊

## Message Complexity?

For example in clique:  $O(n^2)$   
( $O(m)$  in general: each node needs to inform all neighbors when deciding.)

## Independent sets and colorings are related: how?

Each color in a valid coloring constitutes an independent set (but not necessarily a MIS).

## How to compute MIS from coloring?

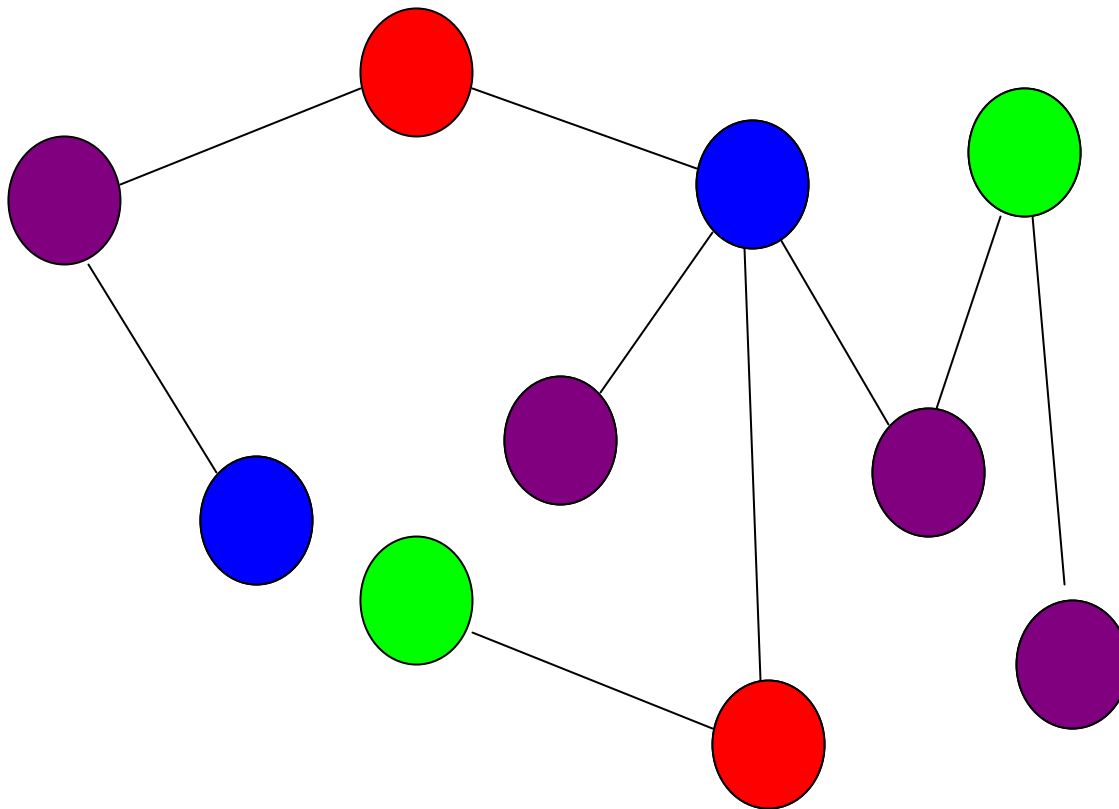
Choose all nodes of **first color**. Then for any **additional color**, add **in parallel** as many nodes as possible!

## Why, and implications?

# Coloring vs MIS

---

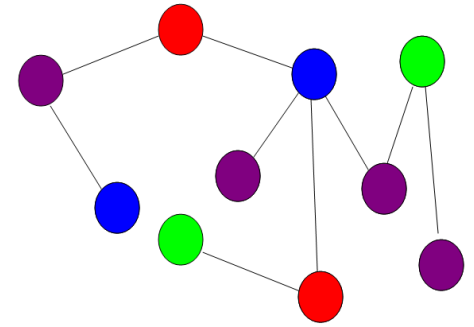
Valid coloring:



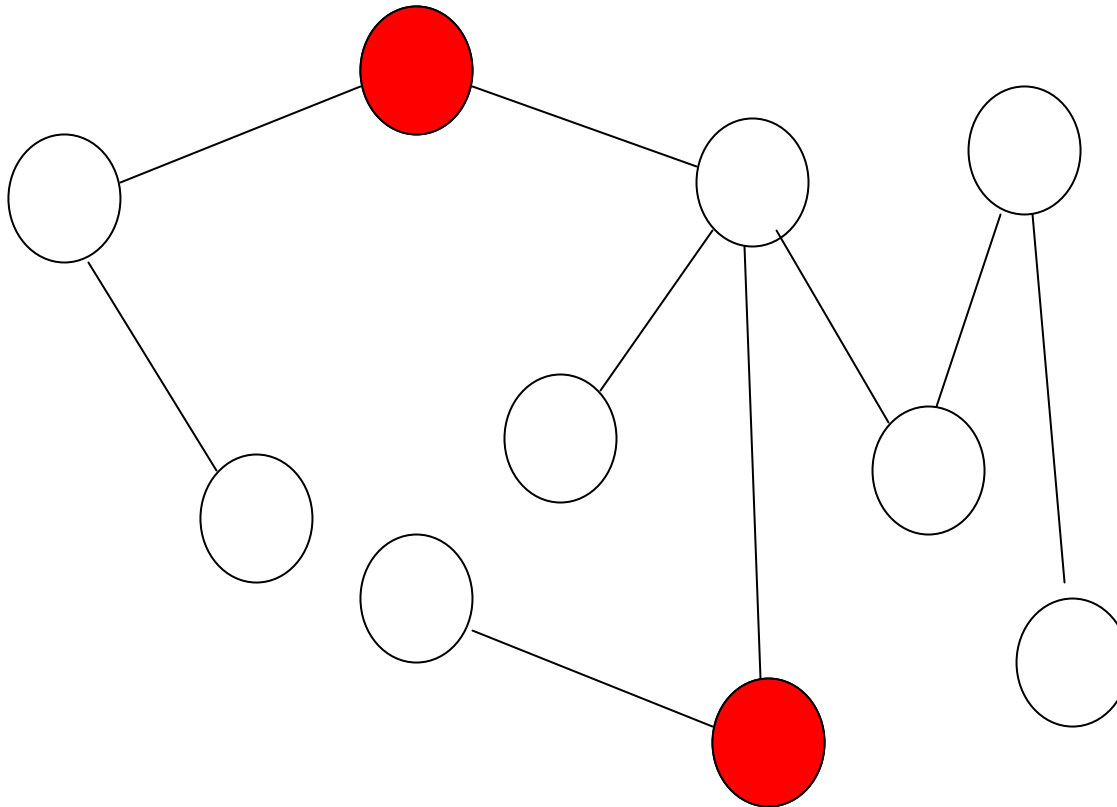


# Coloring vs MIS

---



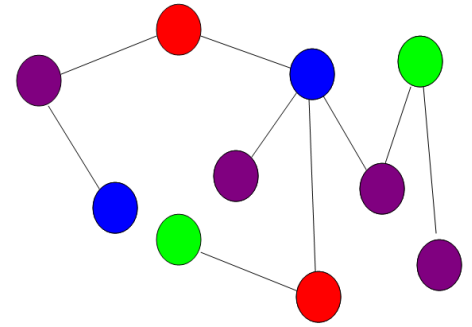
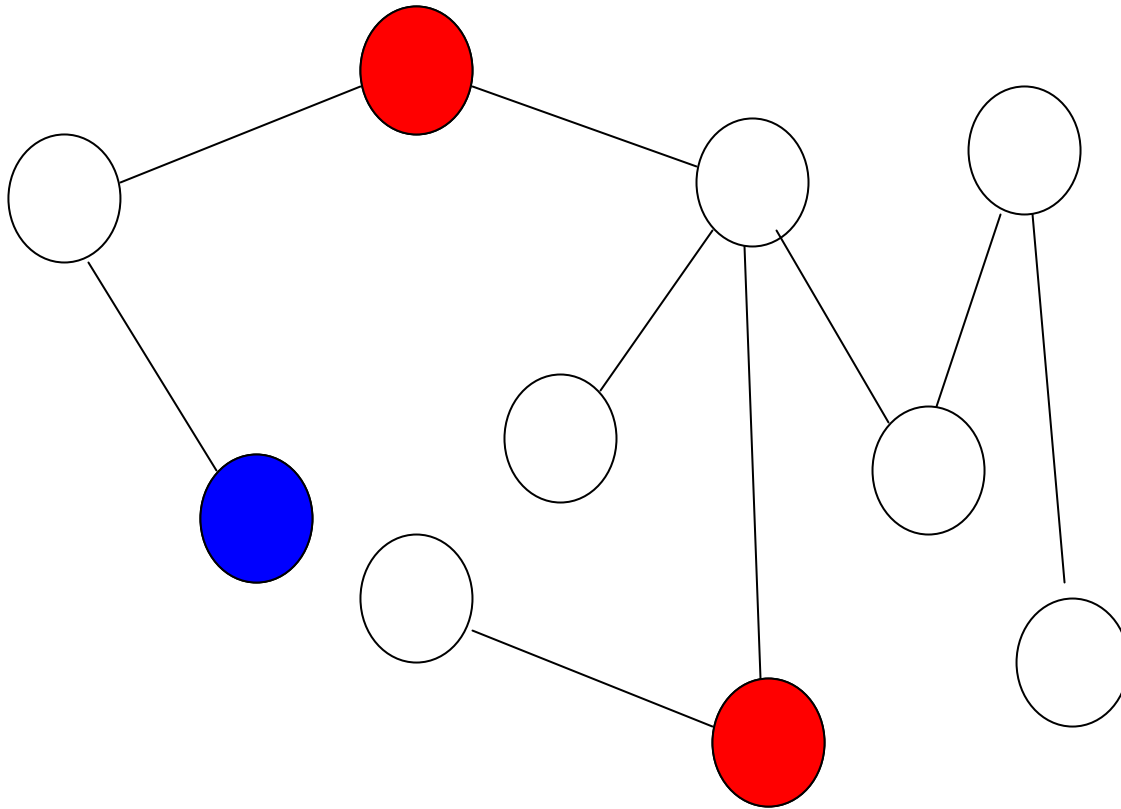
Independent set:



# Coloring vs MIS

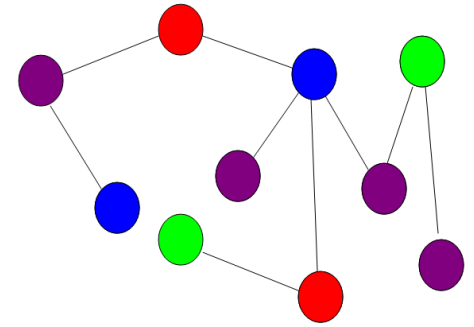
---

Add all possible blue:

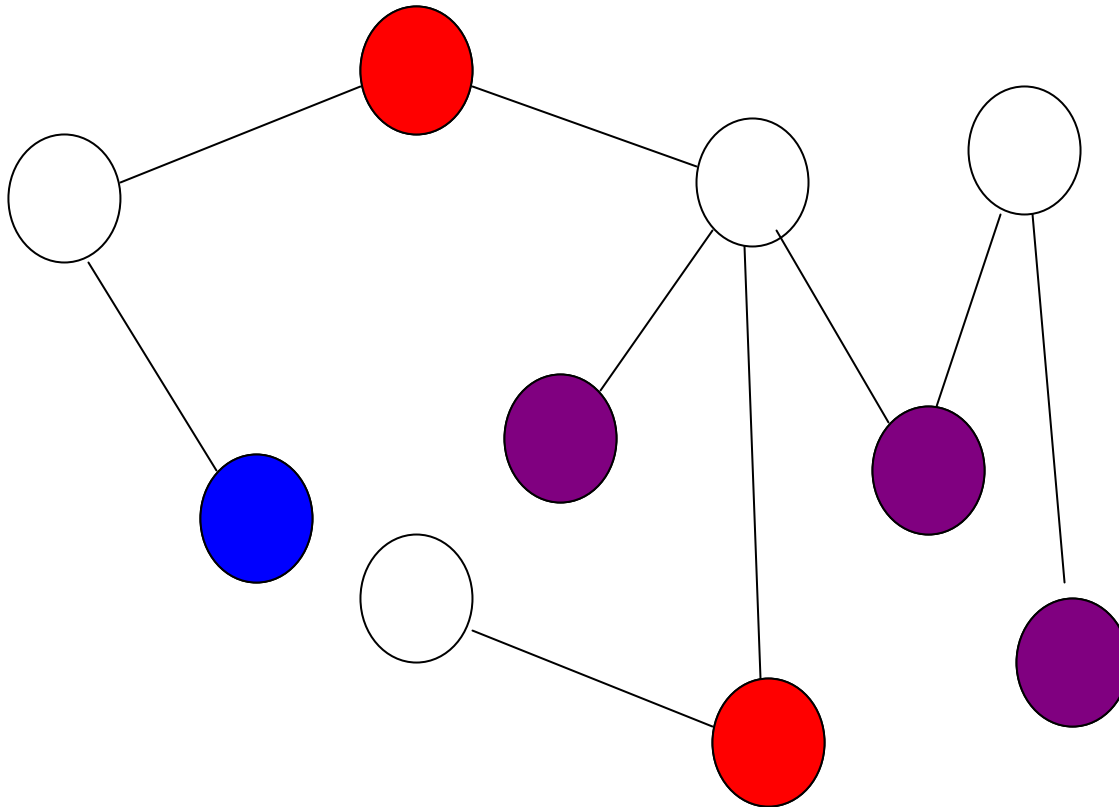


# Coloring vs MIS

---



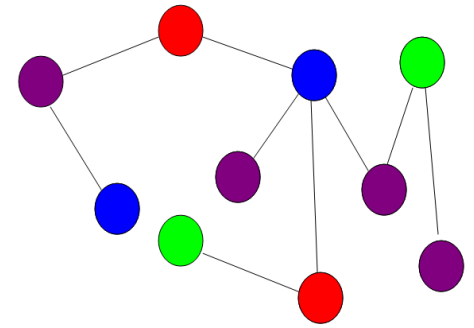
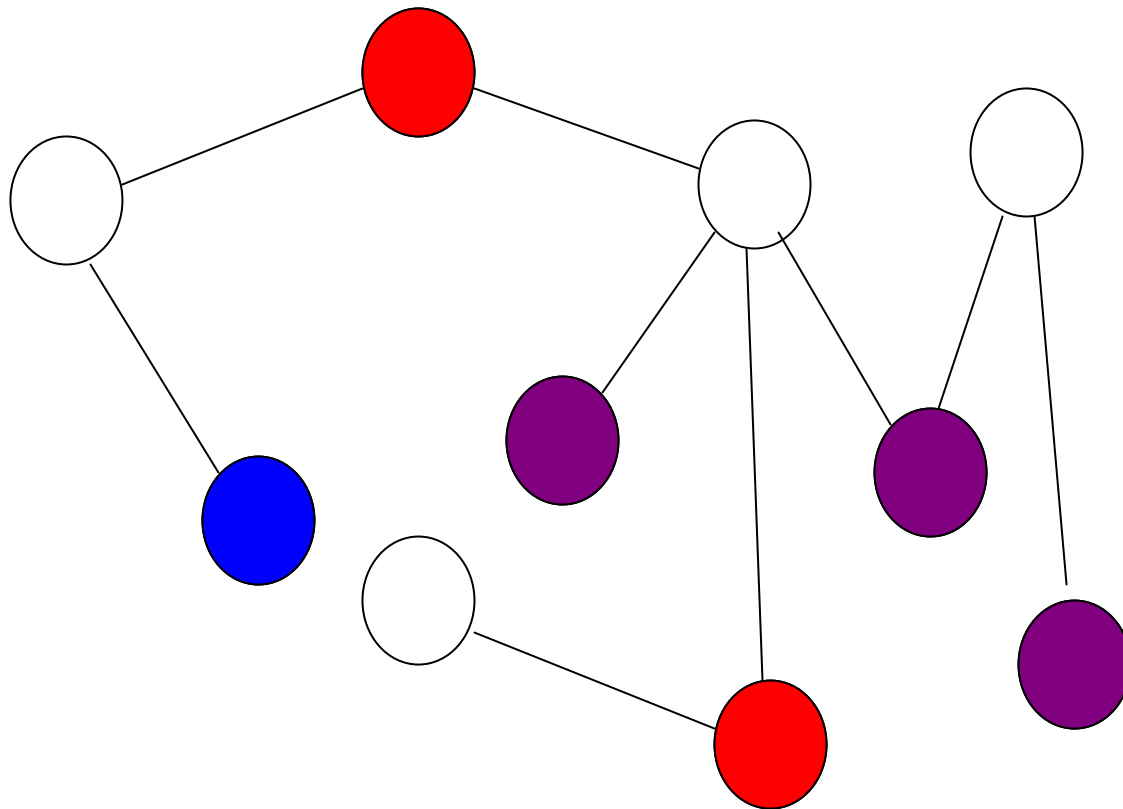
Add all possible violet:



# Coloring vs MIS

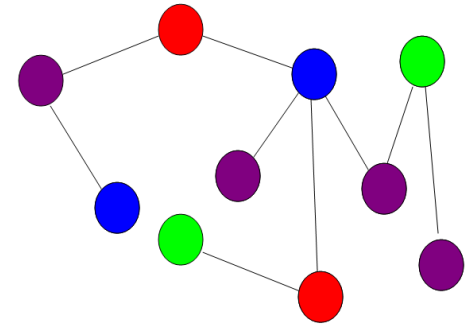
---

Add all possible green:

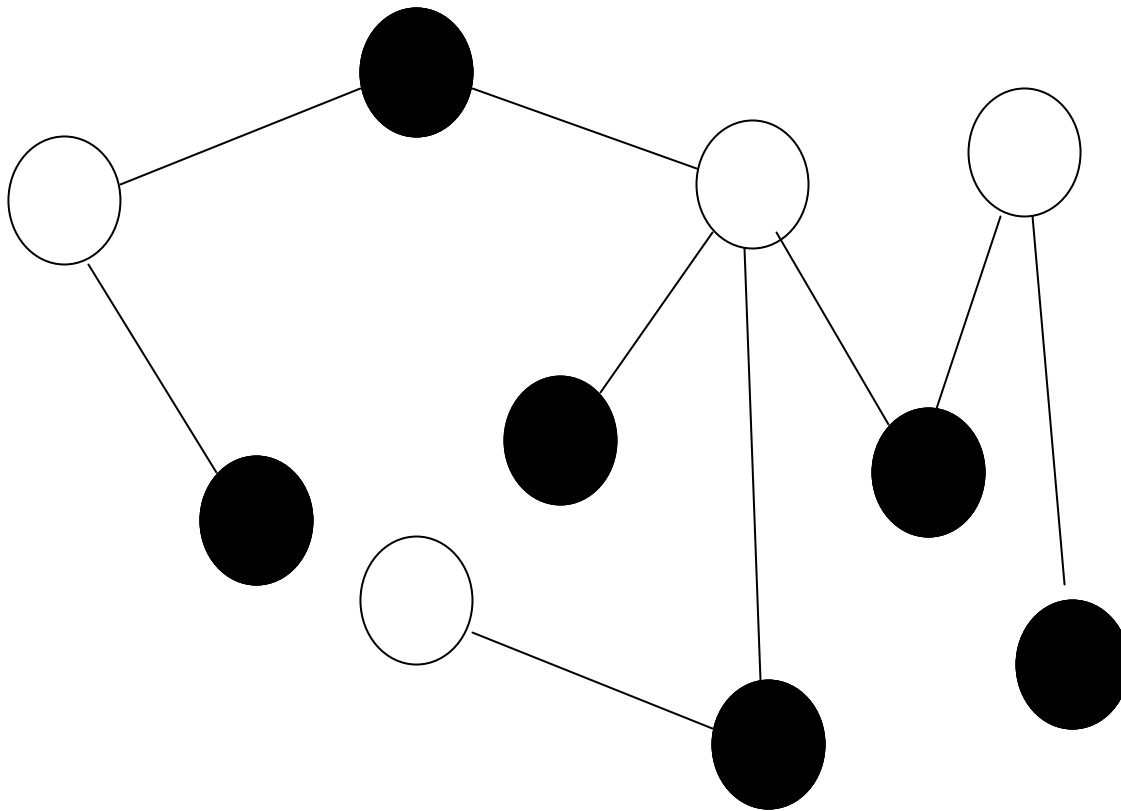


# Coloring vs MIS

---



That's all: MIS!



Analysis of algorithm?

## Why does algorithm work?

Same color: all nodes independent, can add them in parallel without conflict (not adding two conflicting nodes concurrently).

## Runtime?

### Lemma

Given a coloring algorithm with runtime  $T$  that needs  $C$  colors, we can construct a MIS in time  $C+T$ .

## What does it imply for MIS on trees?

We can color trees in  $\log^*$  time and with 3 colors, so:

### **MIS on Trees**

There is a deterministic MIS on trees that runs in distributed time  $O(\log^* n)$ .

# Better MIS Algorithms

---

Any ideas?

## Takeaway

If you can't find fast deterministic algorithms,  
try randomization!

Ideas for randomized algorithms?



## Fast MIS (1986)

Proceed in rounds consisting of phases

In a **phase**:

1. each node  $v$  **marks** itself with **probability**  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no **higher degree neighbor** is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. **delete** all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

Why is it correct? Why IS? Why MIS?

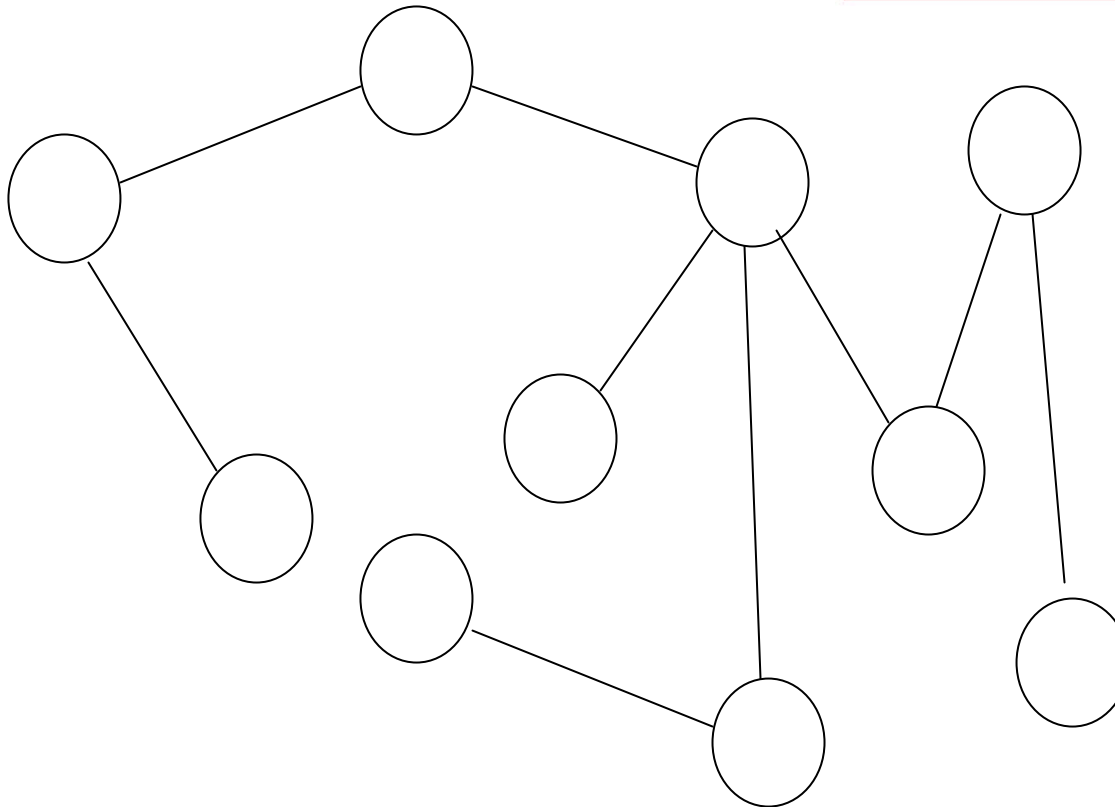
Note: the higher the degree the less likely to mark, but the more likely to join MIS once marked!

Proceed in rounds consisting of phases

In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

## Probability of marking?



# MIS 1986

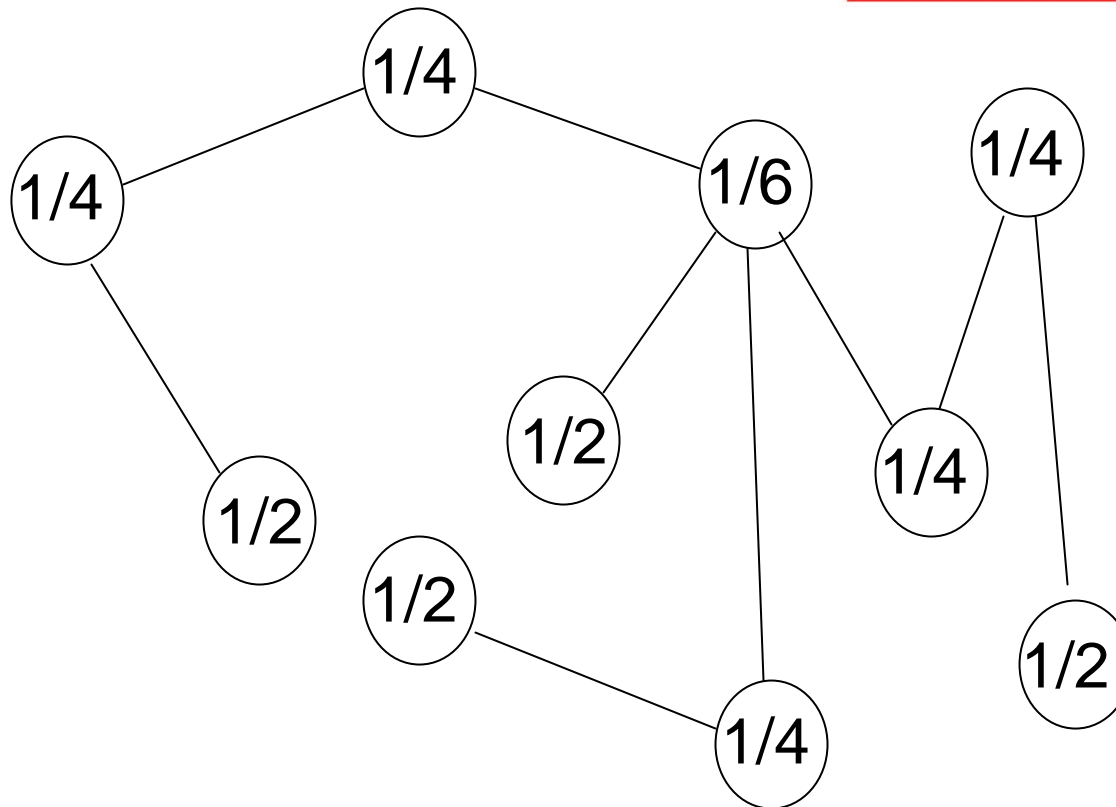
## Fast MIS (1986)

Proceed in rounds consisting of phases

In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

## Probability of marking?



# MIS 1986

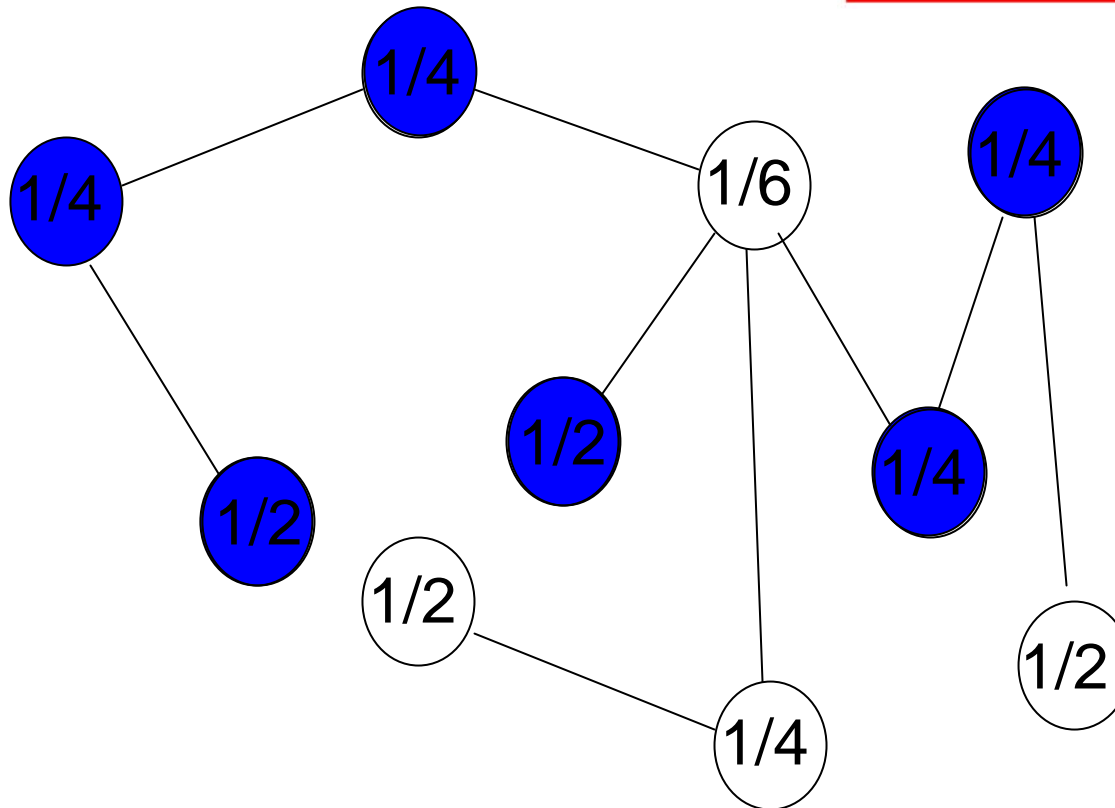
## Marking... Who stays?

### Fast MIS (1986)

Proceed in rounds consisting of phases

In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore



# MIS 1986

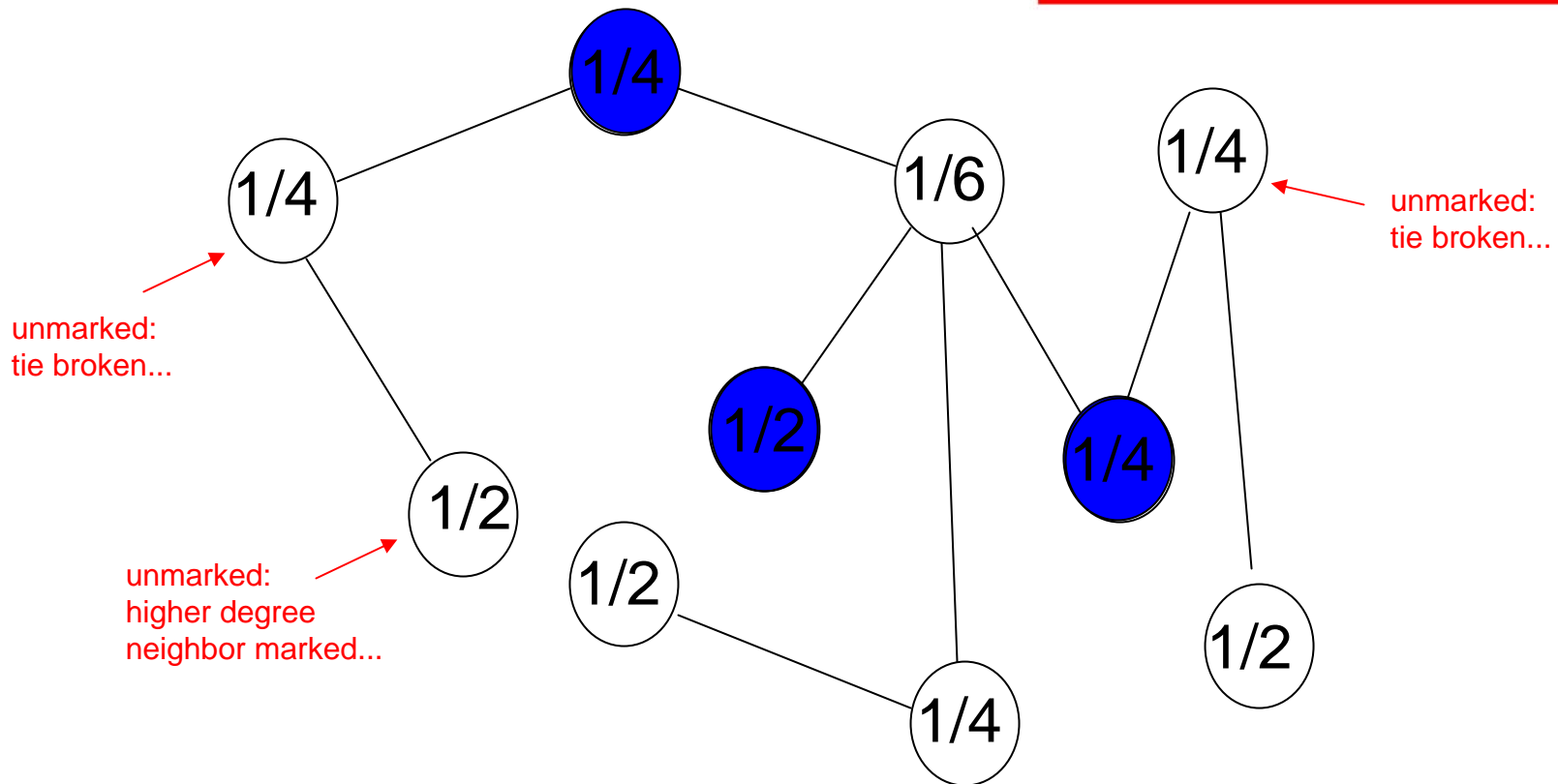
And now?

## Fast MIS (1986)

Proceed in rounds consisting of phases

In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore



# MIS 1986

---

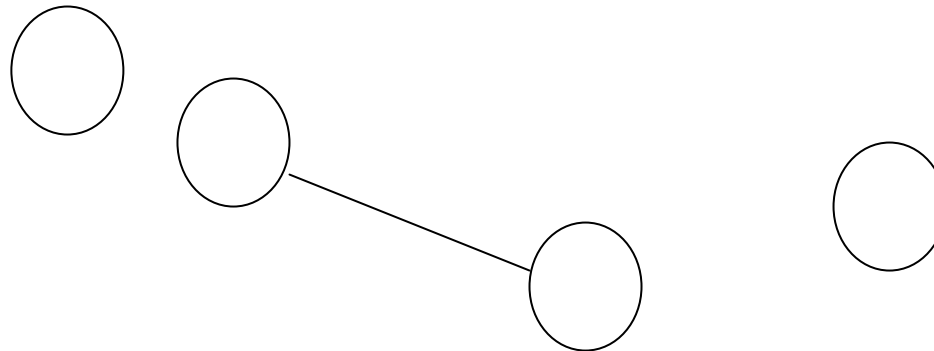
Delete neighborhoods...

## Fast MIS (1986)

Proceed in rounds consisting of phases

In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore



## Fast MIS (1986)

Proceed in rounds consisting of phases

In a **phase**:

1. each node  $v$  **marks** itself with **probability**  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no **higher degree neighbor** is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. **delete** all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

**IS:** Step 1 and Step 2 ensure that node only joins if neighbors do not!

**MIS:** At some time, nodes will mark themselves in Step 1.

## Fast MIS (1986)

Proceed in rounds consisting of phases

In a **phase**:

1. each node  $v$  **marks** itself with **probability**  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no **higher degree neighbor** is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. **delete** all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

Runtime: how fast will algorithm terminate?



# Our Strategy!

---

**We want to show logarithmic runtime. So for example?**

Idea:

Each node is **removed with constant probability** (e.g.,  $\frac{1}{2}$ ) in each round => half of the nodes vanish in each round.

Or: Each **edge is removed** with constant probability in each round! As  $O(\log m) = O(\log n^2) = O(\log n)$

Unfortunately, this is not true... ☹️ Alternative?

A **constant fraction of all nodes** are removed in each step!

E.g., a constant subset of nodes is „**good**“ and a constant fraction thereof is removed...

Or the same for **edges**...

## Joining MIS

Node  $v$  joins MIS in Step 2 with probability  $p \geq ?$

Proof.

On what could it depend?

Marked with probability that depends on degree, i.e.,  $1/2d(v)$ .  
(So at most this...)

In MIS subsequently if degree is largest...  
(This is likely then if degree is small!)

**We will find that marked nodes are likely to join MIS!**

Proceed in rounds consisting of phases

In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

## Joining MIS

Node  $v$  joins MIS in Step 2 with probability  $p \geq 1/(4d(v))$ .

Proof.

Let  $M$  be the set of marked nodes in Step 1.

Let  $H(v)$  be the set of neighbors of  $v$  with higher degree (or same degree and higher identifier).

$$\begin{aligned} P[v \notin \text{MIS} \mid v \in M] &= P[\exists w \in H(v), w \in M \mid v \in M] \\ &= P[\exists w \in H(v), w \in M] \quad // \text{ independent whether } v \text{ is marked or not} \\ &\leq \sum_{w \in H(v)} P[w \in M] \quad // \text{ do not only count exactly one but also multiple} \\ &= \sum_{w \in H(v)} 1/(2d(w)) \quad // \text{ see Joining MIS algorithm} \\ &\leq \sum_{w \in H(v)} 1/(2d(v)) \quad // v\text{'s degree is the lowest one} \\ &\leq d(v)/(2d(v)) = 1/2 \quad // \text{ at most } d(v) \text{ higher neighbors...} \end{aligned}$$

**Marked nodes are likely to be in MIS!**

So

$$\begin{aligned} P[v \in \text{MIS}] &= P[v \in \text{MIS} \mid v \in M] \cdot P[v \in M] \\ &\geq \frac{1}{2} \cdot \frac{1}{2d(v)} \end{aligned}$$

QED

# Recall Our Strategy!

---

**We want to show logarithmic runtime. So for example?**

Idea:

Each node is removed with constant probability (e.g.,  $\frac{1}{2}$ ) in each round => half of the nodes vanish in each round.

Or: Each edge is removed with constant probability in each round! As  $O(\log m) = O(\log n^2) = O(\log n)$

Unfortunately, this is not true... ☹️ Alternative?

**Let's try this:**

**A constant fraction of all nodes are removed in each step!**

**E.g., a constant subset of nodes is „good“ and a constant fraction thereof is removed...**

Or the same for edges...

**How to define good nodes?!**

**Node with low degree neighbors!**

**(Why? Likely to be removed as neighbors are likely to be marked and hence join MIS...)**

Proceed in rounds consisting of phases

In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

## Good&Bad Nodes

A node  $v$  is called *good* if

$$\sum_{w \in N(v)} 1/(2d(w)) \geq 1/6.$$

What does it mean?

A good node has neighbors of low degree. Likely to be removed when neighbor joins MIS!

## Good Nodes

A good node  $v$  will be removed in Step 3 with probability

$$p \geq 1/36.$$

Proof?

# Analysis (1)

„Assets“:

## Proof („Good Nodes“).

Goal:

### Good Nodes

A good node  $v$  will be removed in Step 3 with probability  
 $p \geq 1/36$ .

### Joining MIS

Node  $v$  joins MIS in Step 2 with probability  $p \geq 1/(4d(v))$ .

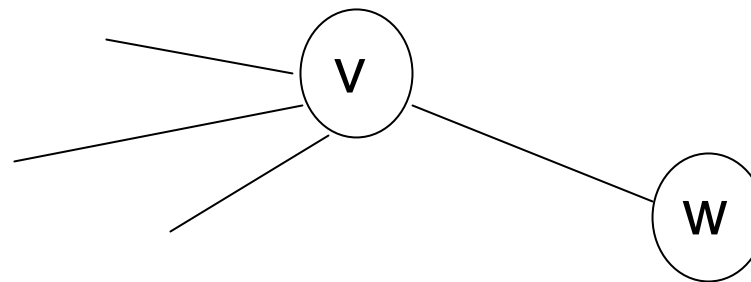
### Good&Bad Nodes

A node  $v$  is called *good* if

$$\sum_{w \in N(v)} 1/(2d(w)) \geq 1/6.$$

If  $v$  has a neighbor  $w$  with  $d(w) \leq 2$ ?

Done: „Joining MIS“ lemma implies that prob. to remove at least  $1/8$  since neighbor  $w$  will join...



So let's focus on neighbors with degree **at least 3**: thus for any neighbor  $w$  of  $v$  we have  $1/(2d(w)) \leq 1/6$ .

# Analysis (2)

---

„Assets“:

## Proof („Good Nodes“).

Goal:

### Good Nodes

A good node  $v$  will be removed in Step 3 with probability  
 $p \geq 1/36$ .

### Joining MIS

Node  $v$  joins MIS in Step 2 with probability  $p \geq 1/(4d(v))$ .

### Good&Bad Nodes

A node  $v$  is called *good* if

$$\sum_{w \in N(v)} 1/(2d(w)) \geq 1/6.$$

**So neighbors have degree at least 3...**

Then, for a good node  $v$ , there must be a subset  $S \subseteq N(v)$  such that

$$1/6 \leq \sum_{w \in S} 1/(2d(w)) \leq 1/3.$$

Why?

By taking all neighbors we have at least  $1/6$  (Definition), and we can remove individual nodes with a granularity of at least  $1/6$  (degree at least 3).

# Analysis (3)

## Good Nodes

A good node  $v$  will be removed in Step 3 with probability

$$p \geq 1/36.$$

## Proof („Good Nodes“).

Let  $R$  be event that  $v$  is removed (e.g., if neighbor joins MIS).

$$P[R] \geq P[\exists u \in S, u \in \text{MIS}] \quad // \text{ removed e.g., if neighbor joins}$$

$$\geq \sum_{u \in S} P[u \in \text{MIS}] - \sum_{u, w \in S} P[u \in \text{MIS and } w \in \text{MIS}] \quad // \text{ why?}$$

By truncating the inclusion-exclusion principle...:

Probability that there is one is sum of probability for all individual minus probability that two enter, plus...

just derived!

$$P[R] \geq \sum_{u \in S} P[u \in \text{MIS}] - \sum_{u, w \in S; u \neq w} P[u \in M \text{ and } w \in M]$$

$$\geq \sum_{u \in S} P[u \in \text{MIS}] - \sum_{u \in S} \sum_{w \in S} P[u \in M] \cdot P[w \in M]$$

independent but count same node double in sum...

see Joining MIS lemma

$$\geq \sum_{u \in S} \frac{1}{4d(u)} - \sum_{u \in S} \sum_{w \in S} \frac{1}{2d(u)} \frac{1}{2d(w)}$$

see algorithm

$$\geq \sum_{u \in S} \frac{1}{2d(u)} \left( \frac{1}{2} - \sum_{w \in S} \frac{1}{2d(w)} \right) \geq \frac{1}{6} \left( \frac{1}{2} - \frac{1}{3} \right) = \frac{1}{36}.$$

QED



## Good Nodes

**We just proved:**

A good node  $v$  will be removed in Step 3 with probability

$$p \geq 1/36.$$

Cool, good nodes have  
Constant probability! 😊

But what now?

What does it help?

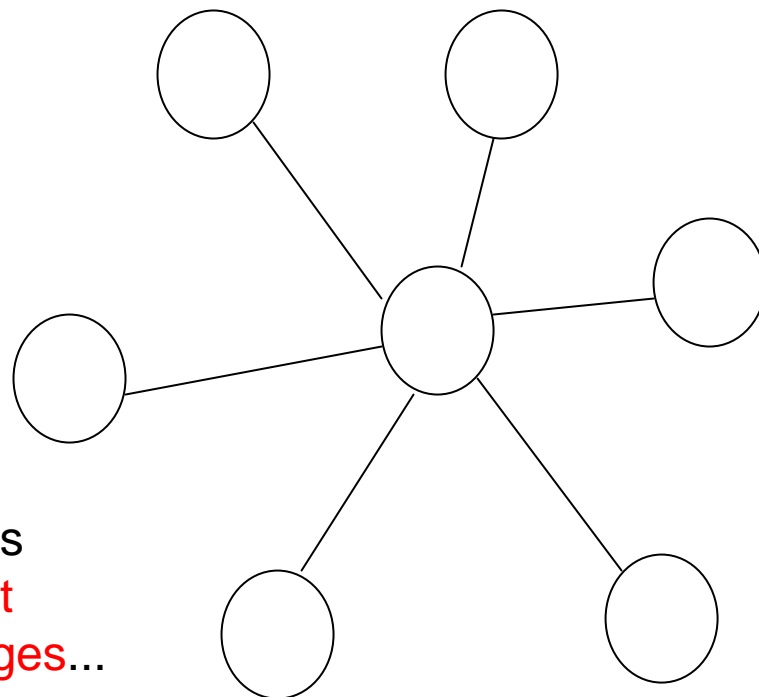
Are many nodes good in a graph?

Example: in star graph,  
only single node is good... 😞

But: there are many „good edges“...

How to define good edges?

Idea: edge is removed if either of its endpoints  
are removed! So good if **at least one endpoint**  
is a good node! And there are **many such edges**...



Proceed in rounds consisting of phases

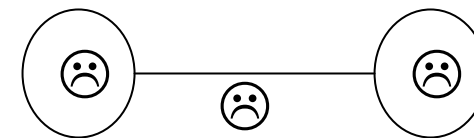
In a phase:

1. each node  $v$  marks itself with probability  $1/2d(v)$  where  $d(v)$  denotes the current degree of  $v$
2. if no higher degree neighbor is marked,  $v$  joins MIS; otherwise,  $v$  unmarks itself again (break ties arbitrarily)
3. delete all nodes that joined the MIS plus their neighbors, as they cannot join the MIS anymore

## Good&Bad Edges

An edge  $e=(u,v)$  called *bad* if both  $u$  and  $v$  are bad (not good). Else the edge is called good.

A bad edge is incident to two nodes with neighbors of high degrees.



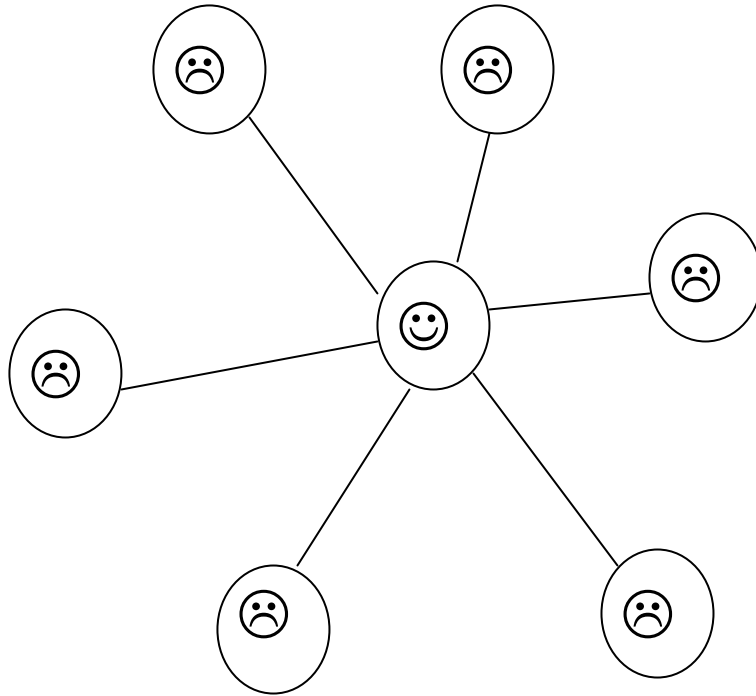
## Good Edges

At least half of all edges are good, at any time.

Proof?

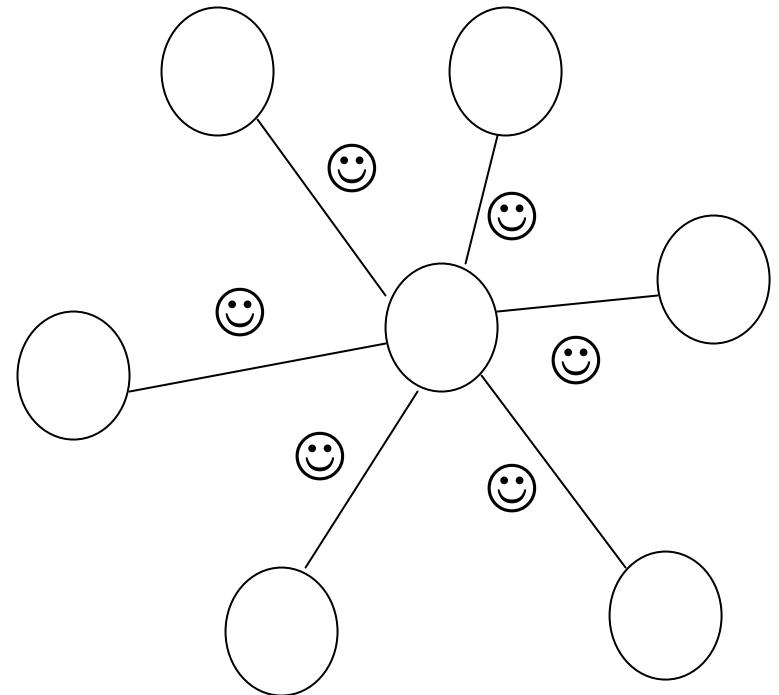
# Analysis

---



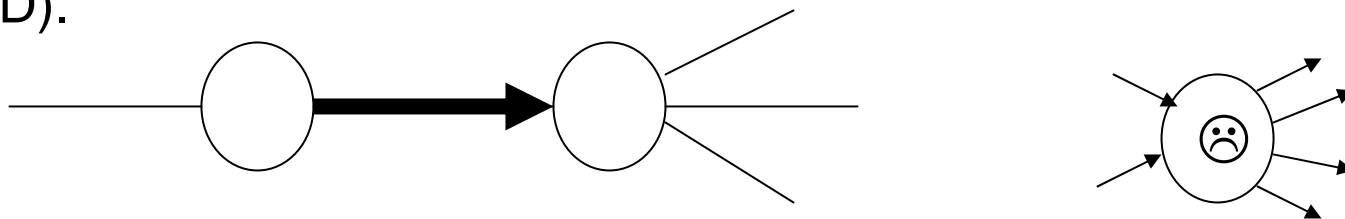
... but many good edges!

Not many good nodes...



# Analysis

Idea: Construct an **auxiliary graph**! Direct each edge **towards higher** degree node (if both nodes have same degree, point it to one with higher ID).



## Helper Lemma

A bad node  $v$  has out-degree at least twice its indegree.

### Proof („Helper Lemma“).

Assume the opposite: at least one third of the neighbor nodes (let's call them  $S$ ) have degree at most  $d(v)$ . But then

$$\sum_{w \in N(v)} \frac{1}{2d(w)} \geq \sum_{w \in S} \frac{1}{2d(w)} \geq \sum_{w \in S} \frac{1}{2d(v)} \geq \frac{d(v)}{3} \frac{1}{2d(v)} = \frac{1}{6}$$

only subset...      Def. of S      Assumption

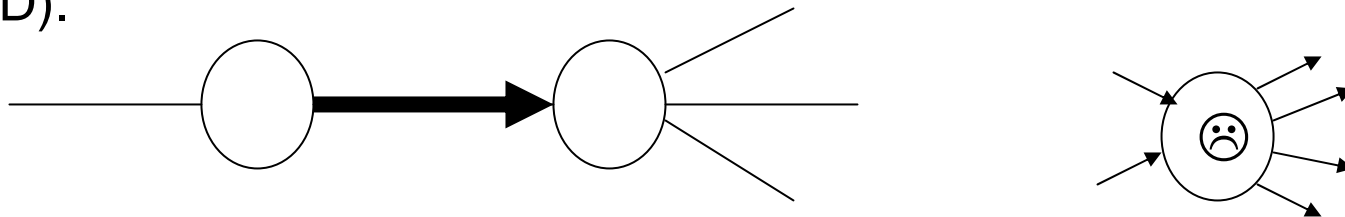
**Contradiction:**  
v would be good!

QED

# Analysis

---

Idea: Construct an **auxiliary graph**! Direct each edge **towards higher** degree node (if both nodes have same degree, point it to one with higher ID).



## Helper Lemma

A bad node  $v$  has out-degree at least twice its indegree.

### So what?

The number of edges into bad nodes can be at most half the number of all edges!

So at least half of all edges are directed **into good nodes**!

And they are good! 😊

## Fast MIS (1986)

Fast MIS terminates in expected time  $O(\log n)$ .

### Proof („Fast MIS“)?

We know that a good node will be deleted with constant probability in Step 3. And with it, a good edge (by definition)!

Since at least half of all the edges are good (and thus have at least one good incident node), a **constant fraction of edges** will be deleted in each phase.

(Note that  $O(\log m) = O(\log n)$ .)

QED

# Back to the future: Fast MIS from 2009...!

Even simpler algorithm!

## Fast MIS (2009)

Proceed in rounds consisting of phases

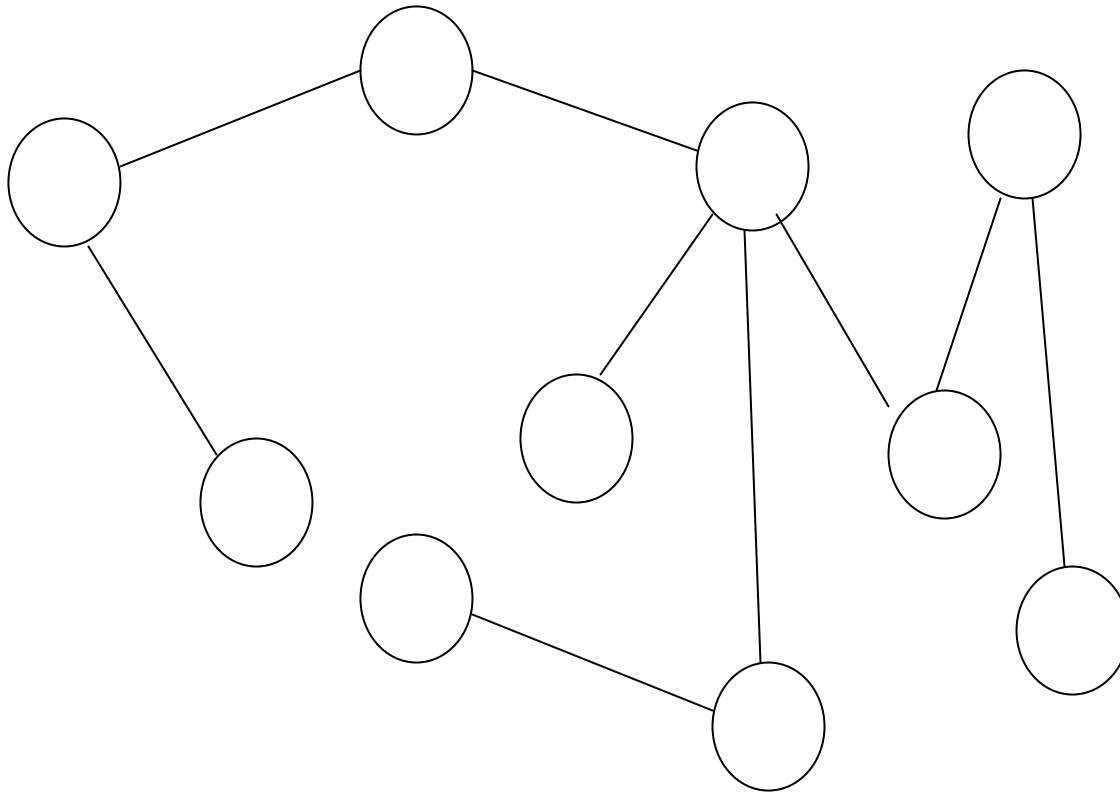
In a **phase**:

1. each node chooses a random value  $r(v) \in [0,1]$  and sends it to its neighbors.
2. If  $r(v) < r(w)$  for all neighbors  $w \in N(v)$ , node  $v$  enters the MIS and informs the neighbors
3. If  $v$  or a neighbor of  $v$  entered the MIS,  $v$  **terminates** (and  $v$  and edges are **removed**), otherwise  $v$  enters next phase!



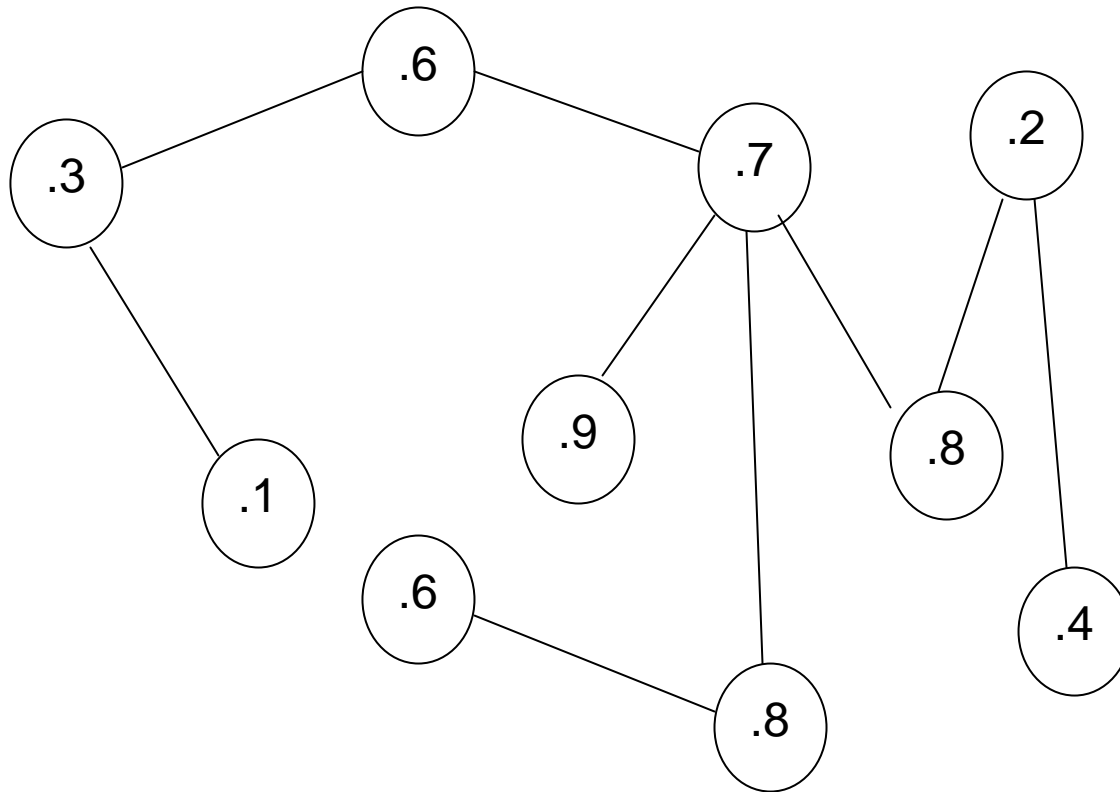
# Fast MIS from 2009...

---



# Fast MIS from 2009...

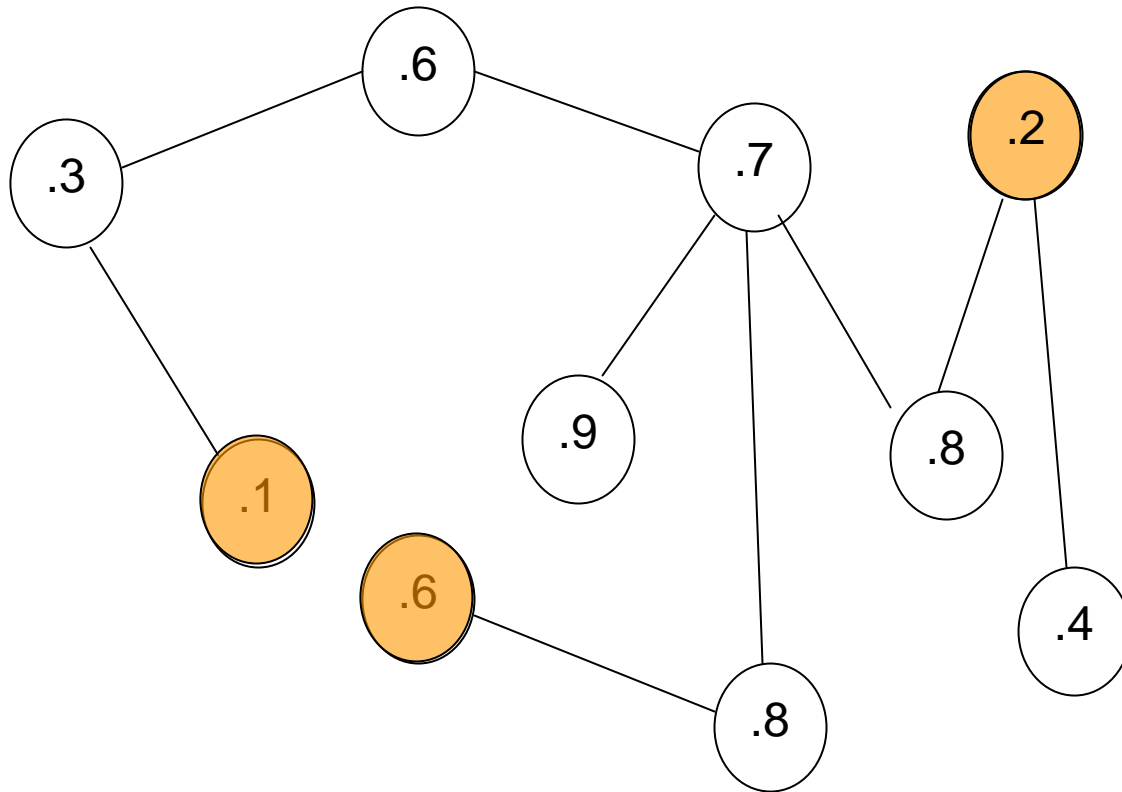
---



**Choose random values!**

# Fast MIS from 2009...

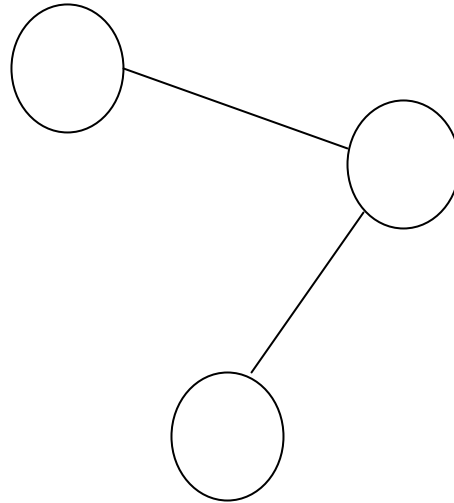
---



**Min in neighborhood => IS!**

# Fast MIS from 2009...

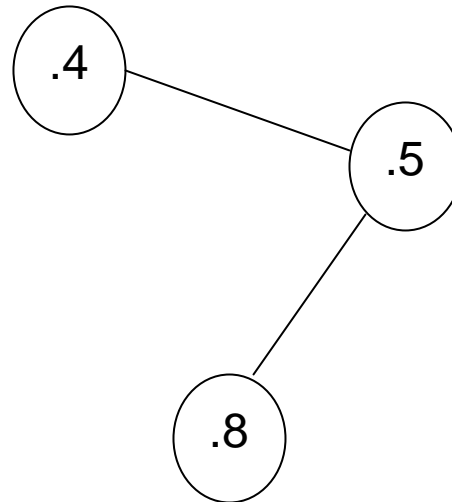
---



**Remove neighborhoods...**

# Fast MIS from 2009...

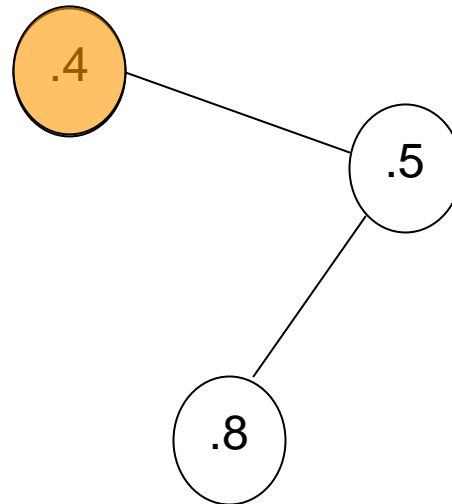
---



**Choose random values!**

# Fast MIS from 2009...

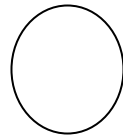
---



**Min in neighborhood  $\Rightarrow$  IS!**

# Fast MIS from 2009...

---



**Remove neighborhoods...**

# Fast MIS from 2009...

---

.1

**Choose random values!**



# Fast MIS from 2009...

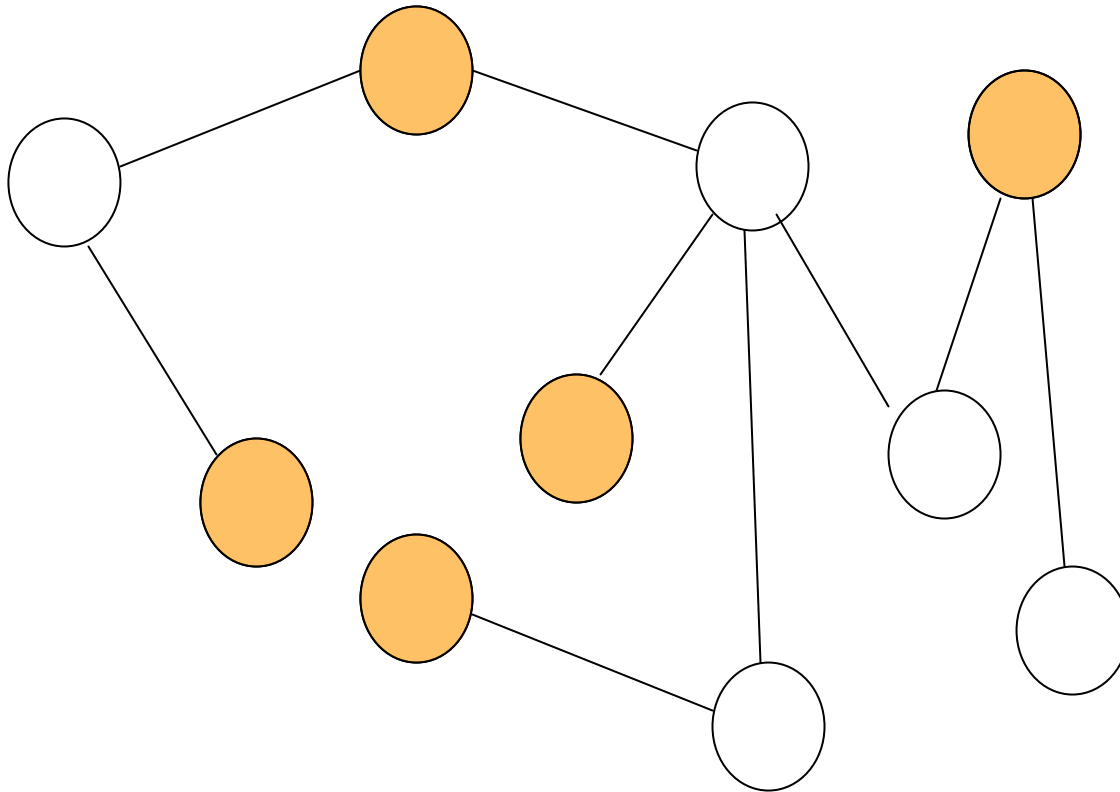
---

.1

**lowest value => IS**

# Fast MIS from 2009...

---



**... done: MIS!**

## Fast MIS (2009)

Proceed in rounds consisting of phases

In a **phase**:

1. each node chooses a random value  $r(v) \in [0,1]$  and sends it to its neighbors.
2. If  $r(v) < r(w)$  for all neighbors  $w \in N(v)$ , node  $v$  enters the MIS and informs the neighbors
3. If  $v$  or a neighbor of  $v$  entered the MIS,  $v$  **terminates** (and  $v$  and edges are **removed**), otherwise  $v$  enters next phase!

Why is it correct? Why IS?

Step 2: if  $v$  joins, neighbors do not

Step 3: if  $v$  joins, neighbors will *never* join again

## Fast MIS (2009)

Proceed in rounds consisting of phases

In a **phase**:

1. each node chooses a random value  $r(v) \in [0,1]$  and sends it to its neighbors.
2. If  $r(v) < r(w)$  for all neighbors  $w \in N(v)$ , node  $v$  enters the MIS and informs the neighbors
3. If  $v$  or a neighbor of  $v$  entered the MIS,  $v$  **terminates** (and  $v$  and edges are **removed**), otherwise  $v$  enters next phase!

## Why MIS?

Node with largest random value will always join the MIS, so there is always progress.

## Fast MIS (2009)

Proceed in rounds consisting of phases

In a **phase**:

1. each node chooses a random value  $r(v) \in [0,1]$  and sends it to its neighbors.
2. If  $r(v) < r(w)$  for all neighbors  $w \in N(v)$ , node  $v$  enters the MIS and informs the neighbors
3. If  $v$  or a neighbor of  $v$  entered the MIS,  $v$  **terminates** (and  $v$  and edges are **removed**), otherwise  $v$  enters next phase!

Runtime?

# Analysis: Recall „Linearity of Expectation“

---

**Theorem 5.9** (Linearity of Expectation). *Let  $X_i$ ,  $i = 1, \dots, k$  denote random variables, then*

$$\mathbb{E} \left[ \sum_i X_i \right] = \sum_i \mathbb{E} [X_i].$$


*Proof.* It is sufficient to prove  $\mathbb{E} [X + Y] = \mathbb{E} [X] + \mathbb{E} [Y]$  for two random variables  $X$  and  $Y$ , because then the statement follows by induction. Since

$$\begin{aligned} P[(X, Y) = (x, y)] &= P[X = x] \cdot P[Y = y | X = x] \\ &= P[Y = y] \cdot P[X = x | Y = y] \end{aligned}$$

we get that

$$\begin{aligned} \mathbb{E} [X + Y] &= \sum_{(X, Y) = (x, y)} P[(X, Y) = (x, y)] \cdot (x + y) \\ &= \sum_{X=x} \sum_{Y=y} P[X = x] \cdot P[Y = y | X = x] \cdot x \\ &+ \sum_{Y=y} \sum_{X=x} P[Y = y] \cdot P[X = x | Y = y] \cdot y \\ &= \sum_{X=x} P[X = x] \cdot x + \sum_{Y=y} P[Y = y] \cdot y \\ &= \mathbb{E} [X] + \mathbb{E} [Y]. \end{aligned}$$

We sum over all possible  $y$  values for a given  $x$ , so  $=1$



# Analysis? (1)

---

We want to show that also this algorithm has logarithmic runtime! How?

Idea: if per phase a constant fraction of node disappeared, it would hold! (Recall definition of logarithm...)

Again: this is not true unfortunately... ☹️

Alternative proof? Similar to last time?

Show that any *edge* disappears with constant probability!

But also this does not work: edge does not have **constant probability** to be removed!

But maybe edges still vanish quickly...?

Let's estimate the **number of disappearing edges** per round again!

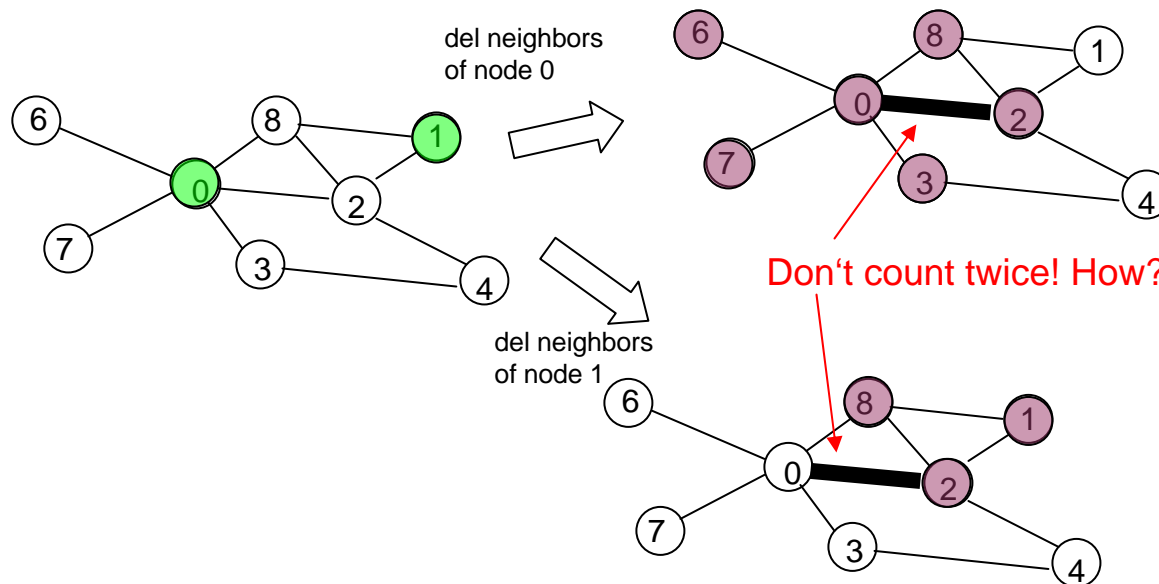
# Analysis? (2)

## Probability of a node $v$ to enter MIS?

Probability = node  $v$  has largest ID in neighborhood, so at least  $1/(d(v)+1)$ ...

... also  $v$ 's neighbors' edges will disappear with this probability, so more than  $d(v)$  edges go away with this probability!

But let's make sure we do not double count edges!



**Idea: only count edges from a neighbor  $w$  when  $v$  is the smallest value even in  $w$ 's neighborhood!  
It's a subset only, but sufficient!**



# Edge Removal: Analysis (1)

## Edge Removal

In expectation, we remove at least half of all the edges in any phase.

### Proof („Edge Removal“)?

Consider the graph  $G=(V,E)$ , and assume **v joins MIS** (i.e.,  $r(v) < r(w)$  for all neighbors  $w$ ).

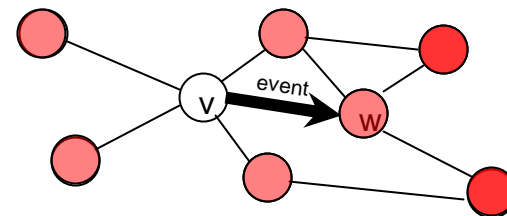
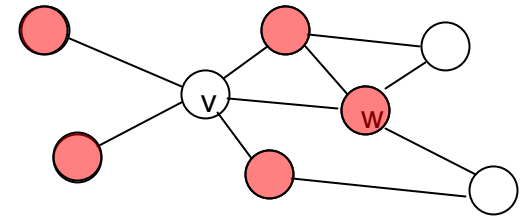
If in addition, it holds that  $r(v) < r(x)$  for all neighbors  $x$  of a neighbor  $w$ , we call this **event (v => w)**.

What is the probability of this event (that  $v$  is minimum also for neighbors of the given neighbor)?

$$P [(v \Rightarrow w)] \geq 1/(d(v)+d(w)),$$

since  $d(v)+d(w)$  is the maximum possible number of nodes adjacent to  $v$  and  $w$ .

If  $v$  joins MIS, all edges  $(w,x)$  will be removed; there are at least  $d(w)$  many.



# Edge Removal: Analysis (2)

## Edge Removal

In expectation, we remove at least half of all the edges in any phase.

### Proof („Edge Removal“)?

How many edges are removed?

Let  $X_{(v \Rightarrow w)}$  denote random variable for number of edges adjacent to  $w$  removed due to event  $(v \Rightarrow w)$ . If  $(v \Rightarrow w)$  occurs,  $X_{(v \Rightarrow w)}$  has value  $d(w)$ , otherwise 0.

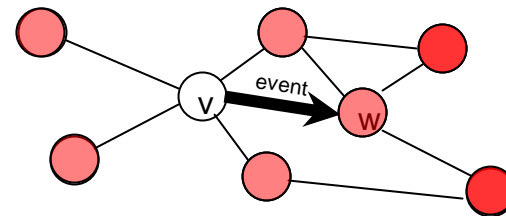
Let  $X$  denote the sum of all these random variables.

So:

$$\begin{aligned}\mathbb{E}[X] &= \sum_{\{v,w\} \in E} \mathbb{E}[X_{(v \rightarrow w)}] + \mathbb{E}[X_{(w \rightarrow v)}] \\ &= \sum_{\{v,w\} \in E} P[\text{Event } (v \rightarrow w)] \cdot d(w) + P[\text{Event } (w \rightarrow v)] \cdot d(v) \\ &\geq \sum_{\{v,w\} \in E} \frac{d(w)}{d(v) + d(w)} + \frac{d(v)}{d(w) + d(v)} \\ &= \sum_{\{v,w\} \in E} 1 = |E|.\end{aligned}$$

So all edges gone in one phase?!

We still overcount!



## Edge Removal: Analysis (3)

### Edge Removal

In expectation, we remove at least half of all the edges in any phase.

#### Proof („Edge Removal“)?

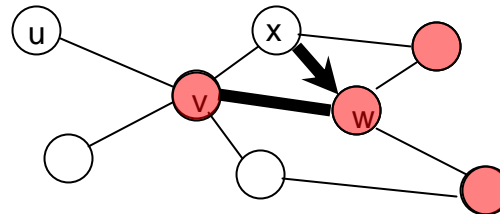
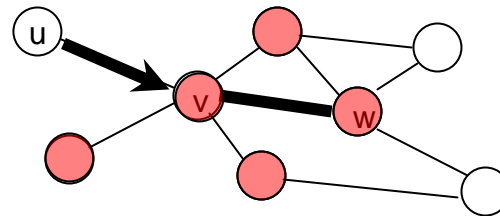
We still overcount:

Edge  $\{v,w\}$  may be counted twice:  
for event  $(u \Rightarrow v)$  and event  $(x \Rightarrow w)$ .

However, it cannot be more than twice, as  
there is at most one event  $(* \Rightarrow v)$  and  
at most one event  $(* \Rightarrow w)$ :

Event  $(u \Rightarrow v)$  means  $r(u) < r(w)$  for all  
 $w \in N(v)$ ; another  $(u' \Rightarrow v)$  would imply  
that  $r(u') > r(u) \in N(v)$ .

**So at least half of all edges vanish!**



**QED**

## **MIS of 2009**

Expected running time is  $O(\log n)$ .

### **Proof („MIS 2009“)?**

Number of edges is cut in two in each round...

**QED**

Actually, the claim even holds with high probability! (see „Skript“)

## Matching

A matching is a **subset  $M$  of edges  $E$**  such that no two edges in  $M$  are adjacent.

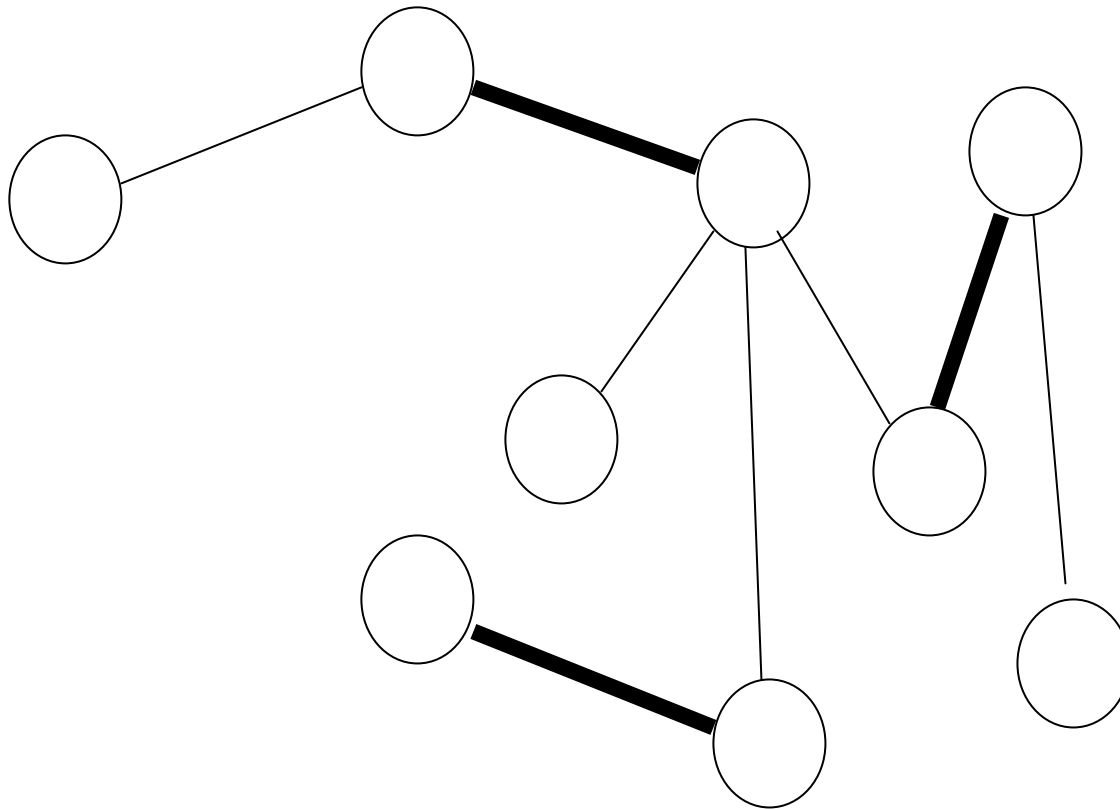
A **maximal** matching cannot be augmented.

A **maximum** matching is the best possible.

A **perfect** matching includes all nodes.

# Excursion: Matchings

---

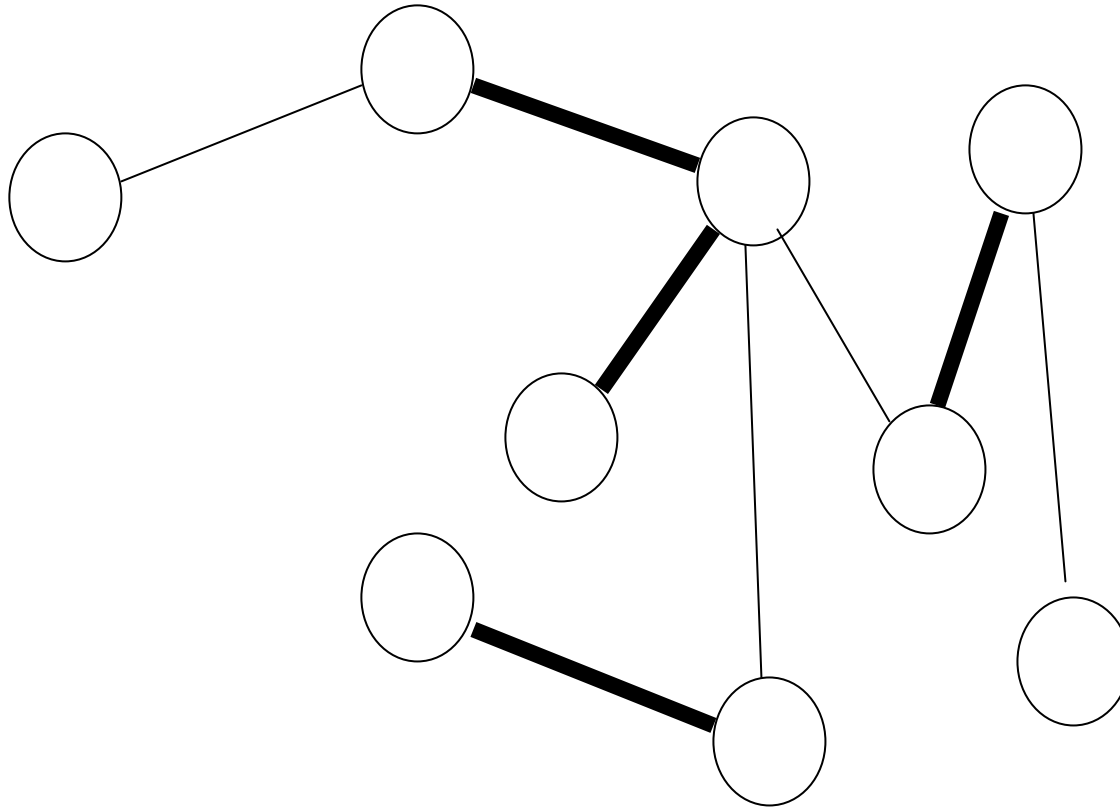


Matching? Maximal? Maximum? Perfect?

Maximal.

# Excursion: Matchings

---

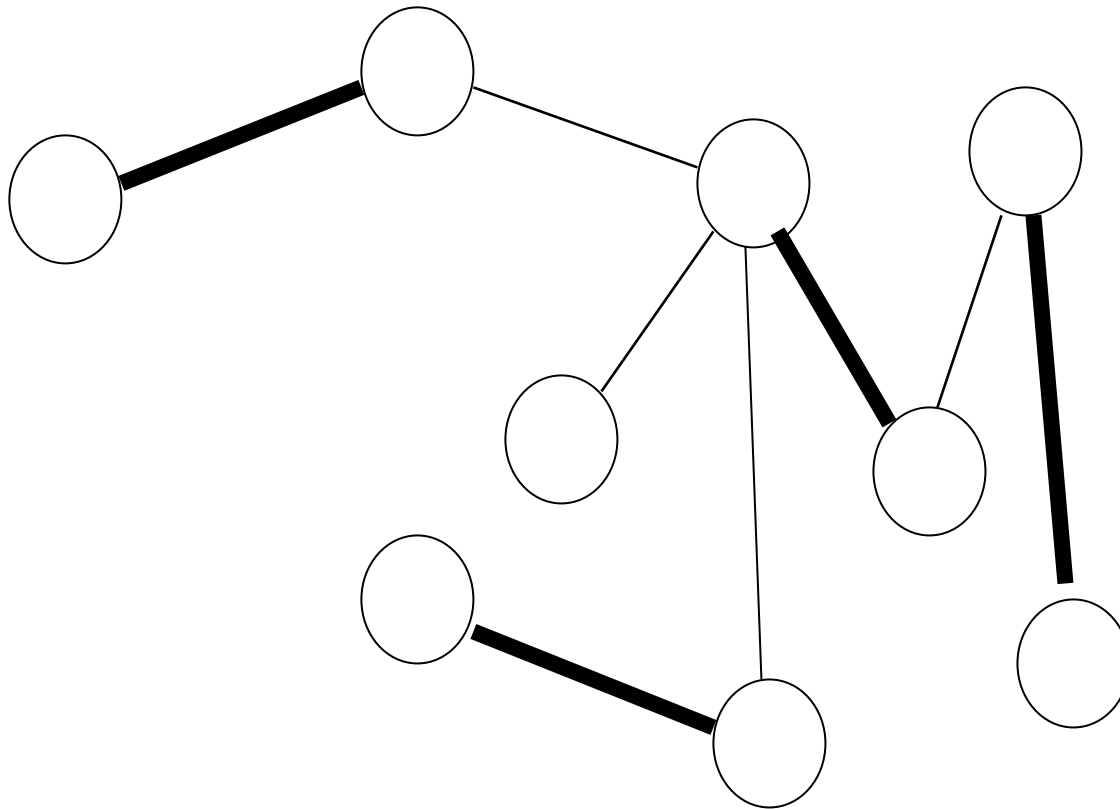


Matching? Maximal? Maximum? Perfect?

Nothing.

# Excursion: Matchings

---



Matching? Maximal? Maximum? Perfect?

Maximum but not perfect.



## Matching

A matching is a **subset  $M$  of edges  $E$**  such that no two edges in  $M$  are adjacent.

A **maximal** matching cannot be augmented.

A **maximum** matching is the best possible.

A **perfect** matching includes all nodes.

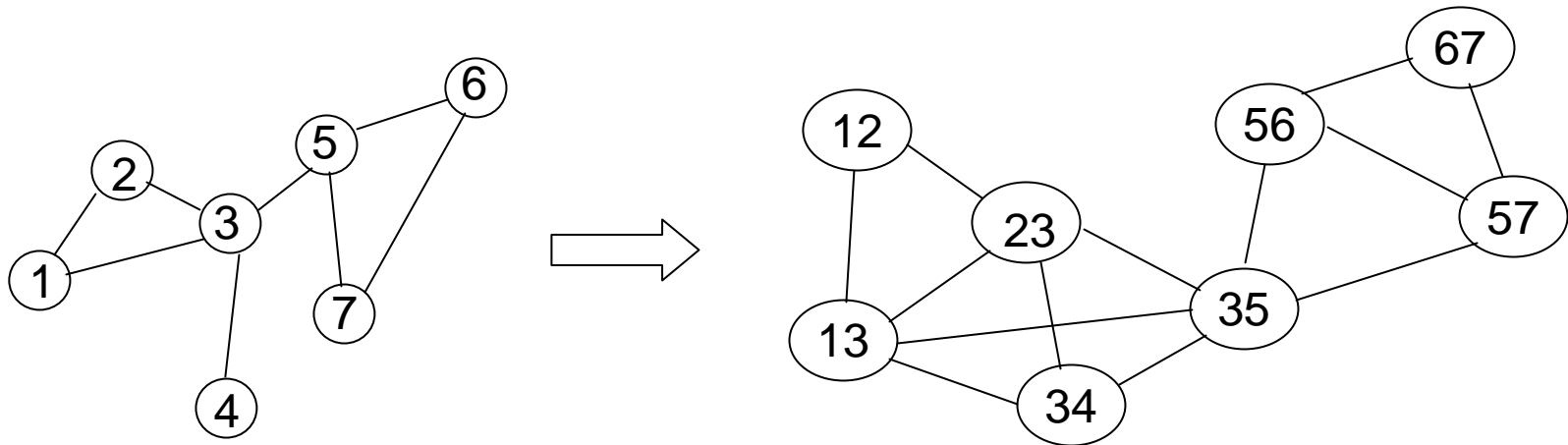
How to compute with an IS algorithm?

# Discussion: Matching

---

An IS algorithm is a matching algorithm! How?

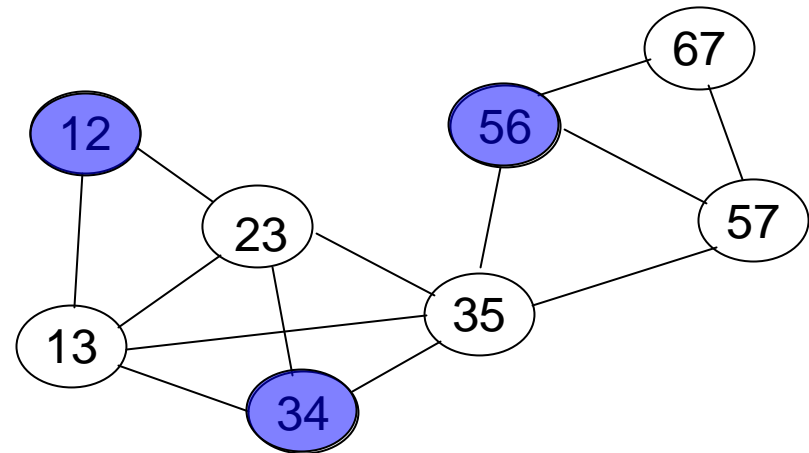
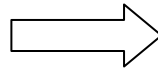
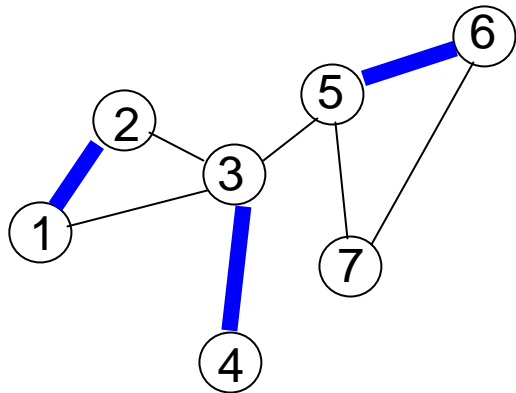
For each edge in original graph make vertex, connect vertices if their edges are adjacent.



# Discussion: Matching

---

MIS = maximal matching: matching does not have adjacent edges!



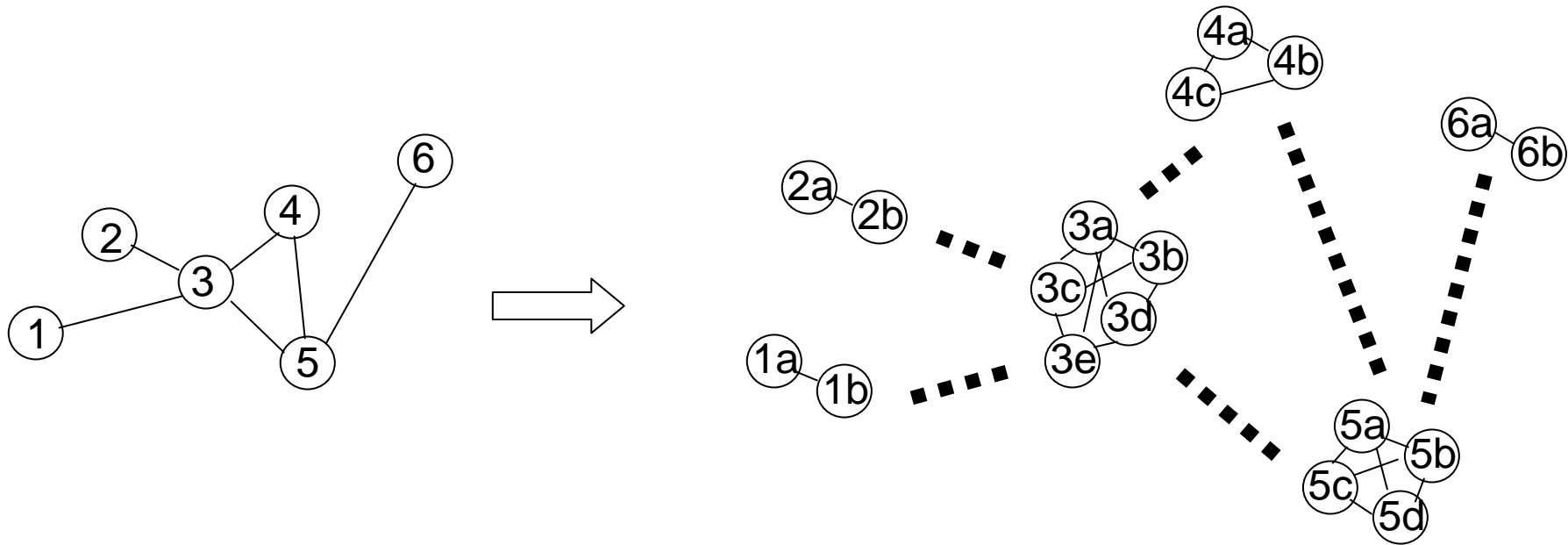
# Discussion: Graph Coloring

How to use a MIS algorithm for graph coloring?

How to use a MIS algorithm for graph coloring?

**Clone each node**  $v$ ,  $d(v)+1$  many times. Connect clones completely and **edges from  $i$ -th clone to  $i$ -th clone**. Then?

Run MIS: if  $i$ -th copy is in MIS, node gets color  $i$ .

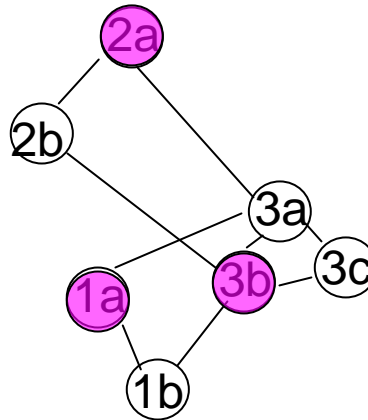
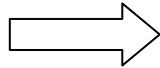
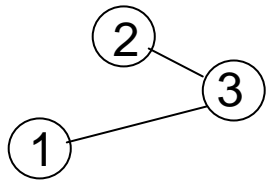


# Discussion: Graph Coloring

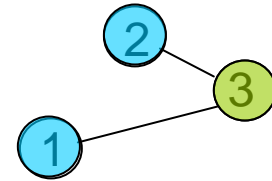
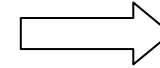
---

Example:

How to use a MIS algorithm for graph coloring?



a => blue  
b => green

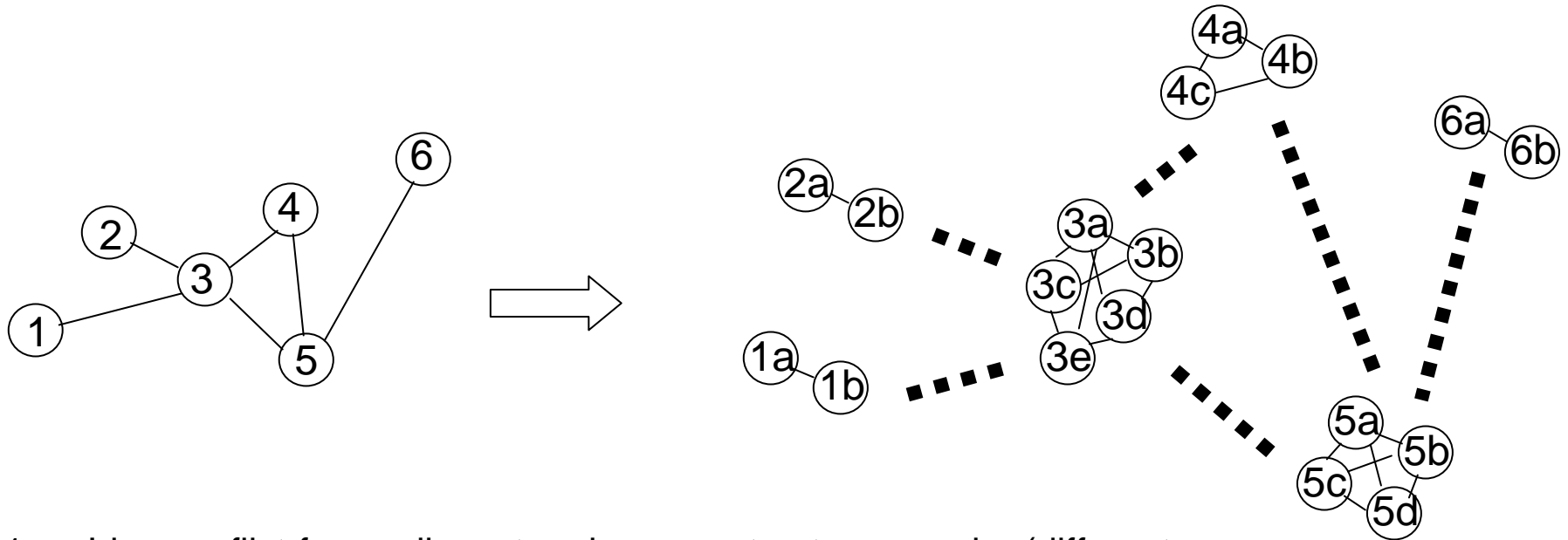


# MIS

# Coloring

# Discussion: Graph Coloring

Why does it work?



1. Idea conflict-free: adjacent nodes cannot get same color (different index in MIS, otherwise adjacent!), and each node has at most one clone in IS, so valid.
2. Idea colored: each node gets color, i.e., each node has a clone in IS: there are only  $d(v)$  neighbor clusters, but our cluster has  $d(v)+1$  nodes...

## Dominating Set

A subset  $D$  of nodes such that each node either is in the dominating set itself, or one of its neighbors is (or both).

How to compute a dominating set?  
See Skript. 😊

Literature for further reading:

- Peleg's book (as always 😊 )

End of lecture