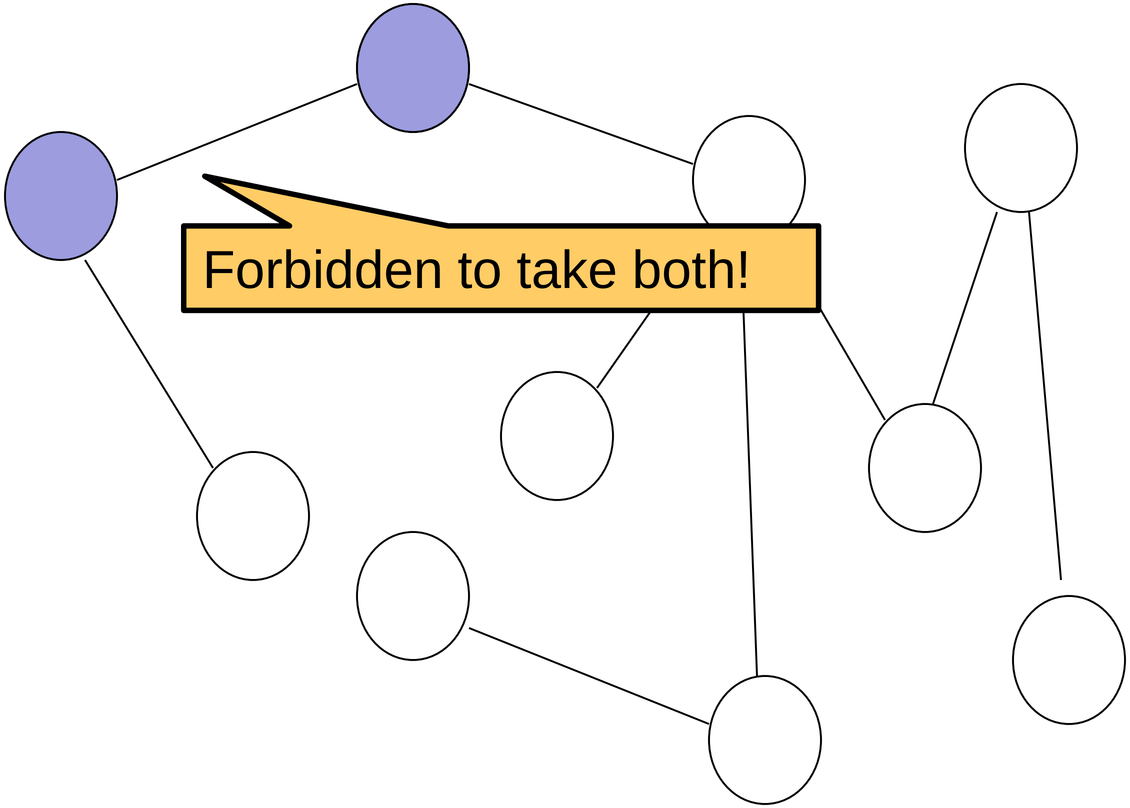
GIAN Course on Distributed Network Algorithms

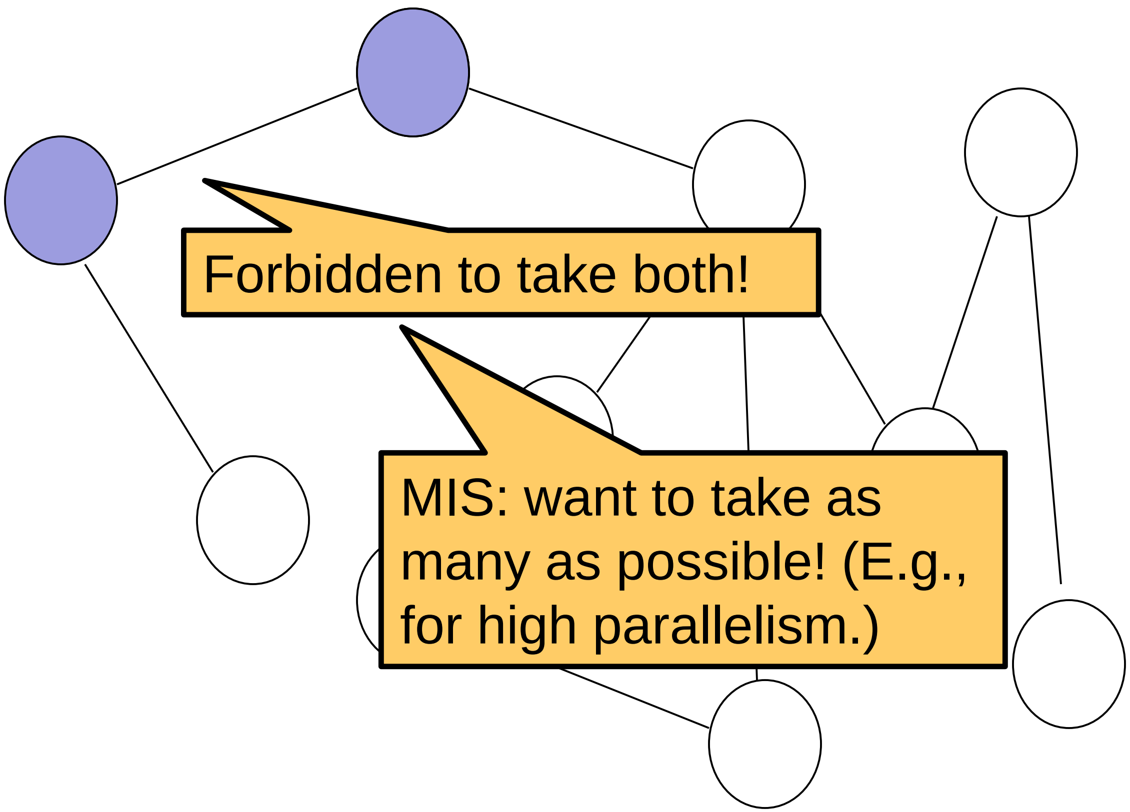
The Power of Randomization

Case Study: Independent Sets

Independent Sets: Set of Non-Neighboring Nodes



Independent Sets: Set of Non-Neighboring Nodes



Case Study: Maximal Independent Sets

Formally:

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.

Case Study: Maximal Independent Sets

Formally:

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.

In non-distributed setting: MIS is easy, MaxIS is hard!

Case Study: Maximal Independent Sets

Formally:

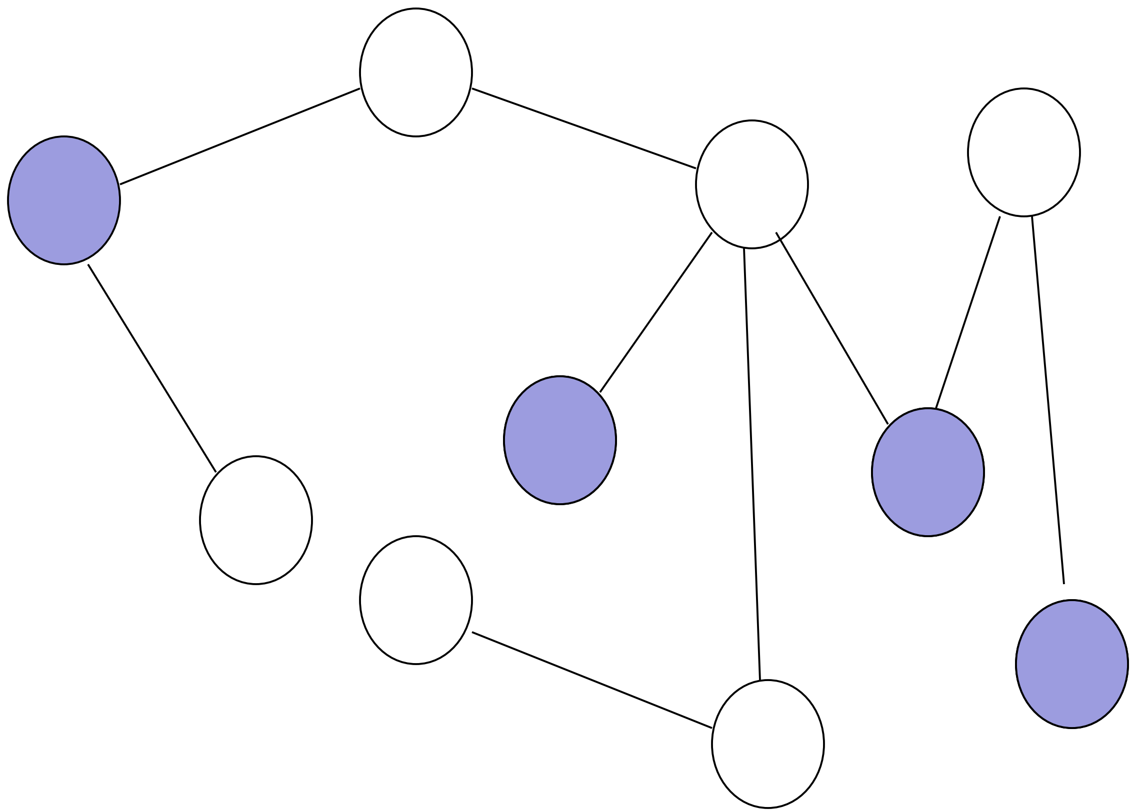
MIS

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.

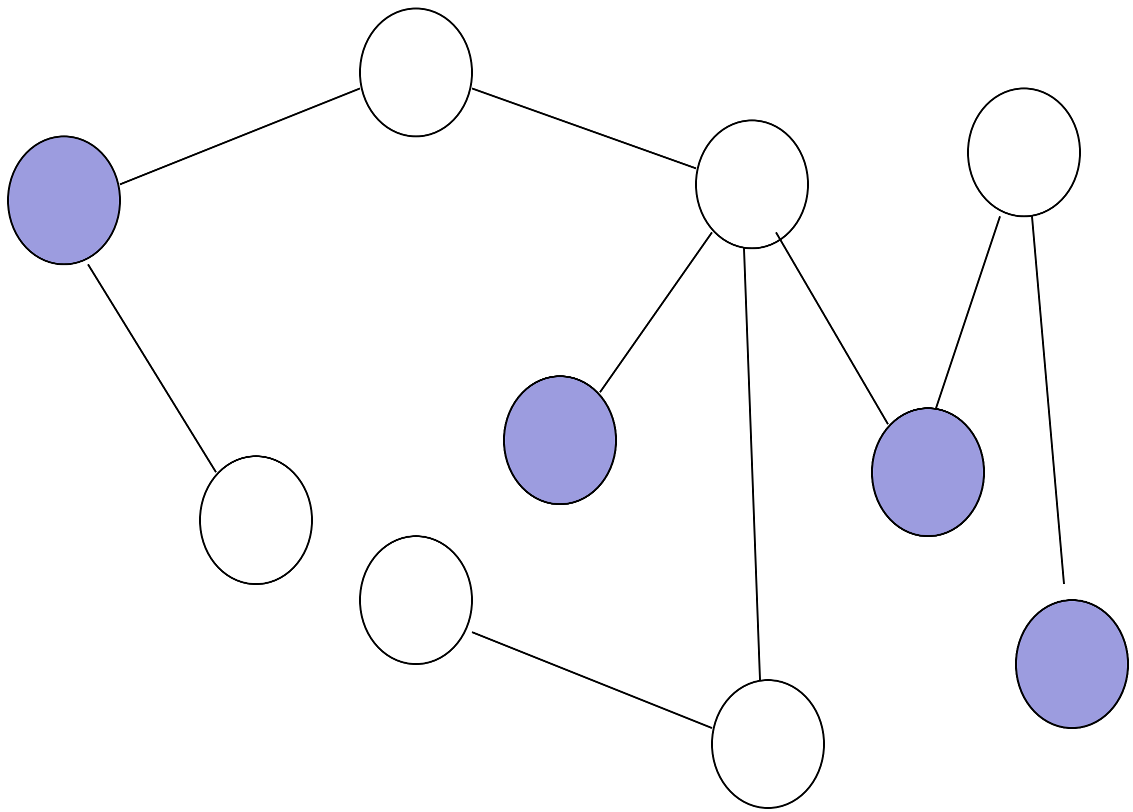
In non-distributed setting: MIS is easy, MaxIS is hard!

Applications: similar to coloring, e.g., symmetry breaking and parallelism. Also building block to compute matchings and load-balancing.

Nothing, IS, MIS, MaxIS?

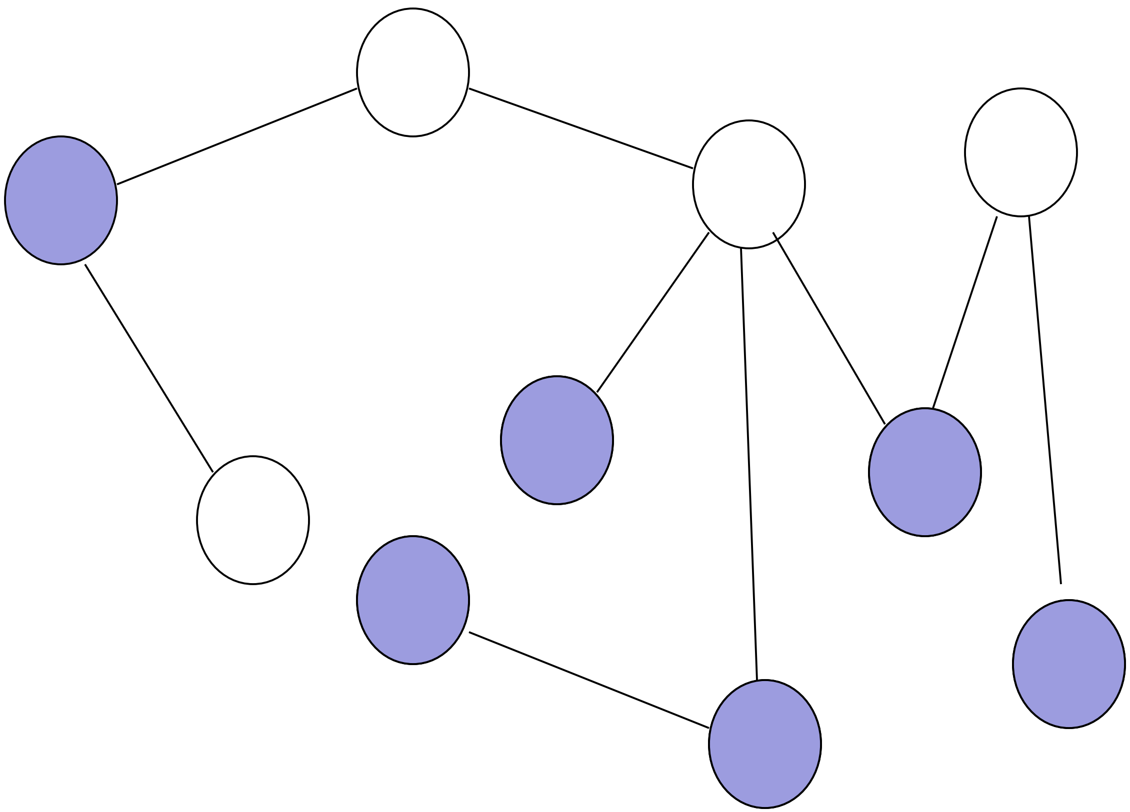


Nothing, IS, MIS, MaxIS?

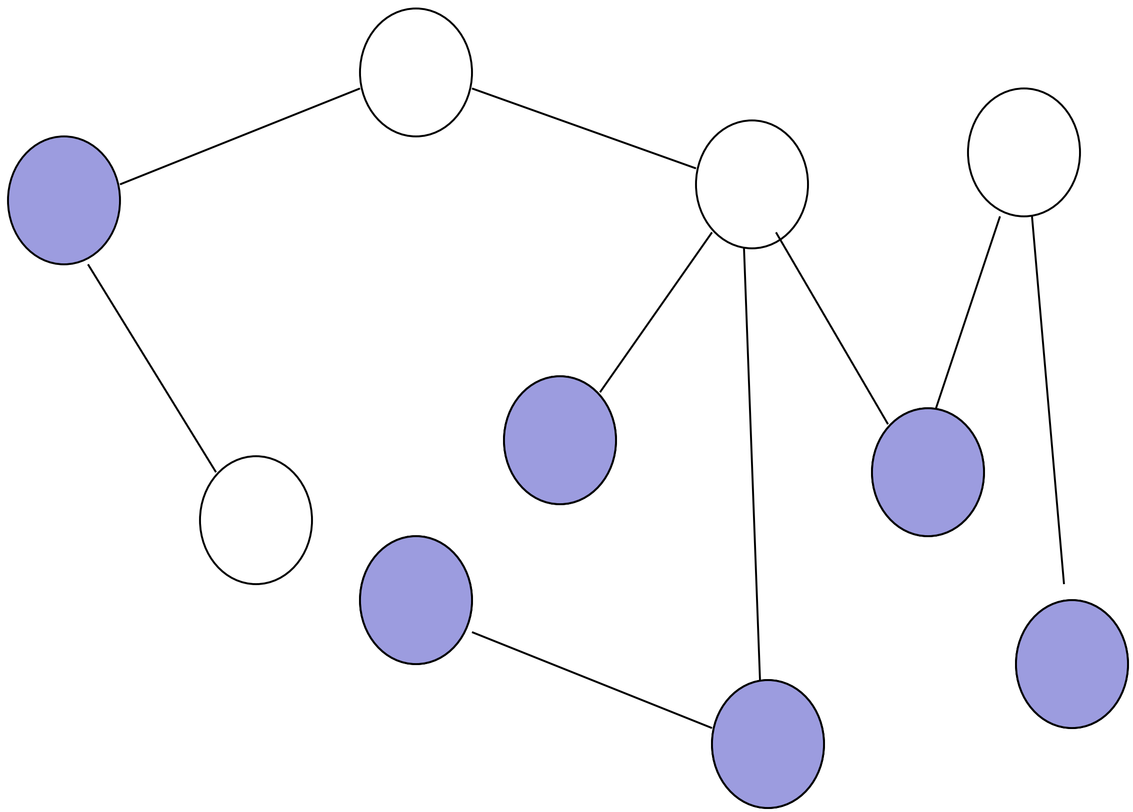


IS but not MIS.

Nothing, IS, MIS, MaxIS?

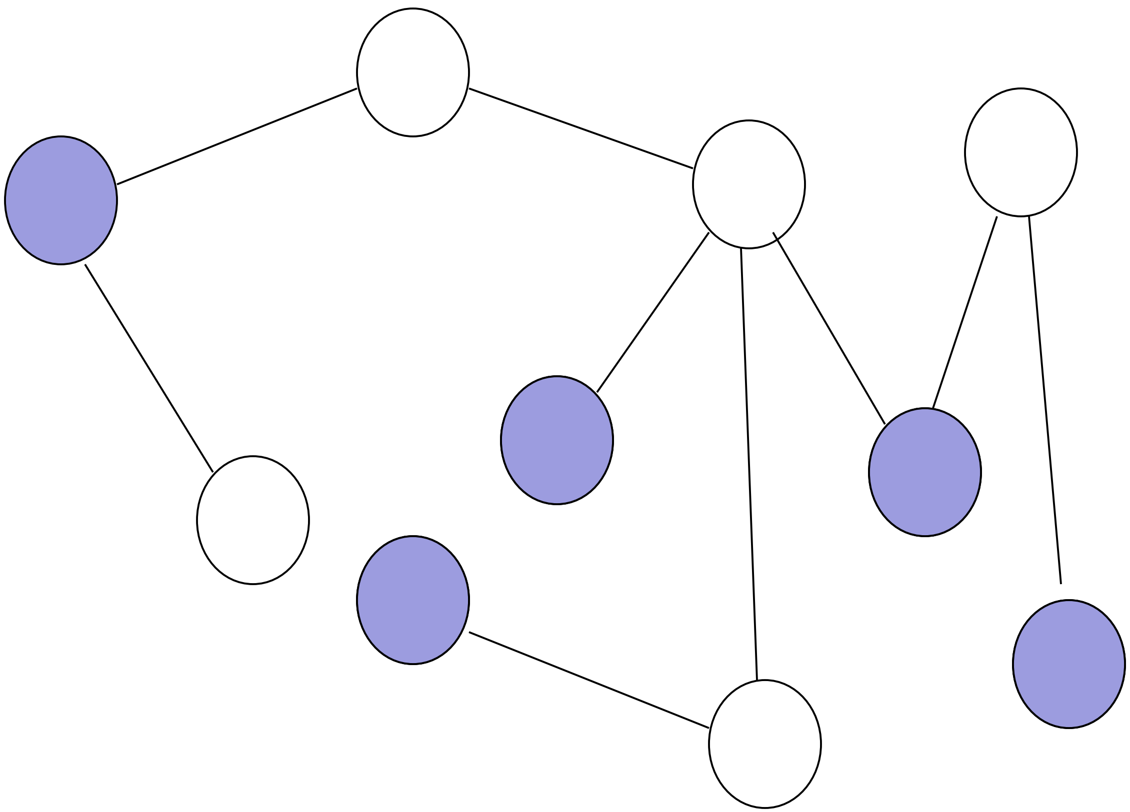


Nothing, IS, MIS, MaxIS?

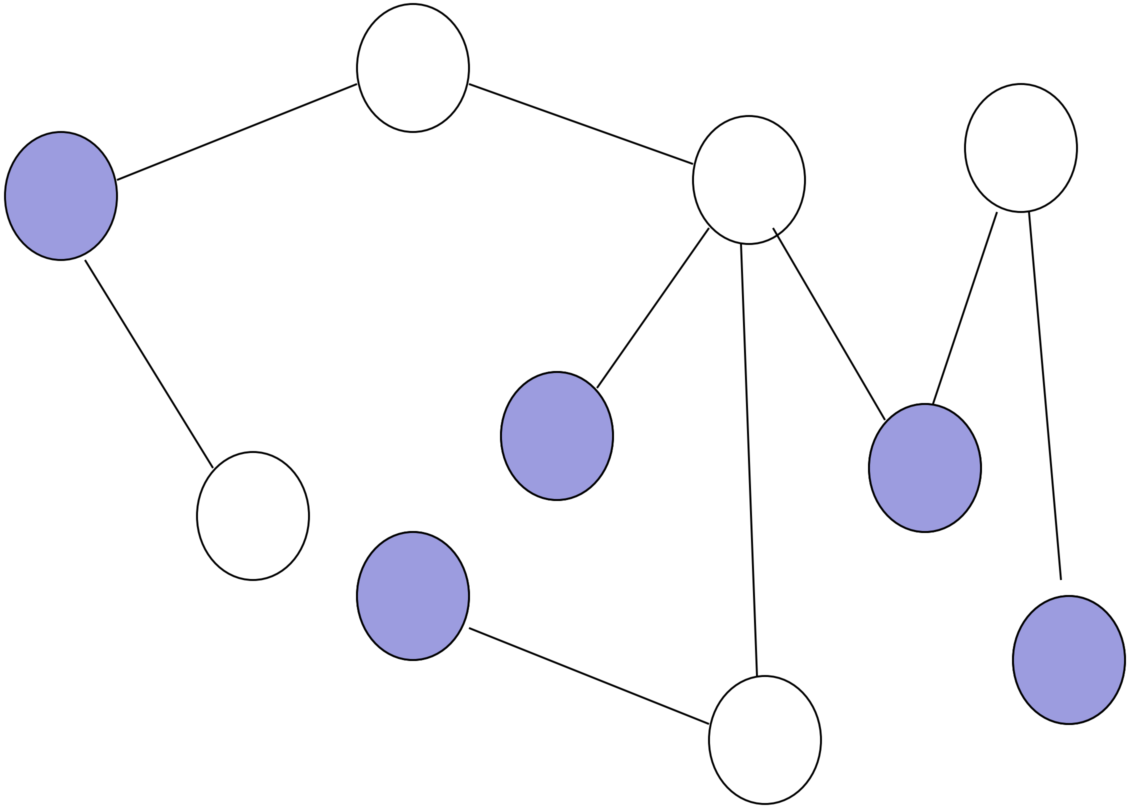


Nothing.

Nothing, IS, MIS, MaxIS?

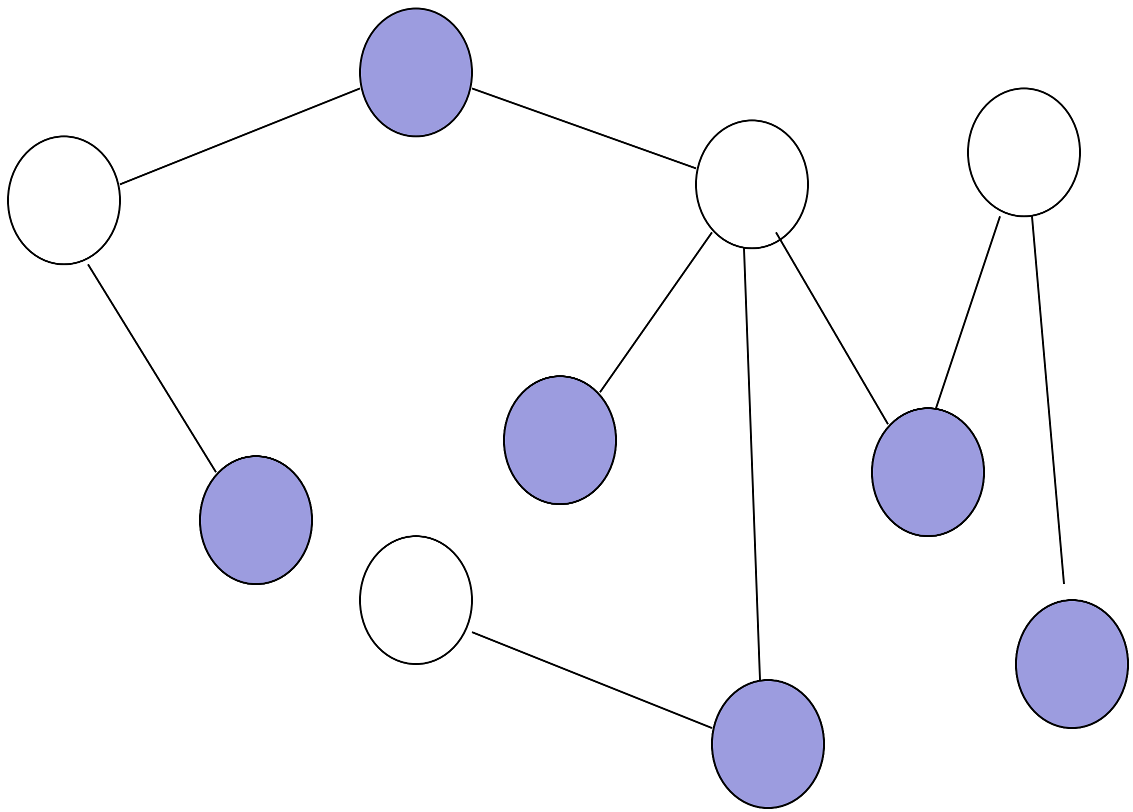


Nothing, IS, MIS, MaxIS?

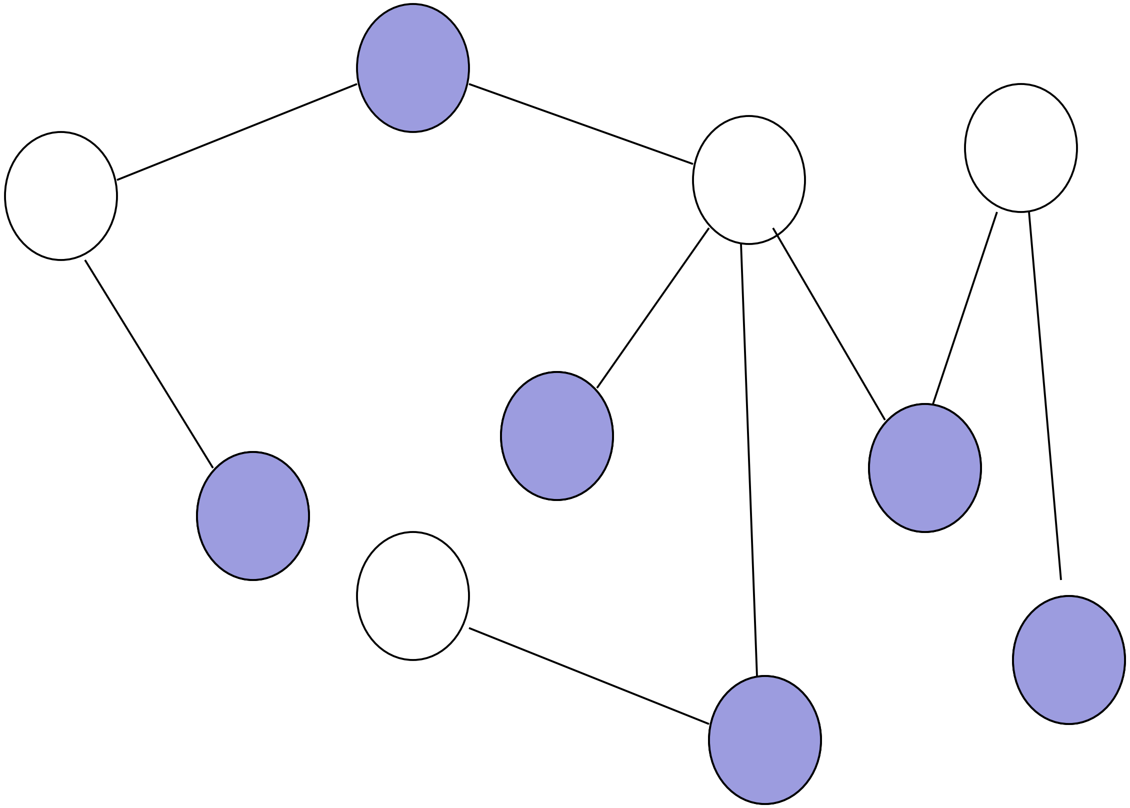


MIS.

Nothing, IS, MIS, MaxIS?



Nothing, IS, MIS, MaxIS?



MaxIS.

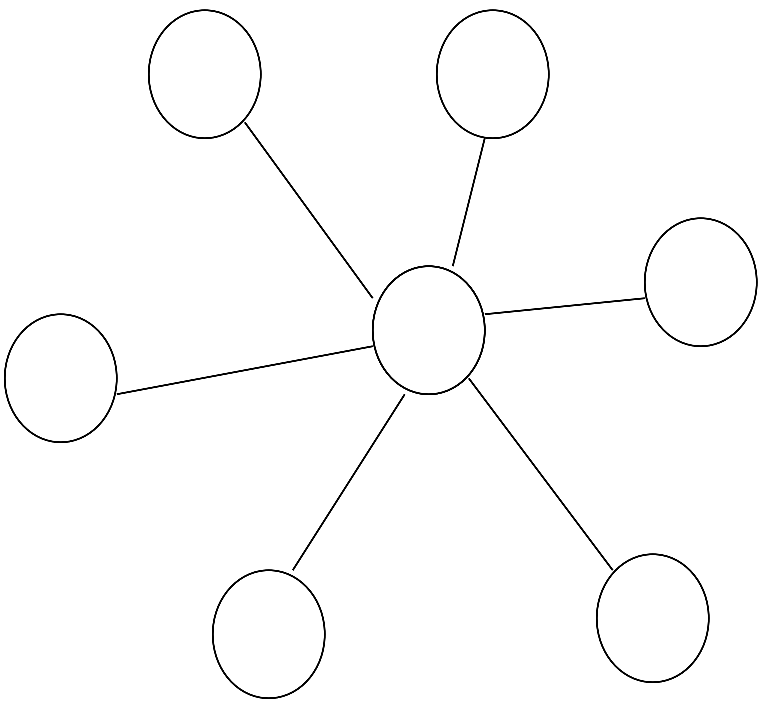
MaxIS is NP-hard!

MaxIS is NP-hard!

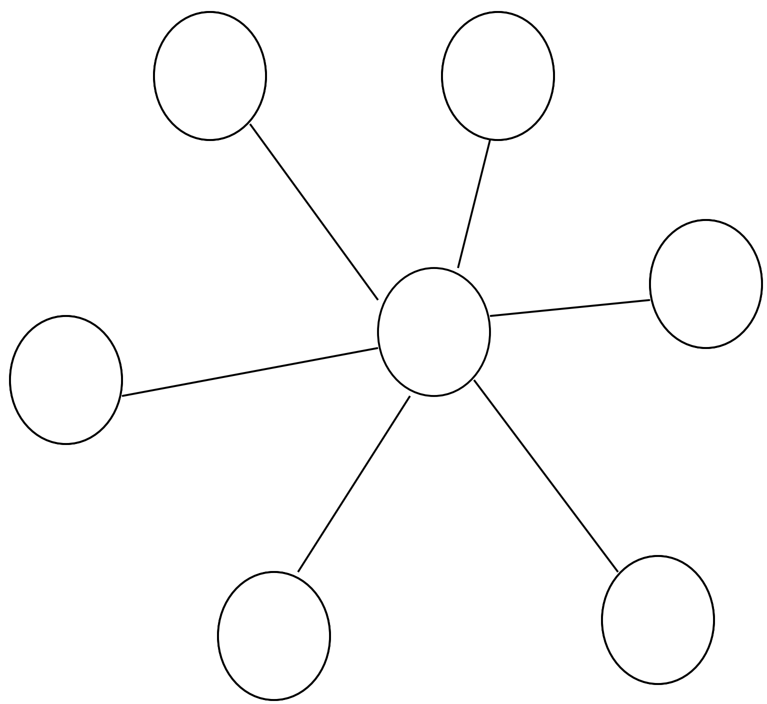
But how much worse can MIS be compared to MaxIS?

MIS vs MaxIS

minimal MIS?



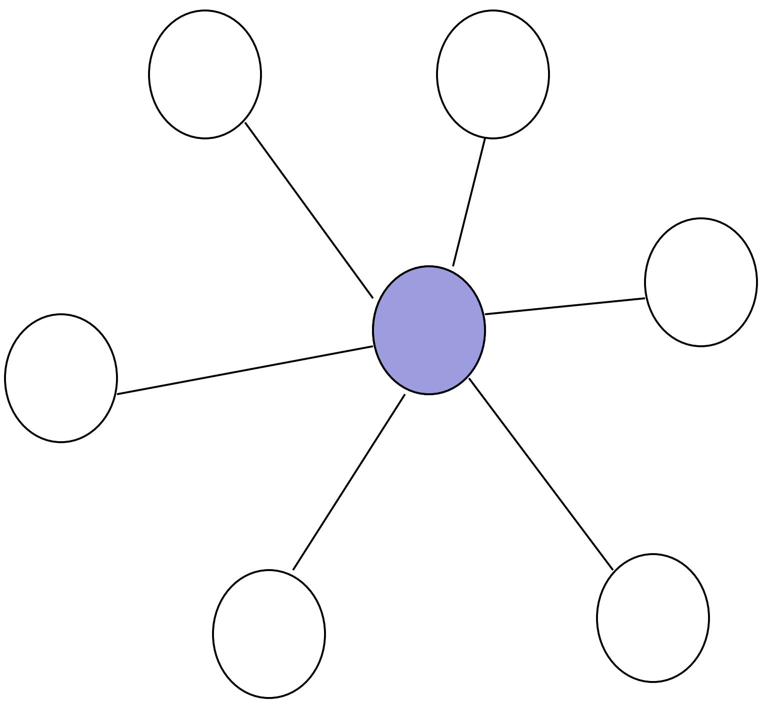
Maximum IS?



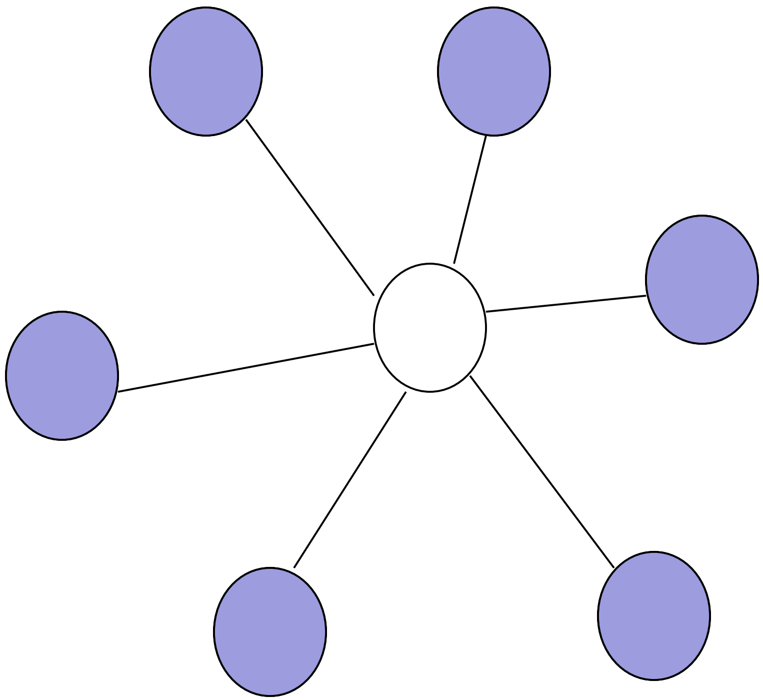
MIS vs MaxIS

MIS is a bad approximation for MaxIS in general!

minimal MIS?



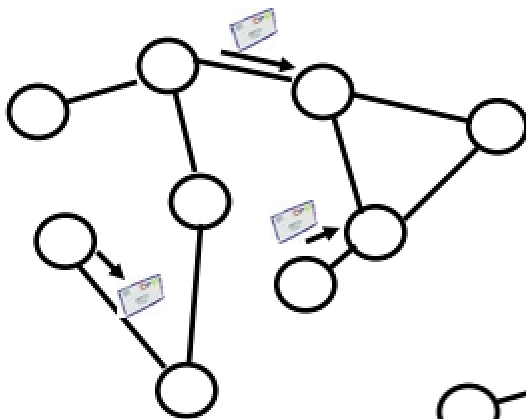
Maximum IS?

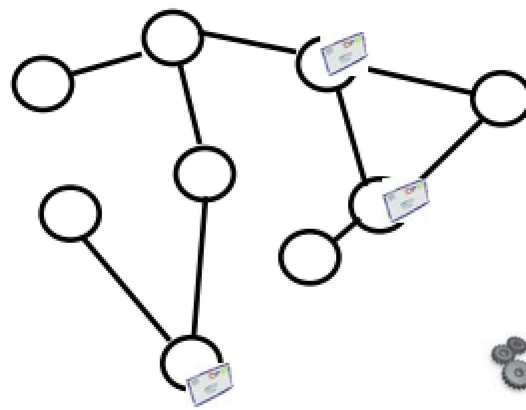


How to compute a MIS in a distributed manner?

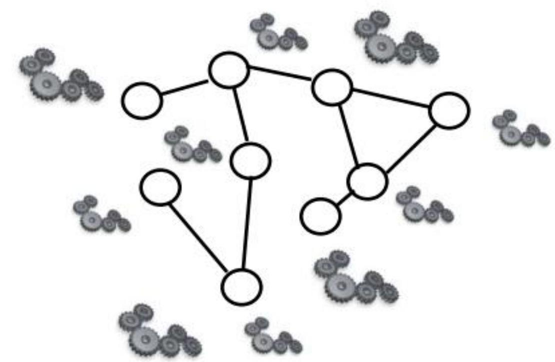
Recall: LOCAL model

Send...



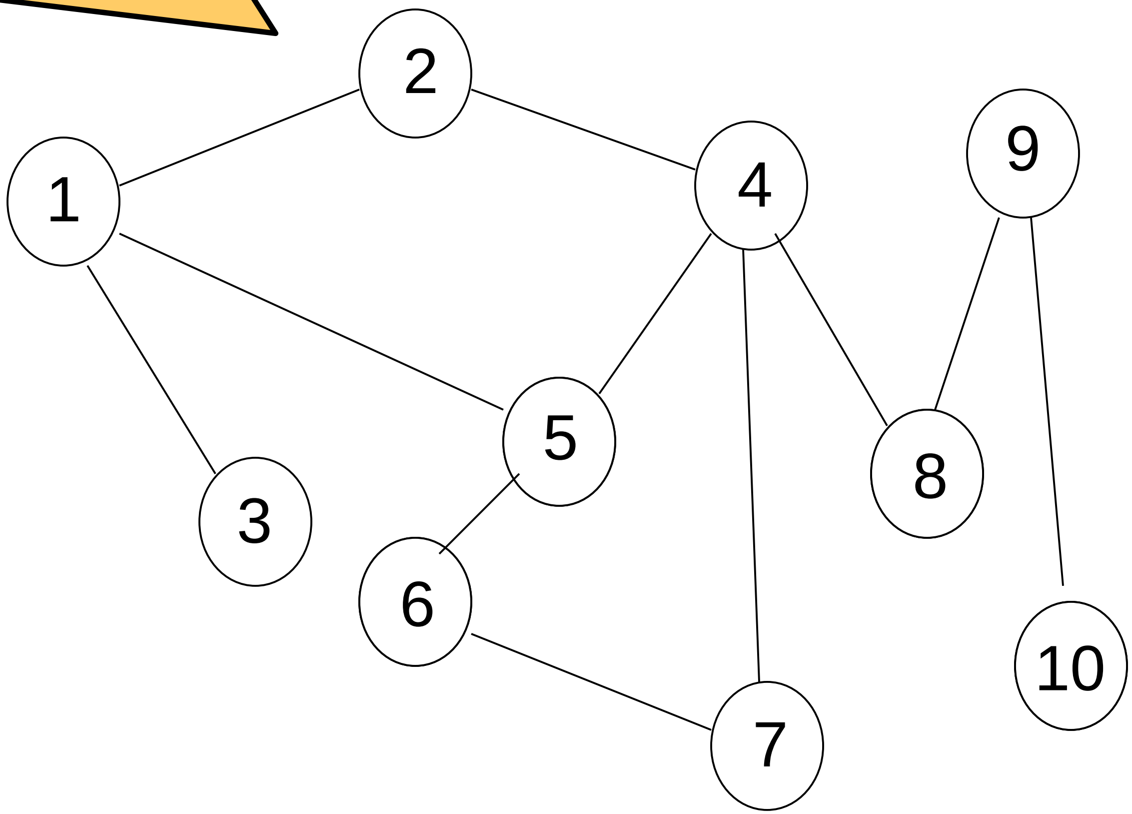


... receive...



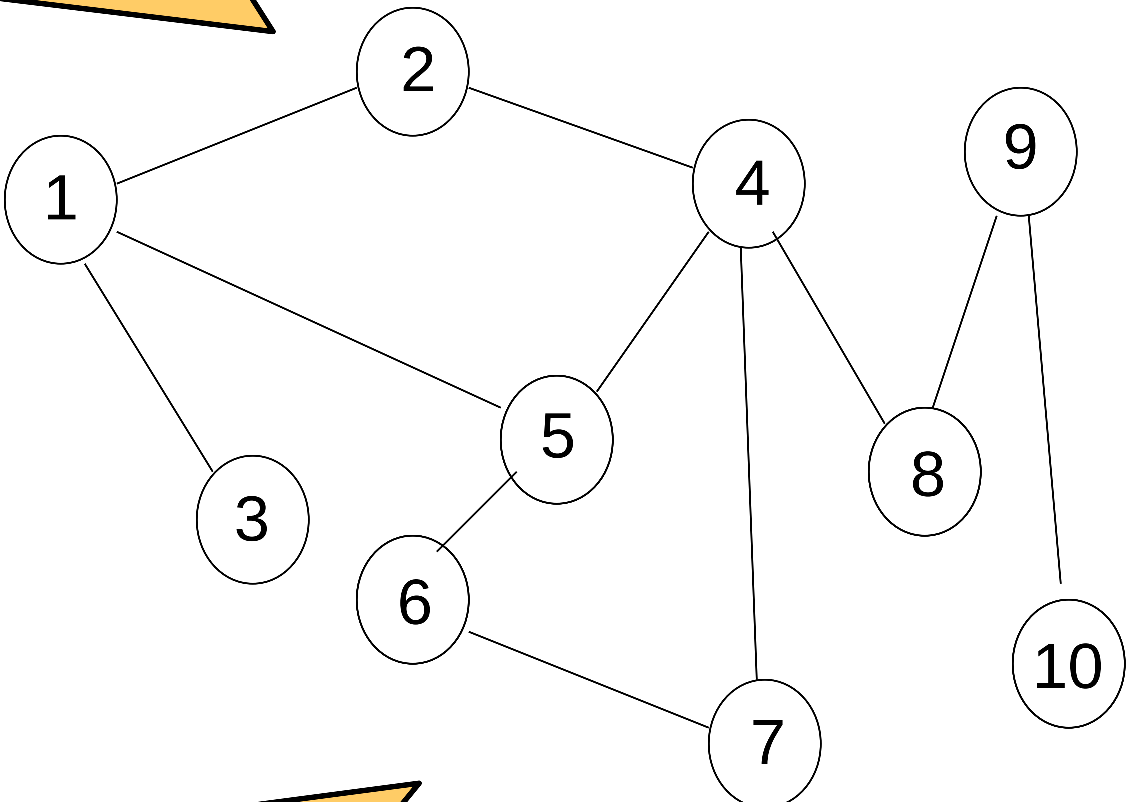
Idea: Distributed Algorithm for MIS?

Challenge: symmetry breaking! Neighboring nodes should not join MIS at the same time!



Idea: Distributed Algorithm for MIS?

Challenge: symmetry breaking! Neighboring nodes should not join MIS at the same time!

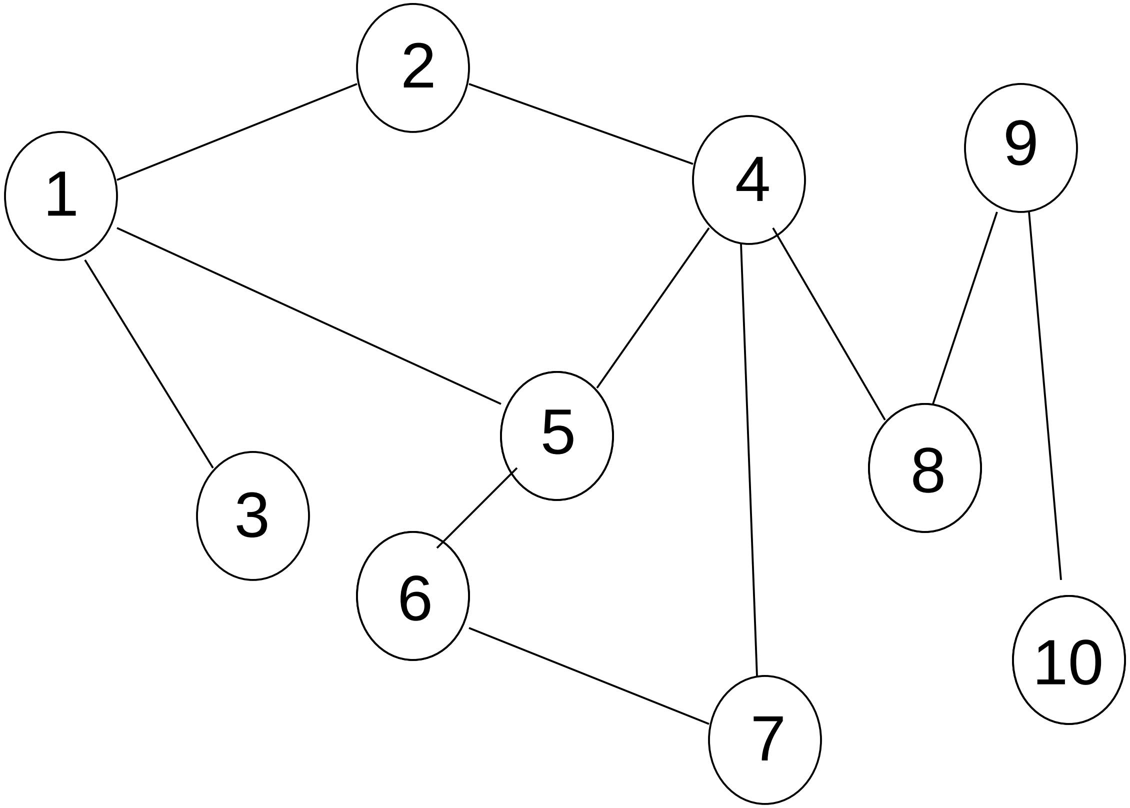


Idea: Make it depend on node IDs (they are unique!):

- Let high ID neighbors decide first!
- Join if no neighbor with larger ID has joined MIS.

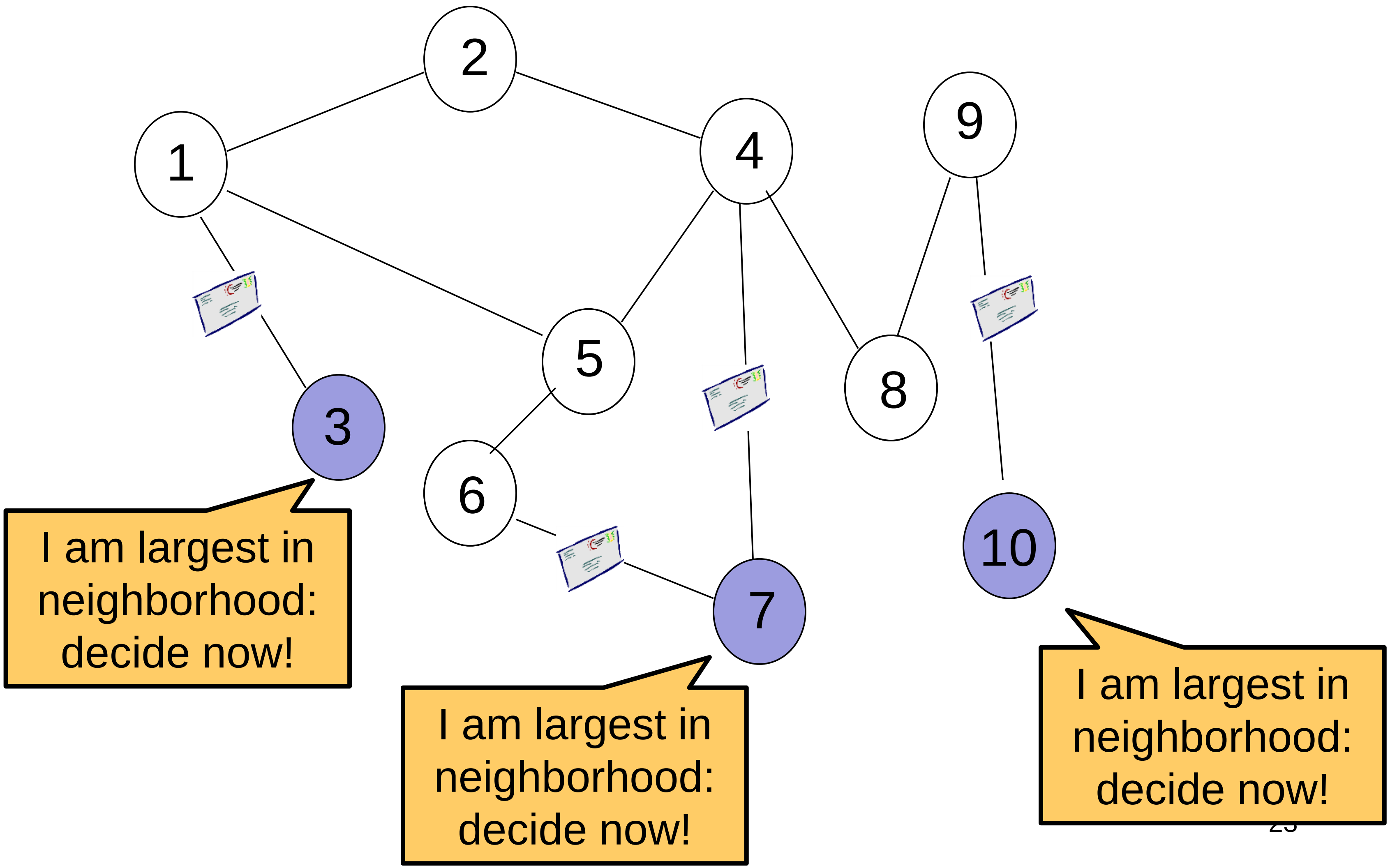
Idea: Distributed Algorithm for MIS?

Round 1:

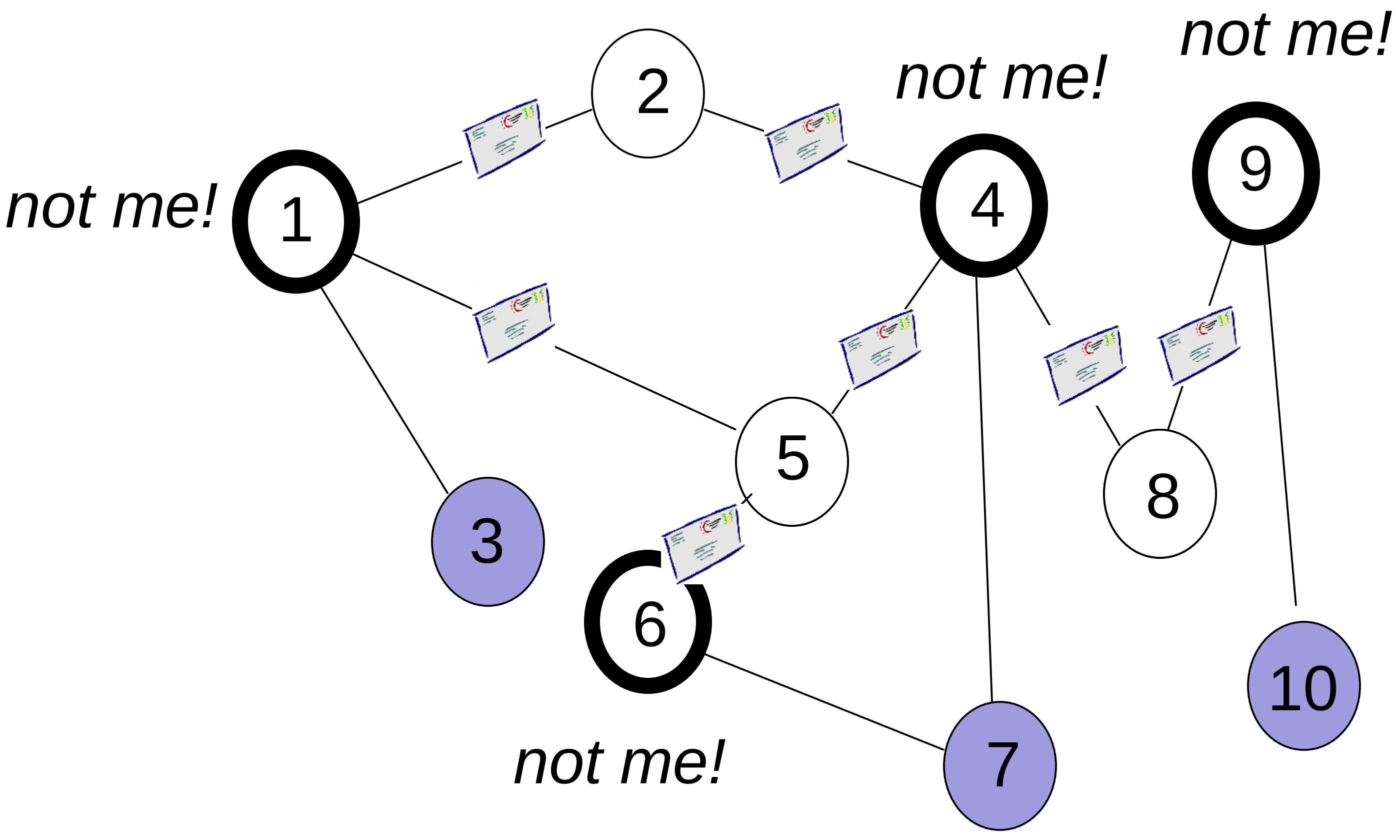


Idea: Distributed Algorithm for MIS?

Round 1:



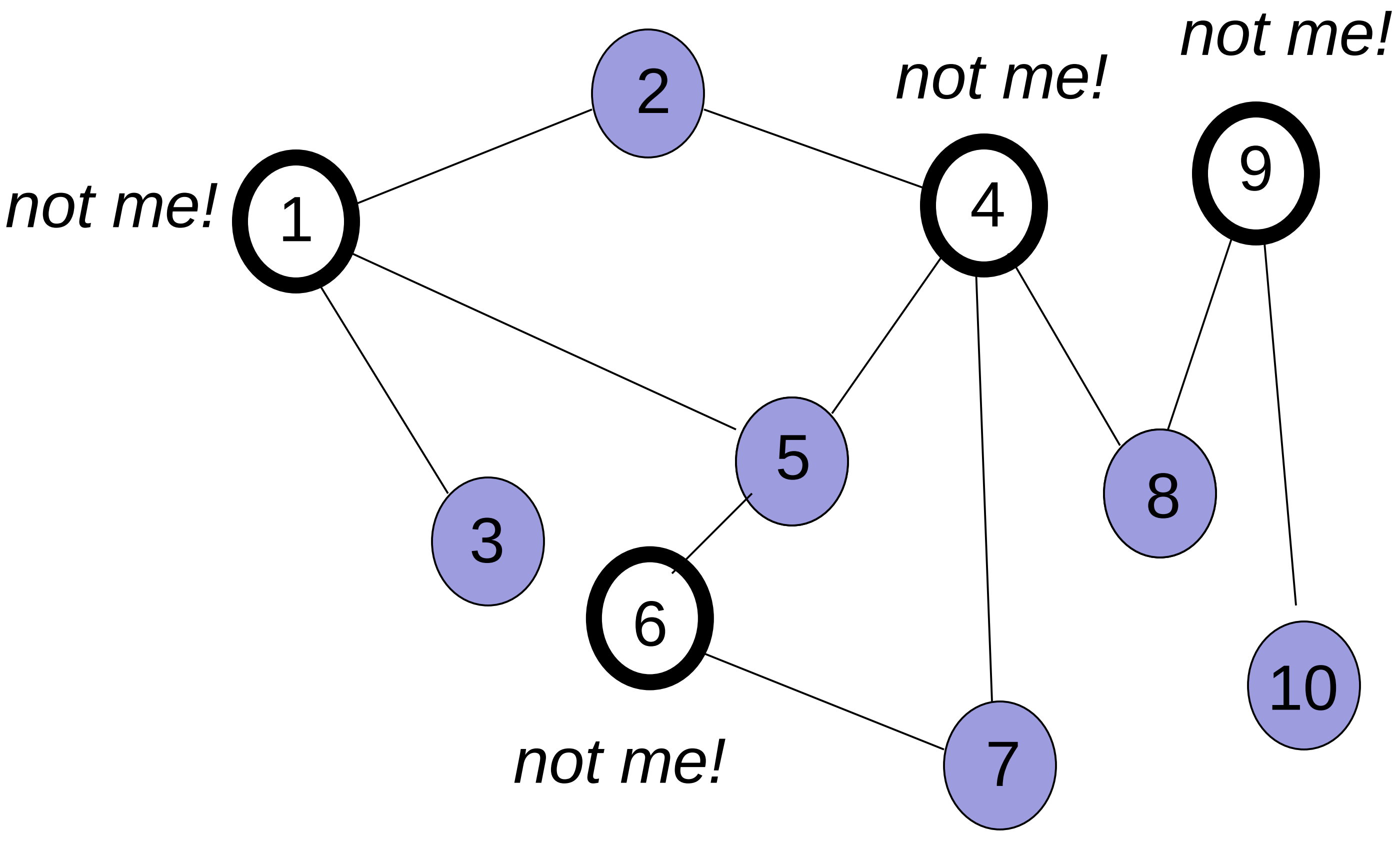
Round 2:



Neighbors of MIS nodes decide not to join!

Idea: Distributed Algorithm for MIS?

Round 3:



Higher-order neighbors have decided: now rest can decide.

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?

Analysis

Time Complexity?

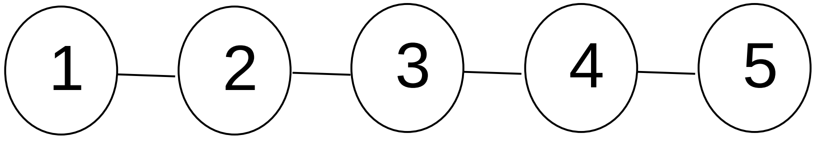
Local Computations?

Message Complexity?

Analysis

Time Complexity?

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



Local Computations?

Fast: join if no higher neighbor has! ☺

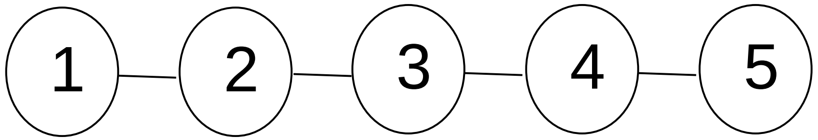
Message Complexity?

O(m) in general: each node needs to inform all neighbors when deciding. (m = number of links)

Analysis

Time Complexity?

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



Can we do it faster?

Local Computations?

Fast: join if no higher neighbor has! ☺

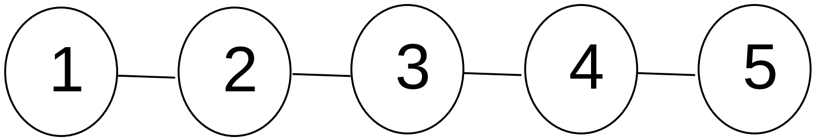
Message Complexity?

O(m) in general: each node needs to inform all neighbors when deciding. (m = number of links)

Analysis

Time Complexity?

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



Can we do it faster?

Local Computations?

Fast: join if no higher neighbo... es! ☺

In general topologies, it is difficult: there are no fast deterministic distributed algorithms. Therefore today: the power of randomization!

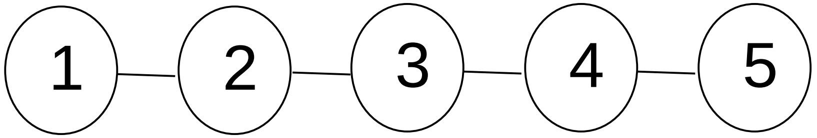
Message Complexity?

O(m) in general: each no...
neighbors when deciding. (m = number of links)

Analysis

Time Complexity?

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



Can we do it faster?

Local Computations?

Fast: join if no higher neighbo es! ☺

In general topologies, it is difficult: there are no fast deterministic distributed algorithms. Therefore today: the power of randomization!

Message Complexity?

O(m) in general: each no
neighbors when deciding. (m = number of links)

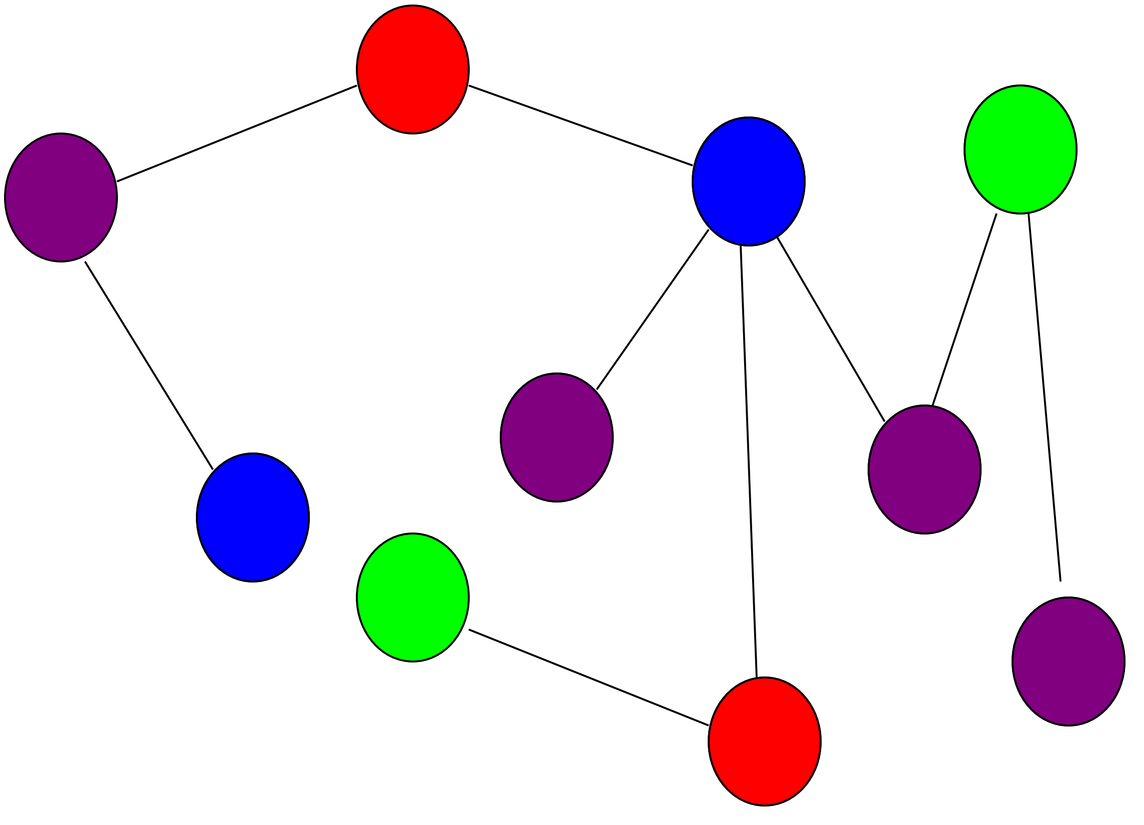
But before that: any ideas how to compute MIS on rooted trees in log* time?

Independent sets on trees in log* time:
Using our coloring algorithm!

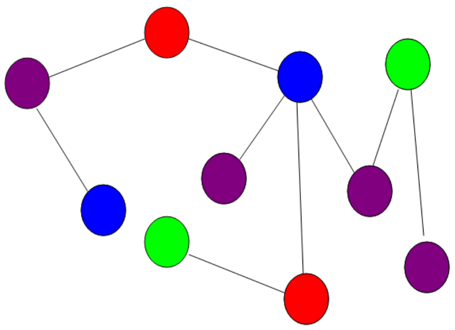
How?

Coloring vs MIS

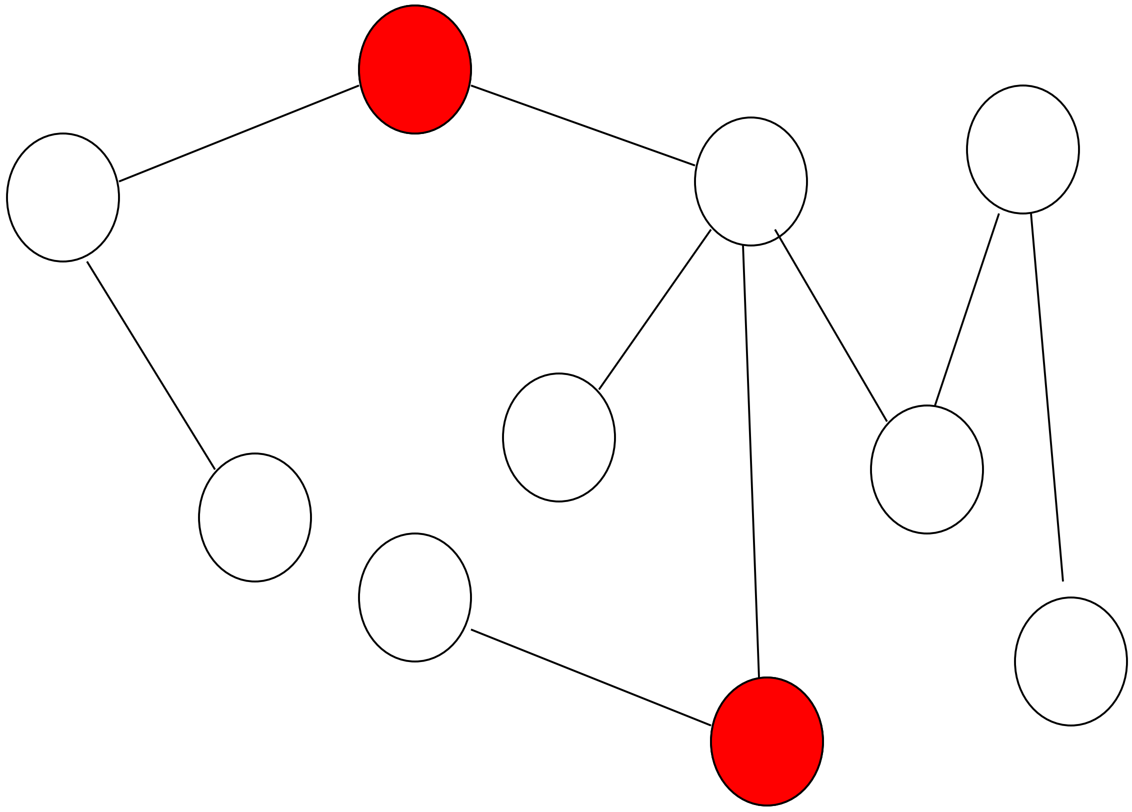
Each color defines an independent set…



Coloring vs MIS



… but not necessarily a maximal one!



Can we compute a MIS given
a distributed coloring algorithm?

Can we compute a MIS given
a distributed coloring algorithm?

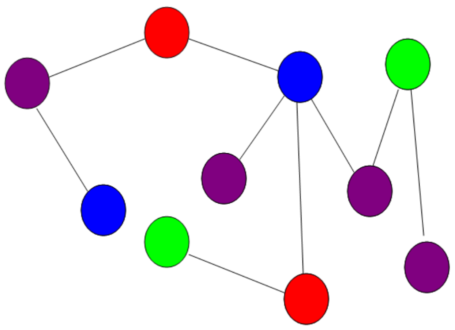
Idea: Choose all nodes of first color (define beforehand: color 0).
Then for any additional color (*one-by-one*), add in parallel as many nodes as possible!

Can we compute a MIS given
a distributed coloring algorithm?

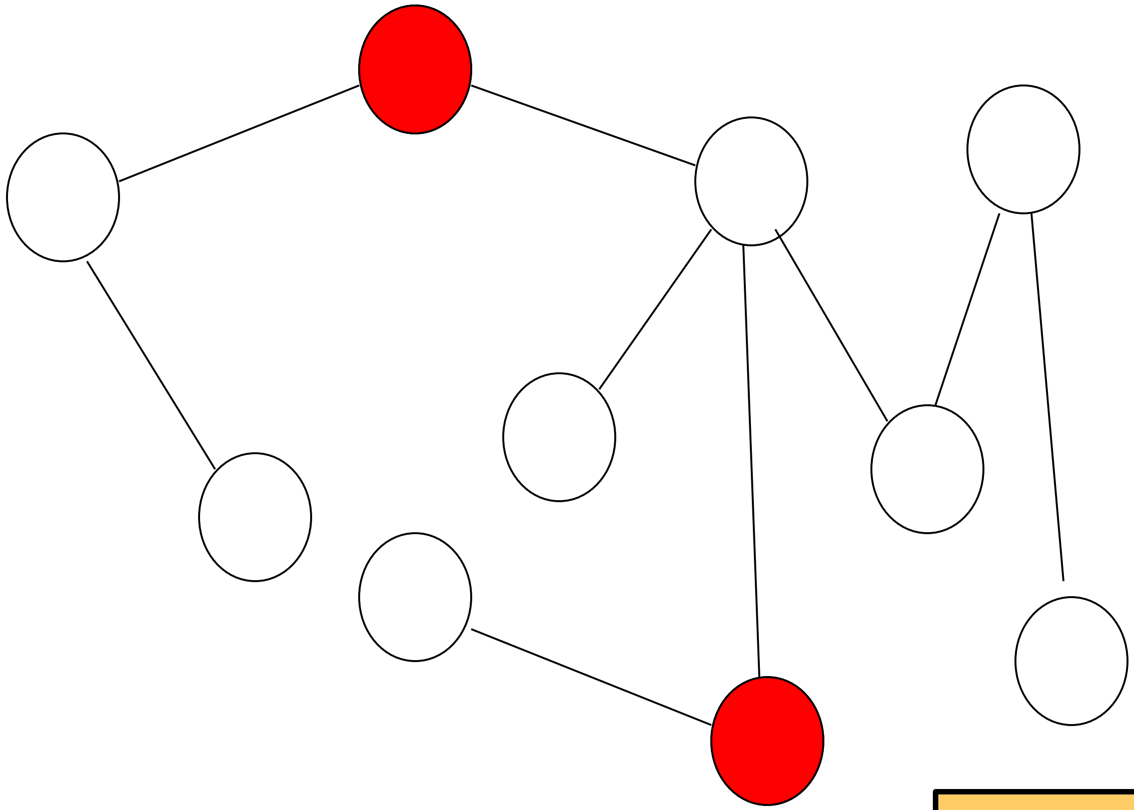
Idea: Choose all nodes of first color (define beforehand: color 0).
Then for any additional color (*one-by-one*), add in parallel as many nodes as possible!

Example where independent sets are useful: can do them in parallel ☺.

From Distributed Coloring to Distributed MIS



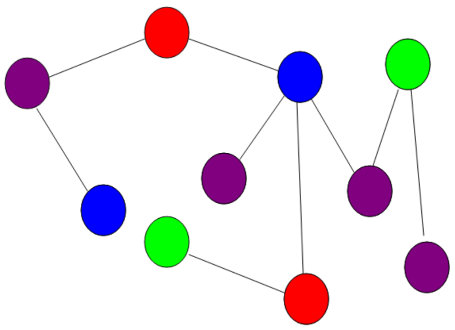
Round 1



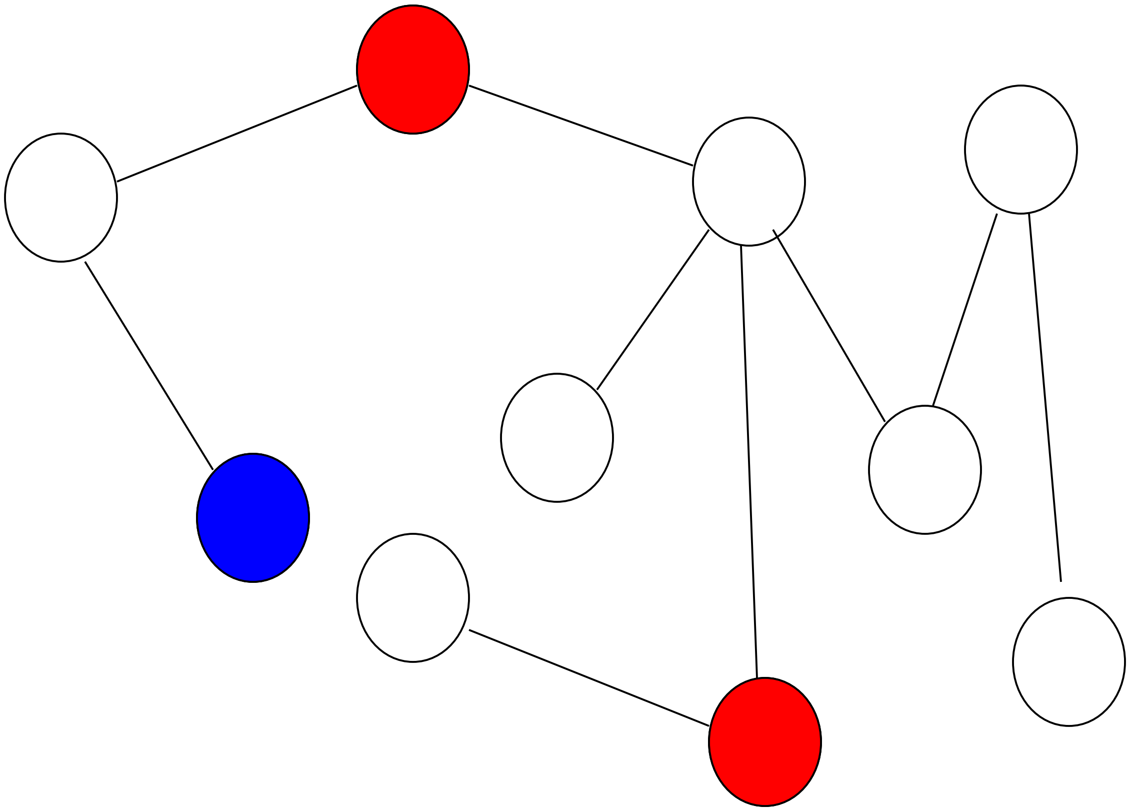
Add Color 0!

Assume: nodes pre-agree on order in which colors are added!

From Distributed Coloring to Distributed MIS

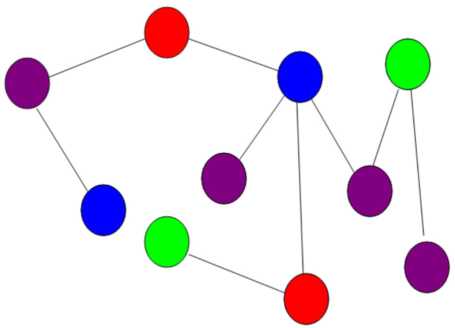


Round 2

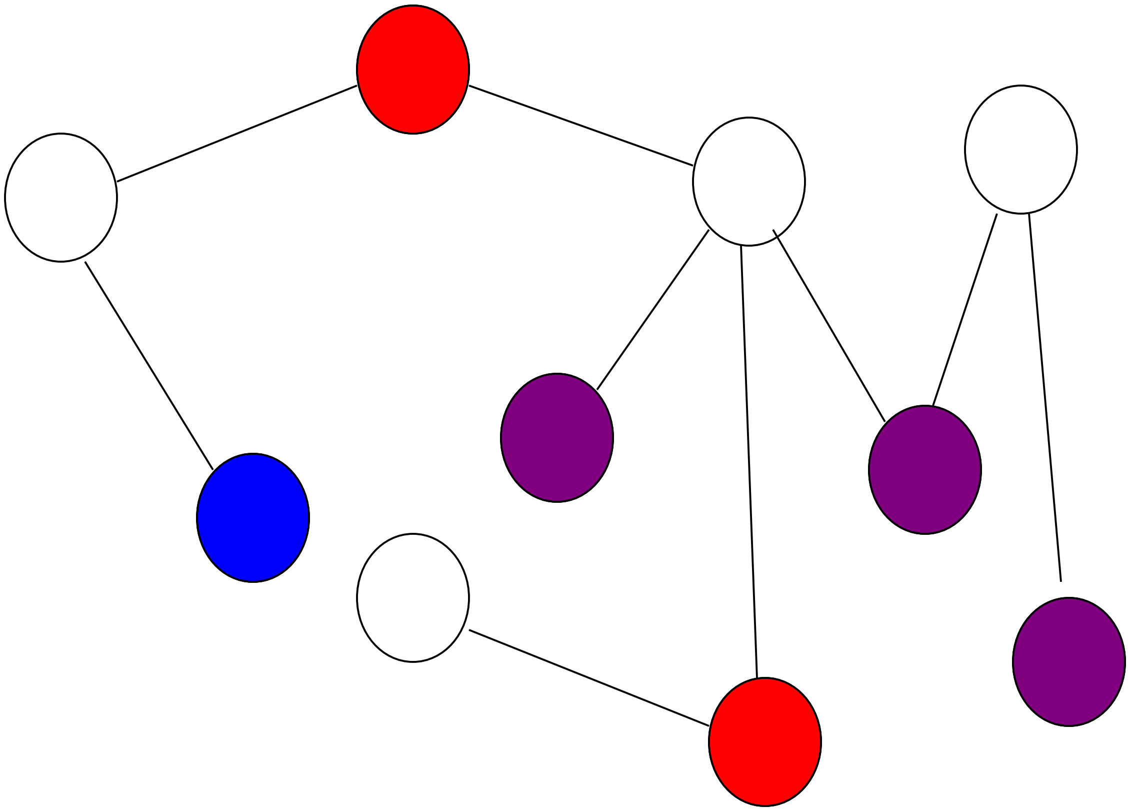


Add Color 1 nodes unless node has MIS neighbor!

From Distributed Coloring to Distributed MIS

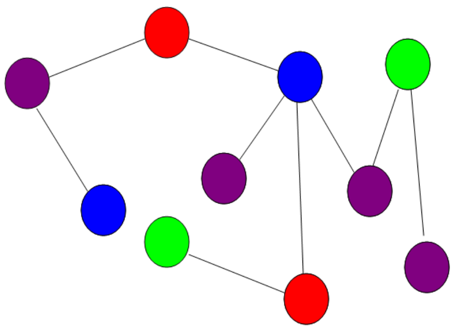


Round 3

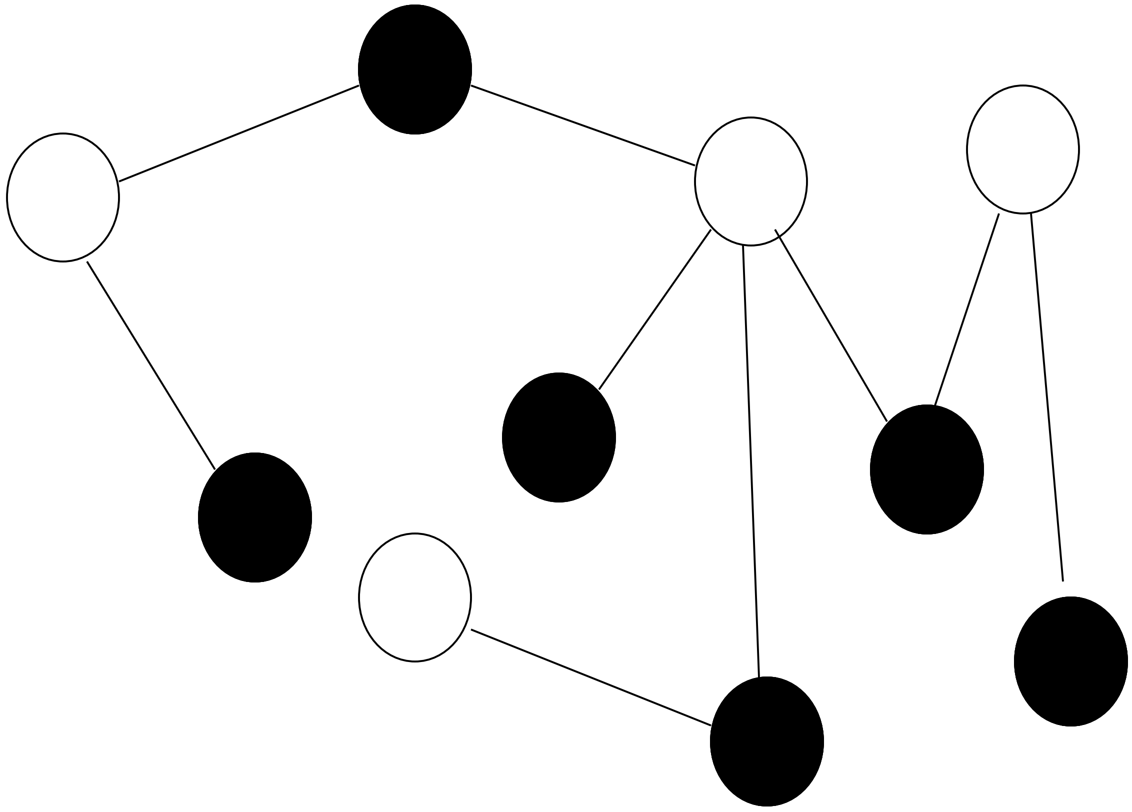


Add Color 2 nodes unless node has MIS neighbor!

From Distributed Coloring to Distributed MIS



Final MIS:



Analysis of algorithm?

Analysis COLOR-To-MIS Algorithm

Why does algorithm work?

Runtime?

Why does algorithm work?

Unicolor nodes are independent, can be added 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.

Implication for MIS on trees

Implication 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 on General Topologies

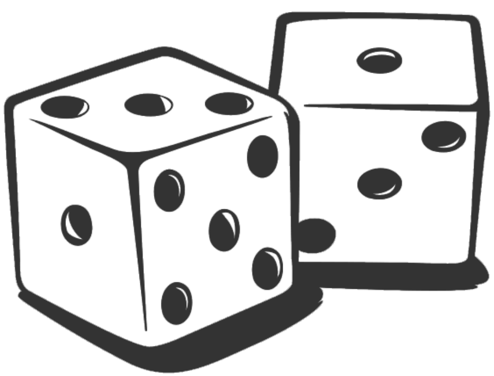
Any ideas?

Better MIS Algorithms on General Topologies

Any ideas?

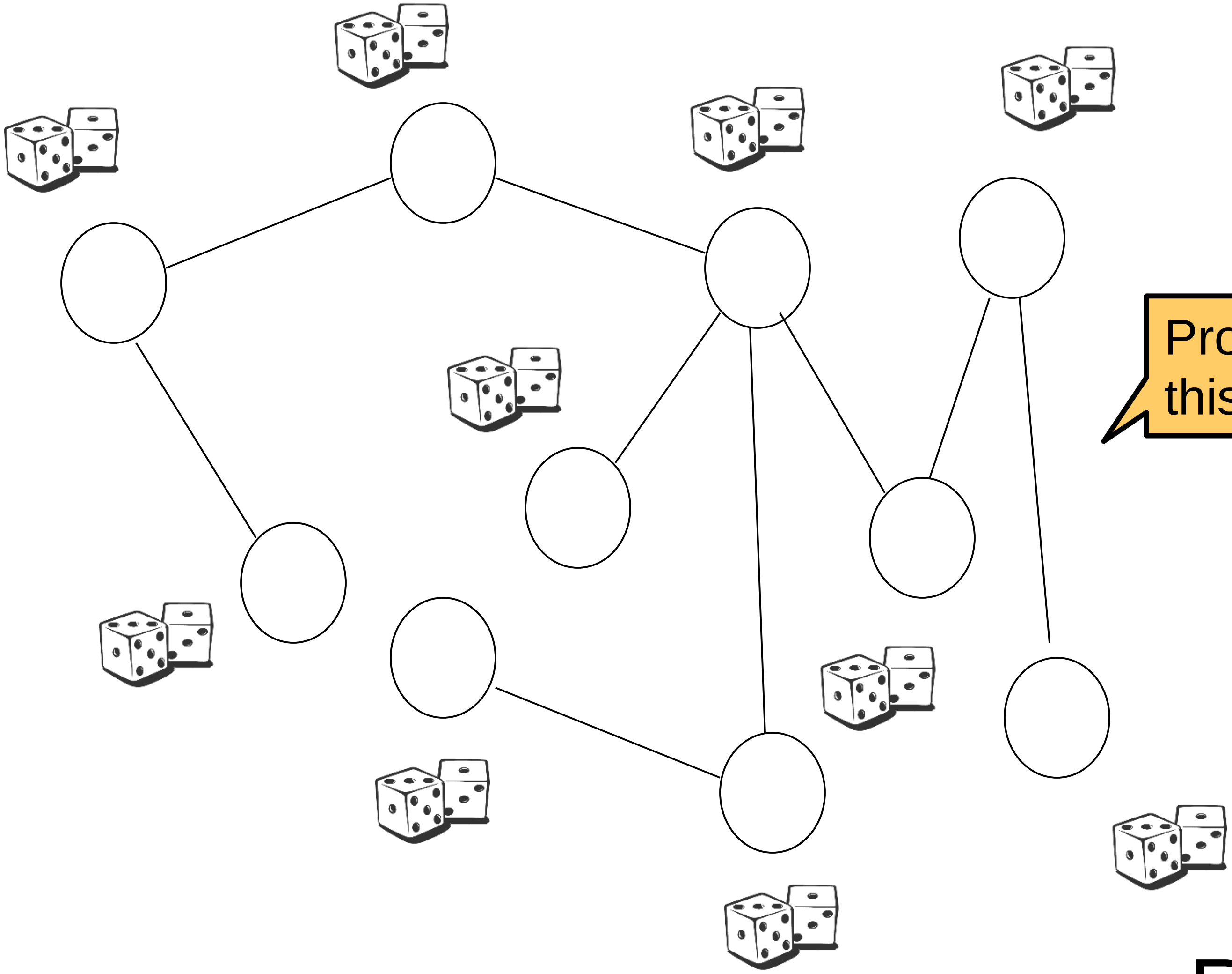
It is difficult on general graphs…

Tipp: If you can‘t find fast deterministic algorithms, try randomization!



Randomized MIS Algorithm

Randomly decide to become MIS node or not with probability 1/(2d(v))!

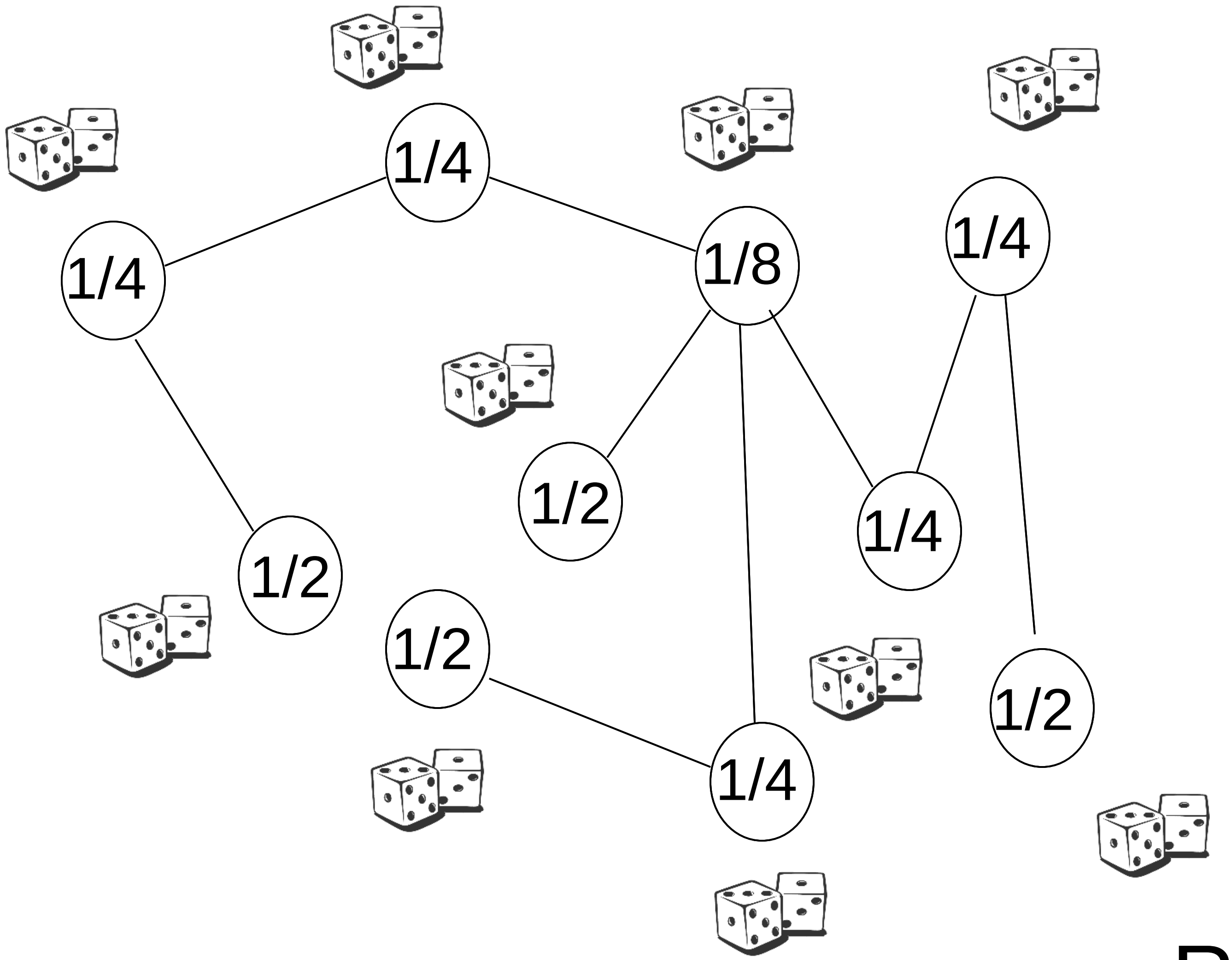


Probabilities in this example?

Round 1

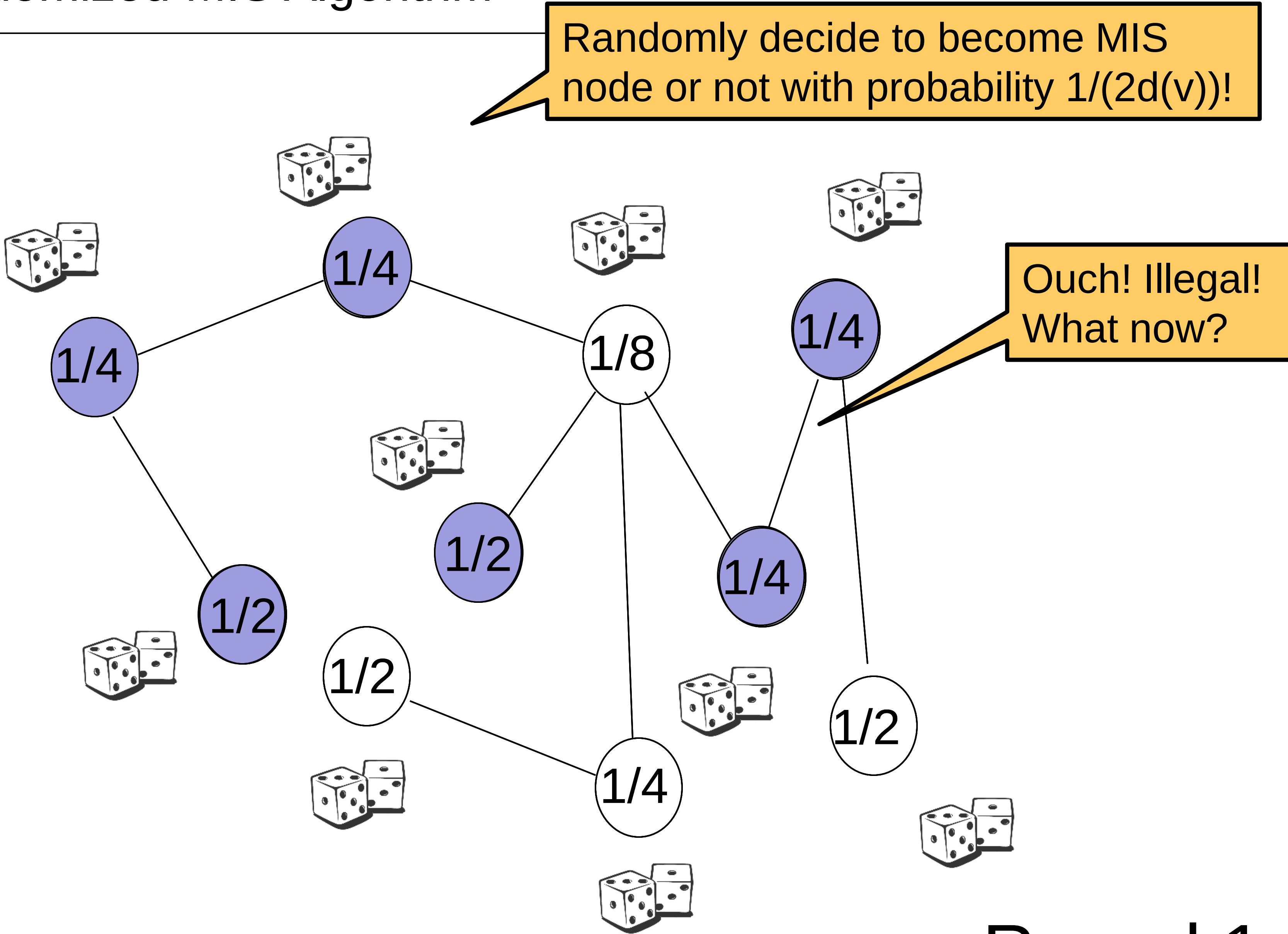
Randomized MIS Algorithm

Randomly decide to become MIS node or not with probability 1/(2d(v))!



Round 1

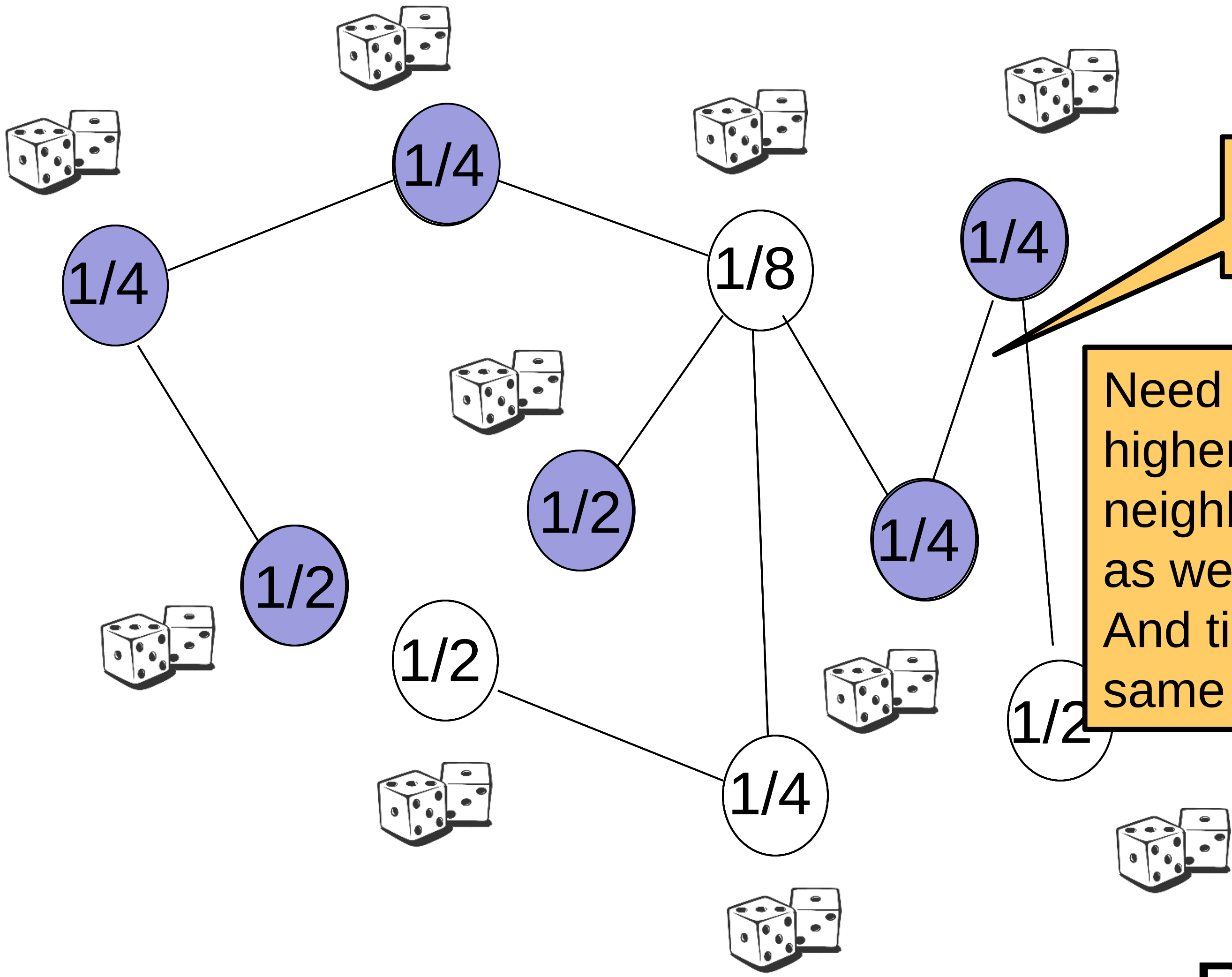
Randomized MIS Algorithm



Round 1

Randomized MIS Algorithm

Randomly decide to become MIS node or not with probability 1/(2d(v))!

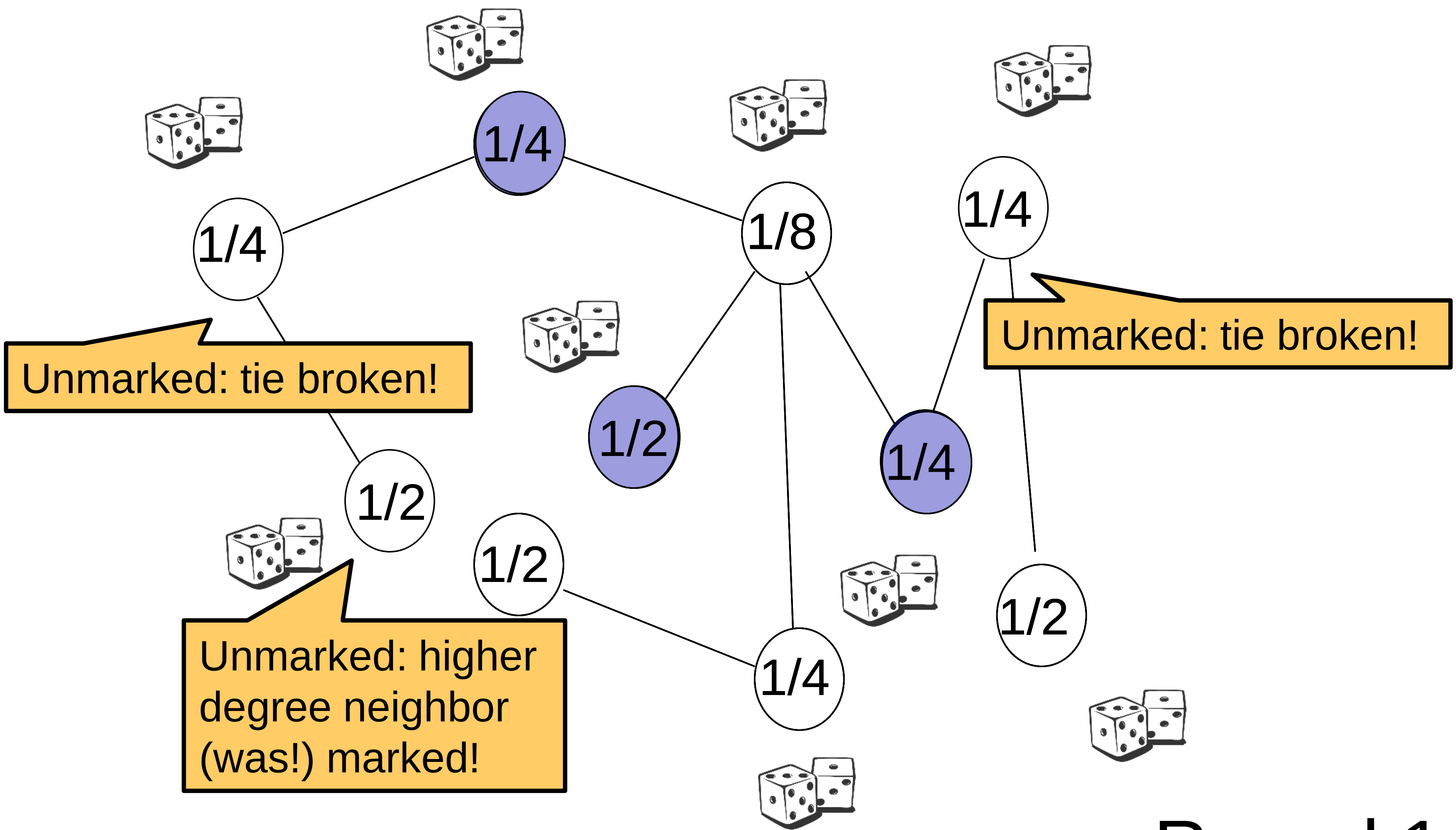


Ouch! Illegal! What now?

Need tie breaker: if higher degree neighbor marked as well, leave MIS. And tie break if same degree.

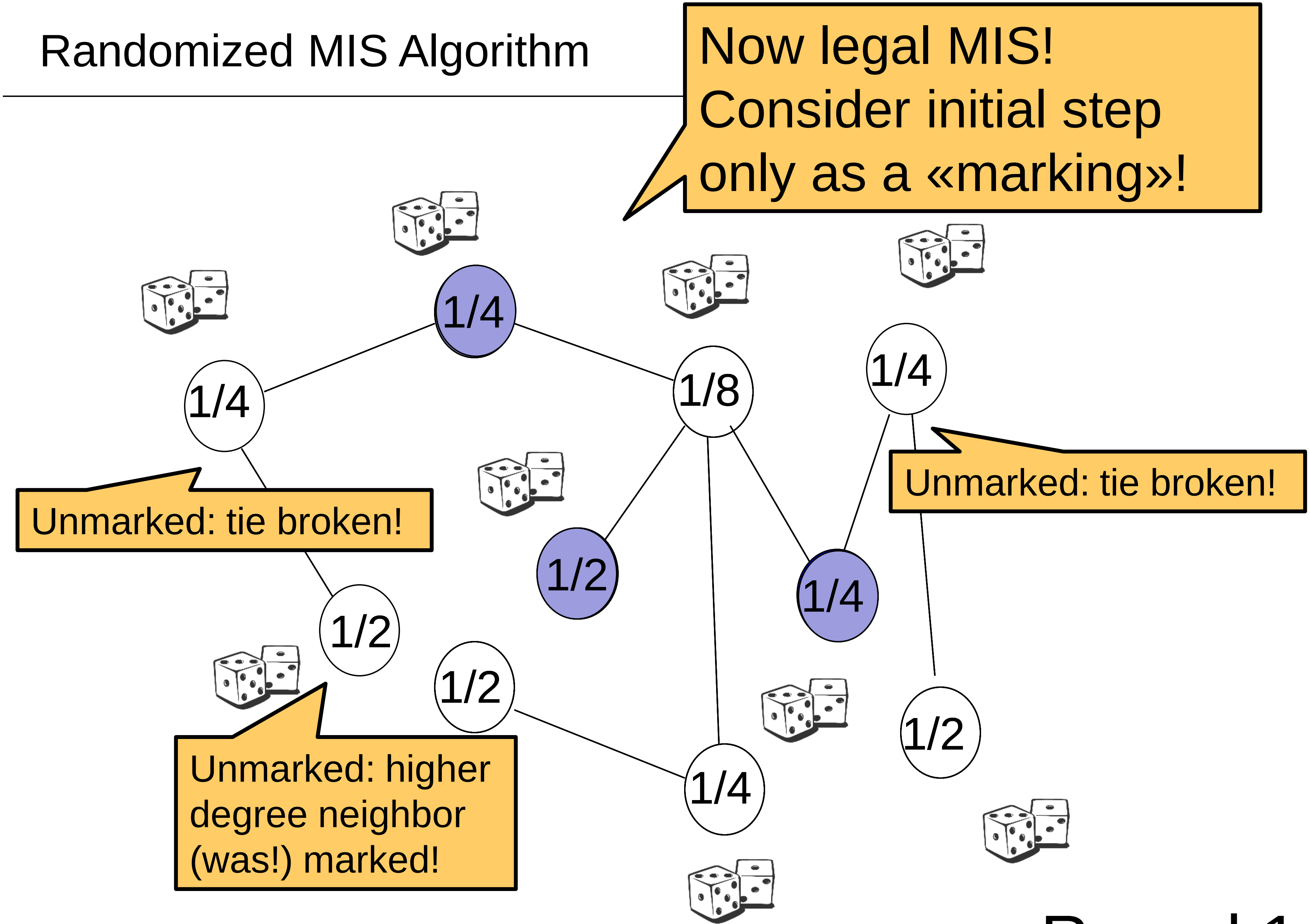
Round 1

Randomized MIS Algorithm



Round 1

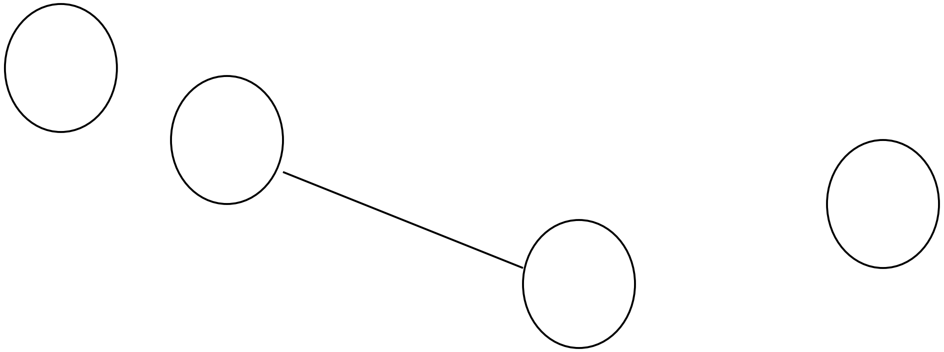
Randomized MIS Algorithm



Round 1

Randomized MIS Algorithm

Now: delete MIS nodes and neighbors: *they are done!*
And proceed with Round 2 like Round 1: toss coin, mark, etc.



Round 2

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

Summary of the Algorithm

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

Summary of the Algorithm

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 legal IS? Because at least one neighbor leaves conflict!

Summary of the Algorithm

Why MIS? Each node without a IS neighbor will eventually mark itself with probability 1.

Fast M

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 legal IS? Because at least one neighbor leaves conflict!

Why MIS? Each node without a IS neighbor will eventually mark itself with probability 1.

Fast M

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 legal IS? Because at least one neighbor leaves conflict!

What about the distributed runtime?

Fast M

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 j ned the MIS plus their hbors, as they canno the MIS anymore

Why MIS? Each node without a IS neighbor will eventually mark itself with probability 1.

Why legal IS? Because at least one neighbor leaves conflict!

Logarithmic: how to prove?

What about the distributed runtime?

The Many Ways to Prove a Logarithmic Runtime

The Many Ways to Prove a Logarithmic Runtime

In general: show that the remaining problem size is divided in two in each round.

log n: How many times do I have to **:2** until <2?

$$n, n/2, n/4, n/8, \ldots, 8, 4, 2, 1$$

$$\underbrace{\hspace{6cm}}_{\log n}$$

The Many Ways to Prove a Logarithmic Runtime

In general: show that the remaining problem size is divided in two in each round.

- Idea 1: Each node is removed with constant probability (e.g., ½) in each round. So half of the nodes vanish in each round.

- Idea 2: Each edge is removed with constant probability in each round! Also works: $O(\log m) = O(\log n^2) = O(\log n)$

log n: How many times do I have to **:2** until <2?

$$n, n/2, n/4, n/8, \ldots, 8, 4, 2, 1$$

$$\xleftrightarrow{\hspace{6em}}$$

log n

The Many Ways to Prove a Logarithmic Runtime

In general: show that the remaining problem size is divided in two in each round.

- Idea 1: Each node is removed with constant probability (e.g., ½) in each round. So half of the nodes vanish in each round.

- Idea 2: E tant probability in each round! Also wor n)

Neither is true! ☹

log n: How many times do I have to **:2** until <2?

n, n/2, n/4, n/8, …, 8, 4, 2, 1

$\longleftrightarrow$

log n

The Many Ways to Prove a Logarithmic Runtime

In general: show that the remaining problem size is divided in two in each round.

- Idea 1: Each node is removed with constant probability (e.g., ½) in each round. So half of the nodes vanish in each round.

- Idea 2: E tant probability in each round! Also wor n)

Neither is true! ☹

log n:

However: A constant fraction of edges are removed with constant probability!

log n

Joining MIS

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

A node, once marked, joins with probabiliy ½.
And marking happens with probability $1/(2d(v))$.

Joining MIS

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

A node, once marked, joins with probabiliy ½.
And marking happens with probability 1/(2d(v)).

Problem: Still a too small probability still (e.g., 1/n).
We need some more definitions!

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

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

Otherwise bad.

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

Joining MIS

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

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

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

Good: With high constant probability! Remains to show that there are many good nodes.

Good Nodes

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

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

Otherwise bad.

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

Good: With high constant probability! Remains to show that there are many good nodes.

Good Nodes

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

Case 1: If v has a neighbor w with $d(w) \leq 2$ easy an due to Joining MIS Lemma: neighbor w joins with probability 1/8.

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

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

Otherwise bad.

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

Good: With high constant probability! Remains to show that there are many good nodes.

Good Nodes

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

Case 1: If v has a neighbor w with $d(w) \leq 2$ easy an due to Joining MIS Lemma: neighbor w joins with probability 1/8.

Case 2: Also easy (s. lecture notes).

We just proved:

Good Nodes

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

$$p \geq 1/36.$$

Good Nodes

We just proved:

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

$$p \geq 1/36.$$

But how many good nodes are there?

We just proved:

Good Nodes

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

$p \geq 1/36.$

But how many good nodes are there?

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

A good node has many neighbors of low degree.

We just proved:

Good Nodes

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

$$p \geq 1/36.$$

But how many good nodes are there?

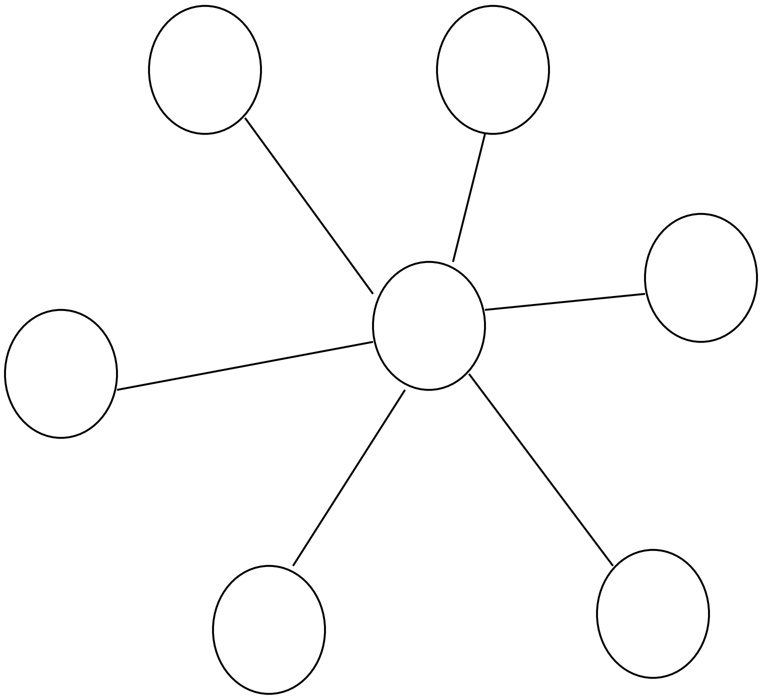
Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

A good node has many neighbors of low degree.



We just proved:

Good Nodes

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

$$p \geq 1/36.$$

But how many good nodes are there?

Good&Bad Nodes

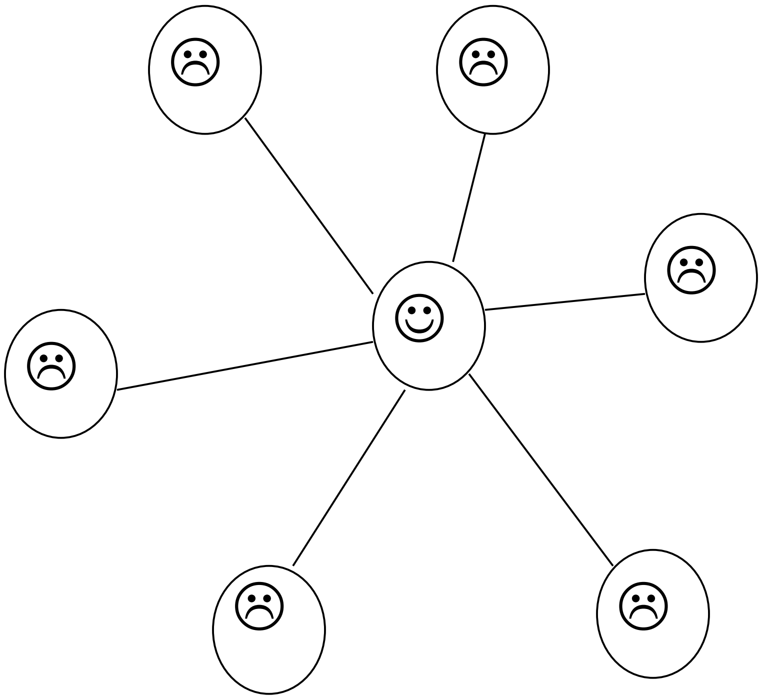
A node v is called *good* if

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

Otherwise bad.

A good node has many neighbors of low degree.

Only one good node!



We just proved:

Good Nodes

A good node v

But many edges have one good node as endpoint! Many «good edges»!

But how many good nodes are there?

Only one good node!

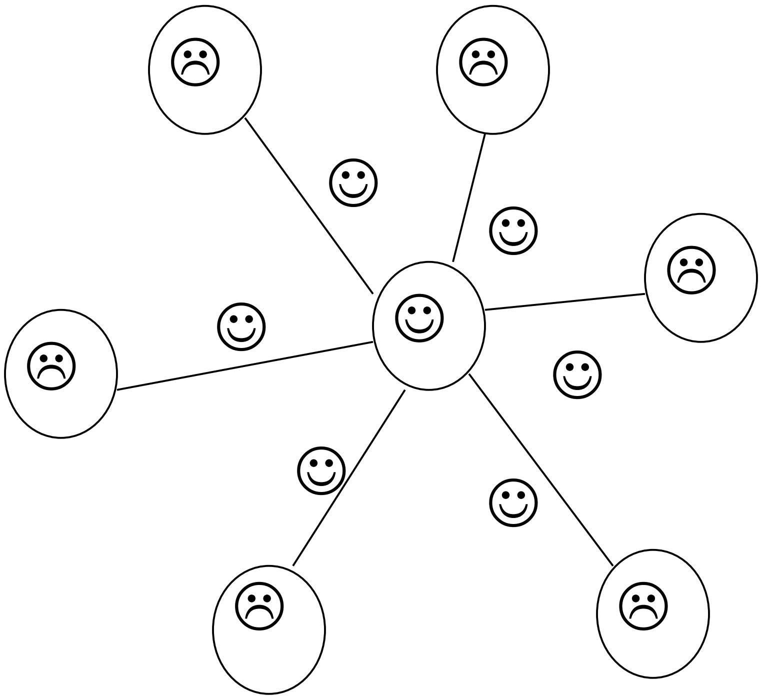
Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

A good node has many neighbors of low degree.

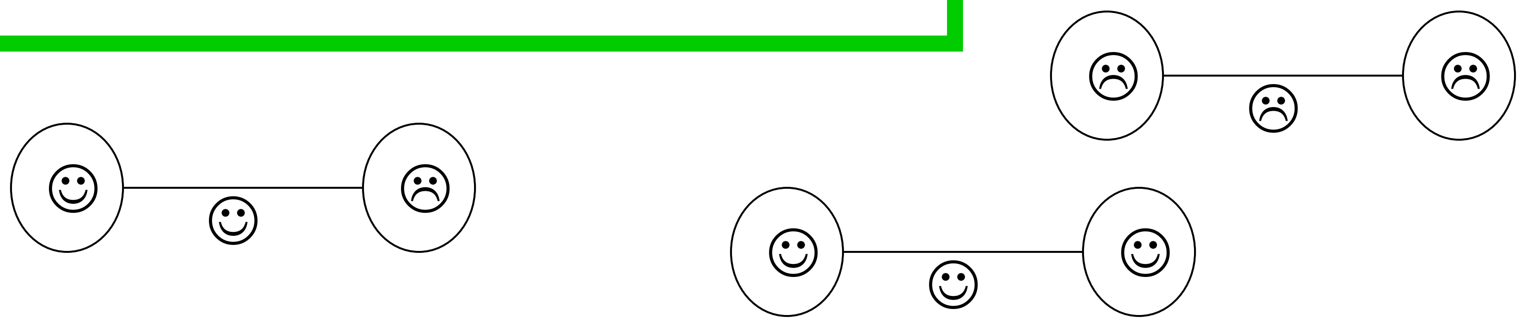


Analysis

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.

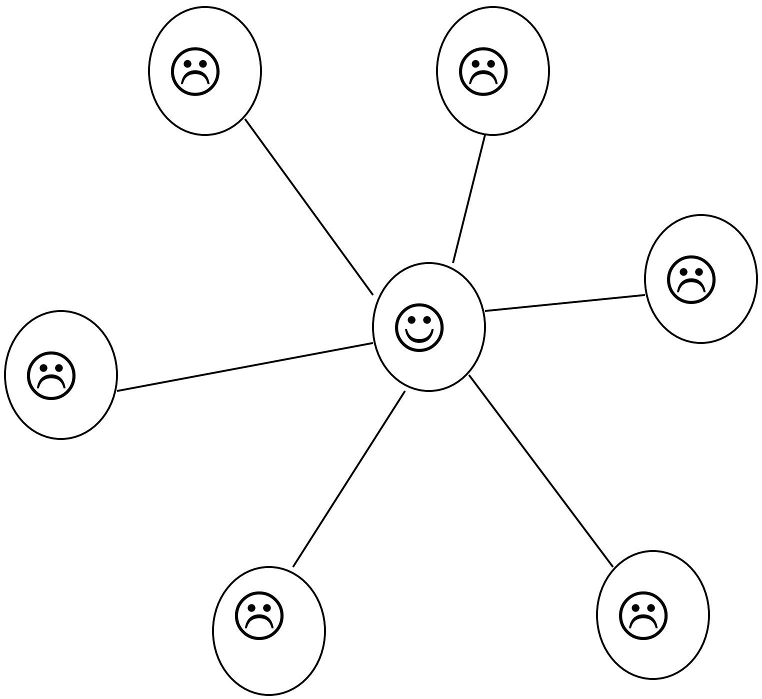
A bad edge is incident to two nodes with neighbors of high degrees.



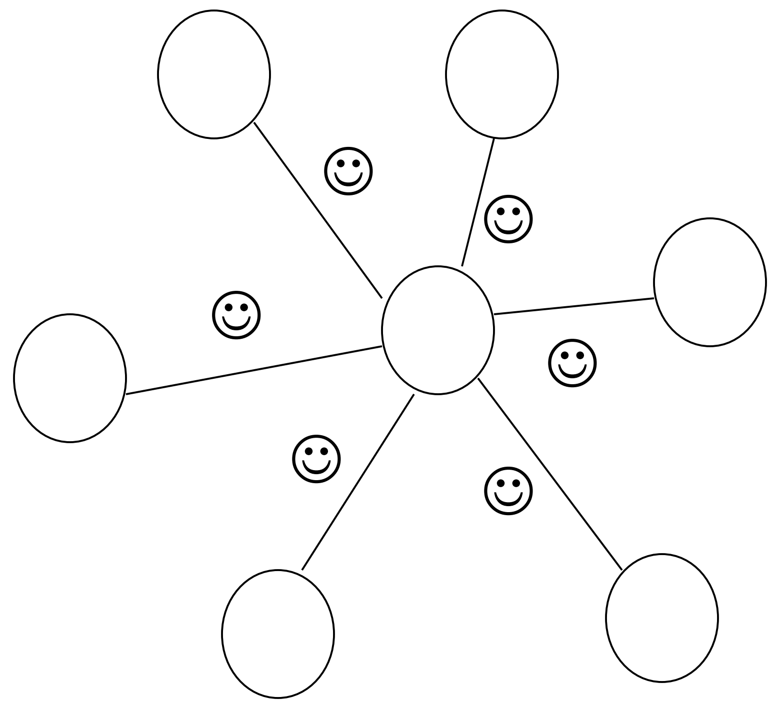
Good Edges

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

Analysis



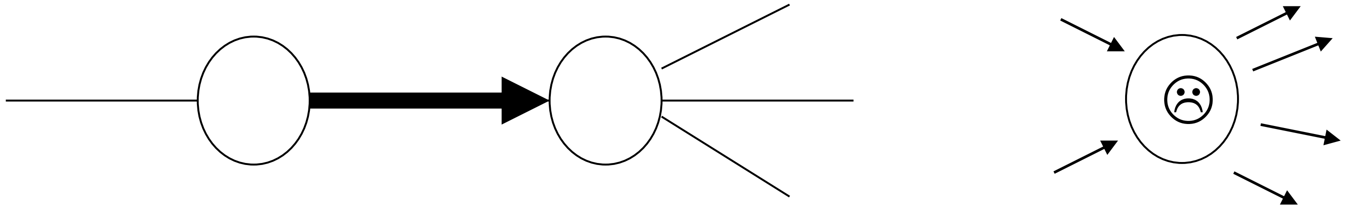
Not many good nodes...



... but many good edges!

Analysis

Idea: Artificially direct each edge towards higher degree node
(if both nodes have same degree, point it to one with higher ID).



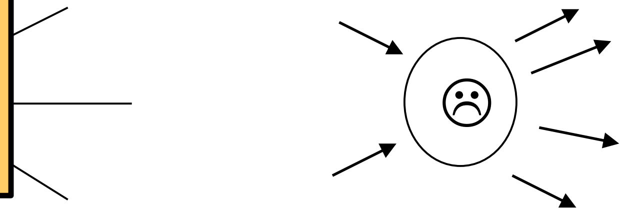
Helper Lemma

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

Analysis

Idea: Artificially direct each edge towards higher degree node
(if both nodes have same degree, point it to one with higher ID).

Easy to see: Otherwise it must have many low-degree neighbors and be good! Contradiction.



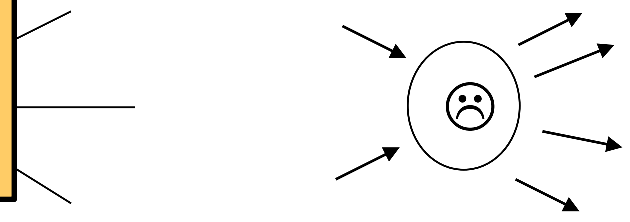
Helper Lemma

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

Analysis

Idea: Artificially direct each edge towards higher degree node
(if both nodes have same degree, point it to one with higher ID).

Easy to see: Otherwise it must have many low-degree neighbors and be good! Contradiction.



Helper Lemma

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

That is great: Since sum of incoming edges = sum of outgoing edges, if the number of edges into bad nodes can be at most half the number of all edges, at least half of all edges are directed into good nodes! And they are good! ☺
So at least half of all edges are good.

Analysis

Fast MIS (1986)

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

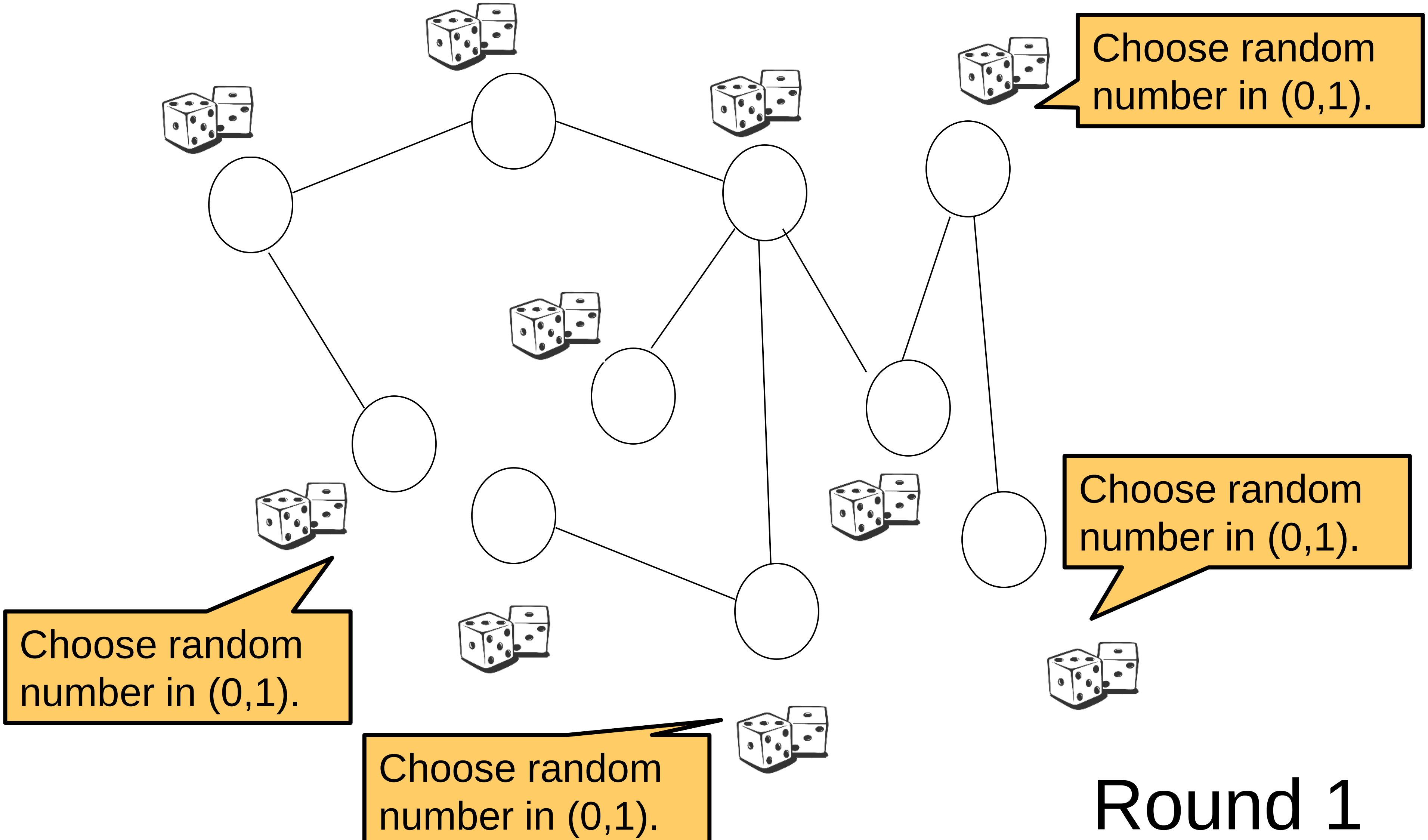
Fast MIS (1986)

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

At least half of all the edges are good, and thus have at least one good incident node which will be deleted with constant probability and so will the edge! A constant fraction of edges will be deleted in each phase. (Note that O(log m)=O(log n).)

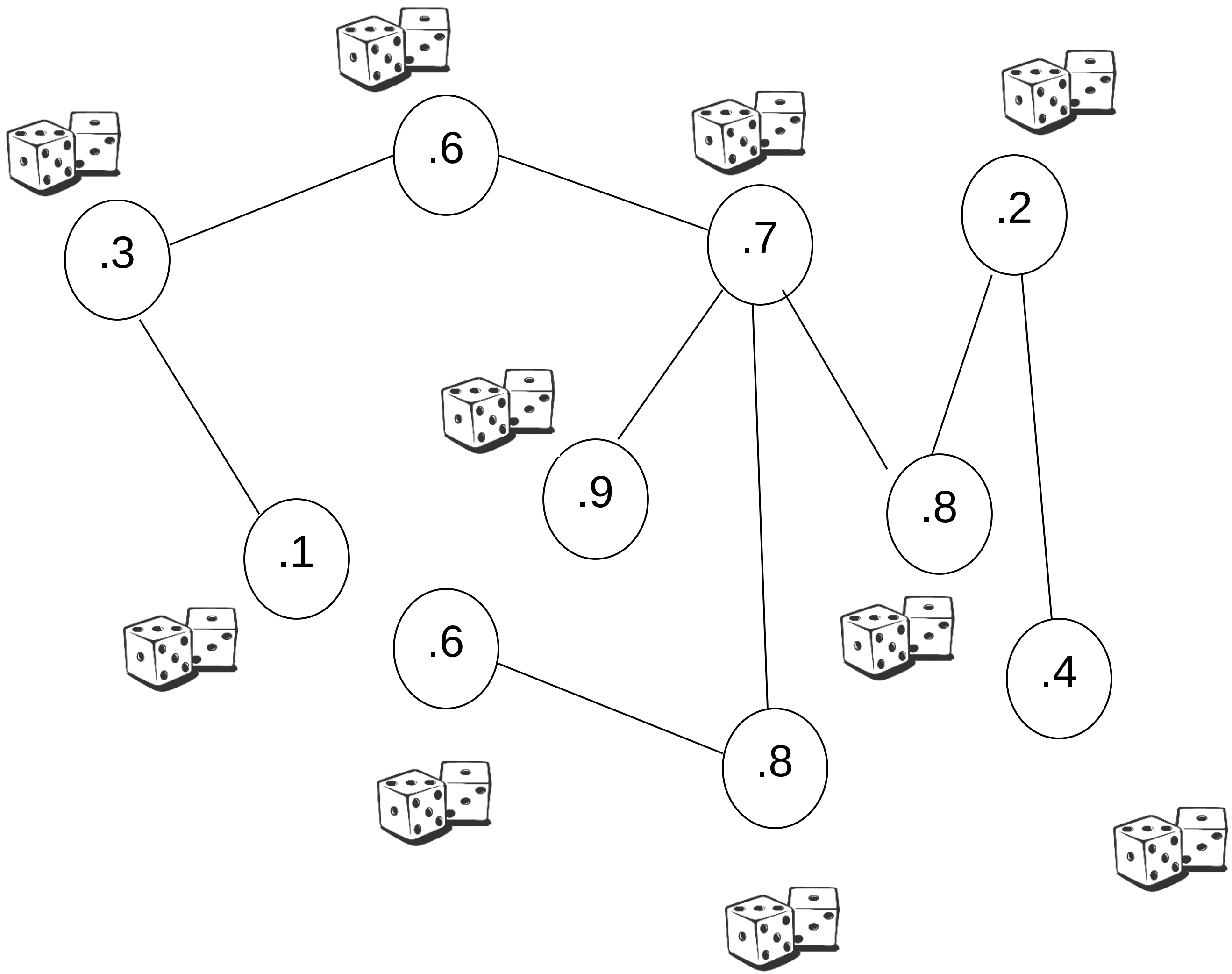
An Even Simpler log(n)-Time MIS Algorithm

An Even Simpler Algorithm



Round 1

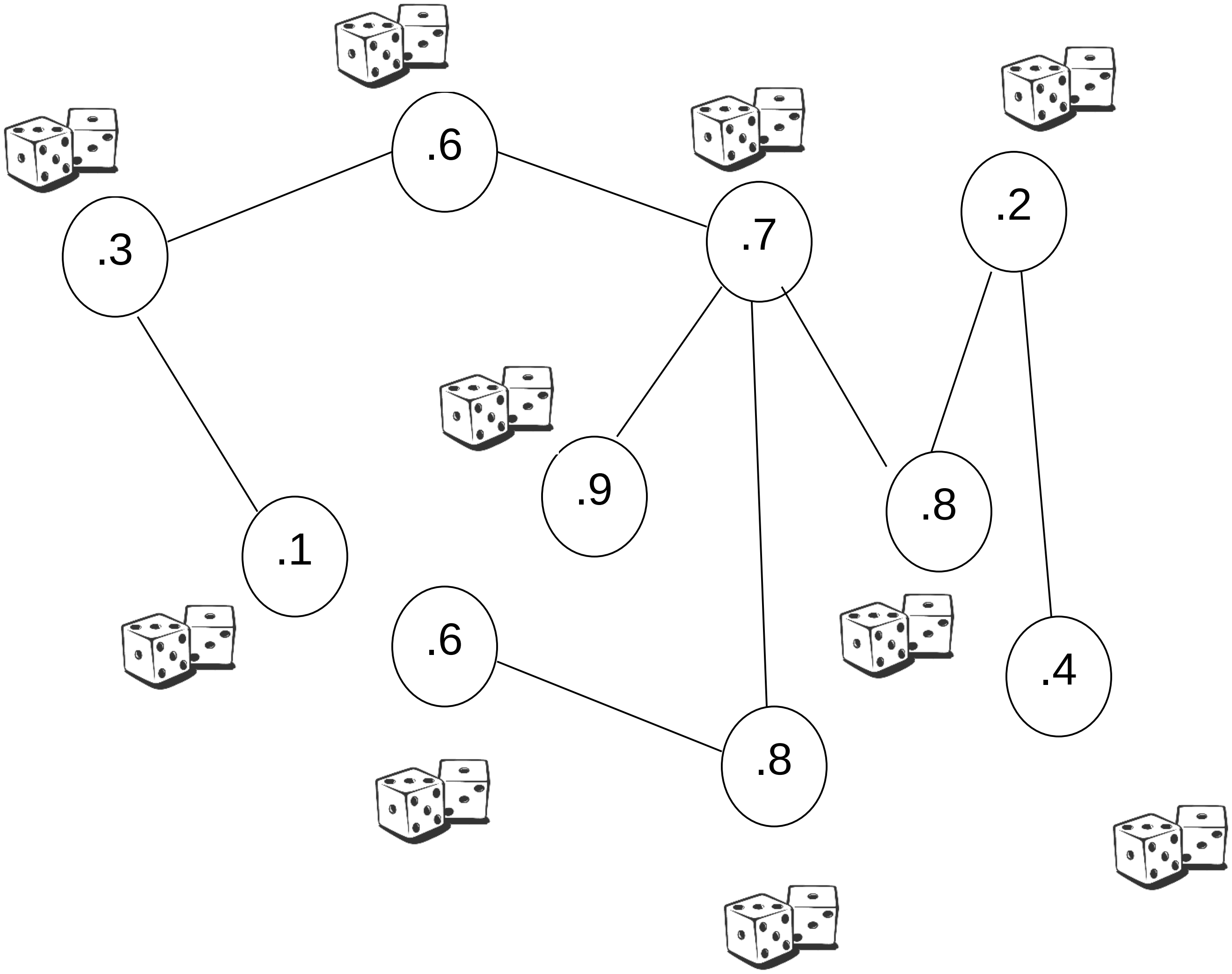
An Even Simpler Algorithm



Round 1

An Even Simpler Algorithm

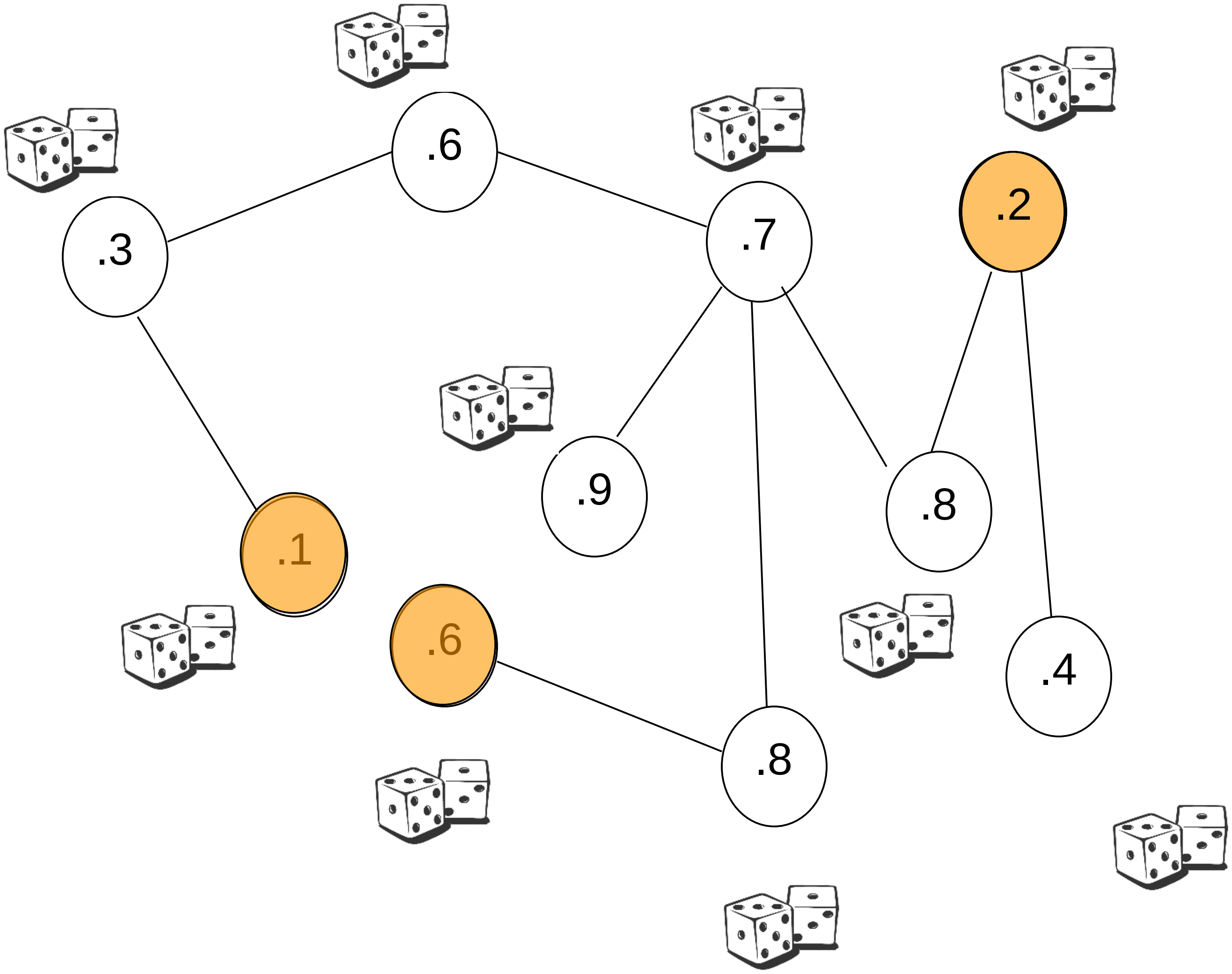
Join MIS if smallest random number in neighborhood!



Round 1

An Even Simpler Algorithm

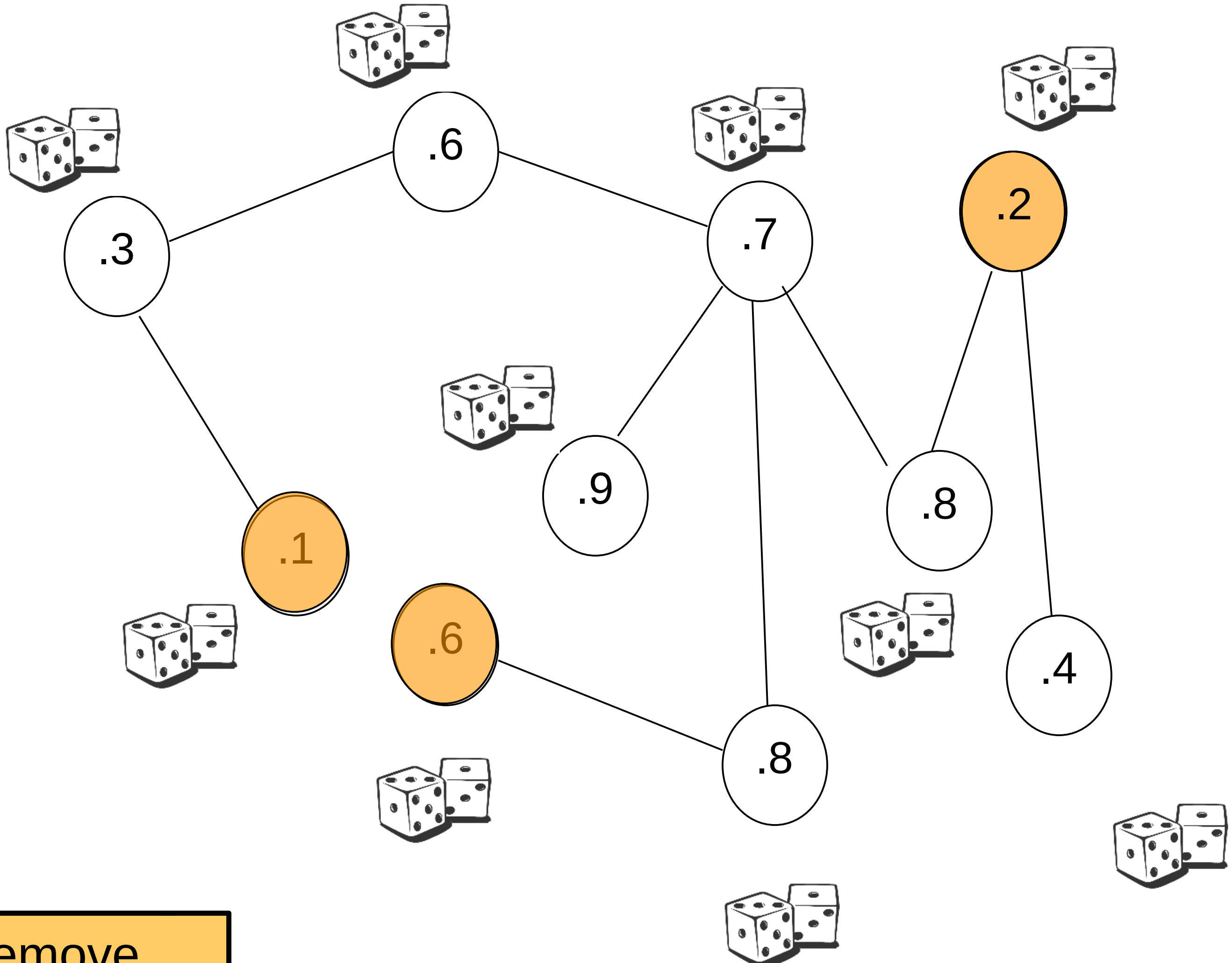
Join MIS if smallest random number in neighborhood!



Round 1

An Even Simpler Algorithm

Join MIS if smallest random number in neighborhood!

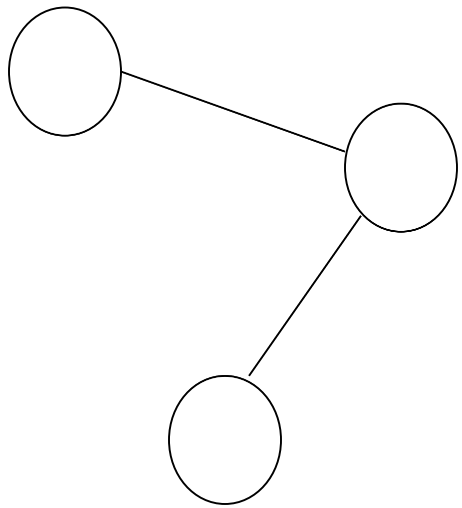


Then remove neighborhood!

Round 1

An Even Simpler Algorithm

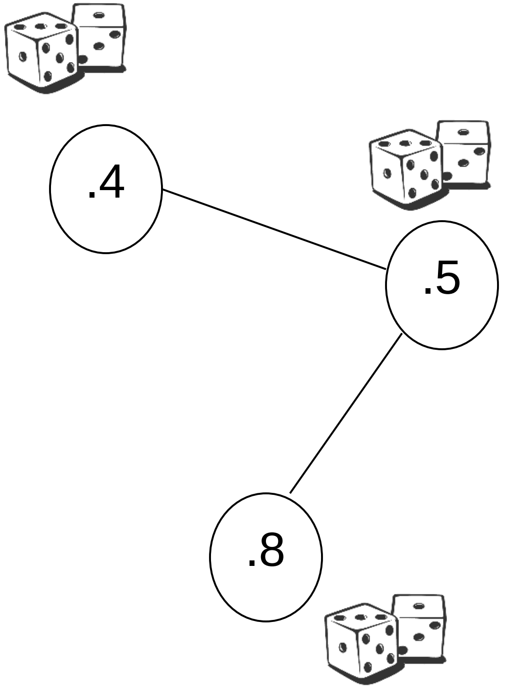
Join MIS if smallest random number in neighborhood!



Then remove neighborhood!

Round 1

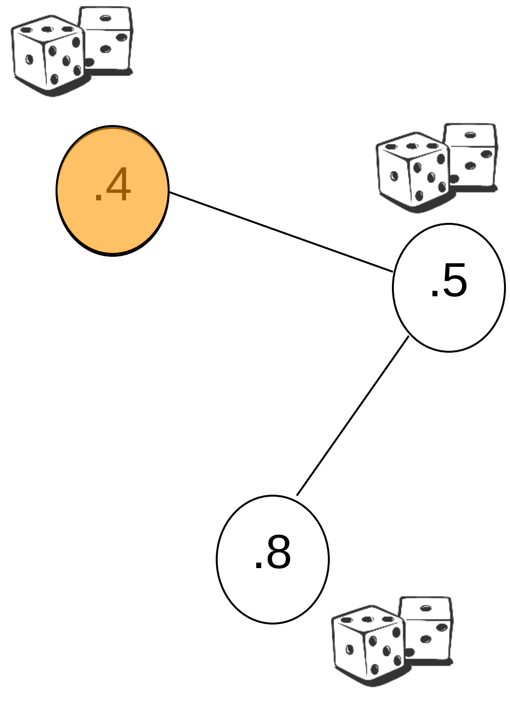
An Even Simpler Algorithm



And continue into Round 2!

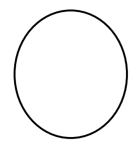
Round 2

An Even Simpler Algorithm



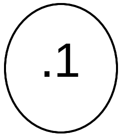
Round 2

An Even Simpler Algorithm



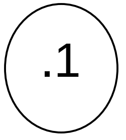
Round 3

An Even Simpler Algorithm



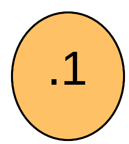
Round 3

An Even Simpler Algorithm



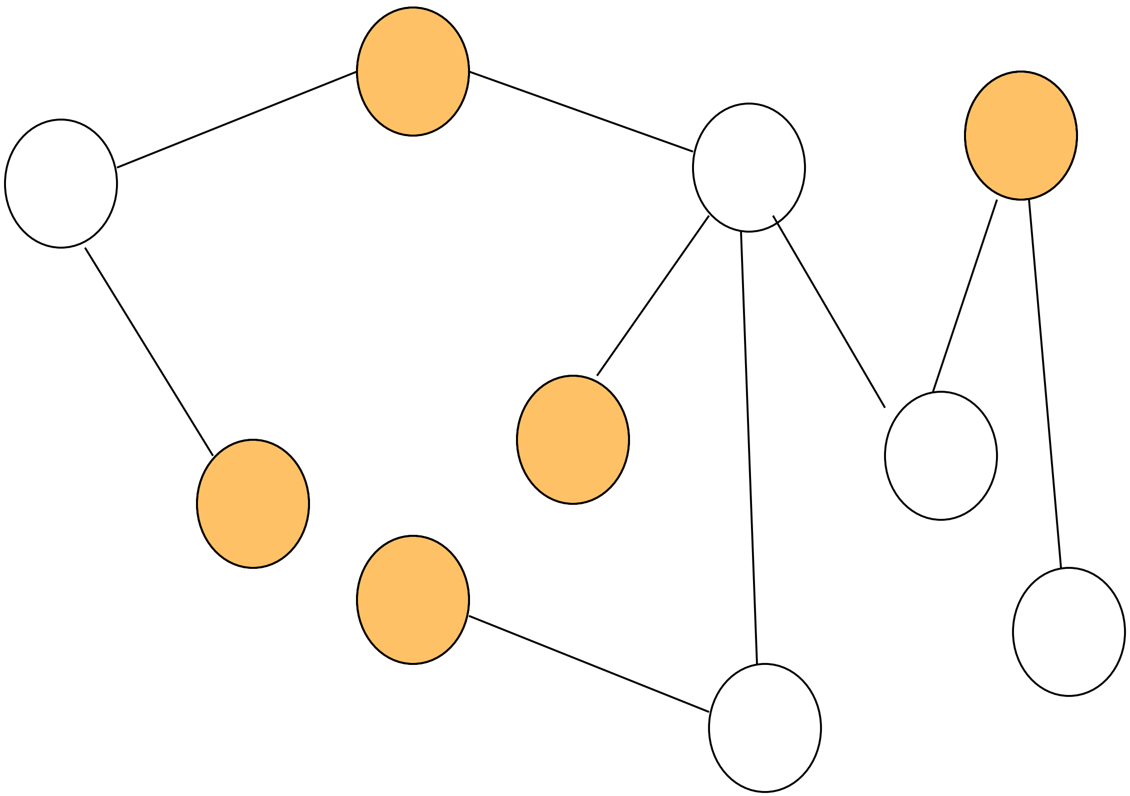
Round 3

An Even Simpler Algorithm



Round 3

An Even Simpler Algorithm



... done: MIS!

An Even Simpler Algorithm

Fast MIS (2009)

Proceed in rounds consisting of phases!

In a phase:

1. each node chooses a random value **r(v)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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 (2009)

Proceed in rounds consisting of phases!

In a phase:

1. each node chooses a random value **r(v)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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 are the resulting nodes independent?

An Even Simpler Algorithm

It is an 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)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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 are the resulting nodes independent?

Fast MIS (2009)

Proceed in rounds consisting of phases!

In a phase:

1. each node chooses a random value **r(v)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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 the independent set maximal?

Fast MIS from 2009...

Fast MIS (2009)

Proceed in rounds consisting of phases!

In a phase:

1. each node chooses a random value **r(v)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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!

It is a MIS:
Node with smallest random value will always join the IS, so there is always progress.

Why is the independent set maximal?

Fast MIS (2009)

Proceed in rounds consisting of phases!

In a phase:

1. each node chooses a random value **r(v)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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!

What is the runtime?

Fast MIS from 2009...

Fast MIS (2009)

Proceed in rounds consisting of phases!

In a phase:

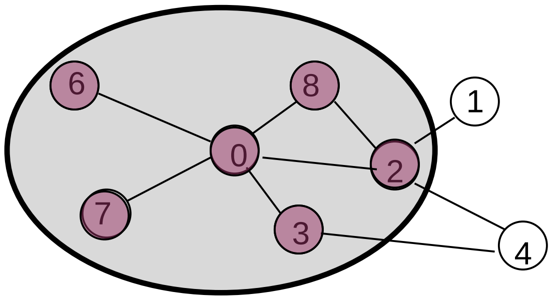
1. each node chooses a random value **r(v)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ N(v), node v enters the MIS and informs the neighbors
3. If v or a neighbor of v entered the MIS, v **terminates** (and v and edges are **removed**), otherwise v enters next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

What is the runtime?

Logarithmic as well!
How to prove?
Remember our tricks…

Idea of the Proof

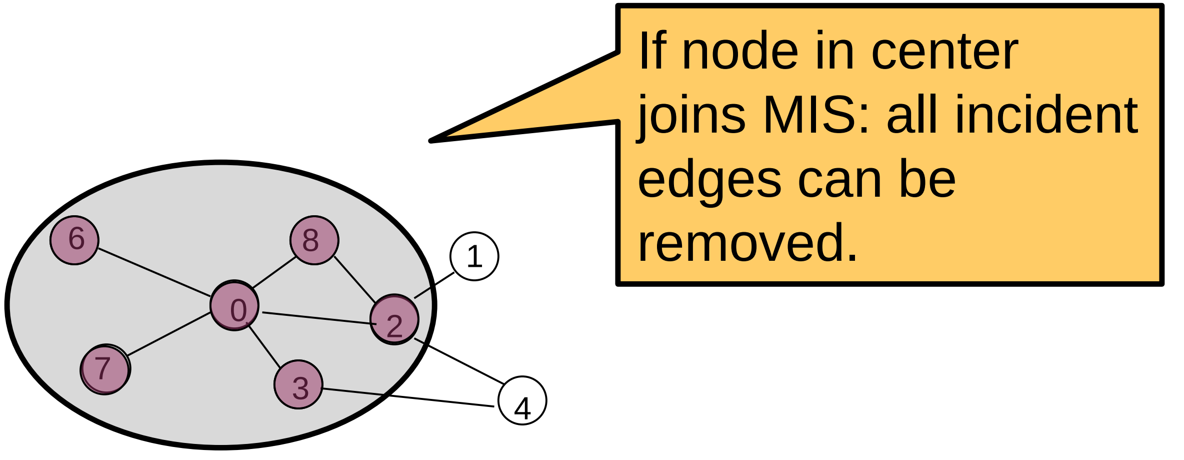
Idea: a constant fraction of edges removed in each round!



If node in center joins MIS: all incident edges can be removed.

Idea of the Proof

Idea: a constant fraction of edges removed in each round!

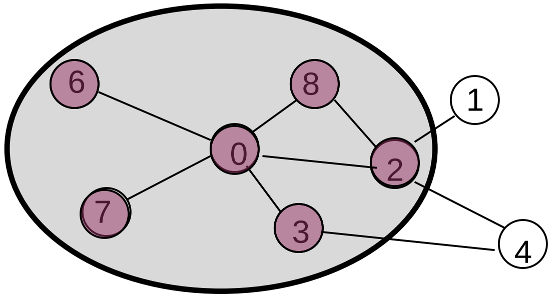


Probability that node v is removed = node v has smallest ID in neighborhood

This is at least 1/(d(v)+1)!

Idea of the Proof

Idea: a constant fraction of edges removed in each round!



If node in center joins MIS: all incident edges can be removed.

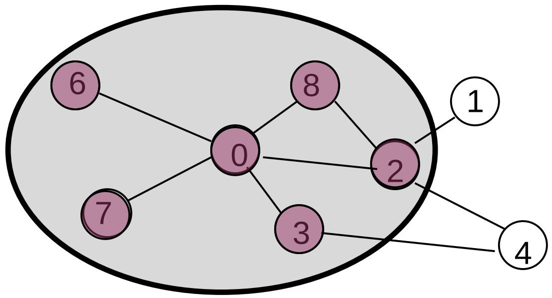
Probability that node v is removed = node v has smallest ID in neighborhood

This is at least 1/(d(v)+1)!

On this occasion, d(v) edges are removed!

Idea of the Proof

Idea: a constant fraction of edges removed in each round!



If node in center joins MIS: all incident edges can be removed.

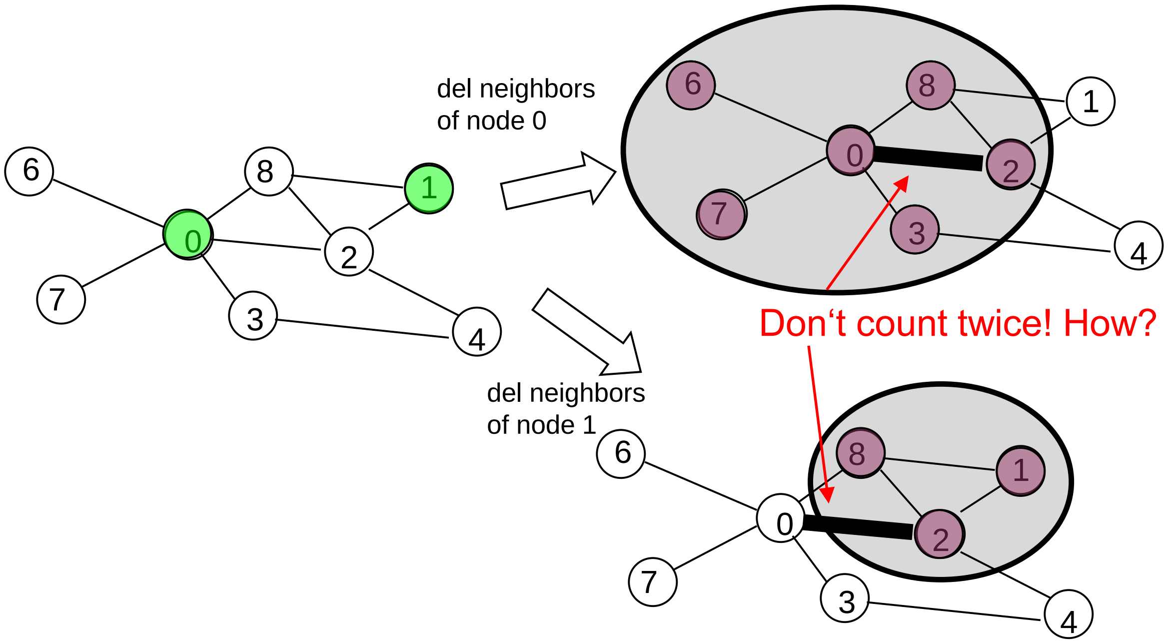
Probability that node v is removed = node v has smallest ID in neighborhood

This is at least 1/(d(v)+1)!

On this occasion, d(v) edges are removed!

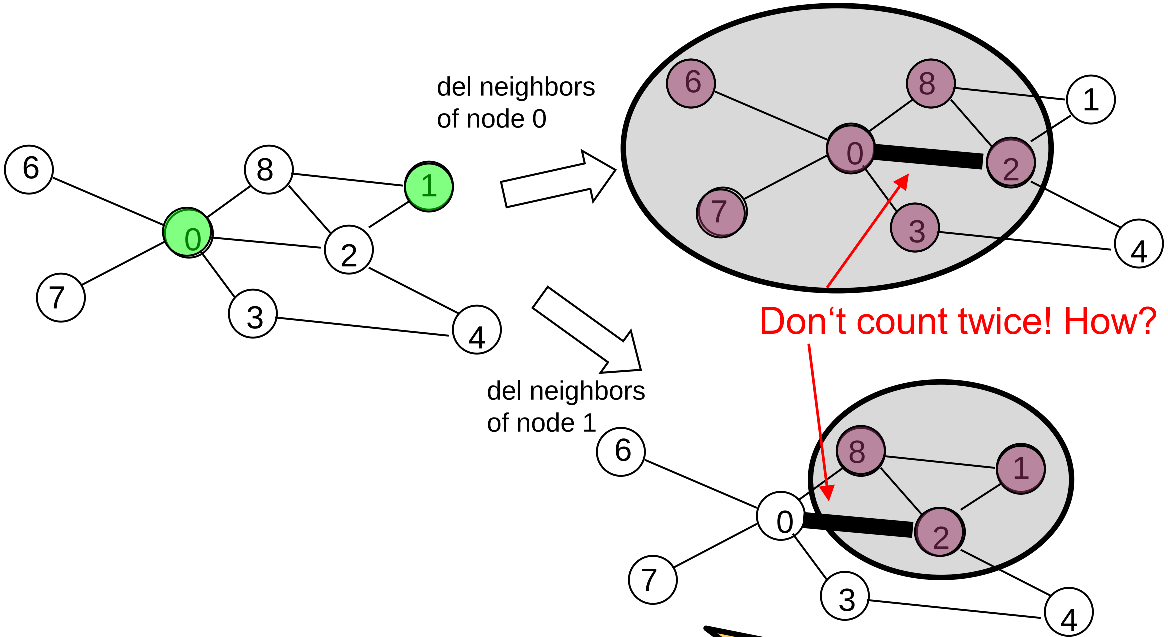
So on expectation (d(v)+1)/d(v)=O(1): great!

Wait a second! Double counting alert!



An edge appears in multiple neighborhoods!

Wait a second! Double counting alert!



An edge appears in

Idea to be on the safe side: 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)

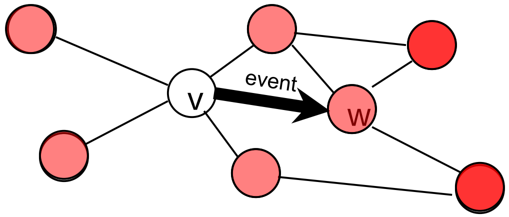
Let us define a more conservative event!

Event (v=>w)

(v=>w): per edge event: node v joins MIS and is even smaller than w's neighbors.

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

$$P\,[(v \Rightarrow w)] \geq ?$$



Edge Removal: Analysis (1)

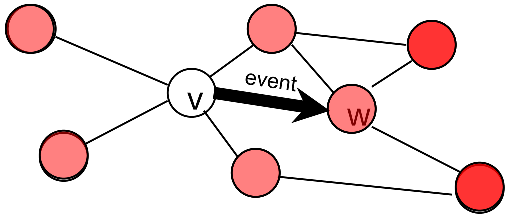
Let us define a more conservative event!

Event (v=>w)

(v=>w): per edge event: node v joins MIS and is even smaller than w's neighbors.

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 (1)

Event (v=>w)

(v=>w): per edge event: node v joins MIS and is even smaller than w's neighbors.

Let us define a more conservative event!

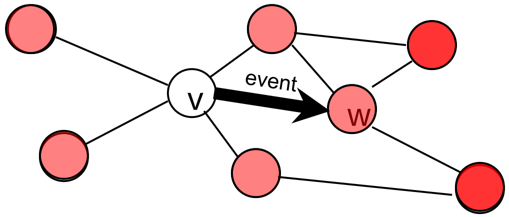
How many edges are removed? In both directions of the event: d(v)+d(w)! Still constant expectation.

nt that

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 (1)

Let us define a more conservative event!

Event (v=>w)

(v=>w): per edge event: node v joins MIS and is even smaller than w's neighbors.

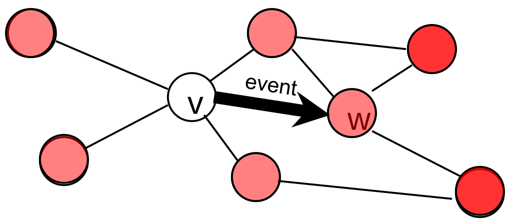
How many edges are removed? In both directions of the event: d(v)+d(w)! Still constant expectation.

nt that

of

the given neighbor?

$$P[(v => w)] \geq 1/(d(v)+d(w))$$



since d(v)+d(w) is the maximum possible num

nodes adjacent to v and w.

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

We still do some double counting, but we are almost there…

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!

MIS of 2009

Expected running time is O(log n).

Proof („MIS 2009“)?

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

QED

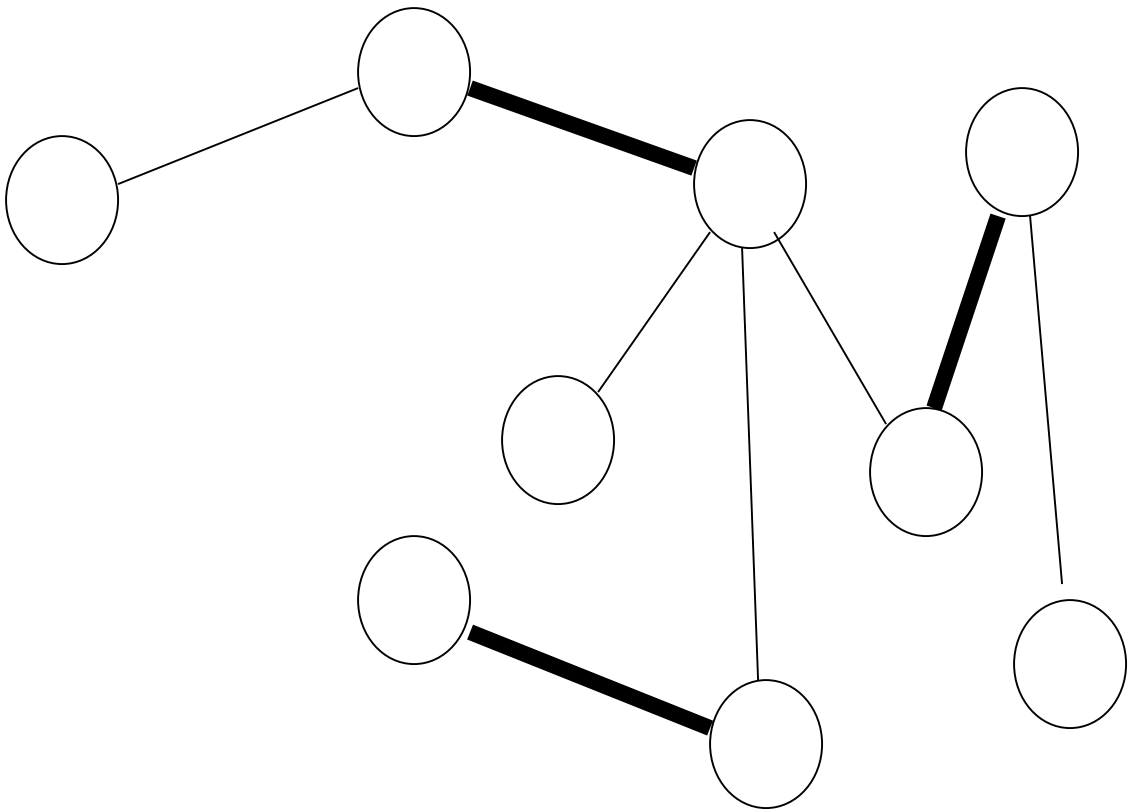
Actually, the claim even holds with h

Distributed MIS can also be used to solve distributed Matching and distributed Coloring problems!

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.

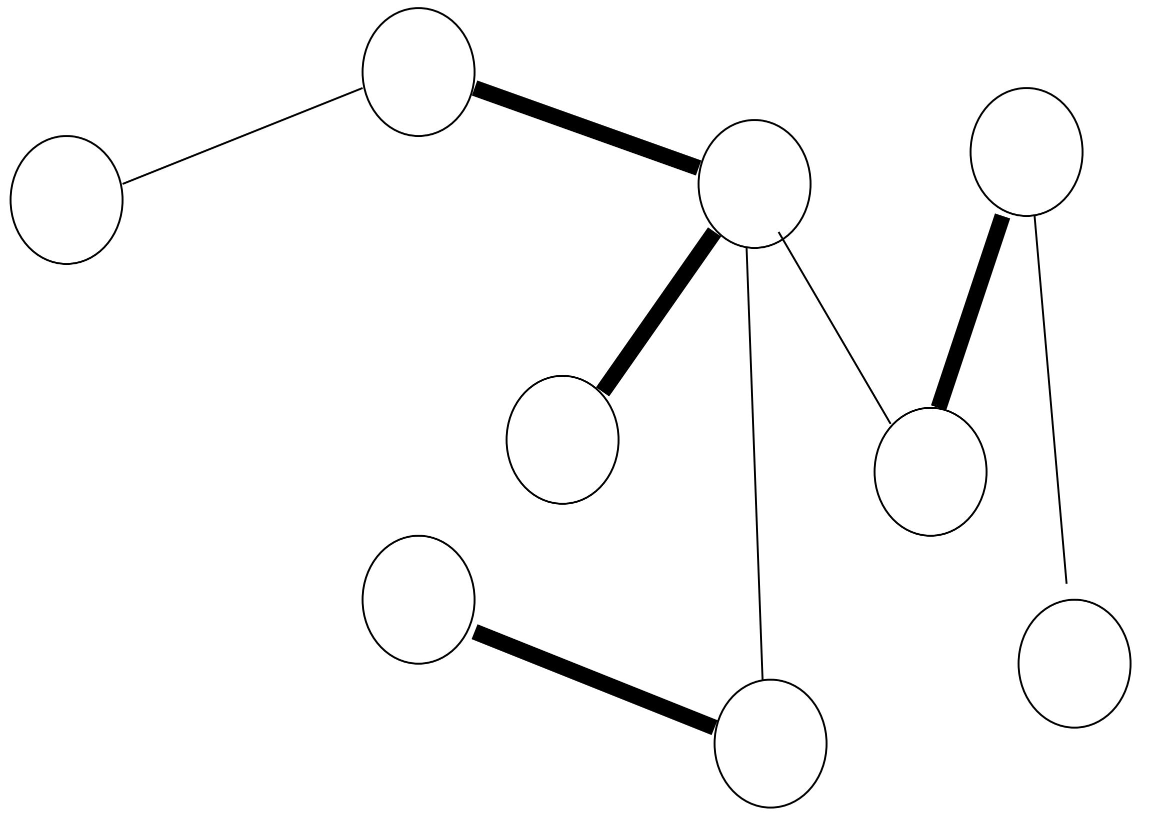
Excursion: Matchings



Matching? Maximal? Maximum? Perfect?

Maximal.

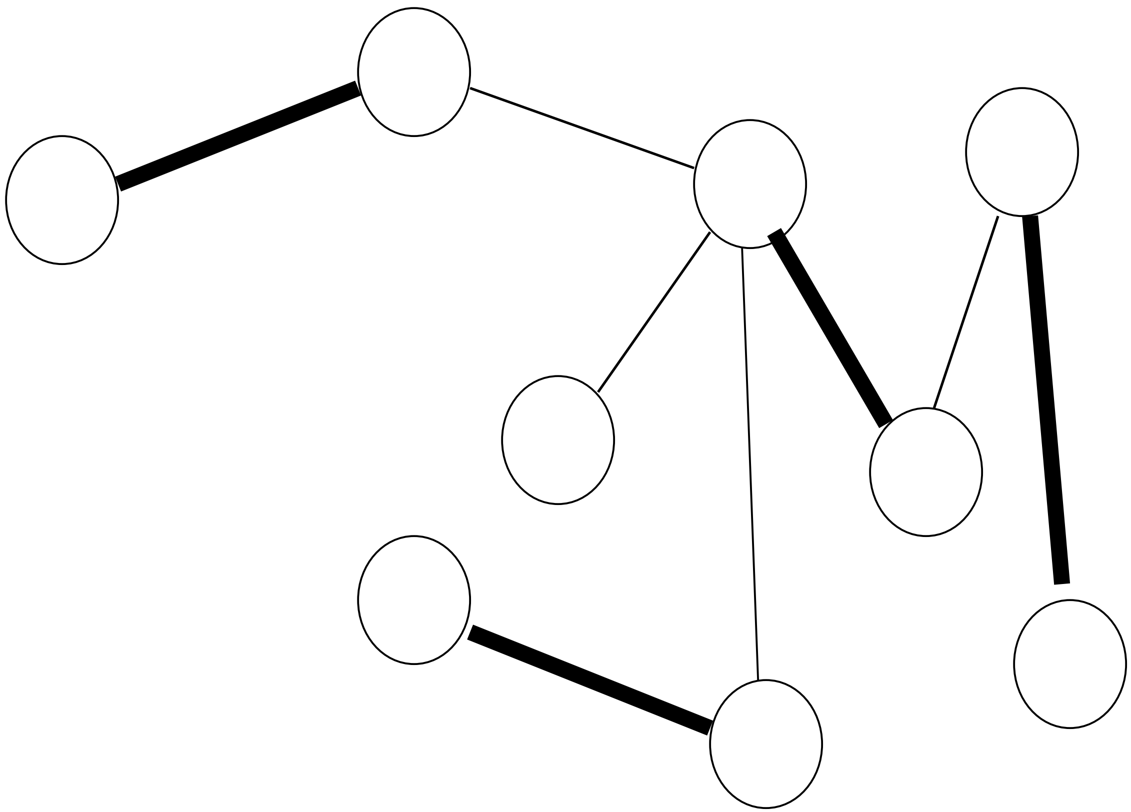
Excursion: Matchings



Matching? Maximal? Maximum? Perfect?

Nothing.

Excursion: Matchings



Matching? Maximal? Maximum? Perfect?

Maximum but not perfect.

Matching

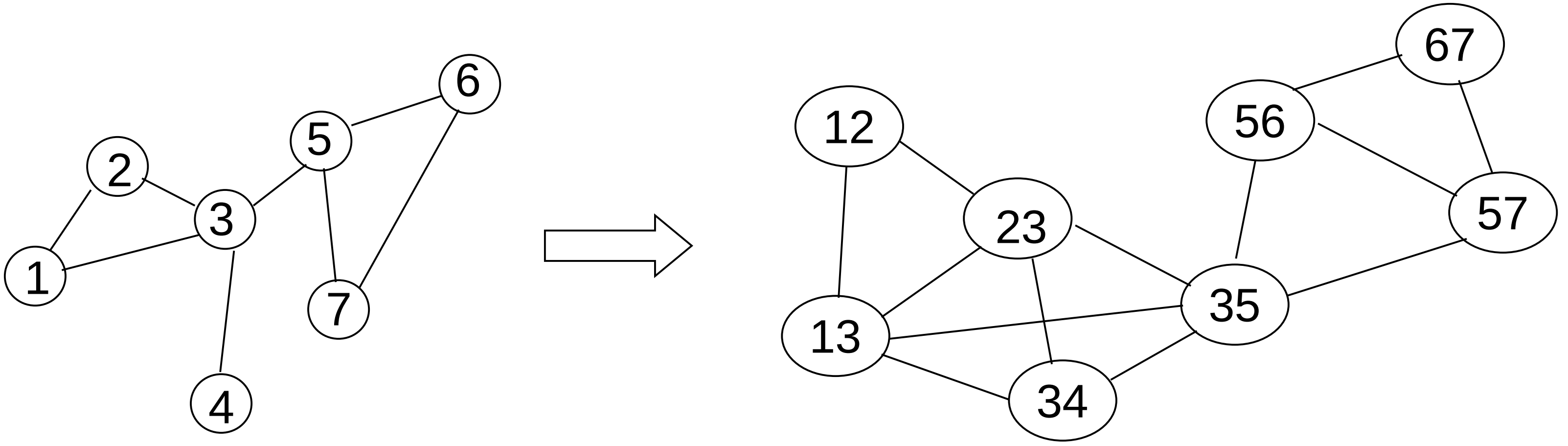
A matching is a subset M of edges E such that no two edges in M are adjacent.
A maximal matching cannot be augmented.
A maximum matching is the best possible.
A perfect matching includes all nodes.

How to compute with an IS algorithm?

Discussion: Matching

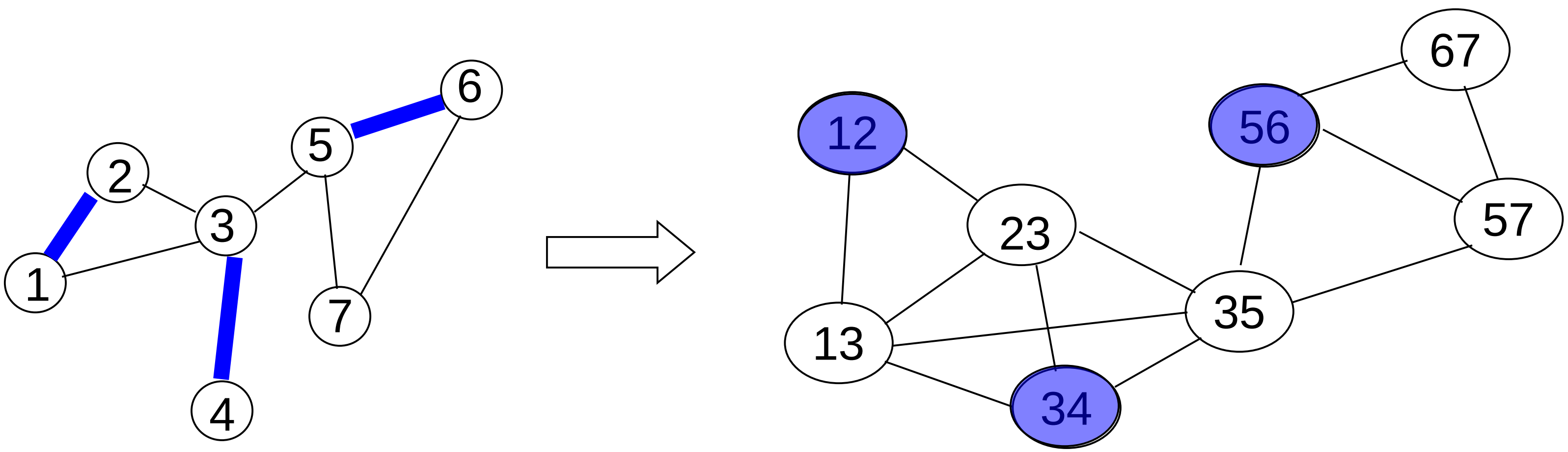
An IS algorithm is a matching algorithm! How?

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!



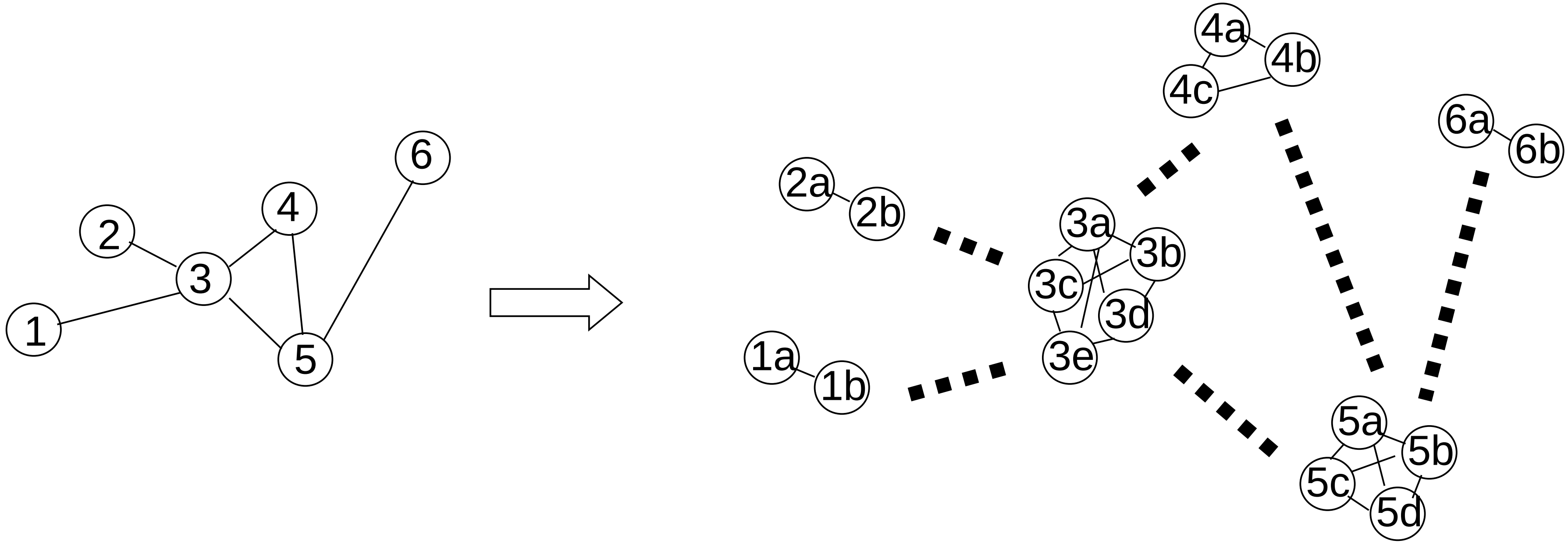
Discussion: Graph Coloring

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 match edges from i-th clone to i-th clone. Then?

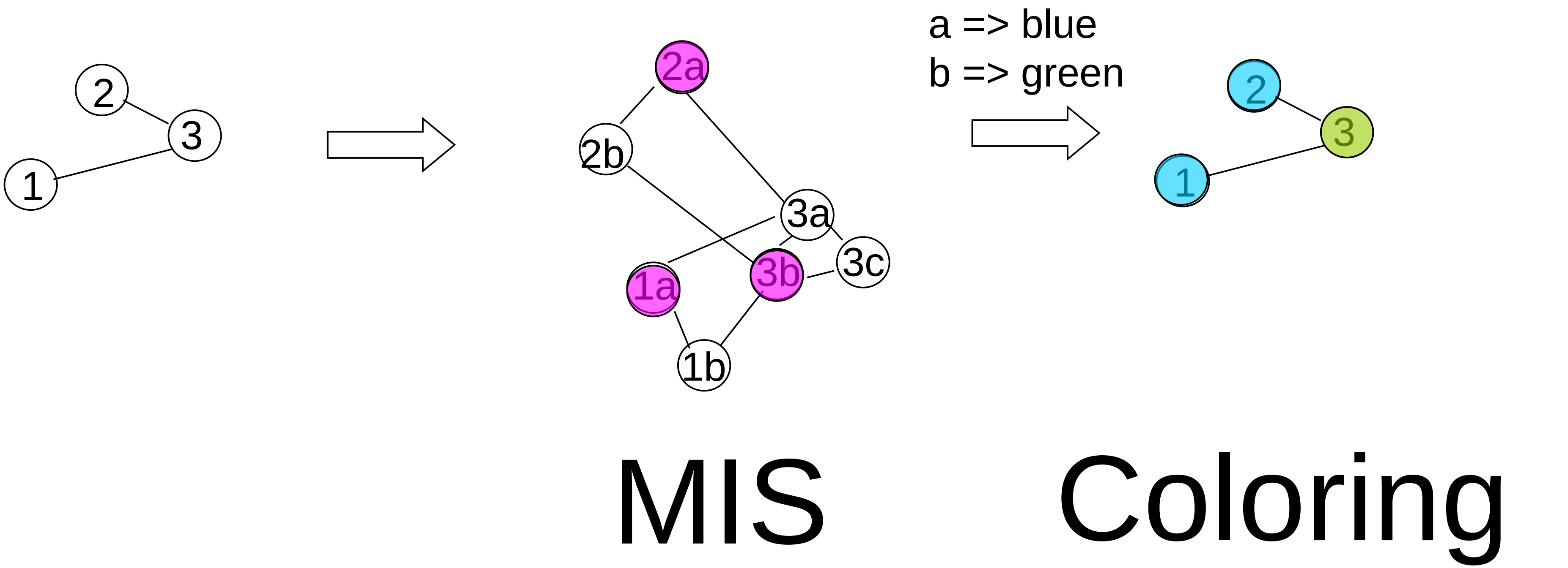
Run MIS: if i-th copy is in MIS, node gets color i.



Discussion: Graph Coloring

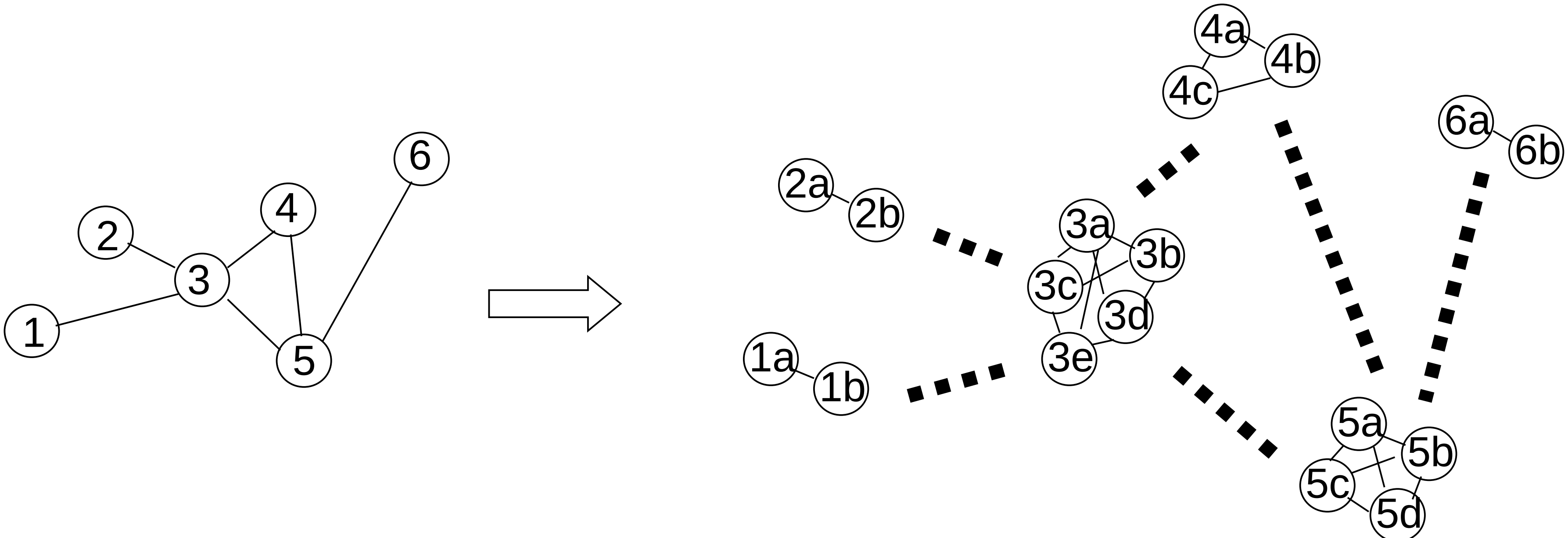
Example:

How to use a MIS algorithm for graph coloring?



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...

Discussion: Dominating Set

Dominating Set

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

How to compute a dominating set?
See lecture notes. ☺

Literature for further reading:

- Peleg‘s book (as always ☺)

End of lecture

Exercise 3: Maximal Independent Set

In the lecture, we discussed a slow but simple deterministic maximal independent set (MIS) algorithm (Algorithm 34) in which the decisions of the nodes are based on their identifiers. The time complexity of this algorithm is $O(n)$.

We might hope that if the nodes with the largest degrees, i.e., the largest number of neighbors, decide to enter the MIS, the set of undecided nodes reduces the most. In the following algorithm we try to exploit the knowledge of the node degrees:

Assume that each node knows its degree and also the degrees of all its neighbors. If a node has a larger degree than all its undecided neighbors, it joins the MIS and informs its neighbors. Once a node v learns that (at least) one of its neighbors joined the MIS, v decides not to join the MIS.

Naturally, the algorithm does not make any progress if two or more neighboring nodes share the largest degree. As this is a difficult problem, we will assume in the following that this situation does not occur, i.e., if a node v has the largest degree, then no neighboring node has the same degree as v.[1]

a) Draw a graph that illustrates that this algorithm has a large time complexity for trees! Give a (non-trivial) lower bound on the (worst-case) time complexity for trees consisting of n nodes!

b) Construct a graph that shows that the time complexity of this algorithm is even worse for arbitrary graphs than for trees! What is the time complexity?

Solution 3: Deterministic Maximal Independent Sets

Lecture: two randomized algorithms!
What about deterministic solutions?

Recall algo from lecture:

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

Unfortunately, not faster than sequential algorithm!
E.g., sorted line: O(n) time.

Solution 3: Deterministic Maximal Independent Sets

How to do better?

Idea: Nodes with high degree should decide first, so many node decide sooner?

Degree MIS

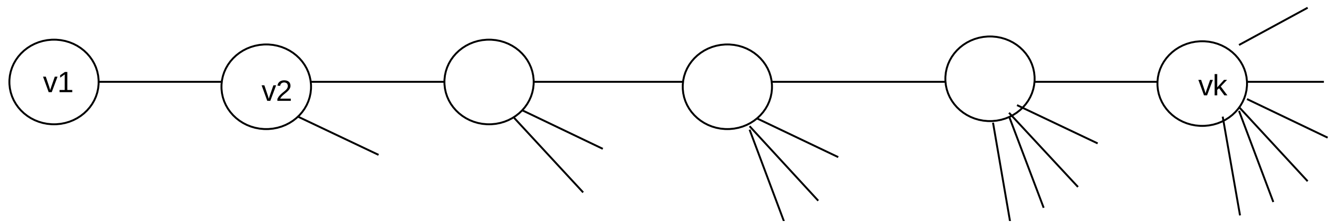
assume nodes v know degrees, also of neighbors
1. If node v has larger degree than undecided neighbors, join MIS and inform neighbors
2. Once a neighbor joins MIS, a node decides not to join MIS

Assume: no node has same degree as neighbors, should only help us, so let's see...

Solution 3: Deterministic Maximal Independent Sets

Degree MIS is bad, even on trees:

- Line with increasing degrees.
- No neighbor has same degree.



1st round: Only v_k joins MIS. (All other nodes lower degree.)
2nd round: Only v_{k-1} decides (not to join).
3rd round: Etc.
k-th round: All nodes have decided. => **Complexity Θ(k)**

What is k?

The number of nodes in this graph is

$$n = k + \sum_{i=1}^{k}(i-1) + 1 = 1 + \sum_{i=1}^{k} i = 1 + \frac{k(k+1)}{2} < \frac{(k+1)^2}{2}.$$

The time complexity is thus $k \geq \sqrt{2n} - 1 \in \Omega(\sqrt{n})$.

Solution 3: Worse on General Trees

Degree MIS is even worse on general graphs:

First note that the $\sqrt{n}$ bound is tight for trees: it cannot be worse than that.

For contradiction, assume a node v0 with degree d(v0) is undecided at time $2\sqrt{n}$. So v0 must have an undecided neighbor v1 with degree d(v1)>d(v0) a time $2\sqrt{n}$-1.

So by induction, there are nodes $v_0, \ldots, v_{2\sqrt{n}}$ such that vi and v_{i+1} are neighboring, and $d(v_i)<d(v_{i+1})$.

So there are at least

$$\sum_{i=0}^{2\sqrt{n}} \delta(v_i) - 1 = \sum_{i=0}^{2\sqrt{n}} i = \frac{2\sqrt{n}(2\sqrt{n}+1)}{2} > 2n$$

nodes in the tree (no neighbors can be shared, since tree cycle-free). Contradiction!

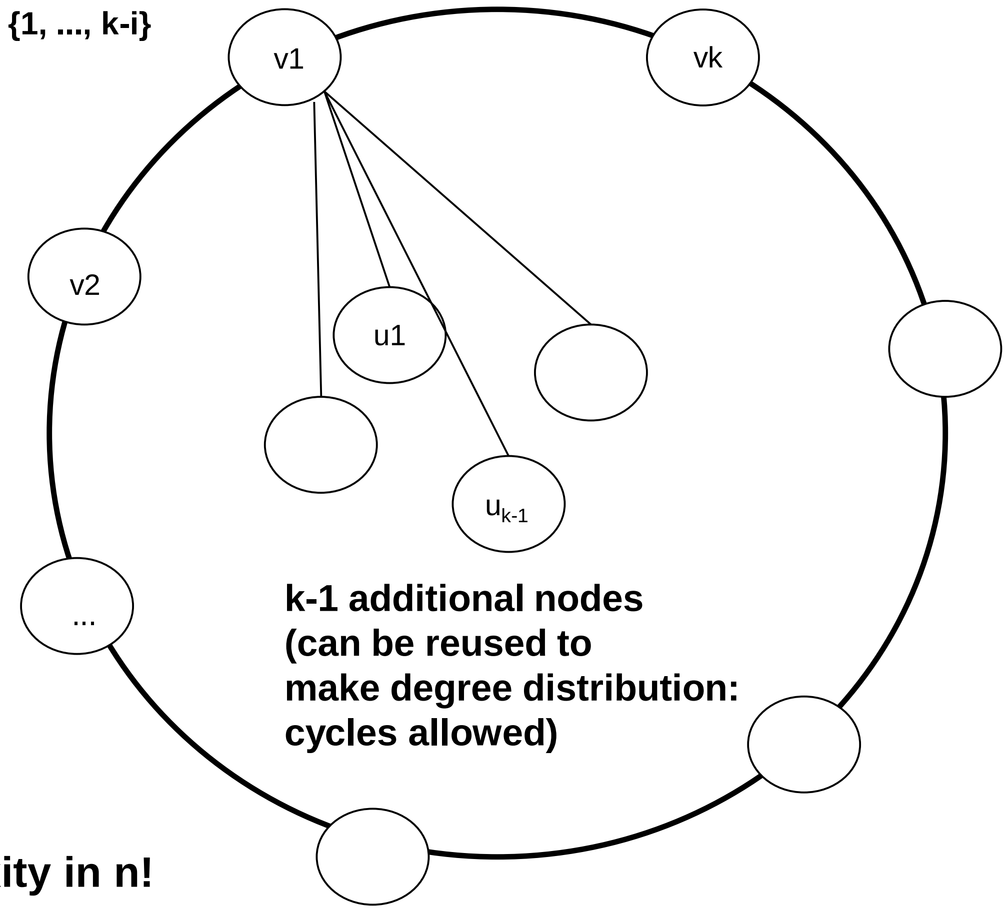
Solution 3: Worse on General Trees

Degree MIS is even worse on general graphs:

Consider the ring:

vi connected to
all uj with j in {1, ..., k-i}

So:
d(vi) = k+2-i
d(uj) = k-j



Execution:

1st round: v1 joins MIS

2nd round: u1, …, u_{k-1}, and v2 and vi cannot join anymore => tell it to neighbors

3rd round: only v3 decides (all other nodes have Undecided higher-degree neighbor)

4th round: only v4 decides.

(k-1) th round: v_{k-1} decides.

Since 2k-1=n, we get a linear complexity in n!

Backup Slides

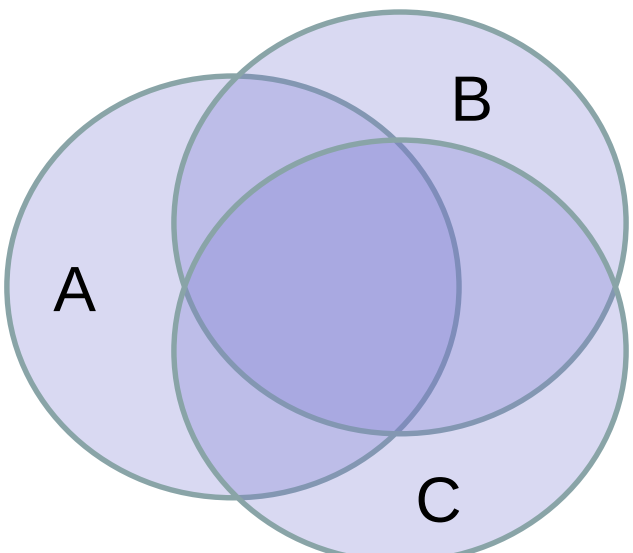
Probability 101

All you will need in the analysis:

Inclusion Exclusion Principle

$$|A \cup B \cup C| = |A| + |B| + |C|$$
$$- |A \cap B| - |A \cap C| - |B \cap C|$$
$$+ |A \cap B \cap C|$$

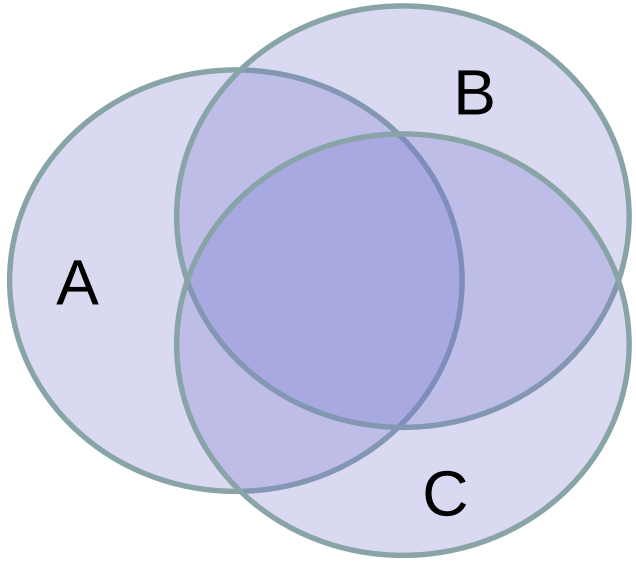
Why?



- Pairs overlaps twice
- But then need to add middle again!

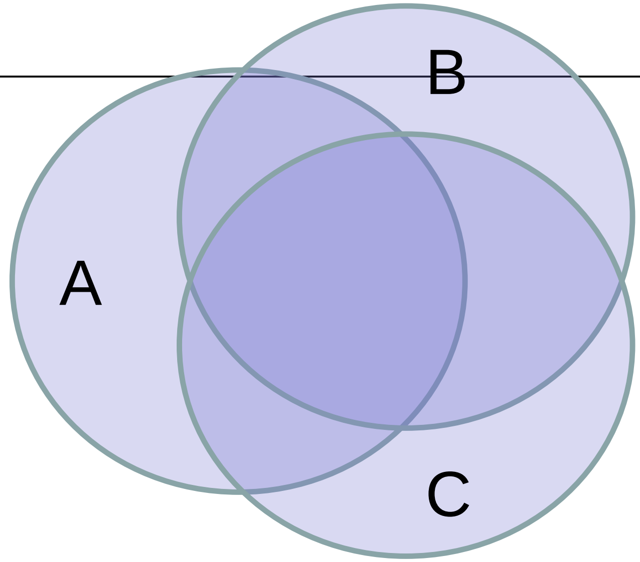
Can be generalized to k sets!

Same for probability:



If disks describe event space:

$$
\begin{aligned}
P[\text{ event} \in A \cup B \cup C] = \; & P[A] + P[B] + P[C] \\
& - P[A \cap B] - P[A \cap C] - P[B \cap C] \\
& + P[A \cap B \cap C]
\end{aligned}
$$



Upper and lower bounds

Can get **upper bound** if omit from minus sign

$$P[\text{ event } \in A \cup B \cup C\ ] \leq P[\ A\ ] + P[\ B\ ] + P[\ C\ ]$$

Can get **lower bound** if omit from plus sign

$$P[\text{ event } \in A \cup B \cup C\ ] \geq P[\ A\ ] + P[\ B\ ] + P[\ C\ ] - P[\ A \cap B\ ] - P[\ A \cap C\ ] - P[\ B \cap C\ ]$$

Let us consider the following Marking Algorithm

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

If $d(v)=0$, assume probabiliy = 1

MIS 1986

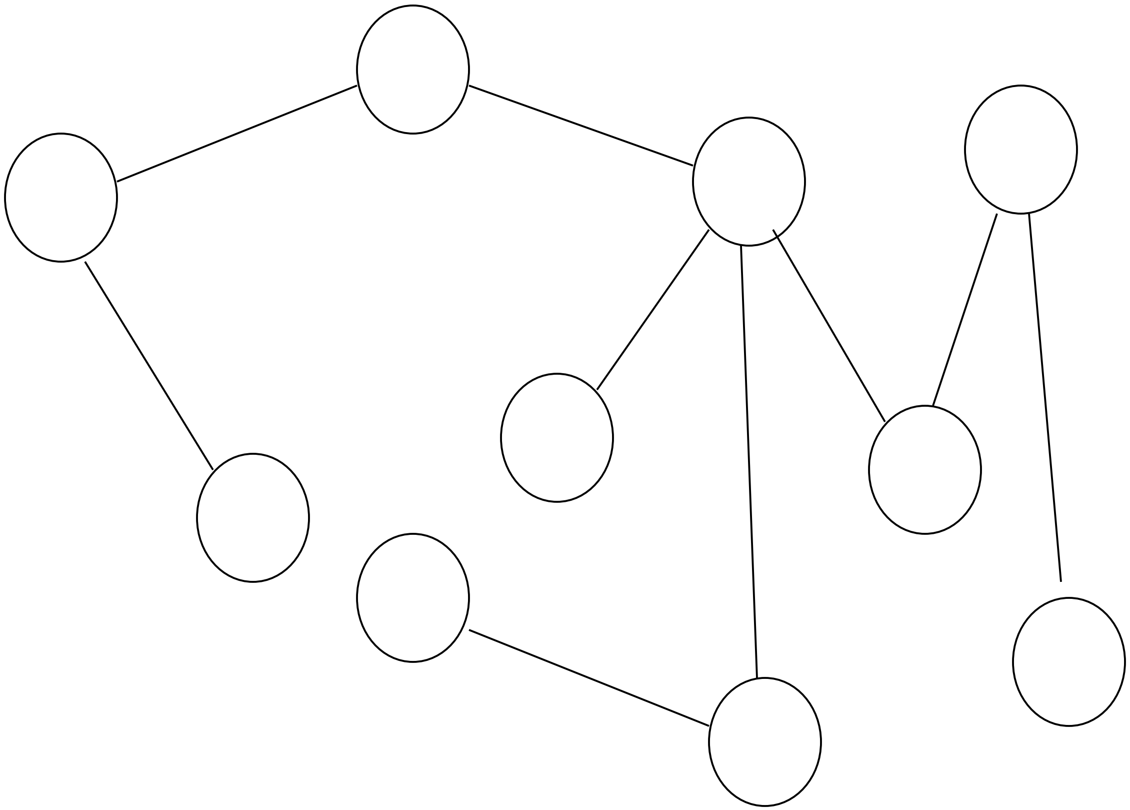
Fast MIS (1986)

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?



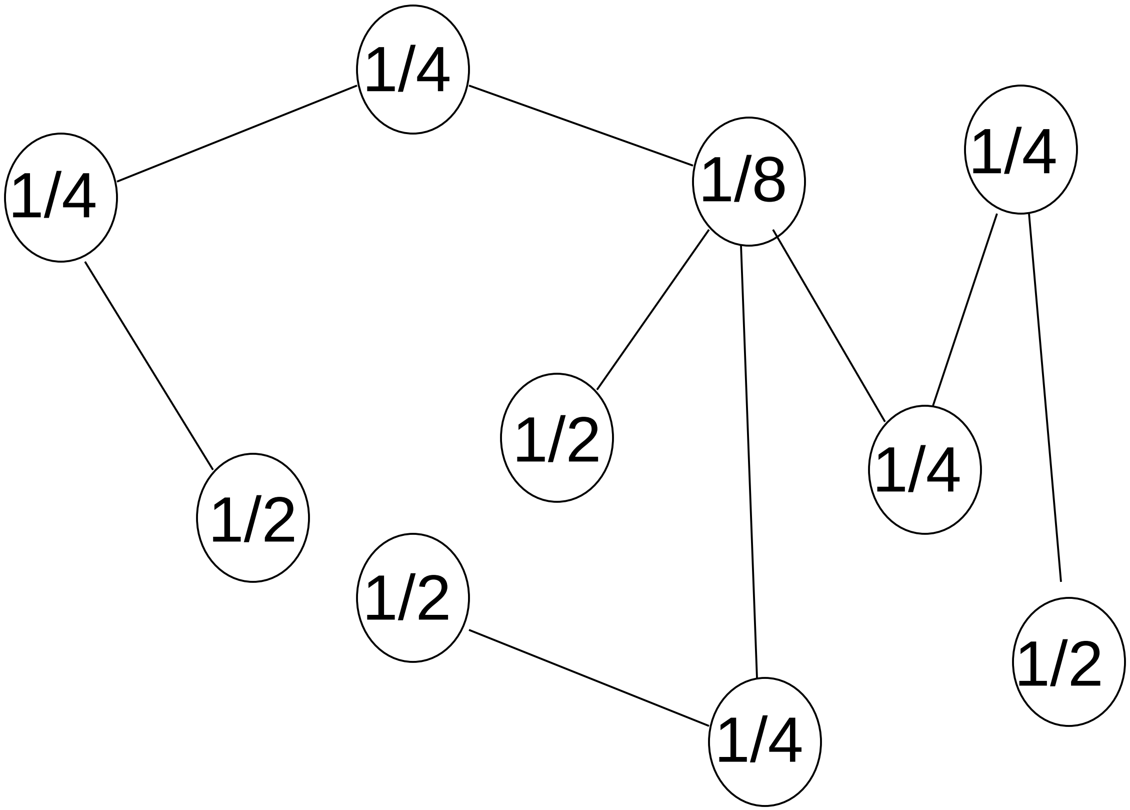
MIS 1986

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

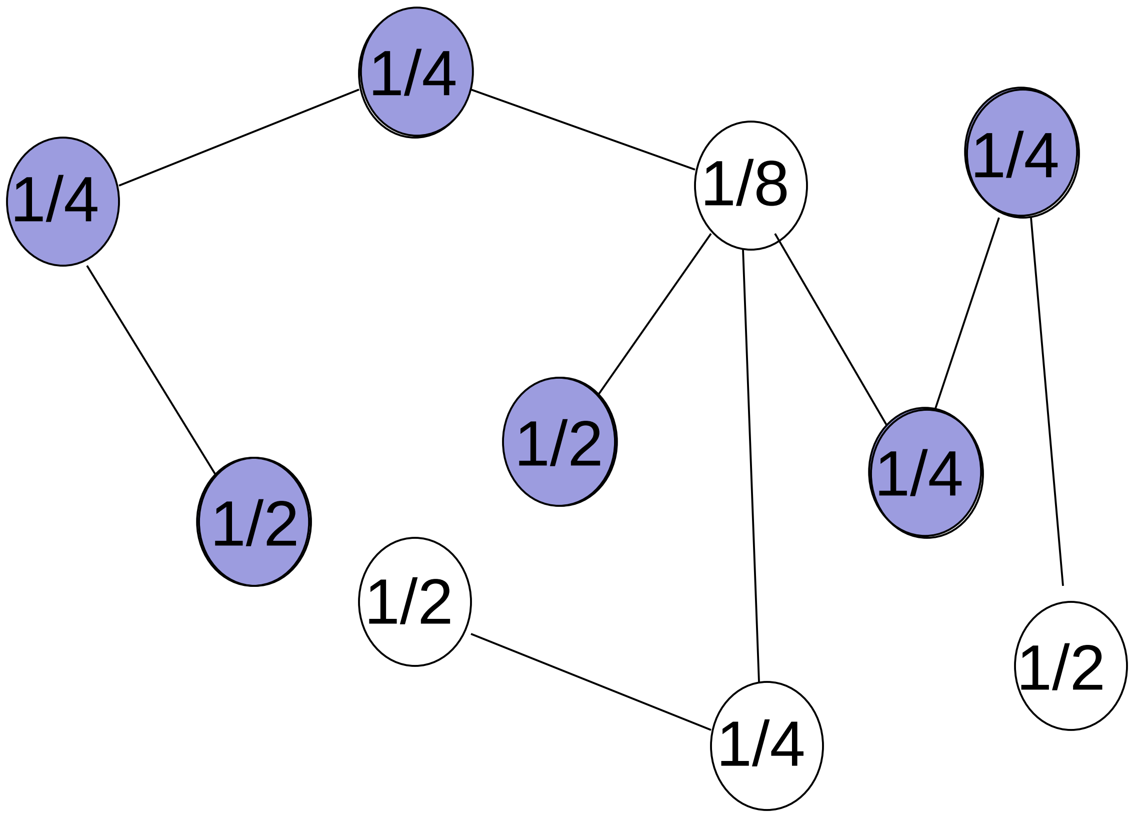
Marking... Who stays?

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



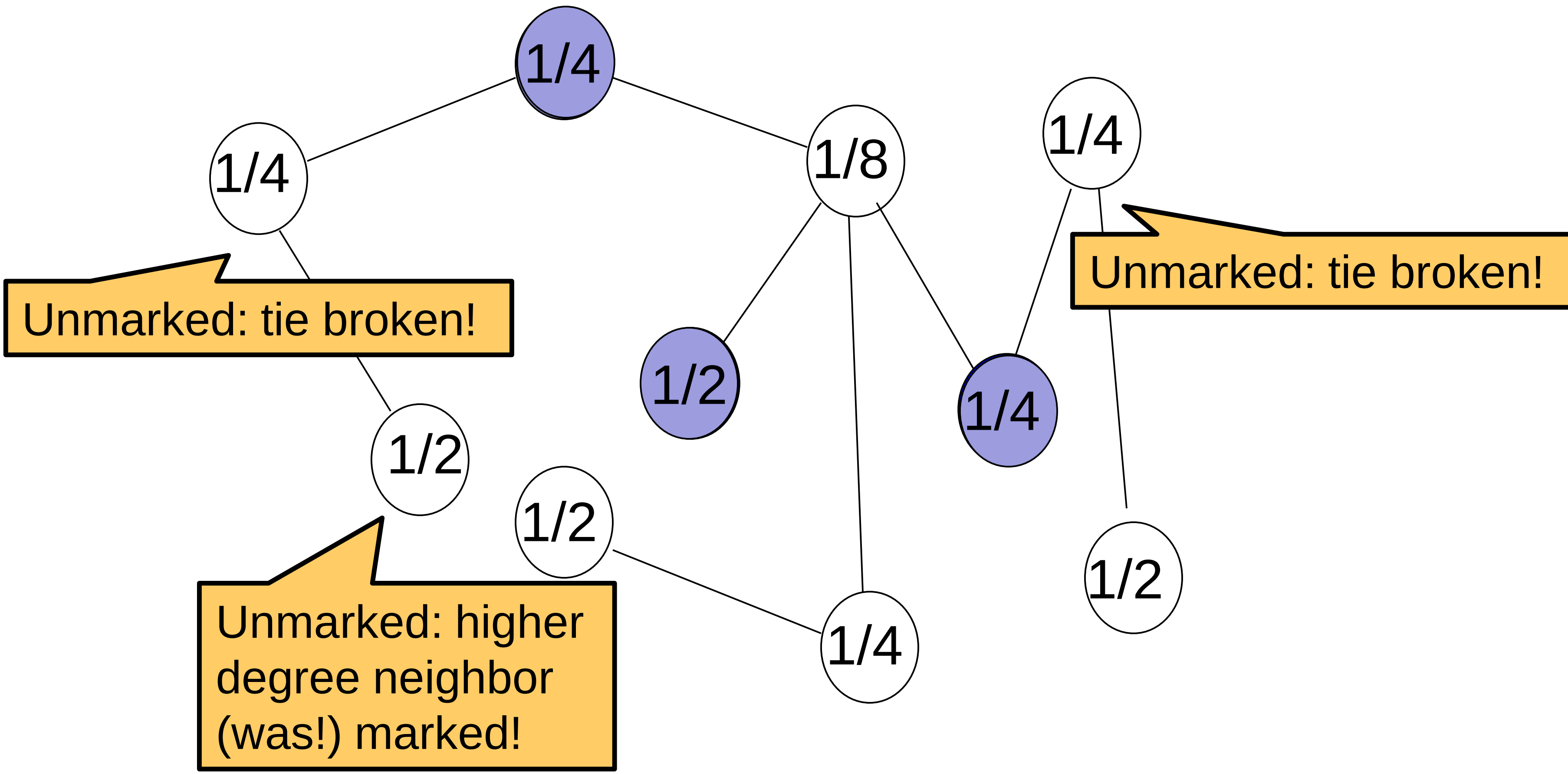
MIS 1986

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



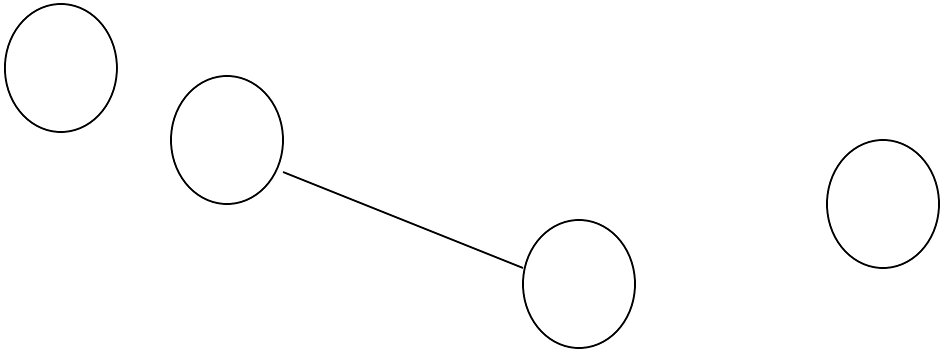
MIS 1986

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

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 from 1986...

Fast MIS (1986)

High degree nodes are unlikely to mark themselves!

hases

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

neighbors, as th

But are likely to win the neighbor competition later and join MIS.

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?

Correctness

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, a they cannot join the MIS anymore

It is an IS: Step 2 ensures that node only joins if neighbors do not (due to degree or tie breaking)!

It is a MIS: At some time, a node without MIS neighbors marks itself 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, a they cannot join the MIS anymore

It is an IS: Step 2 ensu hat
node only joins i
to degree or tie b

What about the distributed runtime?

It is a MIS: At some time, a node without MIS neighbors marks itself in Step 1.

Goal: Proving a logarithmic runtime

There are many ways to show a logrithmic runtime. In general: show that the remaining problem size is divided in two in each round.

- Idea 1: Each node is removed with constant probability (e.g., ½) in each round. So half of the nodes vanish in each round.

- Idea 2: Each edge is removed with constant probability in each round! Also works: $O(\log m) = O(\log n^2) = O(\log n)$

log n: How many times do I have to **:2** until <2?

$$n, n/2, n/4, n/8, \ldots, 8, 4, 2, 1$$

$$\xleftrightarrow{\hspace{6em}}$$

log n

Goal: Proving a logarithmic runtime

There are many ways to show a logrithmic runtime. In general: show that the remaining problem size is divided in two in each round.

- Idea 1: Each node is removed with constant probability (e.g., ½) in each round. So half of the nodes vanish in each round.

- Idea 2: E tant probability in each round! Also wor n)

Neither is true! ☹

log n: How many times do I have to **:2** until <2?

n, n/2, n/4, n/8, …, 8, 4, 2, 1

log n

Goal: Proving a logarithmic runtime

There are many ways to show a logrithmic runtime. In general: show that the remaining problem size is divided in two in each round.

- Idea 1: Each node is removed with constant probability (e.g., ½) in each round. So half of the nodes vanish in each round.

- Idea 2: E[...]tant probability in each round! Also wor[...] n)

Neither is true! ☹

log n:

However: A constant fraction of edges are removed with constant probability!

log n

Analysis

Let us compute the probability p that any node v joins MIS in Step 2!

We know:

- At most 1/2d(v): node only marked with this probability.
- Once marked, needs to join MIS: only if largest degree...

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

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. 1/(2d(v)), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$P[v \notin MIS \mid v \in M] = P[\exists\, w \in H(v), w \in M \mid v \in M]$$

$$= P[\exists\, w \in H(v), w \in M]$$

Definition of algorithm! A neighbor was the reason for v not to join MIS.

$$= \sum_{w \in H(v)} 1/(2d(w))$$

$$\leq \sum_{w \in H(v)} 1/(2d(v))$$

$$\leq d(v)/(2d(v)) = 1/2$$

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. 1/(2d(v)), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$P[v \notin MIS \mid v \in M] = P[\exists\ w \in H(v), w \in M \mid v \in M]$$
$$= P[\exists\ w \in H(v), w \in M]$$
$$\leq \sum_{w \in H(v)} P[w \in M]$$

Probability is independent of whether v is marked or not!

$$\leq \sum_{w \in H(v)} 1/(2d(v))$$
$$\leq d(v)/(2d(v)) = 1/2$$

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$P[v \notin MIS \mid v \in M] = P[\exists\ w \in H(v), w \in M \mid v \in M]$$
$$= P[\exists\ w \in H(v), w \in M]$$
$$\leq \sum_{w \in H(v)} P[w \in M]$$
$$= \sum_{w \in H(v)} 1/(2d(w))$$

Compute union bound (sum) of all these events individually. And ignore neighbor condition.

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$
\begin{aligned}
P[v \notin MIS \mid v \in M] &= P[\exists\, w \in H(v), w \in M \mid v \in M] \\
&= P[\exists\, w \in H(v), w \in M] \\
&\leq \sum_{w \in H(v)} P[w \in M] \\
&= \sum_{w \in H(v)} 1/(2d(w)) \\
&\quad \sum_{w \in H(v)} 1/(2d(v)) \\
&\leq d(v)/(2d(v)) = 1/2
\end{aligned}
$$

Definition of marking algorithm

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$
\begin{aligned}
P[v \notin MIS \mid v \in M] &= P[\exists\, w \in H(v), w \in M \mid v \in M] \\
&= P[\exists\, w \in H(v), w \in M] \\
&\leq \sum_{w \in H(v)} P[w \in M] \\
&= \sum_{w \in H(v)} 1/(2d(w)) \\
&\leq \sum_{w \in H(v)} 1/(2d(v)) \\
&\leq d(v)/(2d(v)) = 1/2
\end{aligned}
$$

Definition of algorithm: degree of w is at least as large as of v.

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$P[v \notin MIS \mid v \in M] = P[\exists\ w \in H(v), w \in M \mid v \in M]$$

$$= P[\exists\ w \in H(v), w \in M]$$

$$\in M]$$

$$(w))$$

$$\leq \sum_{w \in H(v)} 1/(2d(v))$$

$$\leq d(v)/(2d(v)) = 1/2$$

v has at most d(v) neighbors with higher or same degree.

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$
\begin{aligned}
P[v \notin MIS \mid v \in M] &= P[\exists\ w \in H(v), w \in M \mid v \in M] \\
&= P[\exists\ w \in H(v), w \in M] \\
&\leq \sum_{w \in H(v)} P[w \in M] \\
&= \sum_{w \in H(v)} 1/(2d(w)) \\
&\leq \sum_{w \in H(v)} 1/(2d(v)) \\
&\leq d(v)/(2d(v)) = 1/2
\end{aligned}
$$

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$, and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$
\begin{aligned}
P[v \notin MIS \mid v \in M] &= P[\exists\, w \in H(v), w \in M \mid v \in M] \\
&= P[\exists\, w \in H(v), w \in M] \\
&\leq \sum_{w \in H(v)} P[w \in M] \\
&= \sum_{w \in H(v)} 1/(2d(w)) \\
&\leq \sum_{w \in H(v)} 1/(2d(v)) \\
&\leq d(v)/(2d(v)) = 1/2
\end{aligned}
$$

Conclusion: a marked node is likely to join MIS!

Joining MIS

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

To prove this lemma, let **M** be the set of marked nodes in Step 1 (prob. $1/(2d(v))$), and let **H(v)** be the set of neighbors of v with higher degree or same degree and higher identifier.

$$
\begin{aligned}
P[v \notin MIS \mid v \in M] &= P[\exists\ w \in H(v), w \in M \mid v \in M] \\
&= P[\exists\ w \in H(v), w \in M] \\
&\leq \sum_{w \in H(v)} P[w \in M] \\
&= \sum_{w \in H(v)} 1/(2d(w)) \\
&\leq \sum_{w \in H(v)} 1/(2d(v)) \\
&\leq d(v)/(2d(v)) = 1/2
\end{aligned}
$$

Conclusion: a marked node is likely to join MIS!

$$P[v \in MIS] = P[v \in MIS \mid v \in M]\, P[v \in M] \geq 1/2 \cdot 1/(2d(v)) \quad \blacksquare$$

Joining MIS

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

A small probability still (e.g., 1/n).
We need some more definitions!

Joining MIS

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

A small probability still (e.g., 1/n).
We need some more definitions!

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

Joining MIS

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

A small probability still (e.g., 1/n).
We need some more definitions!

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

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

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

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

Good Nodes

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

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

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

Good Nodes

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

Good: With high constant probability! Remains to show that there are many good nodes.

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

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

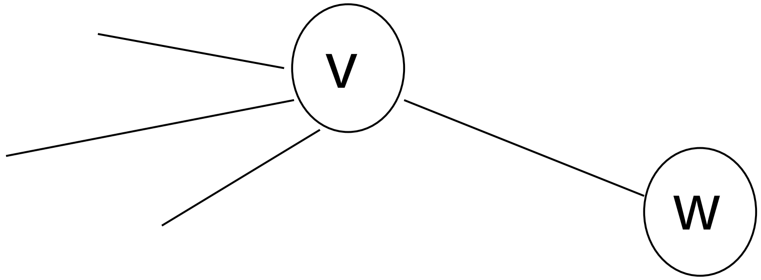
Good: With high constant probability! Remains to show that there are many good nodes.

Good Nodes

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

Case 1: If v has a neighbor w with $d(w) \leq 2$

This case is easy! Our „Joining MIS Lemma“ implies that the probability that this node is removed is at least 1/8: neighbor w joins with probability 1/8.



Recall:

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

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

Otherwise bad.

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

Good Nodes

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

Good: With high constant probability! Remains to show that there are many good nodes.

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

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

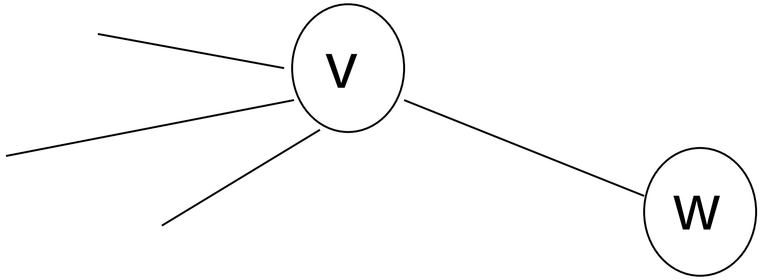
Good: With high constant probability! Remains to show that there are many good nodes.

Good Nodes

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

Case 2: If v has no neighbor w with $d(w) \leq 2$

This case is easy! Our „Joining MIS Lemma“ implies that the probability that this node is removed is at least 1/8: neighbor w joins with probability 1/8.



Recall:

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

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

Otherwise bad.

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

Good: With high constant probability! Remains to show that there are many good nodes.

Good Nodes

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

Case 2: If v has only neighbors w with d(w) > 2

Therefore: for any neighbor w of good node v, we have $1/(2d(w)) \leq 1/6$.
Therefore: fine granularity of summands:

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

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

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

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

Good: With high constant probability! Remains to show that there are many good nodes.

Good Nodes

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

Case 2: If v has only neighbors w with d(w) > 2

Therefore: for any neighbor w of good node v, we have $1/(2d(w)) \leq 1/6$.

Therefore: fine granularity of summands:

By definition of good node.

…e must be a s

We can add neighbors at a granularity of 1/6, see above.

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

Good Nodes

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

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

$$P[R] \geq P[\exists\ u \in S, u \in MIS]$$

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

Good Nodes

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

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

$$P[R] \geq P[\exists\ u \in S,\ u \in MIS]$$

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

Lower bound: *v* removed only if neighbor from this special subset *S* joins

Good Nodes

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

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

$$P[R] \geq P[\exists\ u \in S, u \in MIS]$$

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

By truncating the inclusion-exclusion principle...: Probability that there is one is sum of probability
for all individual ones minus probability that two enter MIS, plus...

Good Nodes

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

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

$P[R] \geq P[\exists\ u \in S, u \in \text{MIS}]$

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

$$
\begin{aligned}
P[R] &\geq \sum_{u \in S} P[u \in \text{MIS}] - \sum_{u,w \in S; u \neq w} P[u \in M \textbf{ and } w \in M] \\
&\geq \sum_{u \in S} P[u \in \text{MIS}] - \sum_{u \in S} \sum_{w \in S} P[u \in M] \cdot P[w \in M] \\
&\geq \sum_{u \in S} \frac{1}{4d(u)} - \sum_{u \in S} \sum_{w \in S} \frac{1}{2d(u)} \frac{1}{2d(w)} \\
&\geq \sum_{u \in S} \frac{1}{2d(u)} \left(\frac{1}{2} - \sum_{w \in S} \frac{1}{2d(w)} \right) \geq \frac{1}{6} \left(\frac{1}{2} - \frac{1}{3} \right) = \frac{1}{36}.
\end{aligned}
$$

■

Good Nodes

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

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

$P[R] \geq P[\exists\ u \in S, u \in \mathrm{MIS}]$

$\geq \sum_{u \in S} P[u \in \mathrm{MIS}] - \sum_{u,w \in S} P[u \in \mathrm{MIS} \text{ and } w \in \mathrm{MIS}]$

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

just derived

$$\sum_{u \in S} P[u \in \mathrm{MIS}] \quad \sum_{u \in S} \sum_{w \in S}$$

Using P[u ϵ M] ≥ P[u ϵ MIS]

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

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

∎

Good Nodes

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

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

$P[R] \geq P[\exists\ u \in S, u \in \text{MIS}]$

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

$$
\begin{aligned}
P[R] &\geq \sum_{u \in S} P[u \in \text{MIS}] - \sum_{u,w \in S;\, u \neq w} P[u \in M \ \textbf{and}\ w \in M] \\
&\geq \sum_{u \in S} P[u \in \text{MIS}] - \sum_{u \in S} \sum_{w \in S} P[u \in M] \cdot P[w \in M] \\
&\geq \sum_{u \in S} \frac{1}{4d(u)} - \sum \cdots \\
&\geq \sum_{u \in S} \frac{1}{2d(u)} \left(\frac{1}{2} - \sum_{w \in S} \frac{1}{2d(w)} \right) \geq \frac{1}{6} \left(\frac{1}{2} - \frac{1}{3} \right) = \frac{1}{36}.
\end{aligned}
$$

Independent but count nodes twice in sum

Good Nodes

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

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

P[R] ≥ P[∃ u ∈ S. u ∈ MIS]

≥ ∑

Joining MIS

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

$P[R]$

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

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

$$\sum_{u\in S} \ldots \sum_{w\in S} \ldots - \frac{1}{3}\Big) = \frac{1}{36}.$$

See Joining MIS lemma

See algorithm

Good Nodes

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

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

P[R] ≥ P[∃ u ϵ S, u ϵ MIS]

≥ $\sum_{u \in S}$ P[u ϵ MIS] - $\sum_{u,w \in S}$ P[u ϵ MIS and w ϵ MIS]

$$
\begin{aligned}
P[R] &\geq \sum_{u \in S} P[u \in \mathrm{MIS}] - \sum_{u,w \in S; u \neq w} P[u \in M \ \textbf{and}\ w \in M] \\
&\geq \sum_{u \in S} P[u \in \mathrm{MIS}] - \sum_{u \in S} \sum_{w \in S} P[u \in M] \cdot P[w \in M] \\
&\geq \sum_{u \in S} \frac{1}{4d(u)} - \sum_{u \in S} \sum_{w \in S} \frac{1}{2d(u)} \frac{1}{2d(w)} \\
&\geq \sum_{u \in S} \frac{1}{2d(u)} \left(\frac{1}{2} - \sum_{w \in S} \frac{1}{2d(w)} \right) \geq \frac{1}{6}\left(\frac{1}{2} - \frac{1}{3}\right) = \frac{1}{36}.
\end{aligned}
$$

Set S has at least cumulative prob. 1/6…

… and at most 1/3.

■

We just proved:

Good Nodes

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

$$p \geq 1/36.$$

We just proved:

Good Nodes

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

$$p \geq 1/36.$$

But how many good nodes are there?

We just proved:

Good Nodes

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

$p \geq 1/36$.

But how many good nodes are there?

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

A good node has many neighbors of low degree.

We just proved:

Good Nodes

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

$$p \geq 1/36.$$

But how many good nodes are there?

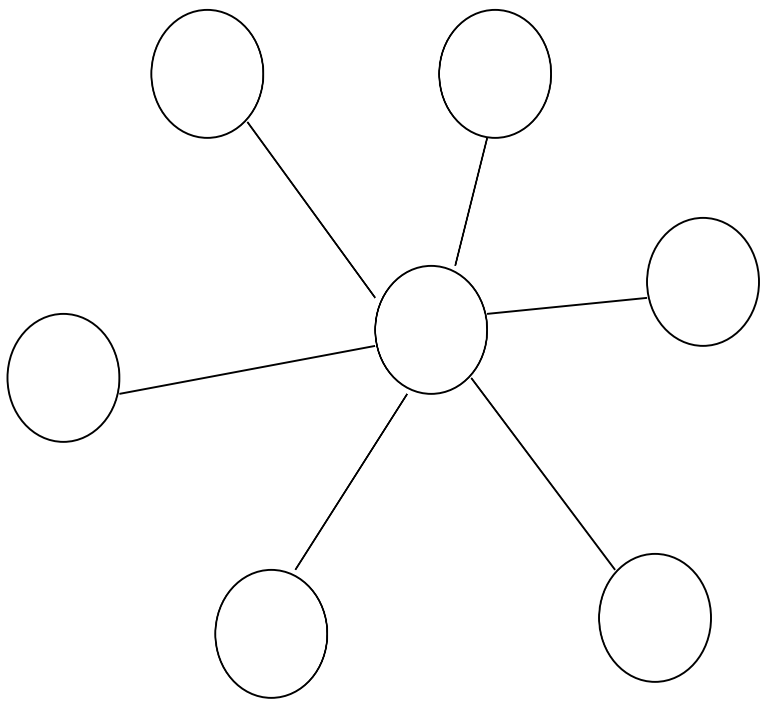
Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.

A good node has many neighbors of low degree.



We just proved:

Good Nodes

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

$$p \geq 1/36.$$

But how many good nodes are there?

Good&Bad Nodes

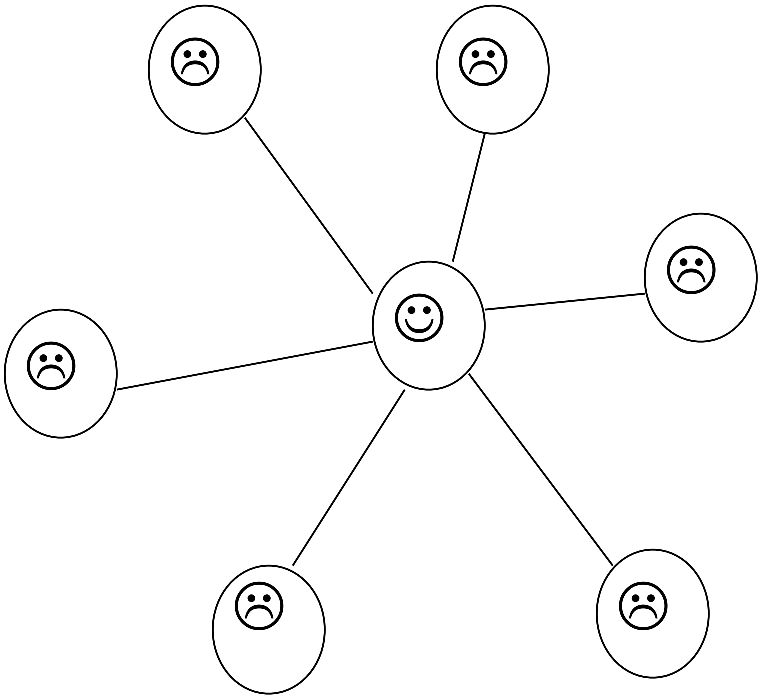
A node v is called *good* if

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

Otherwise bad.

A good node has many neighbors of low degree.

Only one good node!



We just proved:

Good Nodes

A good node v

But many edges have one good node as endpoint! Many «good edges»!

But how many good nodes are there?

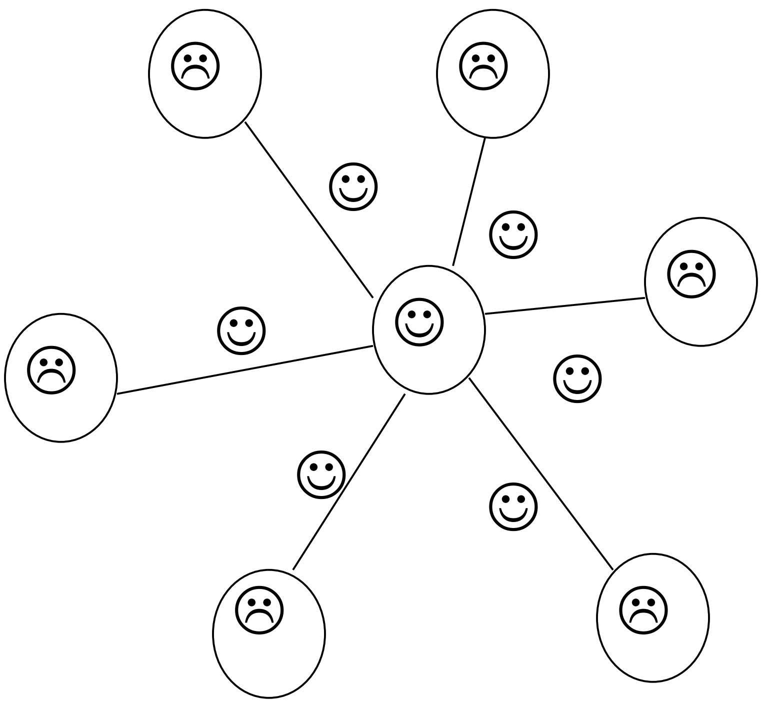
Only one good node!

Good&Bad Nodes

A node v is called *good* if

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

Otherwise bad.



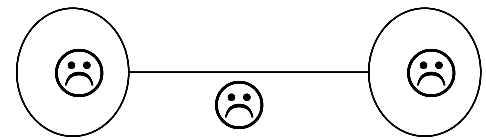
A good node has many neighbors of low degree.

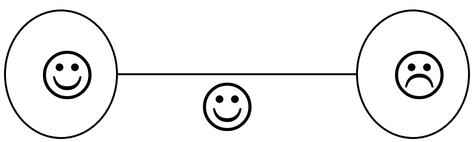
Analysis

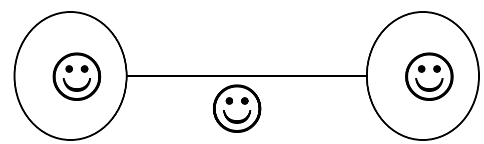
Good&Bad Edges

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.



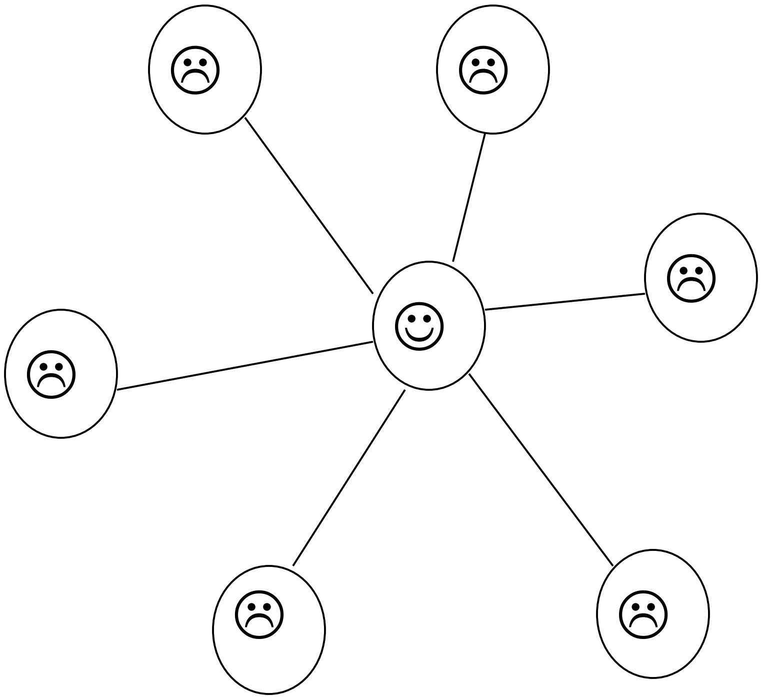




Good Edges

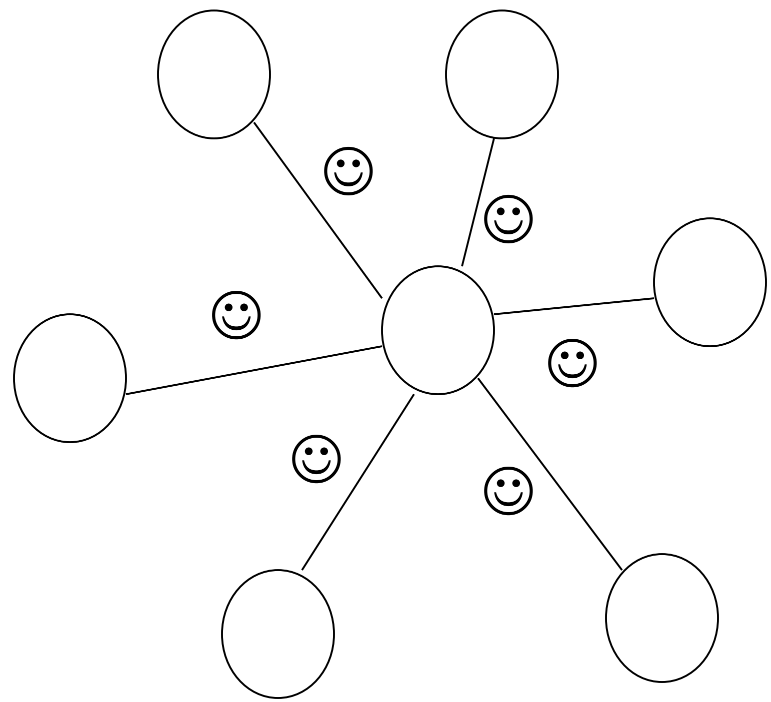
At least half of all edges are good, at any time.

Analysis



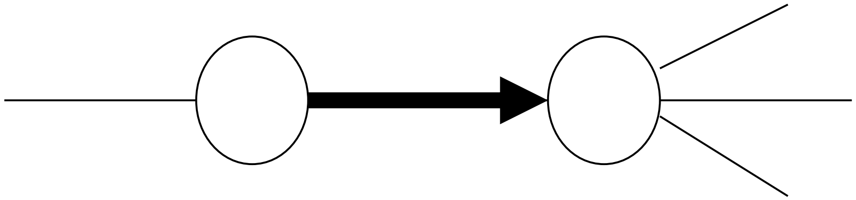
Not many good nodes...

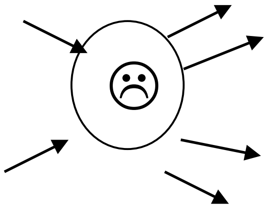
... but many good edges!



Analysis

Idea: Artificially direct each edge towards higher degree node
(if both nodes have same degree, point it to one with higher ID).



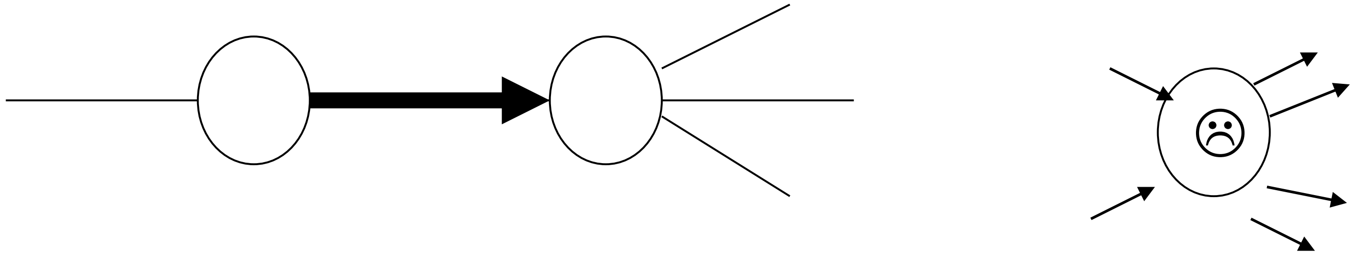


Helper Lemma

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

Analysis

Idea: Artificially direct each edge towards higher degree node
(if both nodes have same degree, point it to one with higher ID).



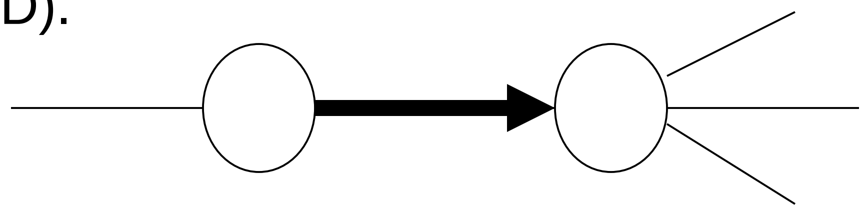
Helper Lemma

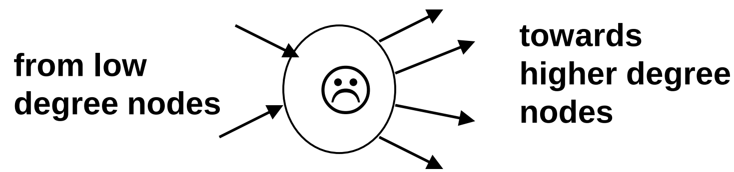
A bad node v has out-degree at least twice its in-degree.

That would be great:
Since sum of incoming edges = sum of outgoing edges,
if the number of edges into bad nodes can be at most half
the number of all edges, at least half of all edges are directed
into good nodes! And they are good! ☺
So at least half of all edges are good.

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“).

Idea: Otherwise it must have many low-degree neighbors and be good!
Assume the opposite: at least d(v)/3 neighbors (let‘s call them $S \subseteq N(v)$) have degree at most d(v) (otherwise v would point to them). But then

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

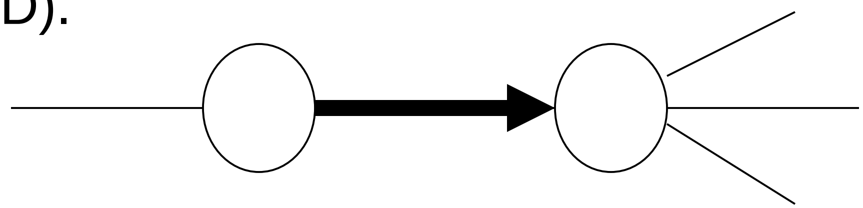
only subset... Def. of S Assumption

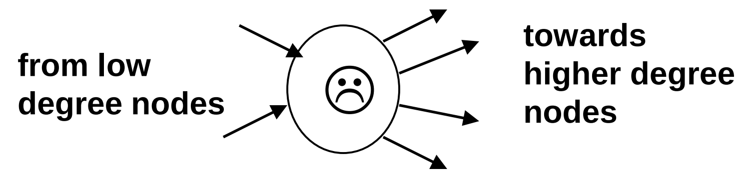
Contradiction: v would be good!

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.

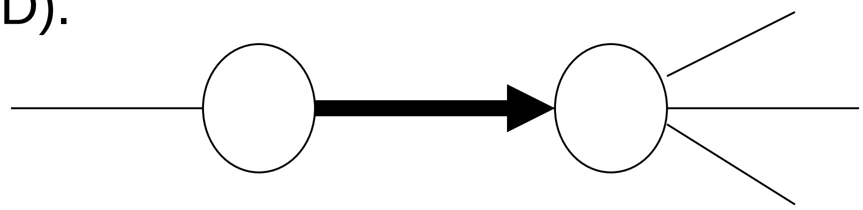
Proof („Helper Lemma“).

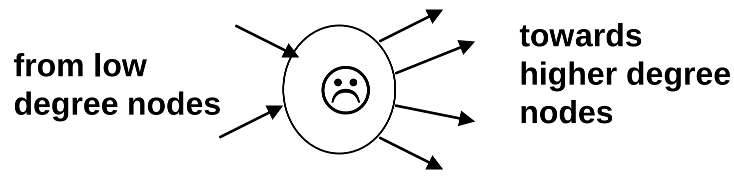
Idea: Otherwise it must have many low-degree neighbors and be good!
Assume the opposite: at least $d(v)/3$ neighbors (let‘s call them $S \subseteq N(v)$) have degree at most $d(v)$ (otherwise v would point to them). But then

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

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“).

Idea: Otherwise it must have many low-degree neighbors and be good!

Assume the opposite: at least d(v)/3 nei

have d(v) (otherwise v w Bu

Only subset

Def. of S

our assumption

Contradiction to good node!

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

Fast MIS (1986)

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

Fast MIS (1986)

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

At least half of all the edges are good, and thus have at least one good incident node which will be deleted with constant probability and so will the edge! A constant fraction of edges will be deleted in each phase. (Note that O(log m)=O(log n).)

Back to the future: Fast MIS from 2009…!

Even simpler algorithm!

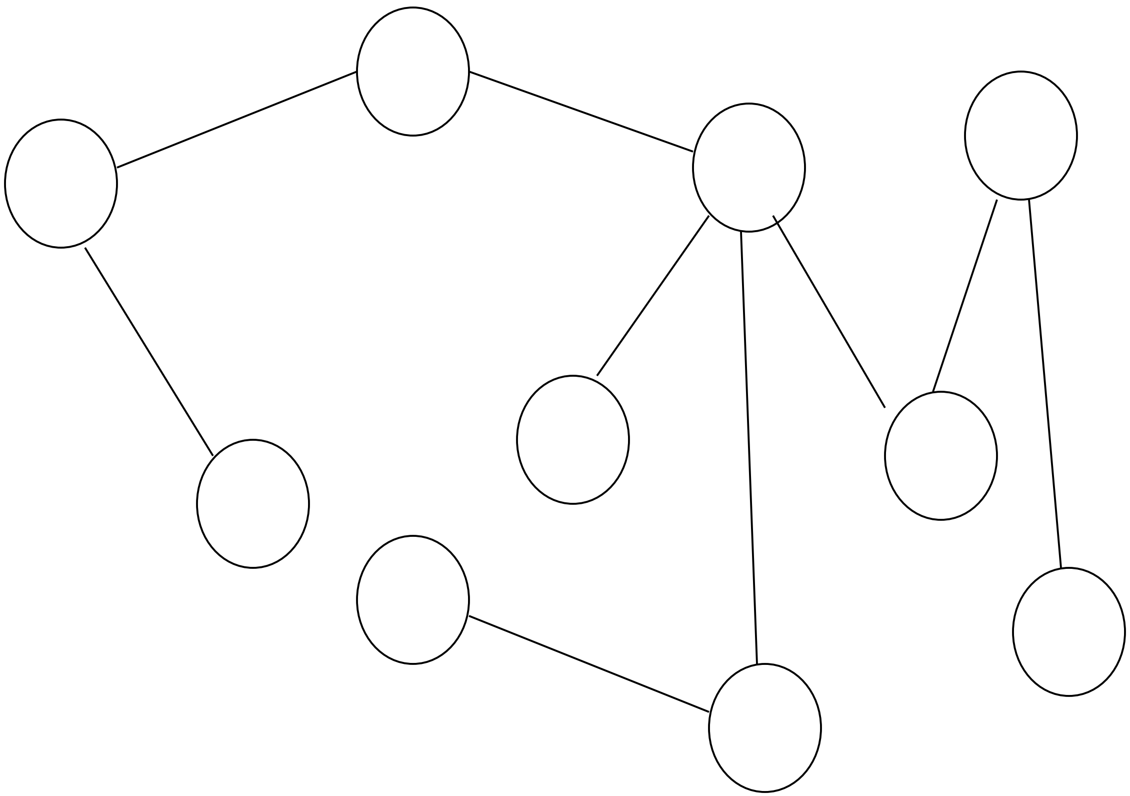
Fast MIS (2009)

Proceed in rounds consisting of phases!

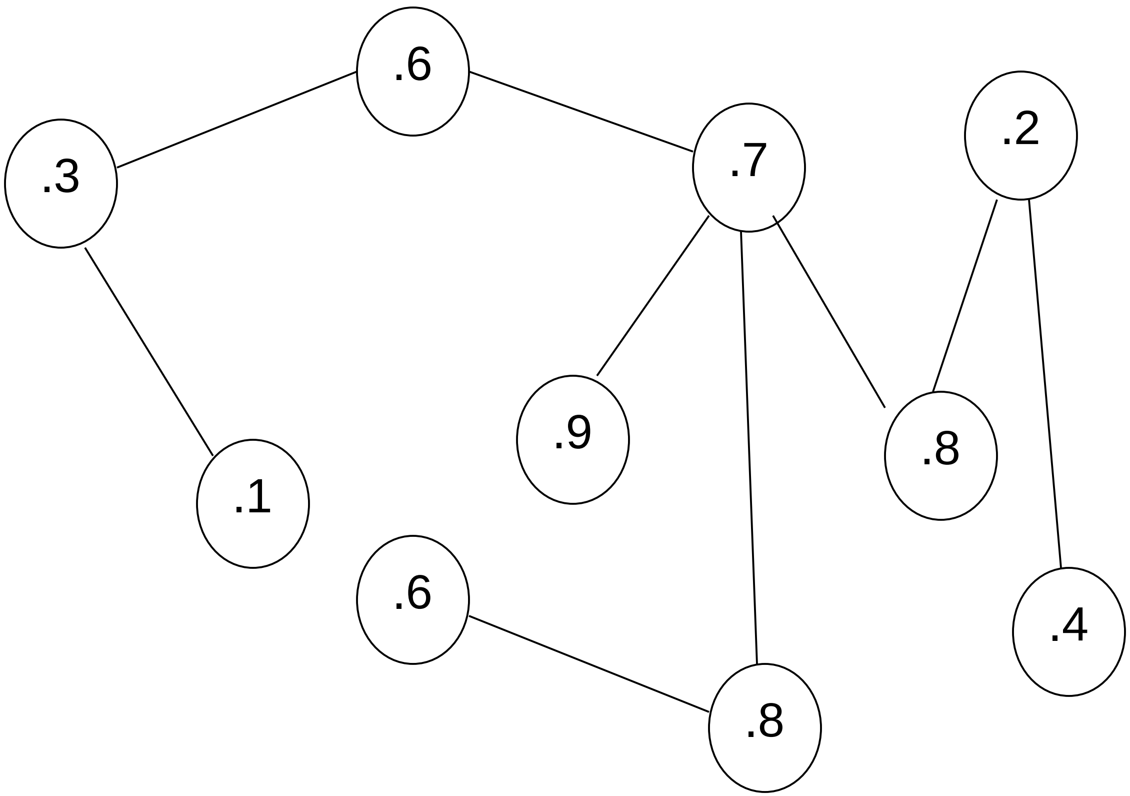
In a phase:

1. each node chooses a random value **r(v)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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...

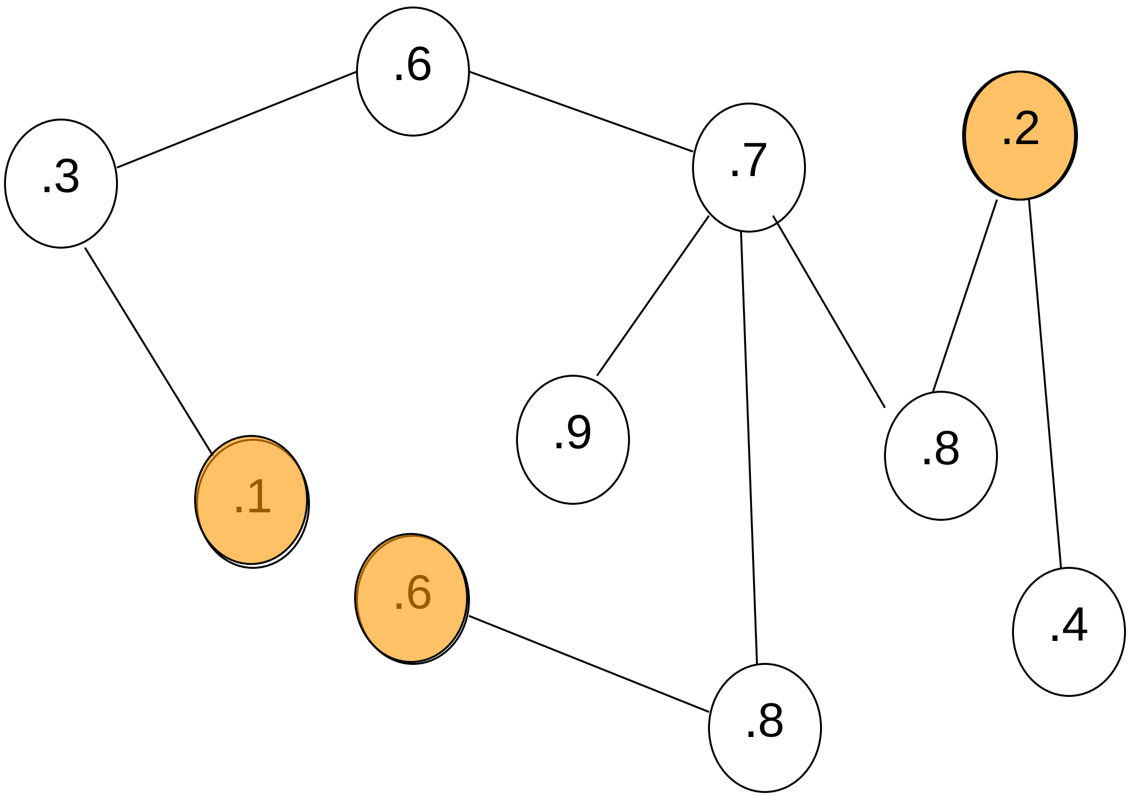


Fast MIS from 2009...



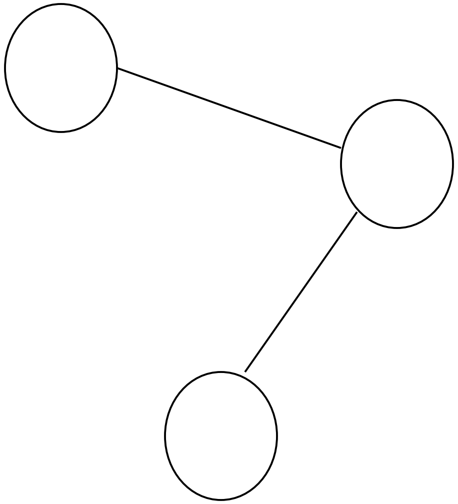
Choose random values!

Fast MIS from 2009...



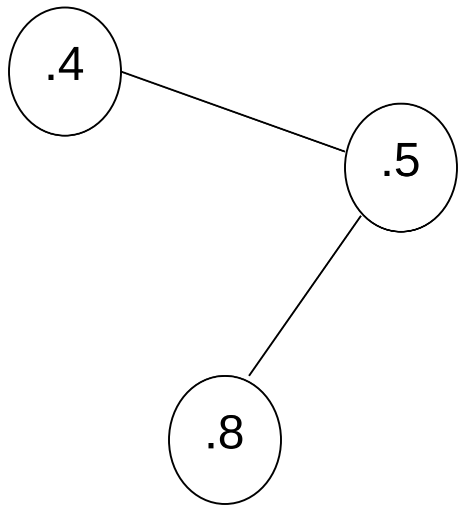
Min in neighborhood: *add to IS!*

Fast MIS from 2009...



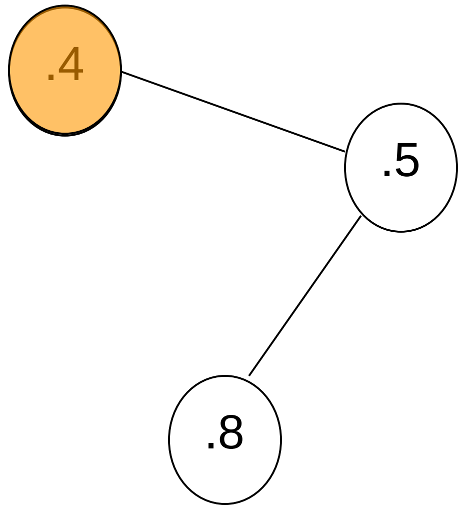
Remove neighborhoods...

Fast MIS from 2009...



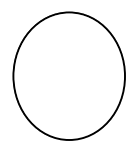
Choose random values!

Fast MIS from 2009...



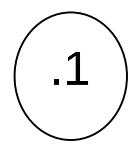
Min in neighborhood: *add to IS!*

Fast MIS from 2009...



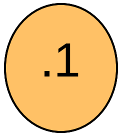
Remove neighborhoods...

Fast MIS from 2009...



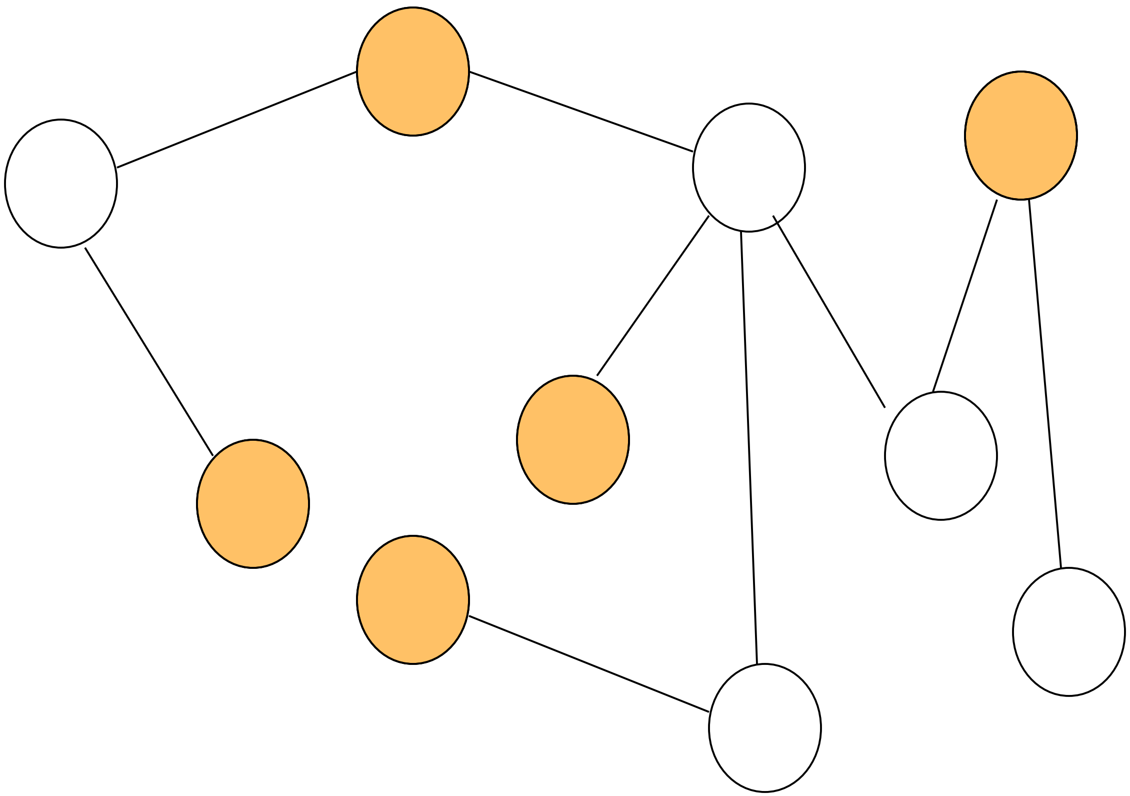
Choose random values!

Fast MIS from 2009...



Min in neighborhood: *add to 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)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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 smallest random value will always join the IS, 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)** ϵ [0,1] and sends it to its neighbors.
2. If **r(v) < r(w)** for all neighbors w ϵ 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*

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

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.

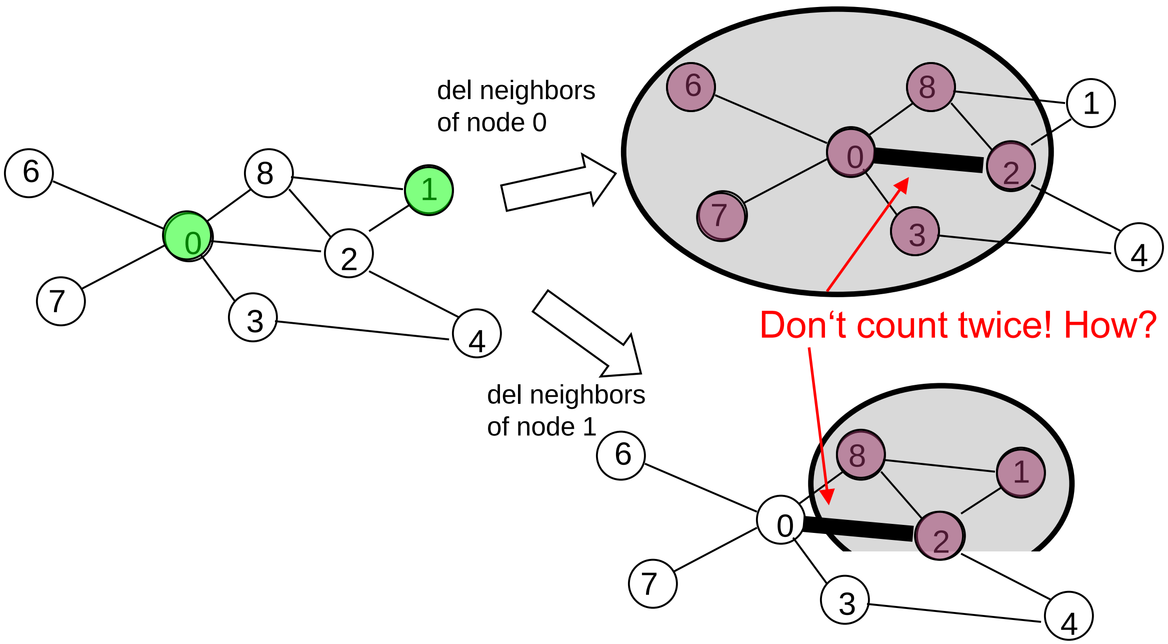
Analysis

Probability of a node v to enter MIS?

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

... also v‘s neighbor’s 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)

Event (v=>w)

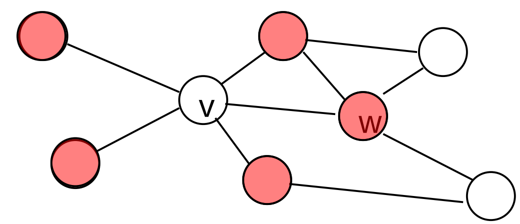
(v=>w): per edge event: node v joins MIS and is even smaller than w’s neighbors.

Edge Removal

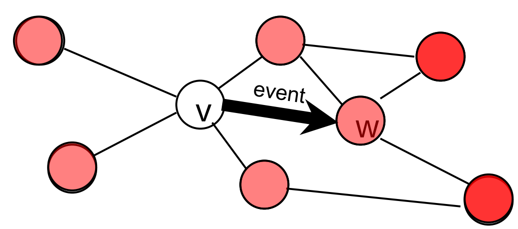
In expectation, we remove at least half of all the edges in any phase.

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)**.

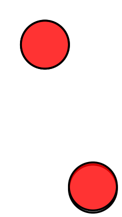


What is the probability of this event (that v is minimum also for neighbors of the given neighbor)?



P [(v => w)] ≥ 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")?

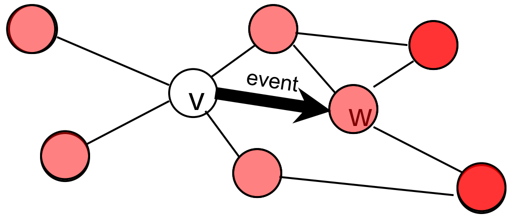
How many edges are removed?

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

Let X denote the sum of all these random variables.

So:

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



So all edges gone in one phase?!
We still overcount!

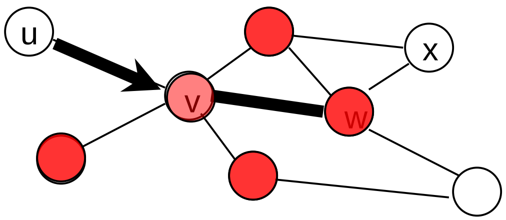
Edge Removal: Analysis (3)

Edge Removal

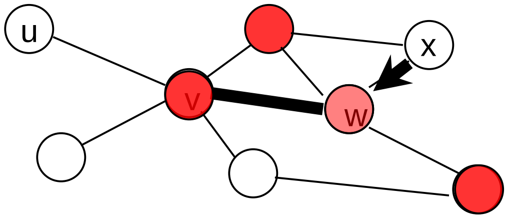
In expectation, we remove at least half of all the edges in any phase.

Proof („Edge Removal“)?

We still overcount:
Edge {v,w} may be counted twice:
for event (u=>v) and event (x=>w).



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



Event (u=>v) means r(u)<r(w) for all w 2 N(v); another (u‘=>v) would imply that r(u‘)>r(u) 2 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!