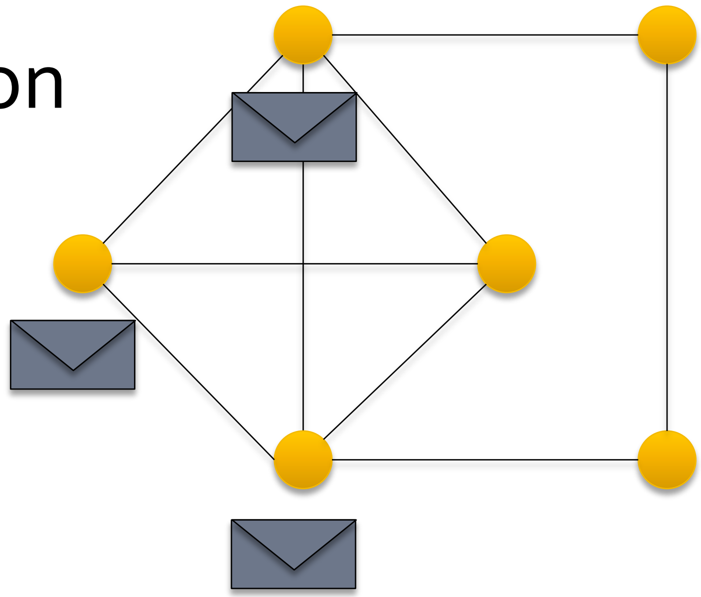
Distributed Algorithms as Combinatorial Structures

Keren Censor-Hillel, Technion
ADGA 2014

Good Morning!

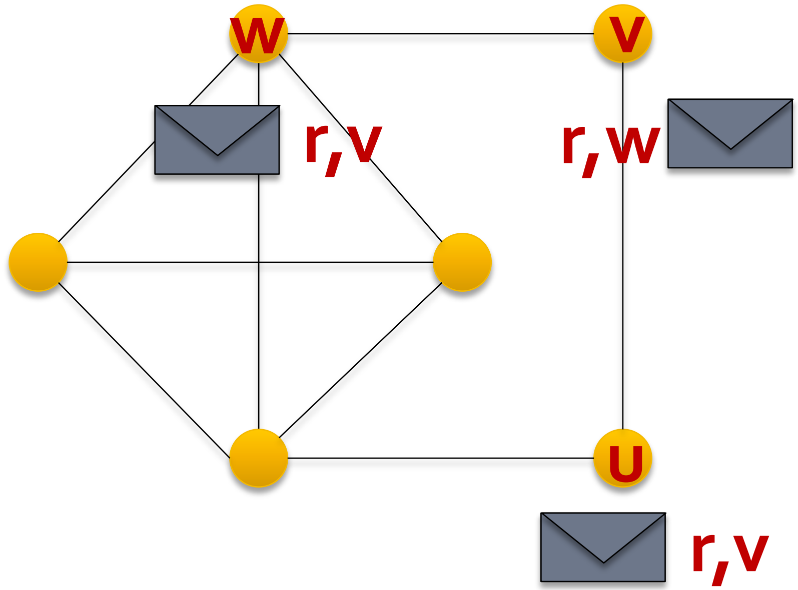
- Today I want to inspect distributed algorithms
- Not their code
- Their execution



Synchronizers

Design algorithms for **synchronous** systems but run them in **asynchronous** systems
[Awerbuch 1985]

- Send **(r+1)**-message after receiving all **r**-messages



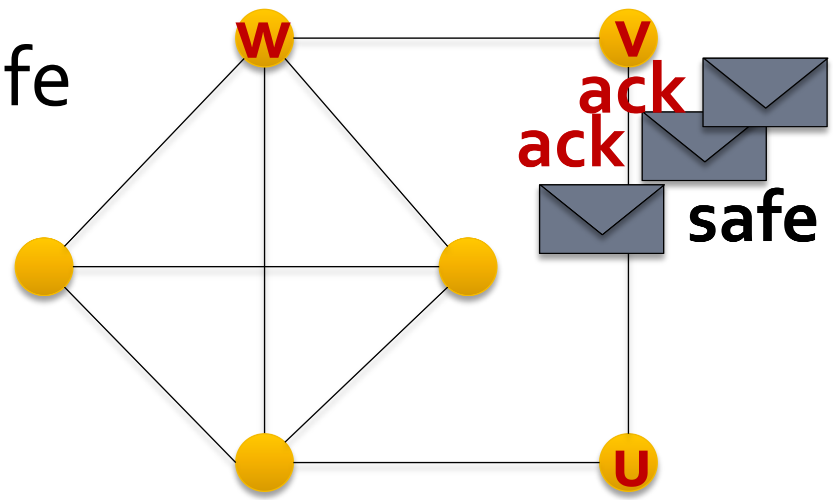
v will wait forever if
u never sends an **r**-message

Sending an empty message increases message complexity

Synchronizers

Send ACKs:

- **v** informs about receiving all ACKs (**v** is **safe**)
- **v** sends **(r+1)**-message
  when all neighbors are safe



Still need a message
from every neighbor

Message overhead is **M=O(|E|)**
Time overhead is **T=O(1)**

Spanners

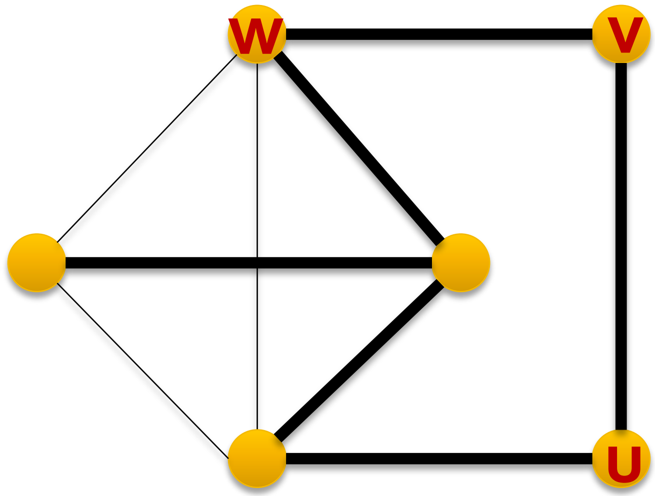
In a **k-spanner** **S** of G:

$d_S(u,v) \le k \cdot d_G(u,v)$

k is the **stretch**

[Peleg, Ullman 1989]

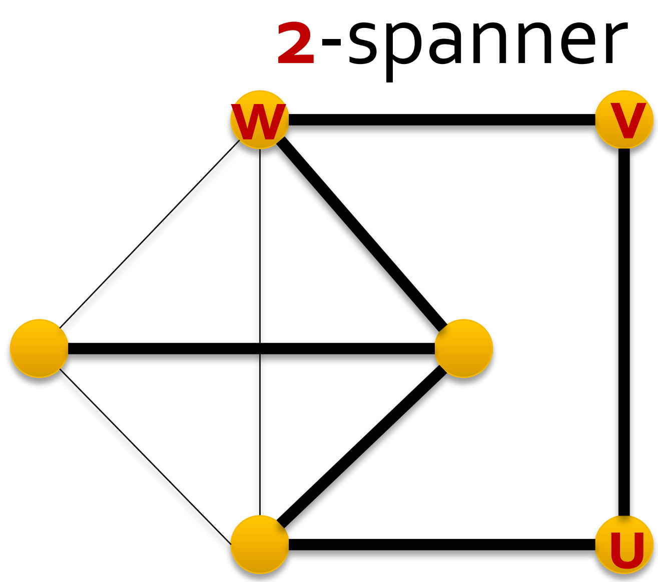
2-spanner



Synchronizers with Spanners

S is a **k**-spanner with **m** edges

- Repeat **k** iterations:
  - Send **safe** messages in the spanner
  - Wait for **safe** messages in the spanner



Iteration **t**: nodes within spanner distance **t** are safe
All neighbors are within distance **k** in the spanner

Synchronizers with Spanners

k-spanner with **m** edges → **synchronizer** with **M=O(km)**, **T=O(k)**

[Peleg, Ullman 1989]

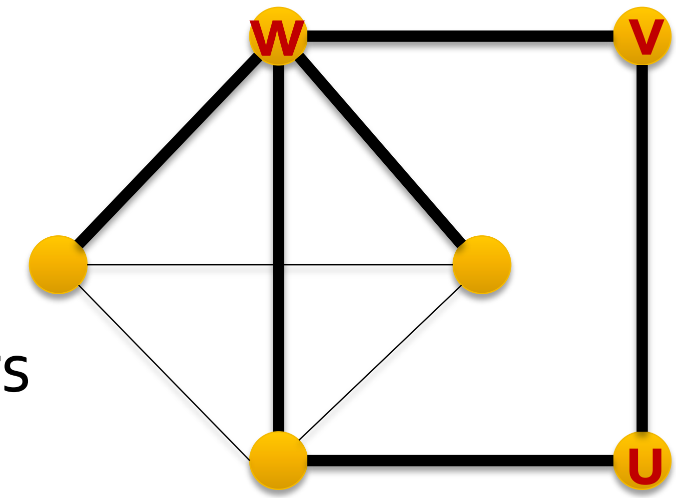
Synchronizers with Spanners

Sync is a synchronizer

Mark all edges used

Information has to pass
between each pair of neighbors

Gives a spanner **S** with **m≤M** and **k≤T**



Synchronizers vs. Spanners

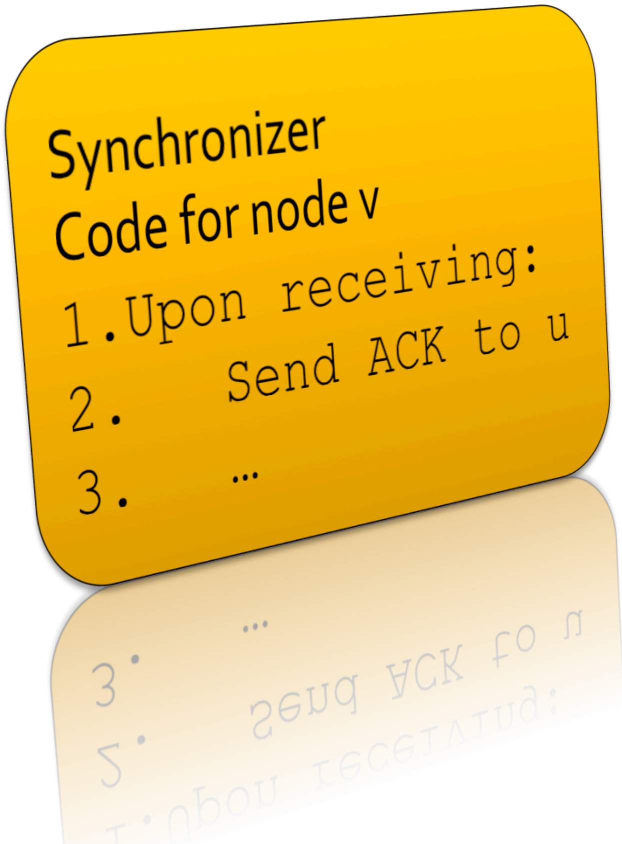
k-spanner with **m** edges →
synchronizer with **M=O(km)**, **T=O(k)**

synchronizer with parameters **M** and **T** →
spanner with **m≤M** and **k≤T**

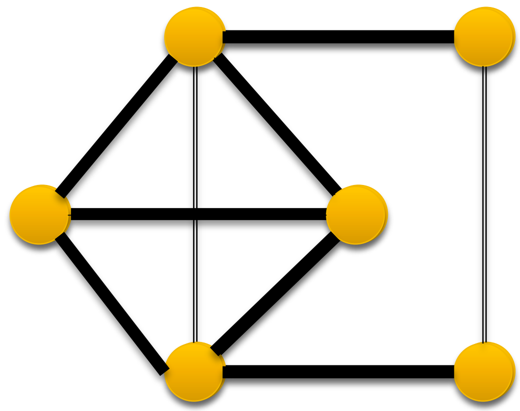
[Peleg, Ullman 1989]

Synchronizers vs. Spanners

Synchronizers

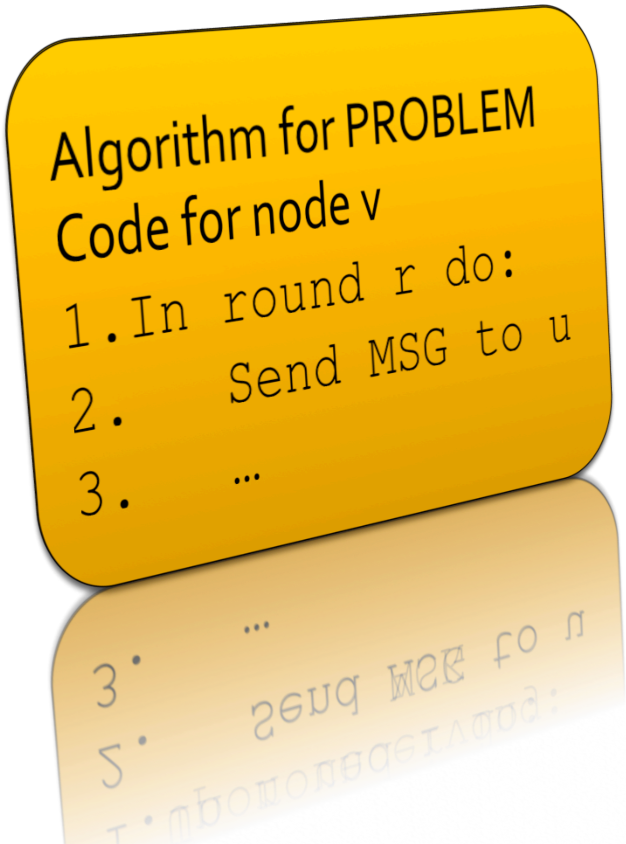


Spanners

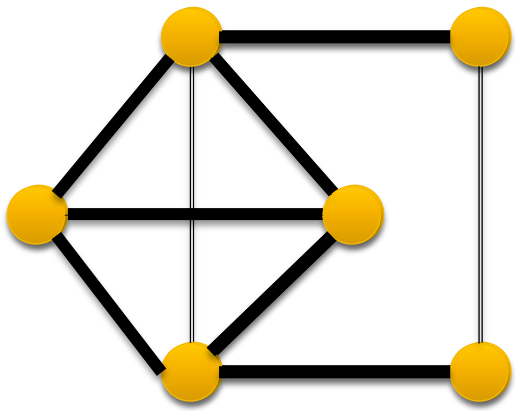


MIS (Most Important Slide)

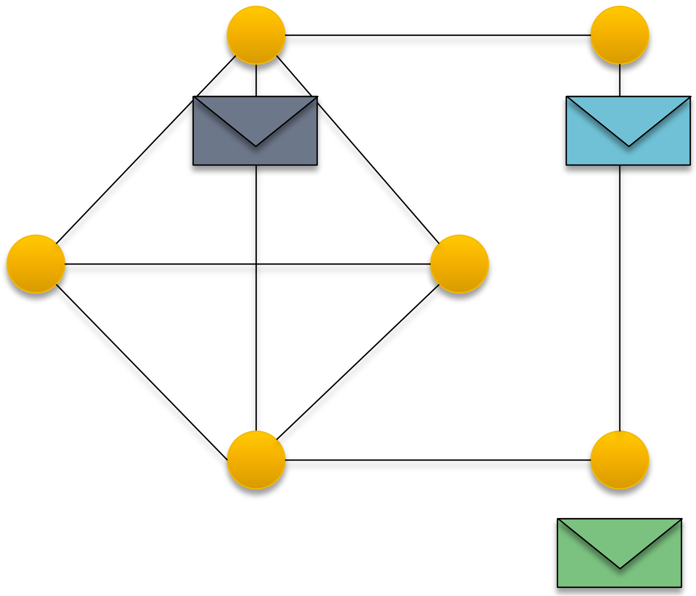
Distributed Algorithms



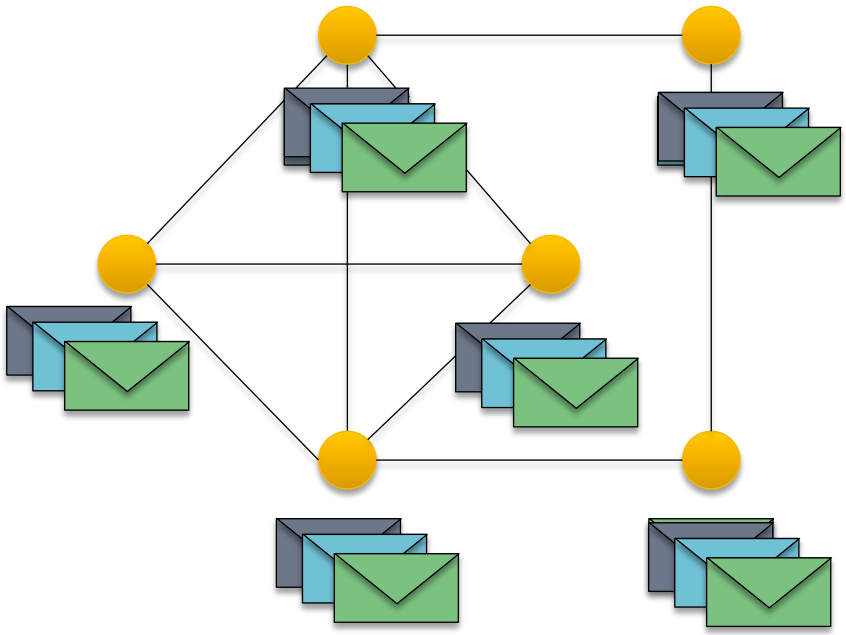
Graph Structures



Multi-Message Broadcast



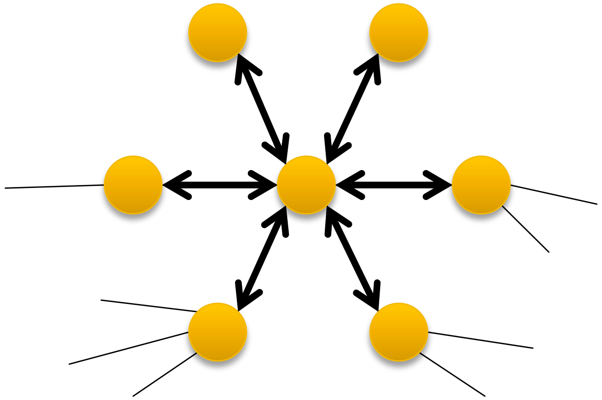
Multi-Message Broadcast



The GOSSIP model

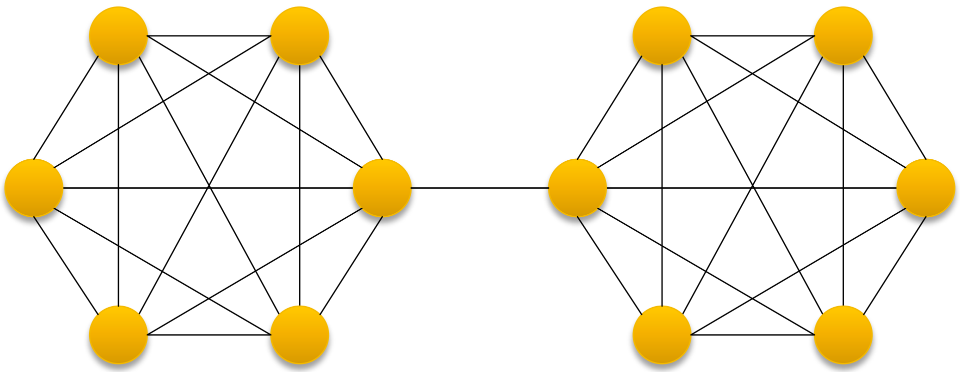
LOCAL round: contact all neighbors
[Linial 1987]

GOSSIP round: contact a single neighbor
[Demers et al. 1987]



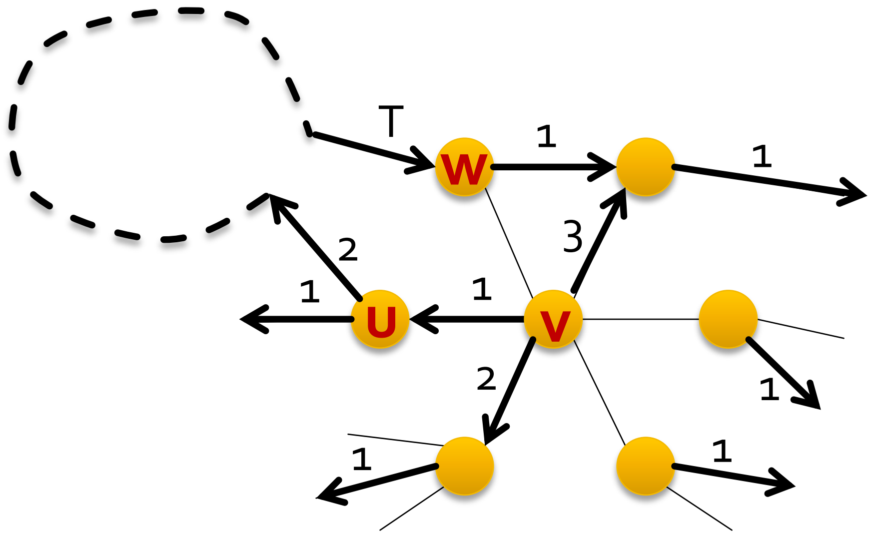
Multi-Message Broadcast in GOSSIP

- The challenge: bottlenecks
  - Simple round-robin performs poorly
  - Random choices depend on conductance: **$\Theta(\log n/\varphi)$**
    [Giakkoupis 2011,
    Chierichetti, Lattanzi, Panconesi 2010]



Neighbor Exchange

- **T** rounds for neighbor exchange → **DT** rounds for multi-message broadcast
- **T**-spanner with **O(nT)** edges and max degree **T** in **T** rounds



Multi-Message Broadcast vs. Spanners

Neighbor exchange in **T** rounds → **spanner** with **m=nT** , **k=T** and max degree **T**

- Neighbor exchange in **$O(\log^3 n)$** rounds **[Censor-Hillel, Hauepler, Kelner, Maymounkov 2012]**
- Neighbor exchange improved to **$O(\log^2 n)$** **[Hauepler 2013]**

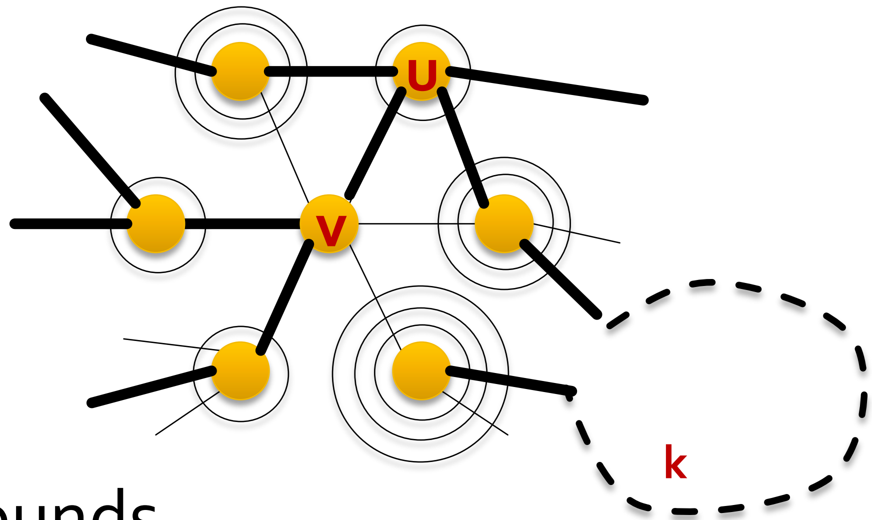
Multi-Message Broadcast with Spanners

S is a **k**-spanner with max degree **d**

- Repeat **k** iterations:
  - Pick new neighbor in **S** (round-robin)

Iteration **t**: reach nodes within spanner distance **t**

All neighbors are within distance **k** in the spanner



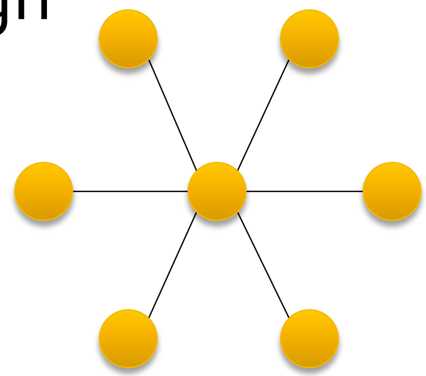
- Neighbor exchange in **dk** rounds
- mm-broadcast in **dkD** rounds

Wish Upon a Star

Not every graph has a small degree spanner

Instead, use a spanner that is “sparse enough”

- Takes **O(1)** rounds to reach the next hop



Hereditary density **δ**:

The maximal density of induced subgraphs

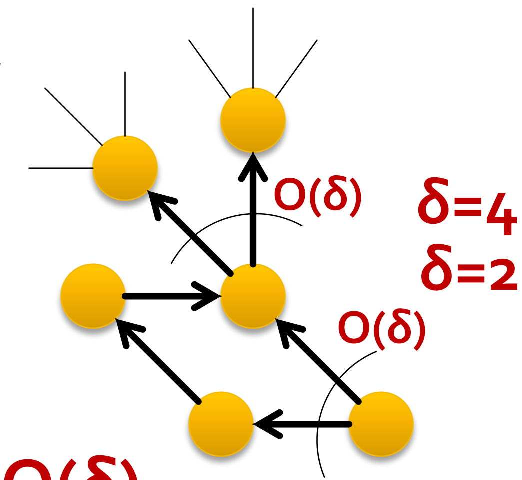
- Any induced subgraph has at most **δ|S|** edges

Can direct edges of **G** with hereditary density **δ** such that max out-degree is **δ**, in **O(δlog n)** rounds

Directing edges

In GOSSIP model:

- Guess **δ**, if have fewer than **δ** remaining edges then contact these neighbors, else double **δ**
  - Terminates when the guess is **O(δ)**



- All edges directed, out-degree at most **O(δ)**
- A constant fraction of nodes in each iteration, otherwise density would be greater than **δ**
- Takes **O(δlog(n))** rounds to direct all edges

Multi-Message Broadcast vs. Spanners

(α,β)-spanner S with hereditary density **δ** → **mm-broadcast** in **$T = \text{polylog} n T_S + \delta \log n + \delta(\alpha D + \beta)$**

[Censor-Hillel, Hauepler, Kelner, Maymounkov 2012]
mm-broadcast in **O(D+polylog n)** rounds

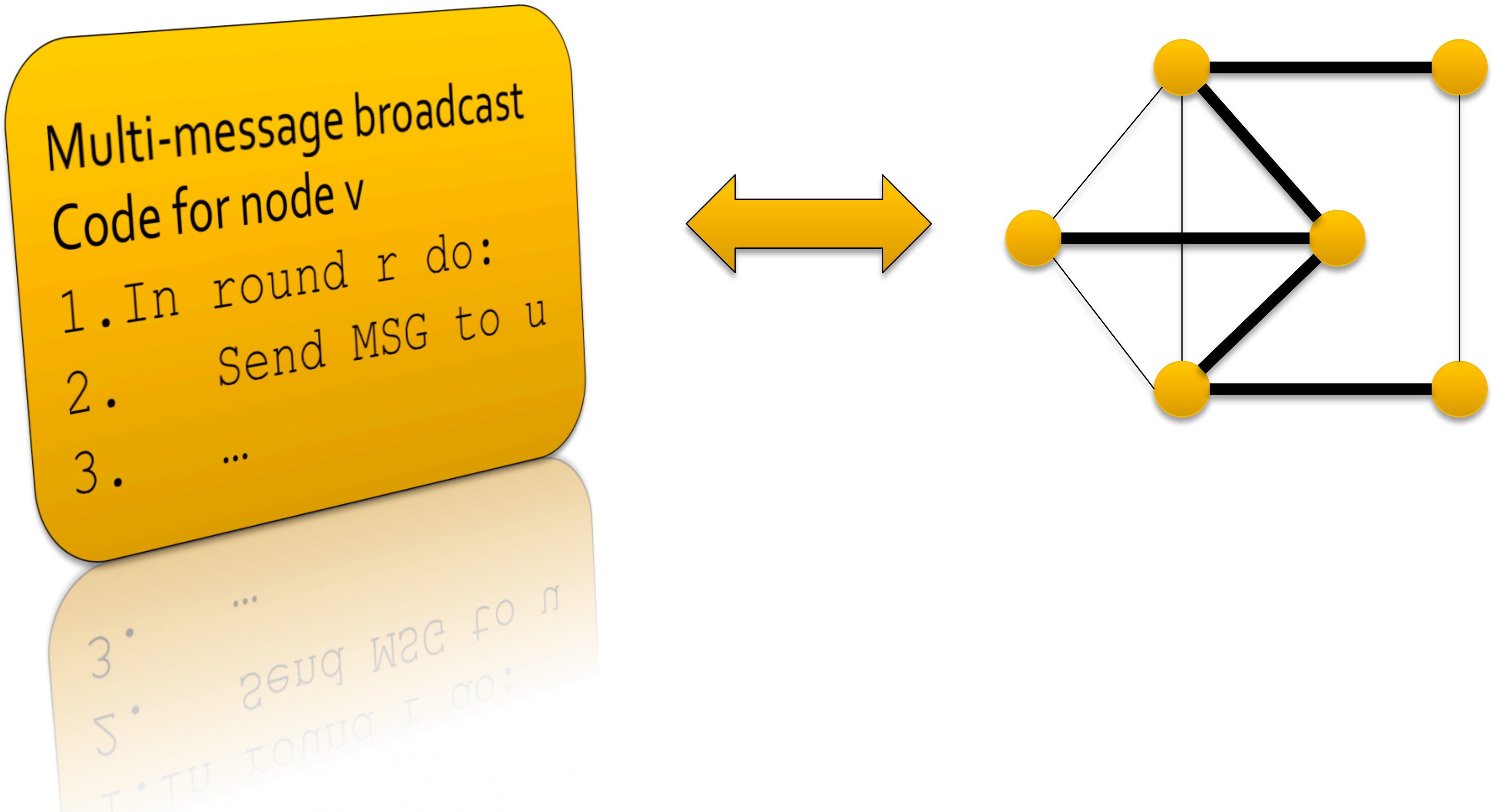
- By simulating a **(O(1),polylog n)**-spanner with **δ=O(1)** construction in **O(polylogn)** rounds **[Pettie 2009]**

- Other **α,β** trade-offs by simulating other spanner constructions **[Dubhashi, Mei, Panconesi, Radhakrishnan, Srinivasan 2005] [Derbel, Gavoille, Peleg, Viennot 2008] [Pettie 2010]**

Multi-Message Broadcast vs. Spanners

Multi-message broadcast

Spanners



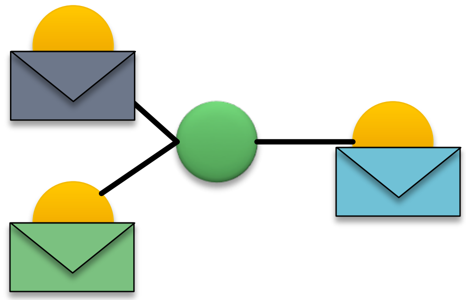
The Congestion Models

Two congestion models:

Edge-congested model

Message size **O(log n)**

Classic CONGEST model **[Peleg 2000]**

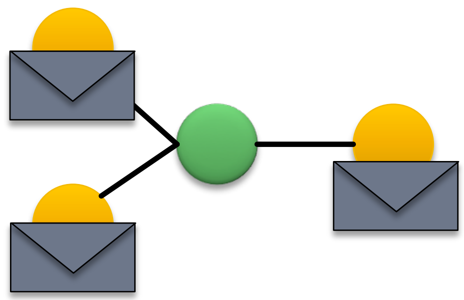


Node-congested model

Message size **O(log n)**

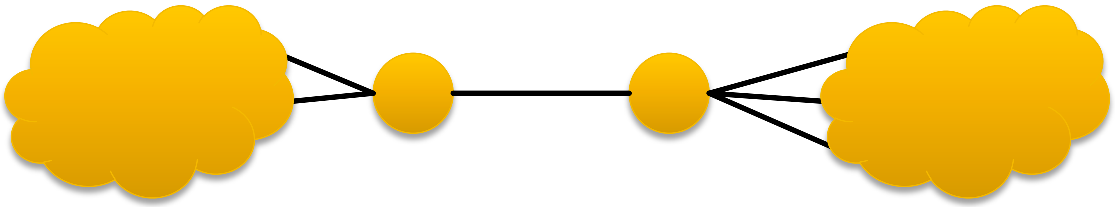
Communication by local broadcast

→ Send same message to all neighbors



Connectivity

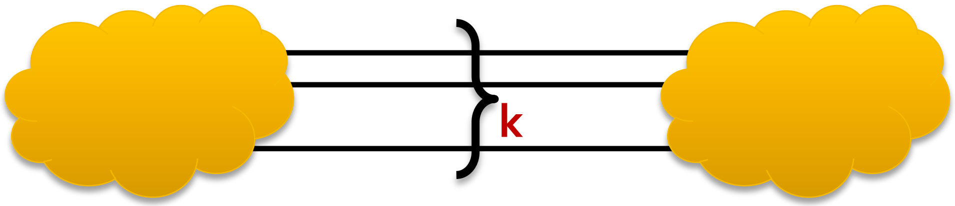
Toy example: bottleneck graph, throughput **≤ 1**



Time is **O(D+N)** for **N** messages **[Topkis 1985]**

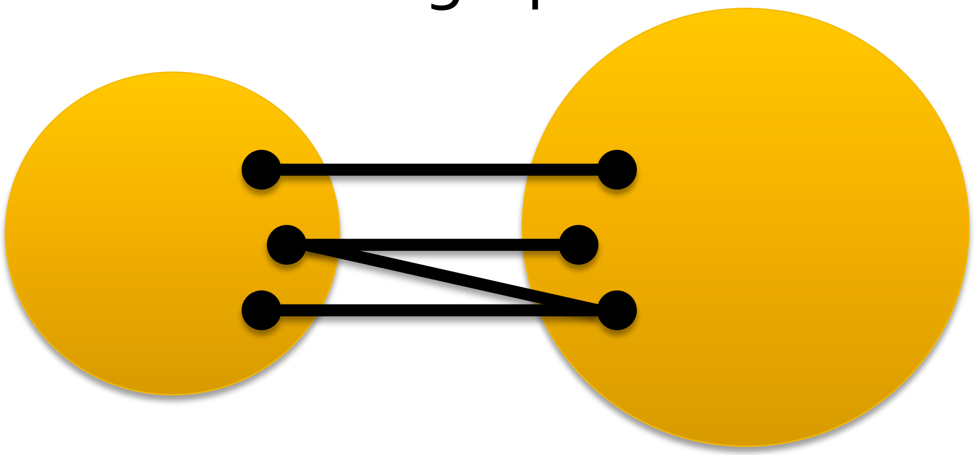
But what if connectivity is larger?

Can we hope for throughput of **k** and shorter broadcast time?



Graph Connectivity

λ-connectivity: removal of any **λ-1** edges (vertices) does not disconnect the graph

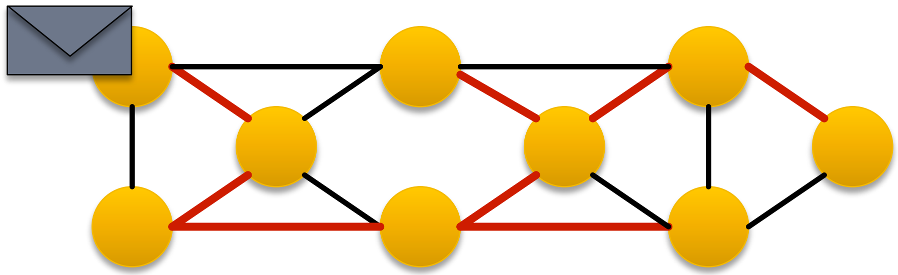


Menger's Theorem: **λ** edge-(vertex-) disjoint paths between every pair of nodes

CONGEST: Spanning Tree Packings

S is a spanning tree

- Send msg **M** on **S**



N messages?

Spanning tree packing: set of ~~edge-disjoint~~ spanning trees

Fractional packing: the **weight** on each edge is at most 1

The **size** of a packing is the total weight of all trees

Spanning tree packing of size **N** → throughput of **Ω(N)**

CONGEST: Spanning Tree Packings

Spanning tree packing of size **N** →
mm-broadcast with throughput of **Ω(N)**

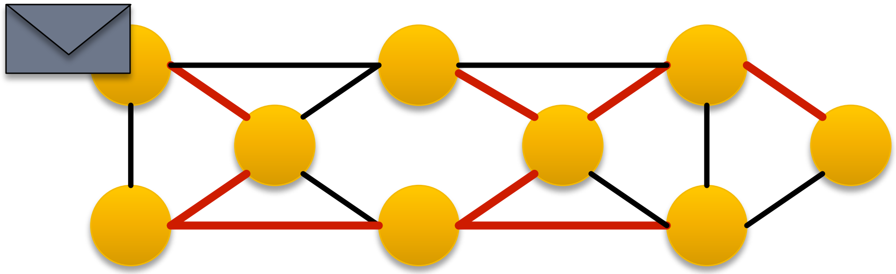
mm-broadcast with throughput of **Ω(N)** →
Spanning tree packing of size **N**

[Censor-Hillel, Ghaffari, Kuhn 2014]

CONGEST: Spanning Tree Packings

MMB-E is a mm-broadcast algorithm

Mark all edges over which msg **M** is sent



M reaches all nodes

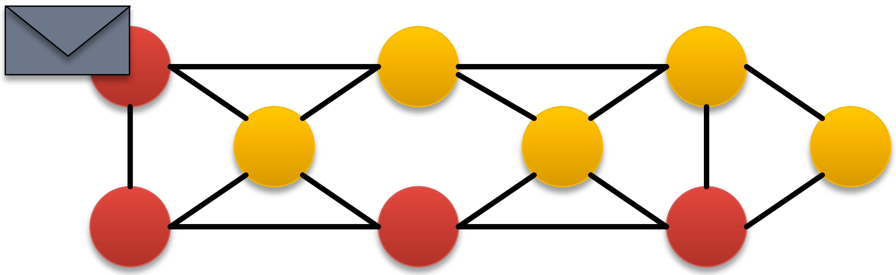
Gives a spanning tree **S** (removing cycles)

Repeat for all msgs: Gives a **spanning tree packing** of size **N**

V-CONGEST: Dominating Tree Packings

S is a dominating tree

- Send msg **M** on **S**



N messages?

Dominating tree packing: set of ~~vertex-disjoint~~ dominating trees

Fractional packing: the **weight** on each vertex is at most 1

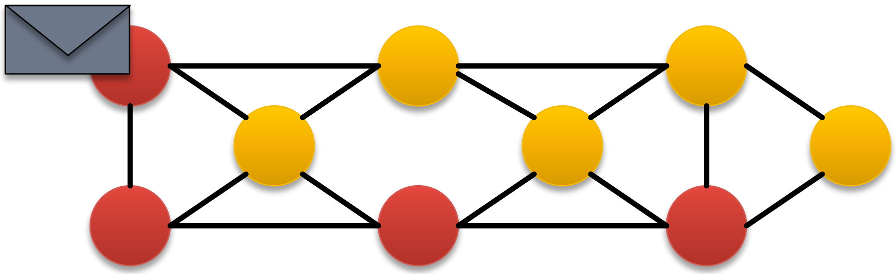
The **size** of a packing is the total weight of all trees

Dominating tree packing of size **N** → throughput of **Ω(N)**

V-CONGEST: Dominating Tree Packings

MMB-V is a mm-broadcast algorithm

Mark all vertices which send msg **M**



M reaches all nodes

Gives a dominating tree **S**

Repeat for all msgs: Gives a **dominating tree packing** of size **N**

V-CONGEST: Dominating Tree Packings

Dominating tree packing of size **N** →
mm-broadcast with throughput of **Ω(N)**

mm-broadcast with throughput of **Ω(N)** →
Dominating tree packing of size **N**

[Censor-Hillel, Ghaffari, Kuhn 2014]

Spanning Tree Packings

Every spanning tree packing of a **λ-edge connected** graph has size at most **λ** (every spanning tree needs to cross a min-cut)

Every **λ**-edge connected graph has a spanning tree packing of size $\lceil (\lambda - 1)/2 \rceil$ (tight)
[Tutte 1961, Nash-Williams 1961, Kundu 1974]

Centralized algorithms for decomposing unweighted graphs in $\tilde{O}\left(\min\left\{mn, m^2/\sqrt{n}\right\}\right)$ time **[Gabow and Westermann 1988]** and weighted graphs in $\tilde{O}(mn)$ time **[Barahona 1995]**

Distributed Spanning Tree Packings

Distributed decomposition of **λ**-edge connected graphs into fractionally edge-disjoint spanning trees with total weight $\lceil(\lambda-1)/2\rceil(1-\varepsilon)$ in $\tilde{O}\left(D+\sqrt{n\lambda}\right)$ rounds

A lower bound of $\Omega\left(D+\sqrt{n/\lambda}\right)$ rounds for such decompositions

[Censor-Hillel, Ghaffari, Kuhn 2014]

Distributed Dominating Tree Packings

Distributed decomposition of **k**-vertex connected graphs into fractionally vertex-disjoint dominating trees with total weight **Ω(k/log n)** in $\tilde{O}\left(D+\sqrt{n}\right)$ rounds

A lower bound of $\Omega\left(D+\sqrt{n/k}\right)$ rounds for such decompositions

More on Spanning Tree Packings

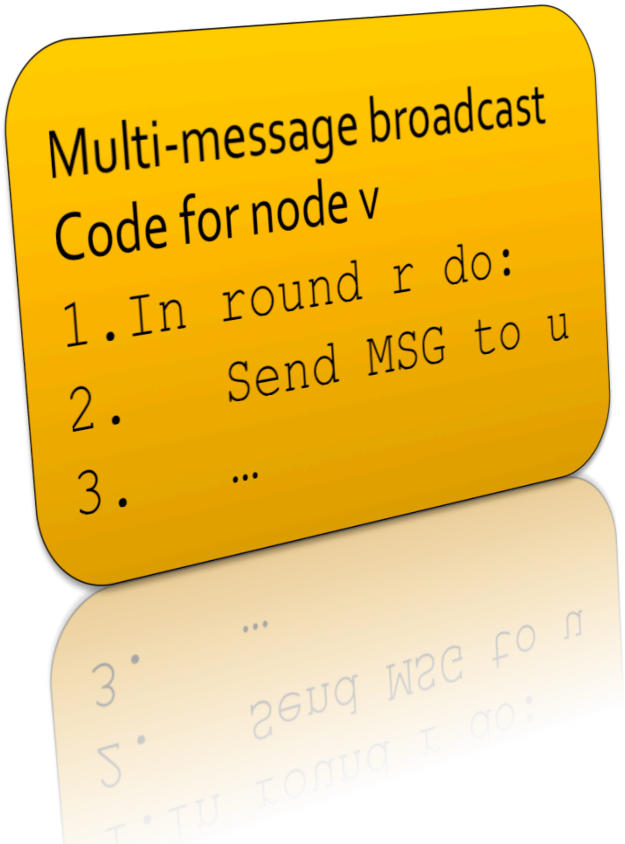
Centralized decomposition of **k**-vertex connected graphs into fractionally vertex-disjoint dominating trees with total weight **Ω(k/log n)** in $\tilde{O}(m)$ time

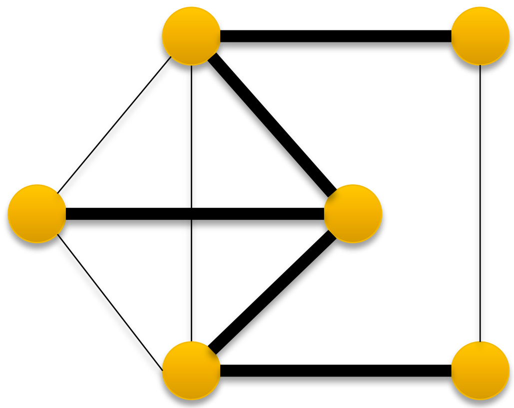
Centralized and distributed **O(log n)-approximation algorithms** for the vertex connectivity of a graph in $\tilde{O}(m)$ time, and $\tilde{O}\left(D+\sqrt{n}\right)$ rounds, respectively

Multi-Message Broadcast vs. Tree Packings

Multi-message broadcast

Tree packings

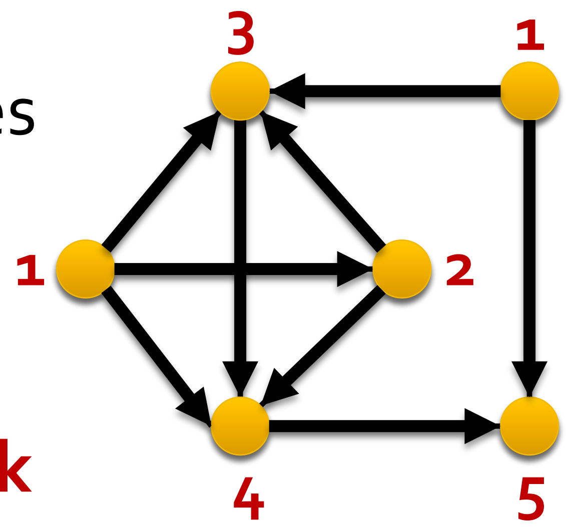




Colorings

Use **k** colors, neighbors have different colors

Acyclic orientation: no directed cycles
Length: length of longest path



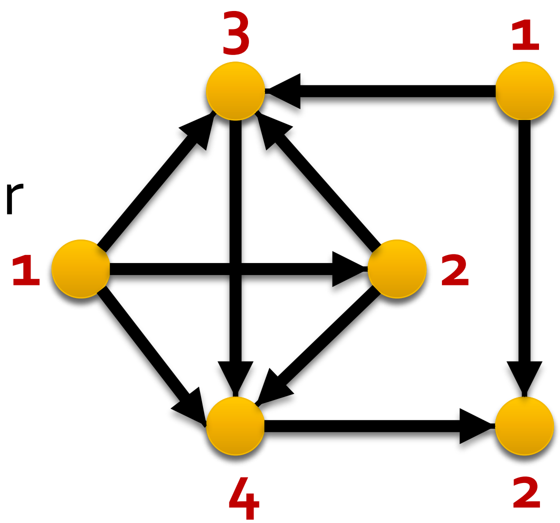
AO is an acyclic orientation of length **k**

- For **i=1,...,k+1**
  - Color **i** each node whose parents have been colored

Colorings vs. Acyclic Orientations

AO is an acyclic orientation, length **k** and in-degree **d**

- For **i=1,…,k+1**
  - Color each node whose parents have been colored with an **unused** color

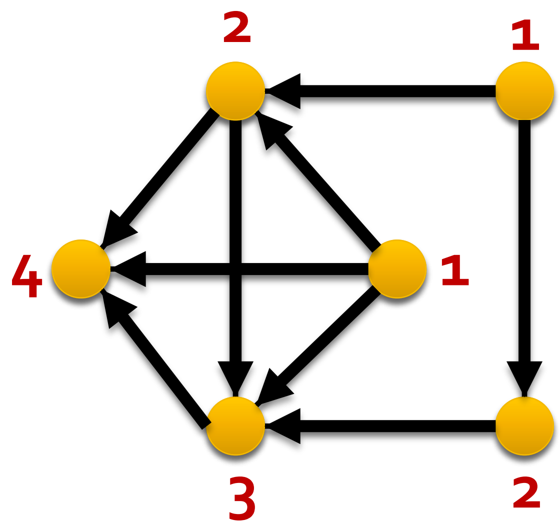


[Gallai, Hasse, Roy, Vitaver 1960's]

Colorings vs. Acyclic Orientations

A legal **k**-coloring gives an acyclic orientation with length **k-1**:

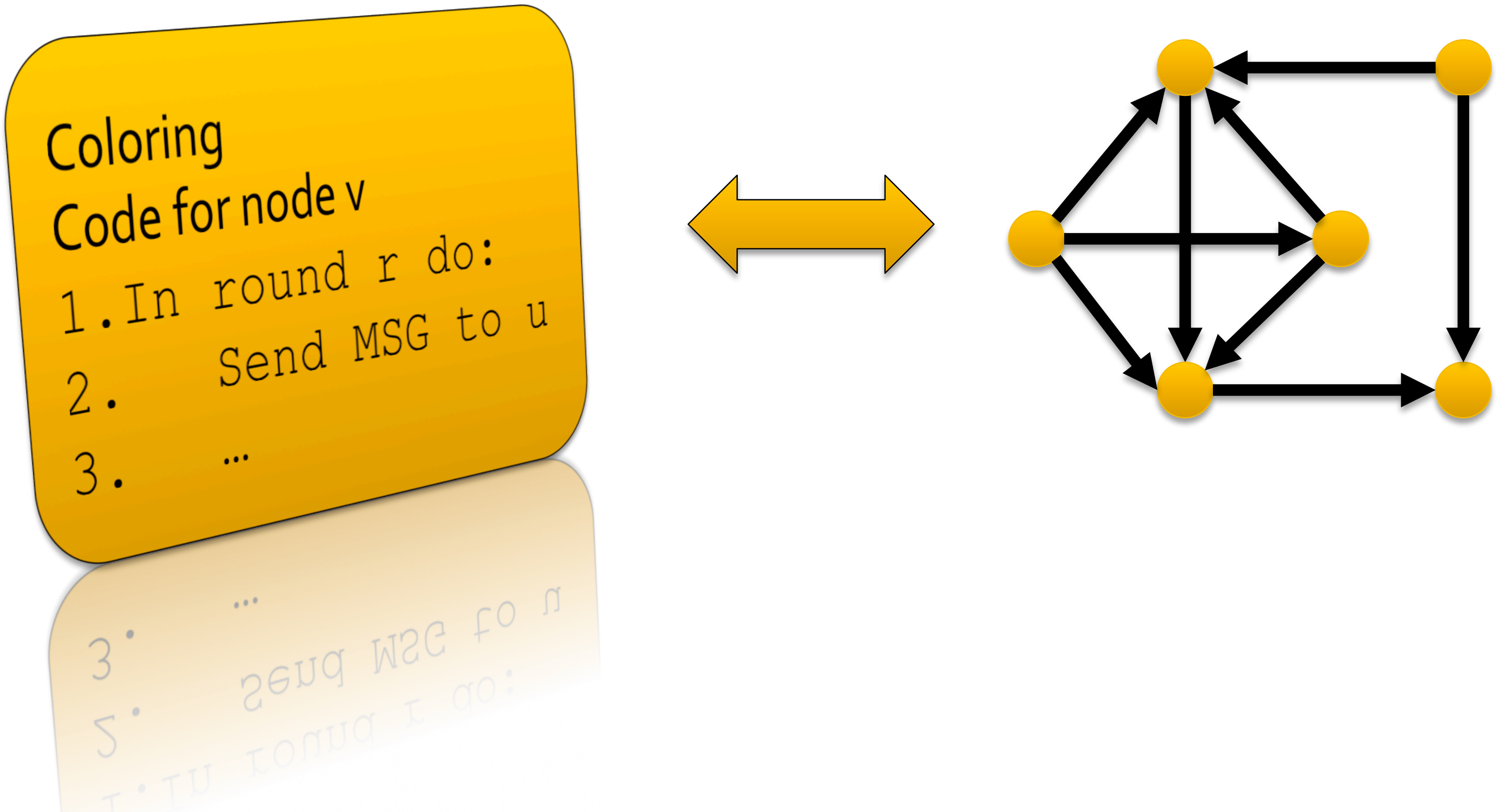
Orient edges from smaller color towards larger color



- Used in coloring algorithms by orienting edges **[Barenboim, Elkin 2008, 2009]**

Coloring vs. Acyclic Orientations

Coloring Acyclic orientations



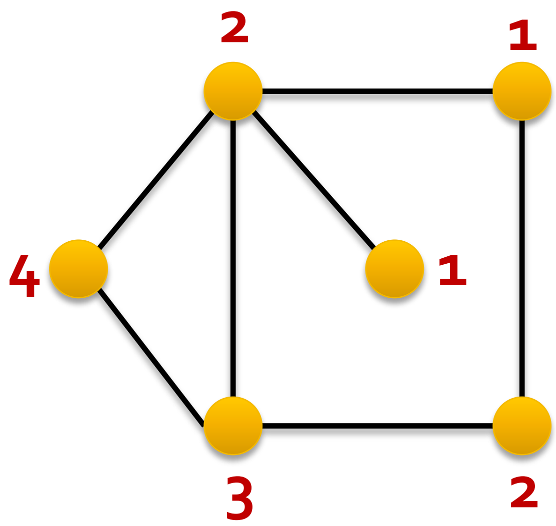
Colorings and MIS

MIS: maximal independent set

A **k**-coloring gives an MIS in **k** rounds.

- In round **i** all remaining nodes with color **i** enter the MIS and inform all their neighbors to drop out

MIS in **T(n,Δ)** rounds gives **(Δ+1)-coloring** in **T(n(Δ+1), 2Δ)** rounds **[Luby 1986]** (different graph)



Summary

Distributed Algorithms

Graph Structures