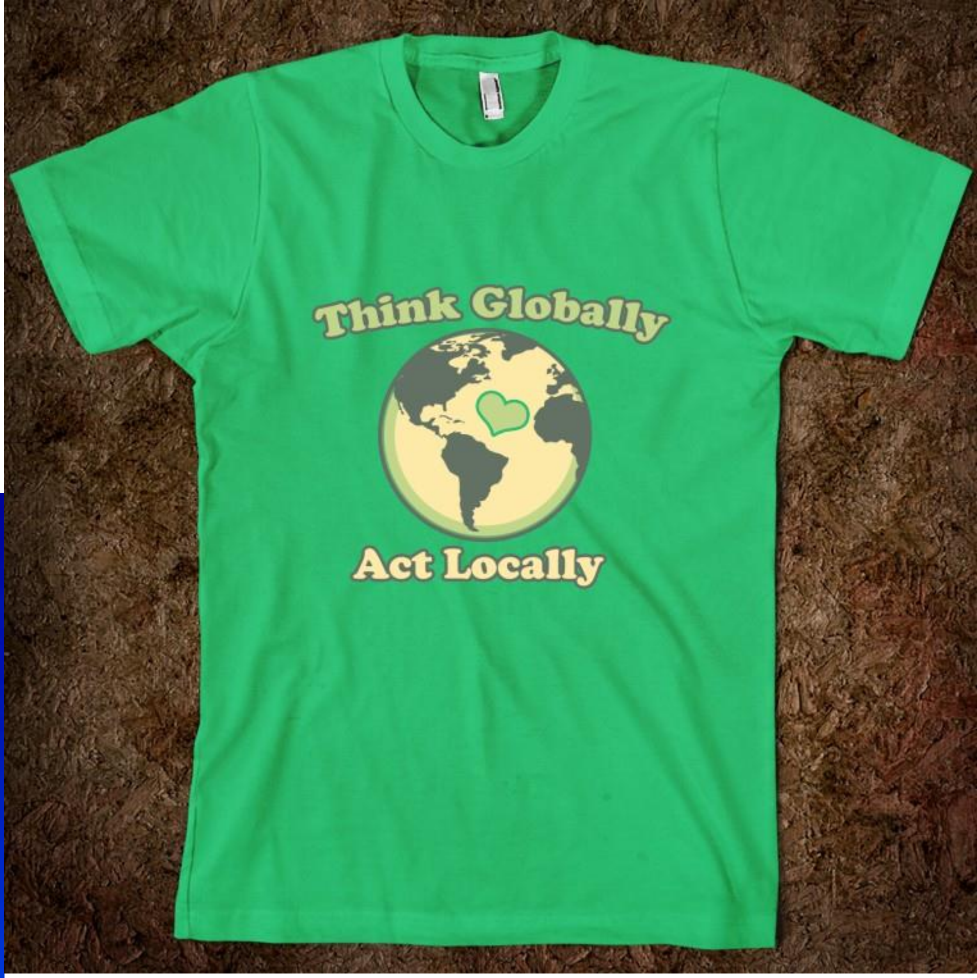
Think Global – Act Local

Roger Wattenhofer


Think global ...
act local.


Think Globally
Act Locally

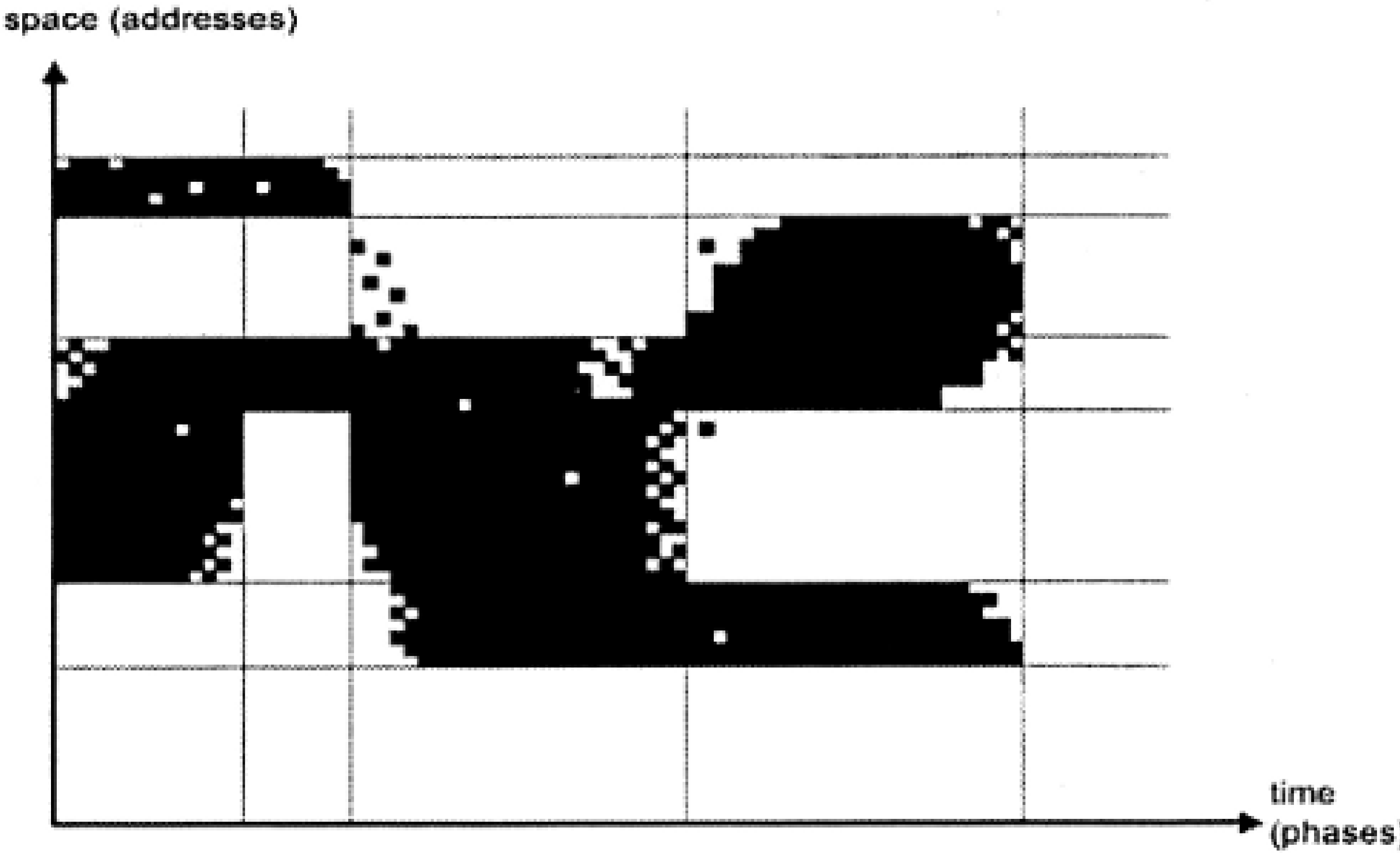

THINK GLOBAL
ACT LOCAL


think
global
act
local

Town Planning *Patrick Geddes*

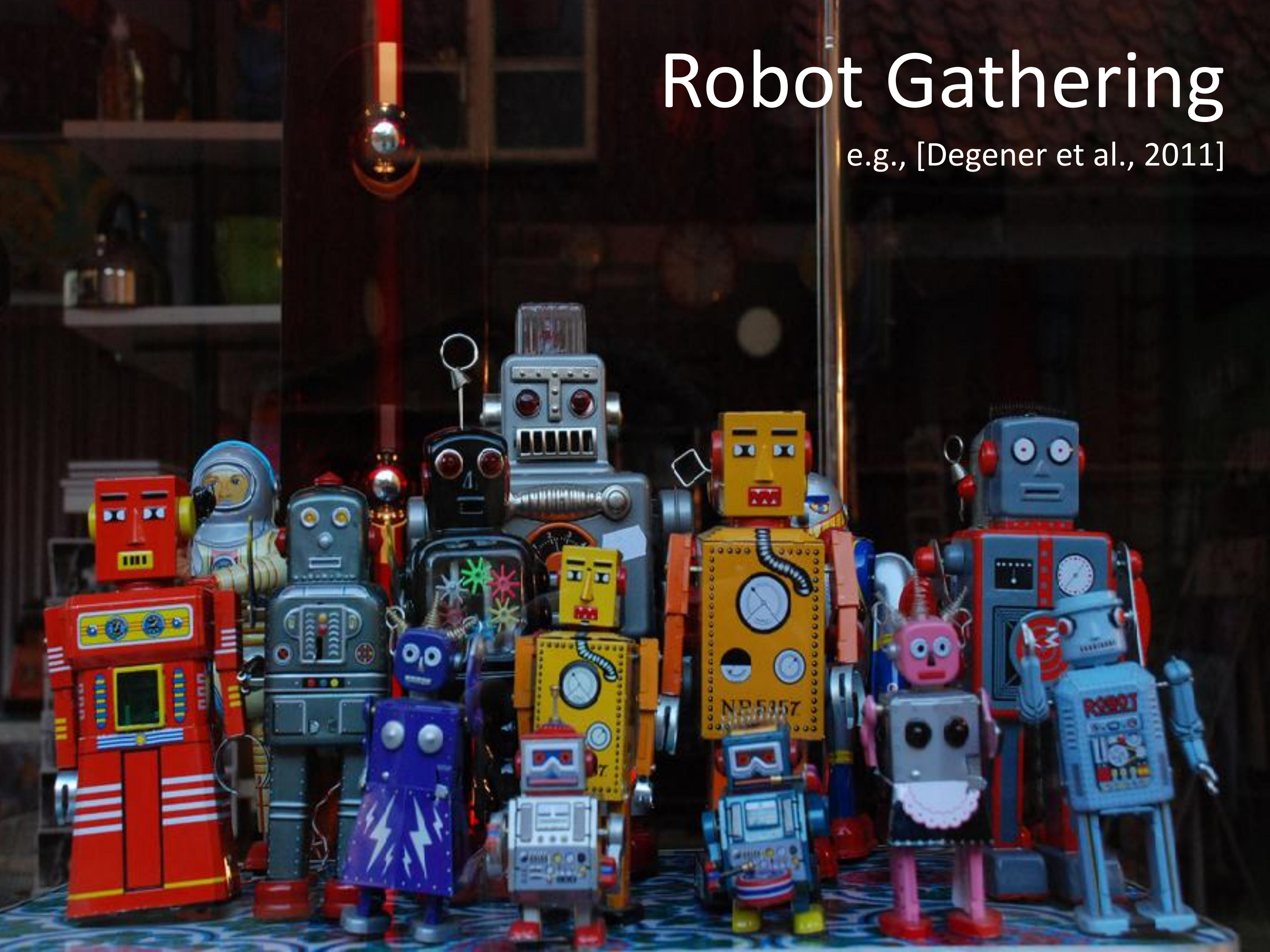

Architecture Buckminster Fuller

Computer Architecture *Caching*

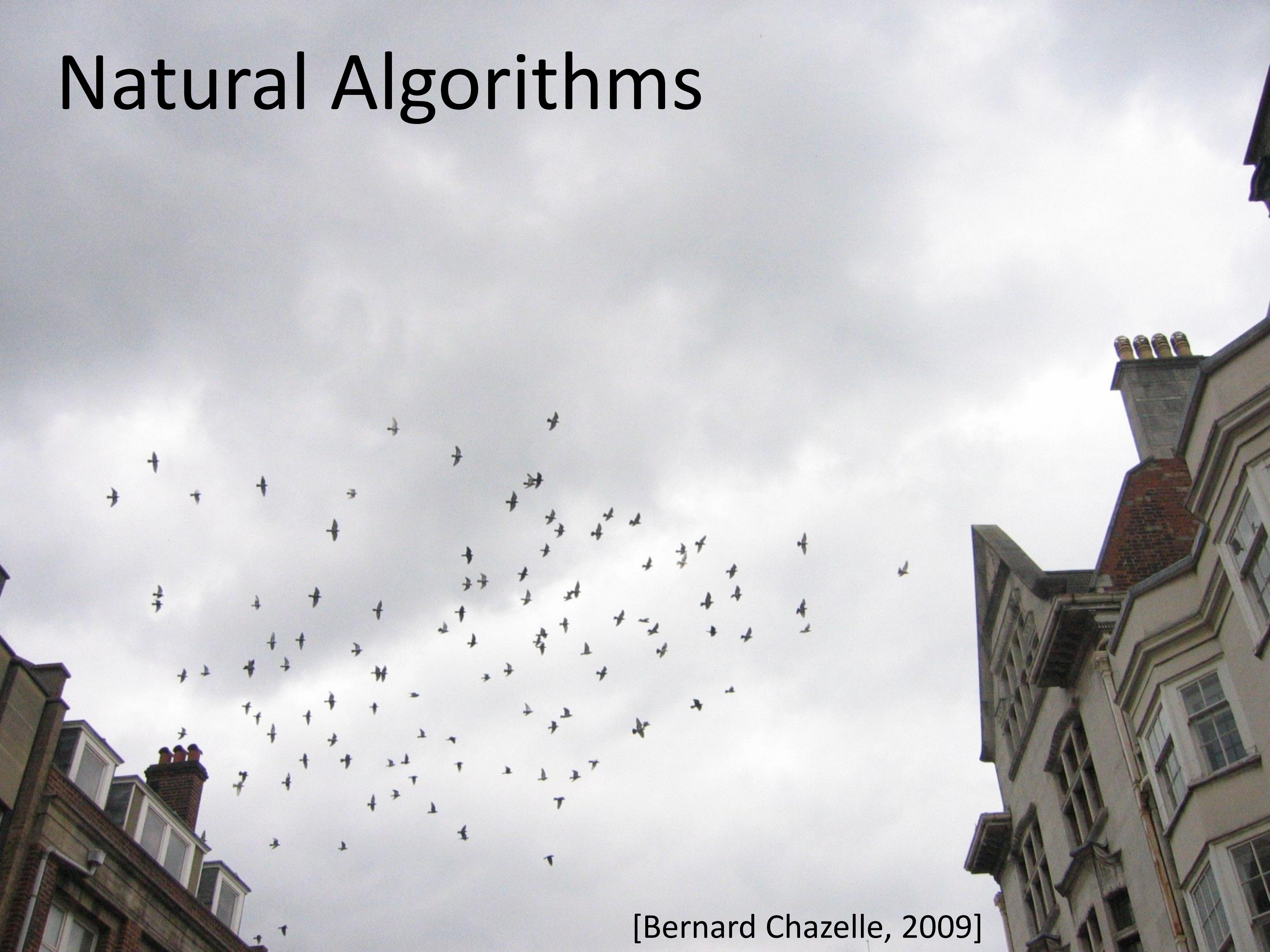


Robot Gathering

e.g., [Degener et al., 2011]



Natural Algorithms



[Bernard Chazelle, 2009]

game theory

Algorithmic Trading

Think Global – Act Local

…is there a theory?

Complexity Theory

Can a Computer Solve Problem P in Time t?

(Think Global – Act Local)

Distributed

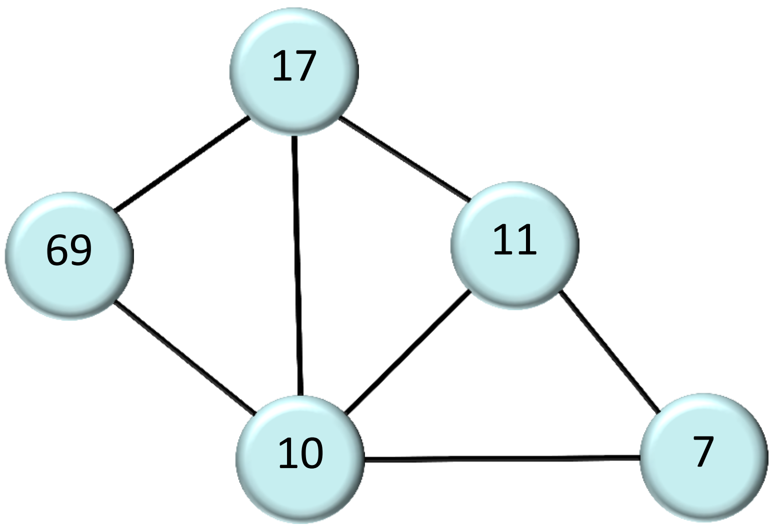
Complexity Theory

Network

Can a ~~Computer~~ Solve Problem P in Time t?

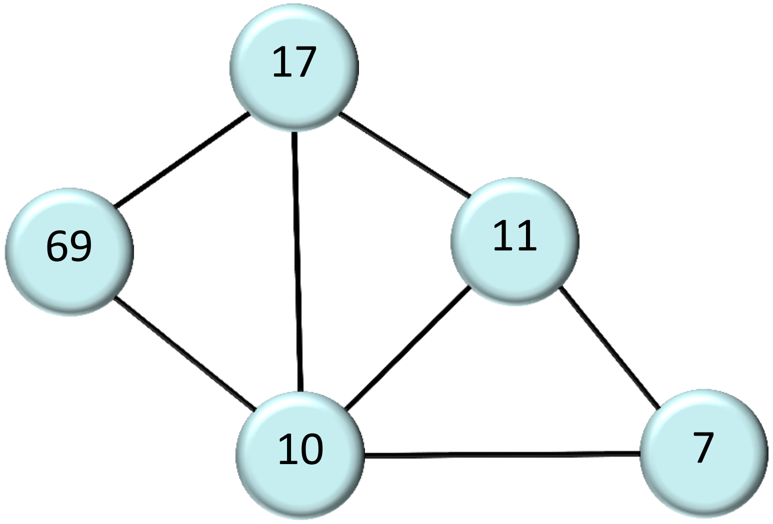
Distributed (Message-Passing) Algorithms

- Nodes are agents with unique ID's that can communicate with neighbors by sending messages. In each synchronous round, every node can send a (different) message to each neighbor.



Distributed (Message-Passing) Algorithms

- Nodes are agents with unique ID's that can communicate with neighbors by sending messages. In each synchronous round, every node can send a (different) message to each neighbor.



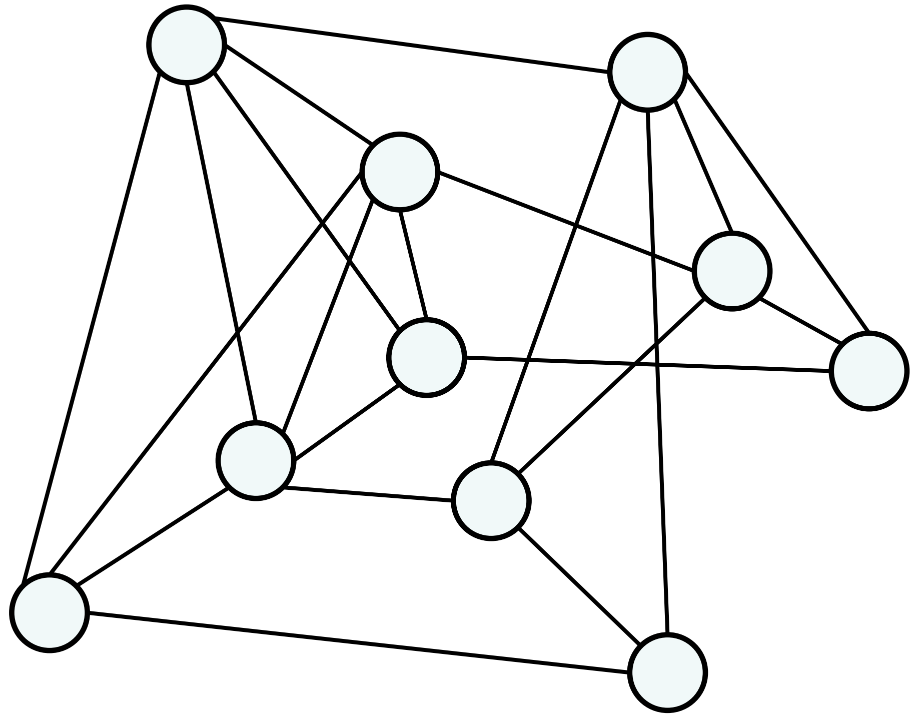
each round:
every node:
1. send msgs
2. rcv msgs
3. compute

- Distributed (Time) Complexity: How many rounds does problem take?

An Example

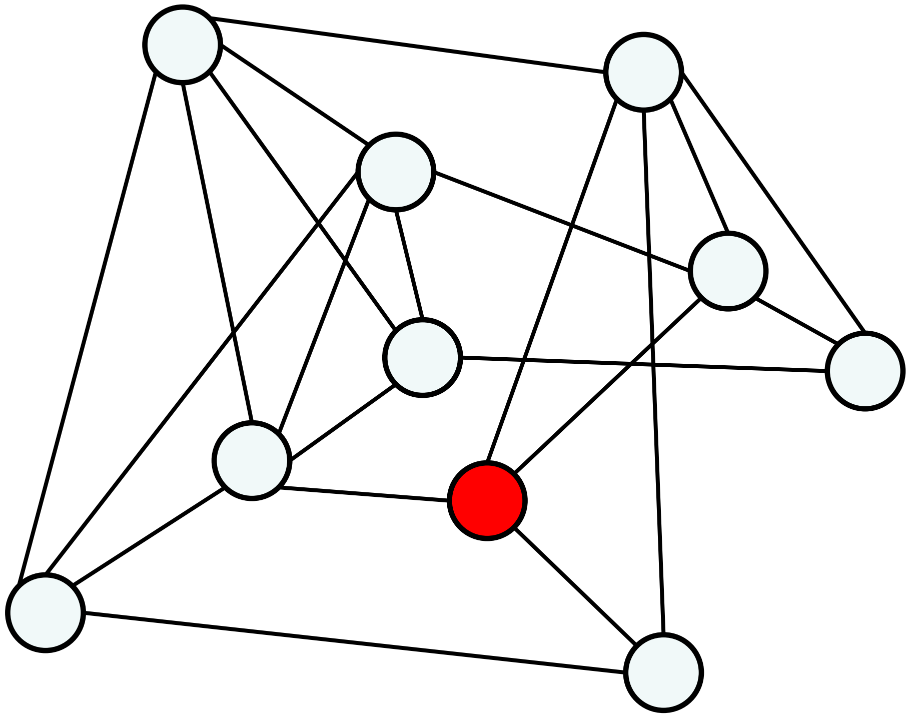
```
each round:
every node:
1. send msgs
2. rcv msgs
3. compute
```

How Many Nodes in Network?



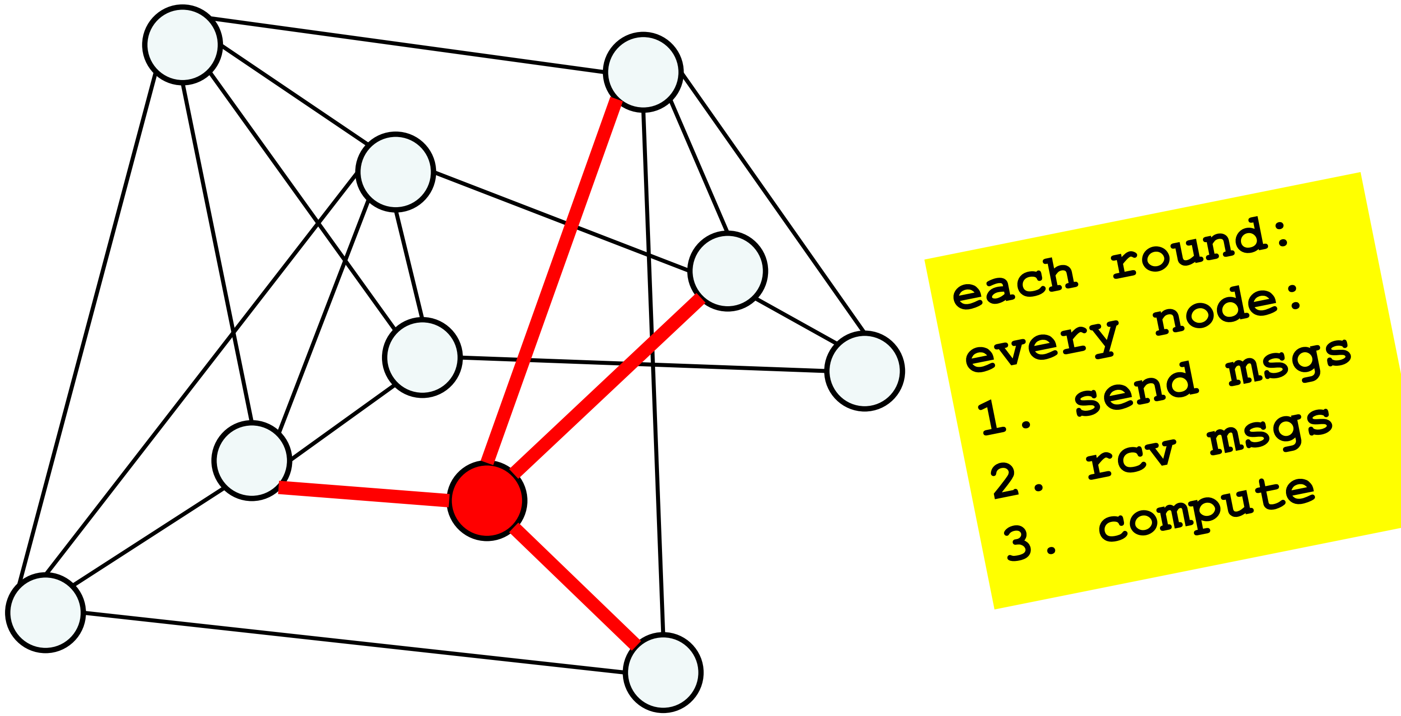
each round:
every node:
1. send msgs
2. rcv msgs
3. compute

How Many Nodes in Network?

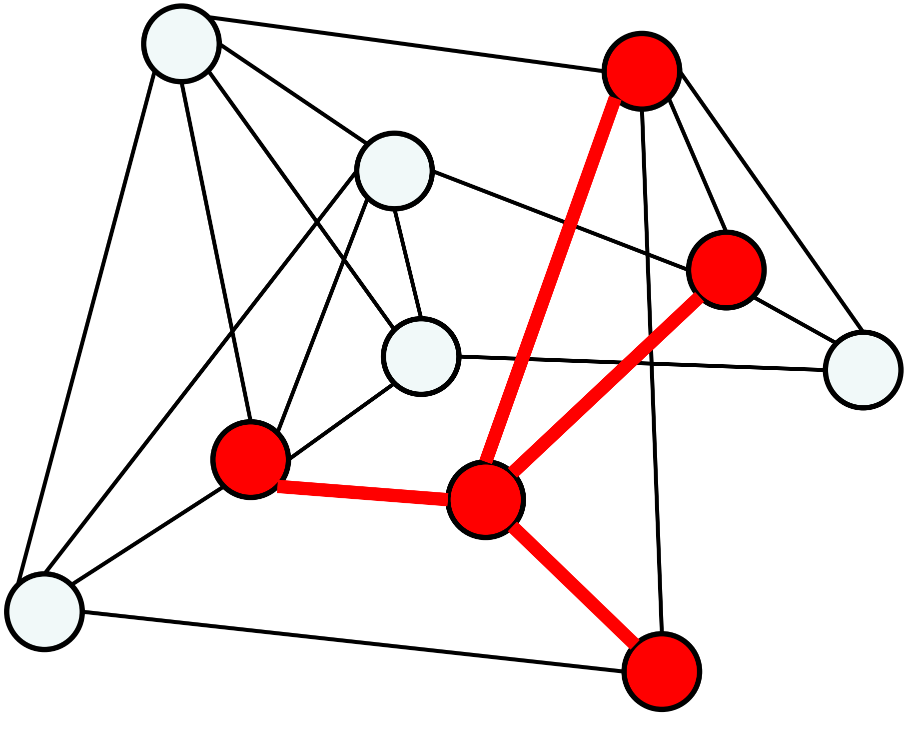


each round:
every node:
1. send msgs
2. rcv msgs
3. compute

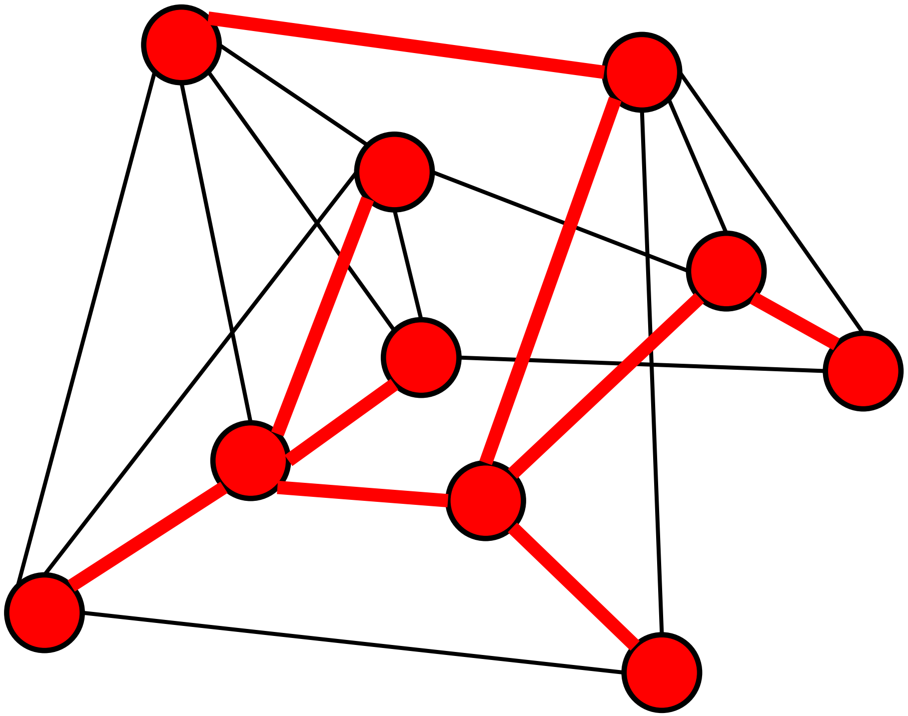
How Many Nodes in Network?



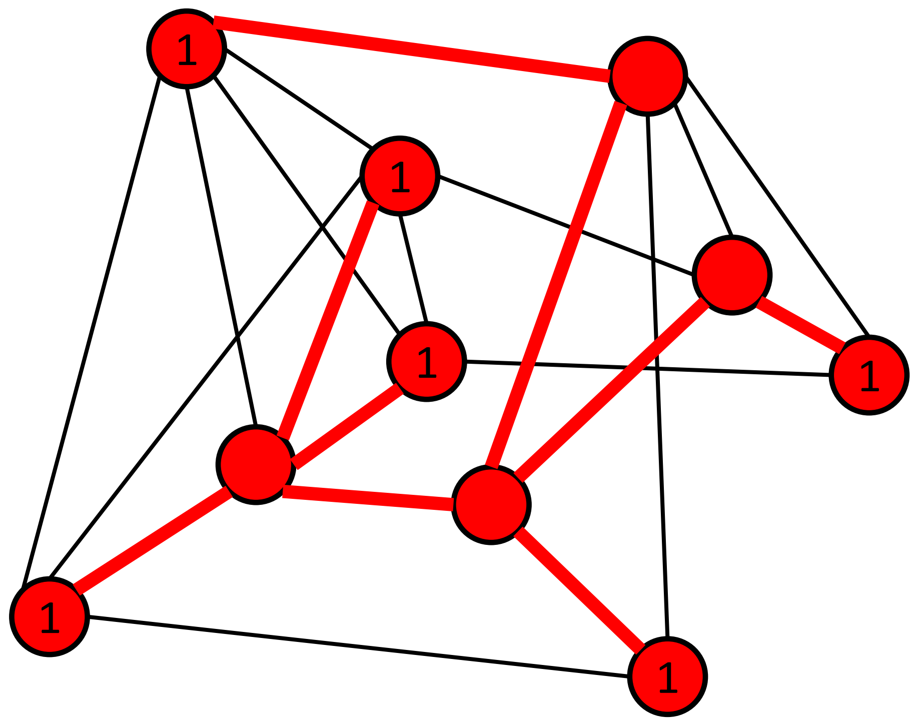
How Many Nodes in Network?



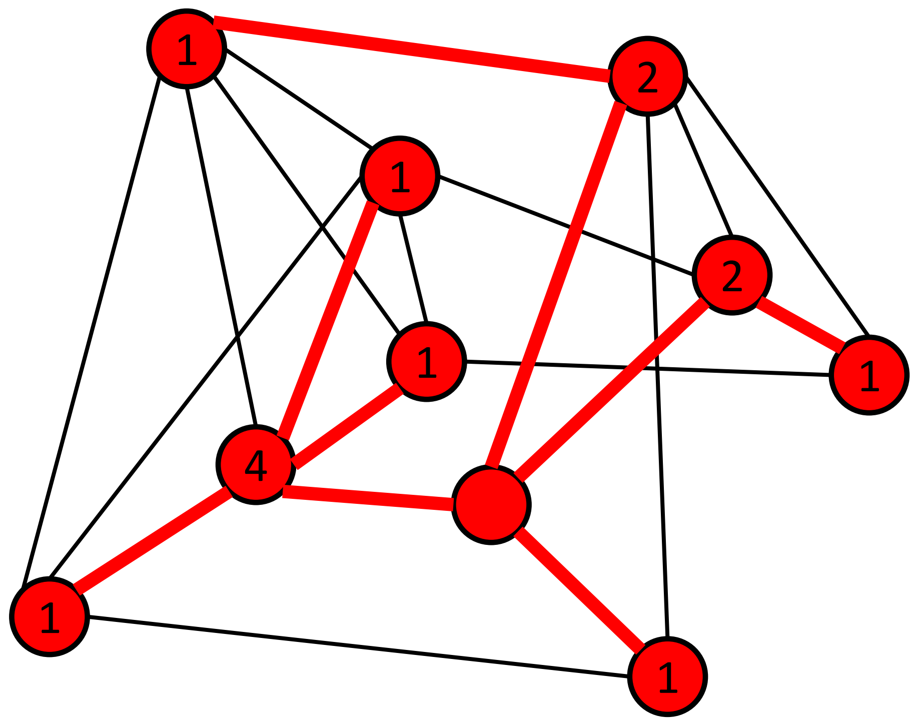
How Many Nodes in Network?



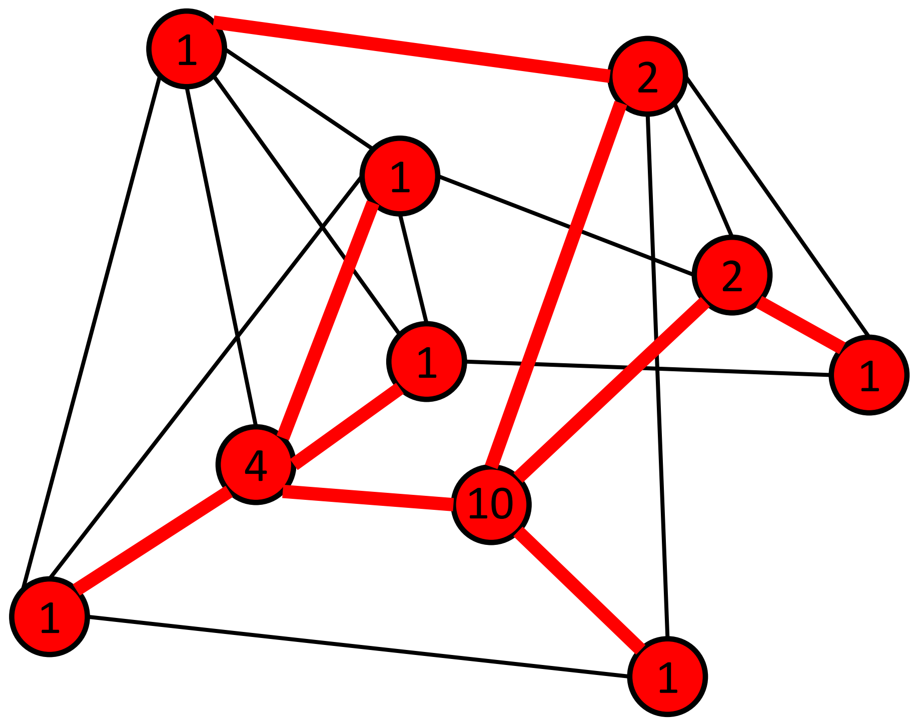
How Many Nodes in Network?



How Many Nodes in Network?

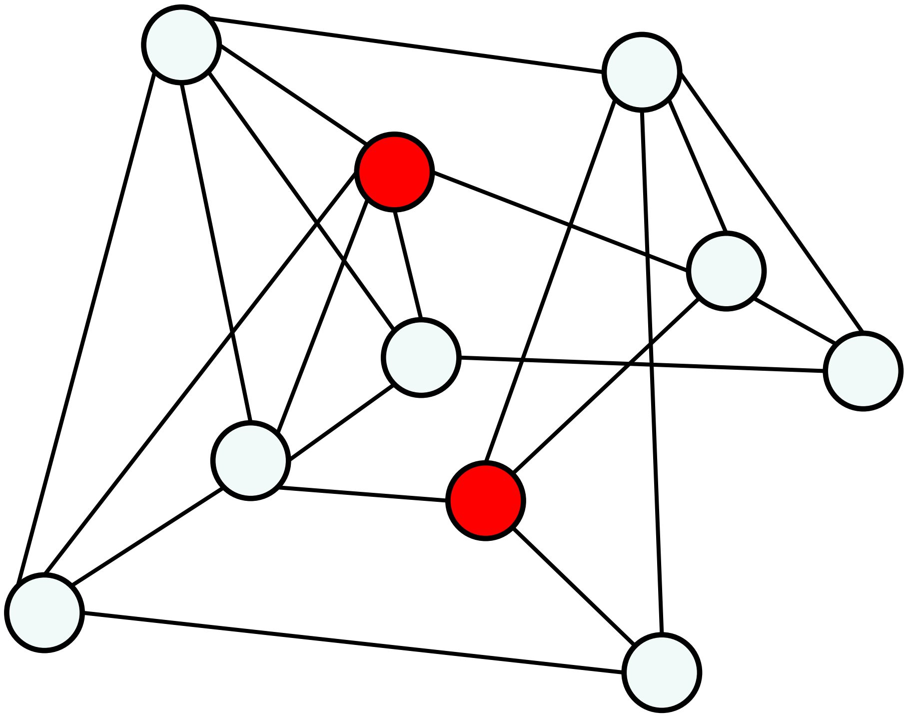


How Many Nodes in Network?



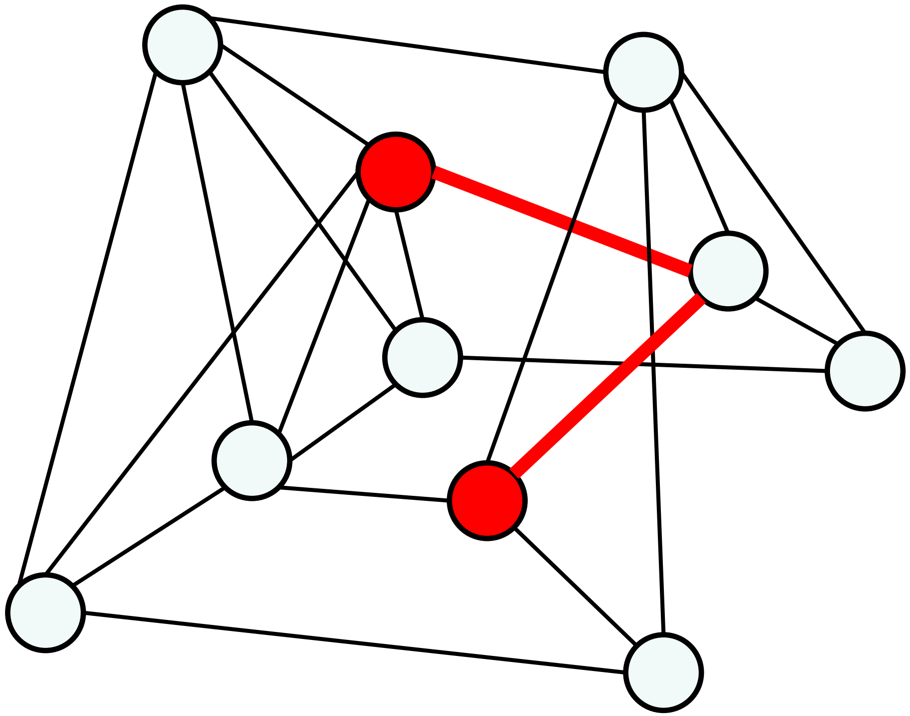
With a simple flooding/echo process, a network can find the number of nodes in time $O(D)$, where D is the diameter (size) of the network.

Diameter of Network?



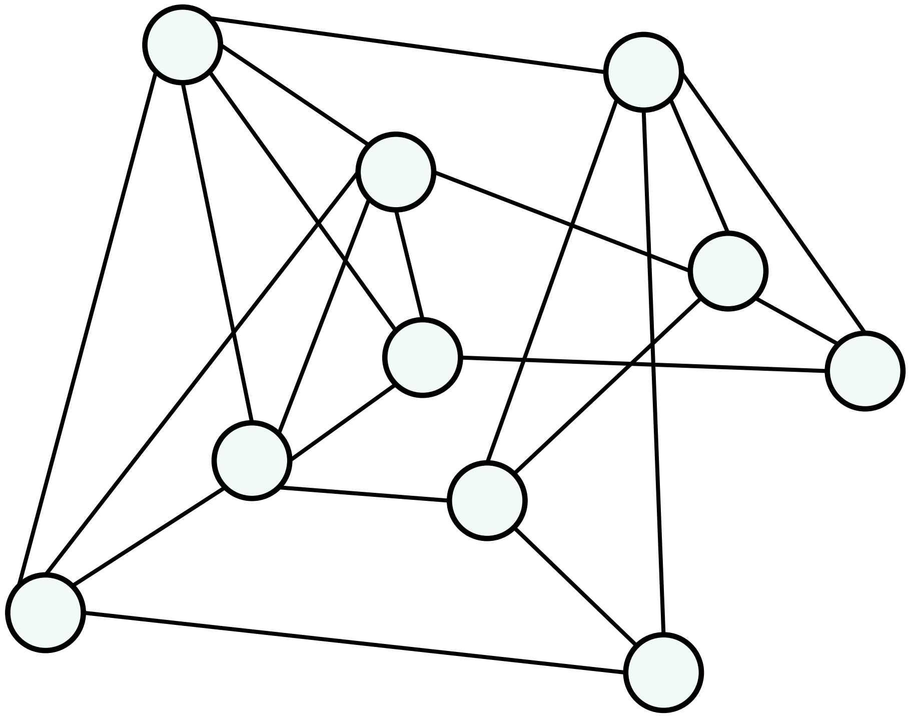
- Distance between two nodes = Number of hops of shortest path

Diameter of Network?



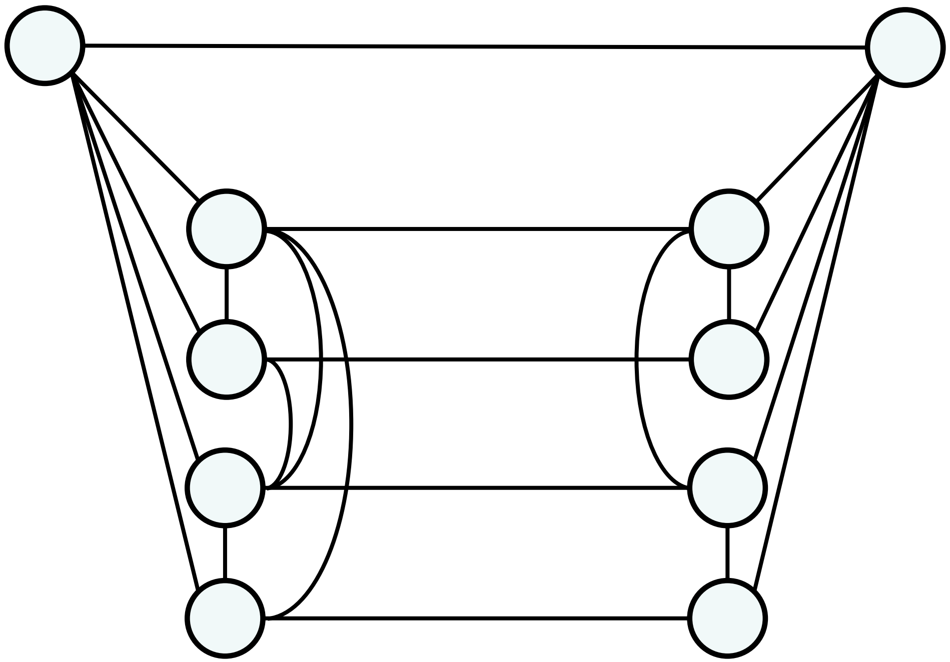
- Distance between two nodes = Number of hops of shortest path

Diameter of Network?

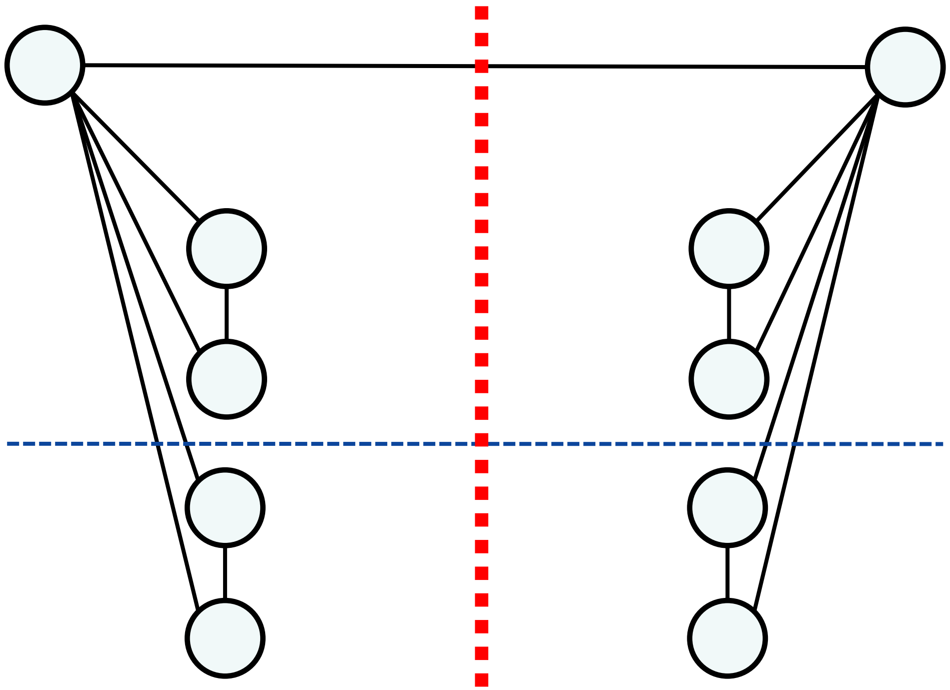


- Distance between two nodes = Number of hops of shortest path
- Diameter of network = Maximum distance, between any two nodes

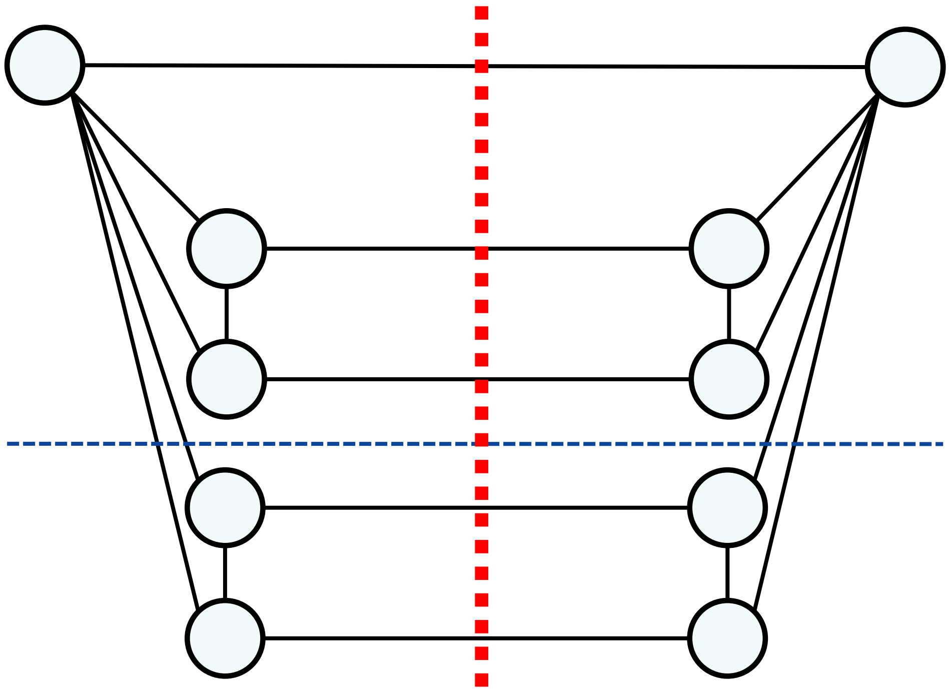
Diameter of Network?



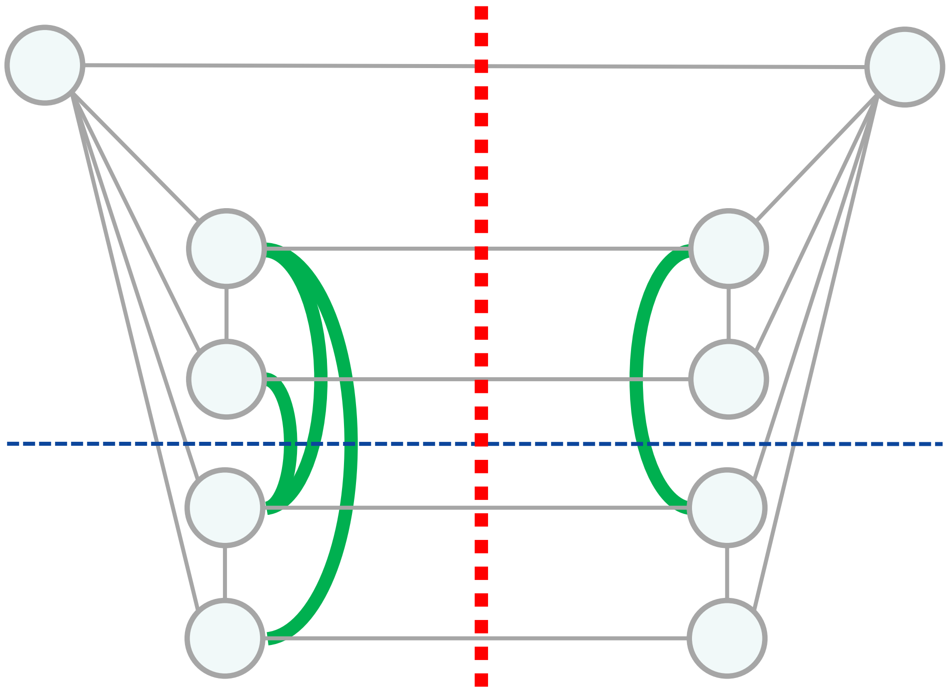
Diameter of Network?



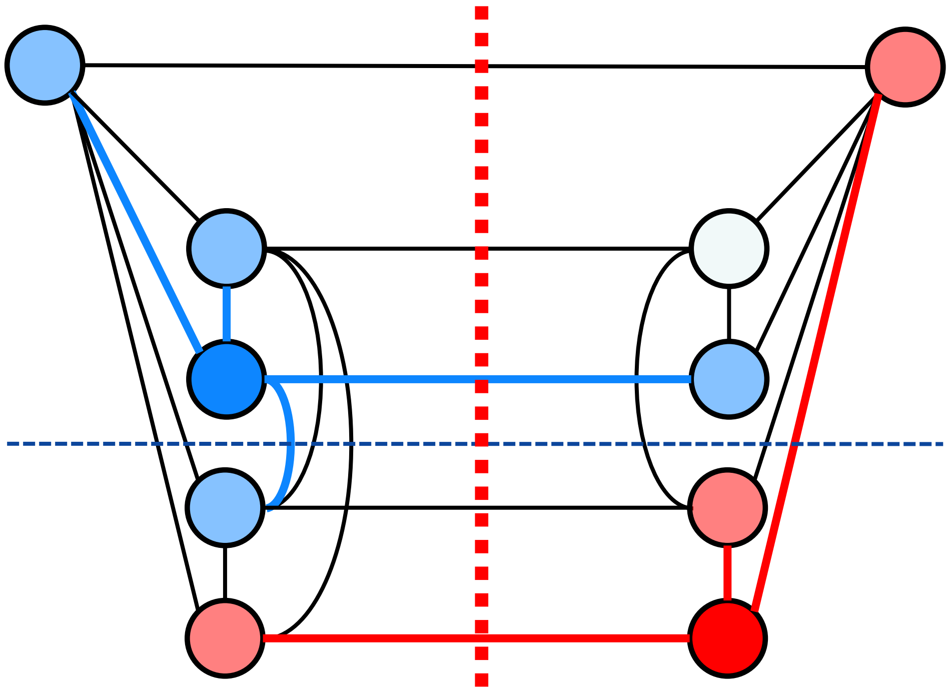
Diameter of Network?



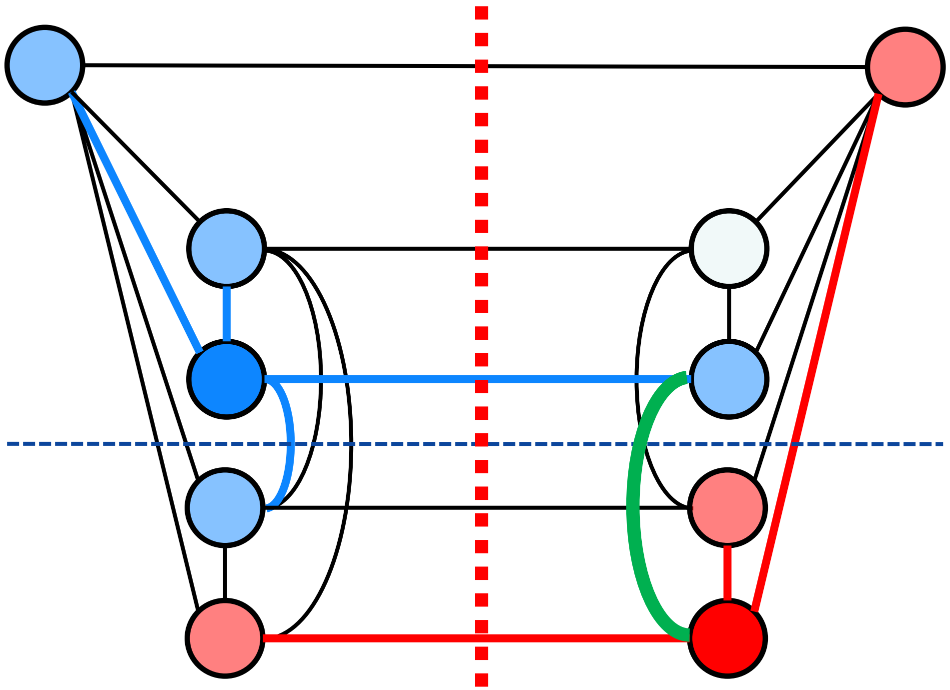
Diameter of Network?



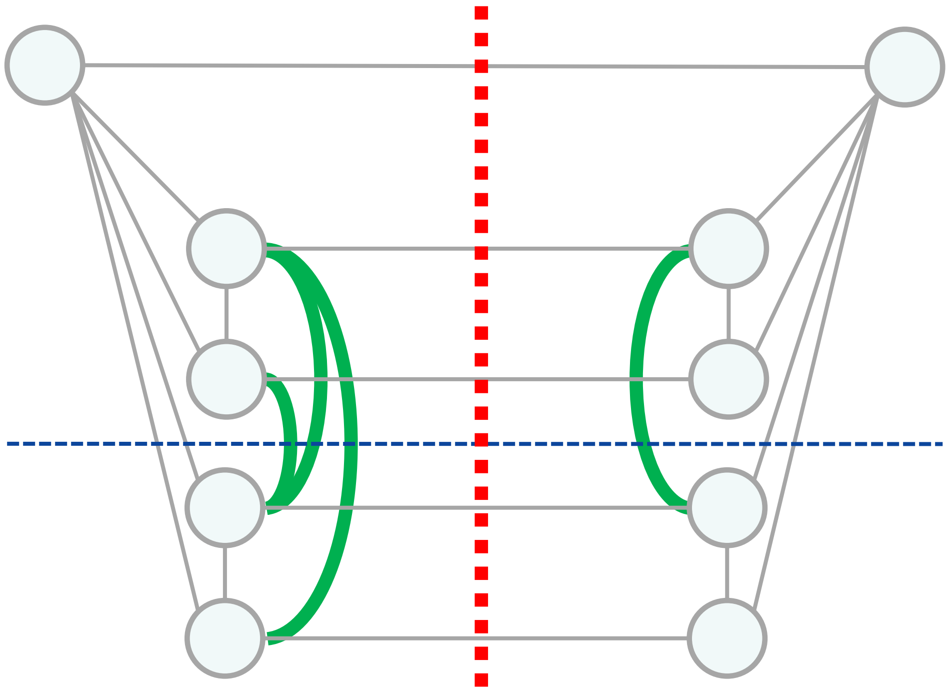
Diameter of Network?



Diameter of Network?

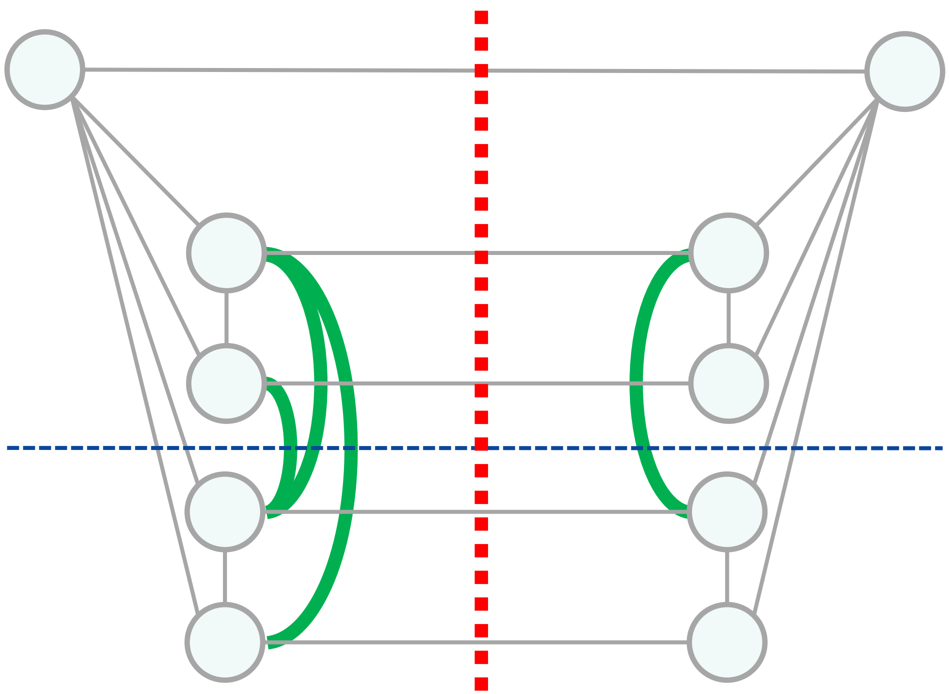


Diameter of Network?



Networks Cannot Compute Their Diameter in Sublinear Time!

(even if diameter is just a small constant)



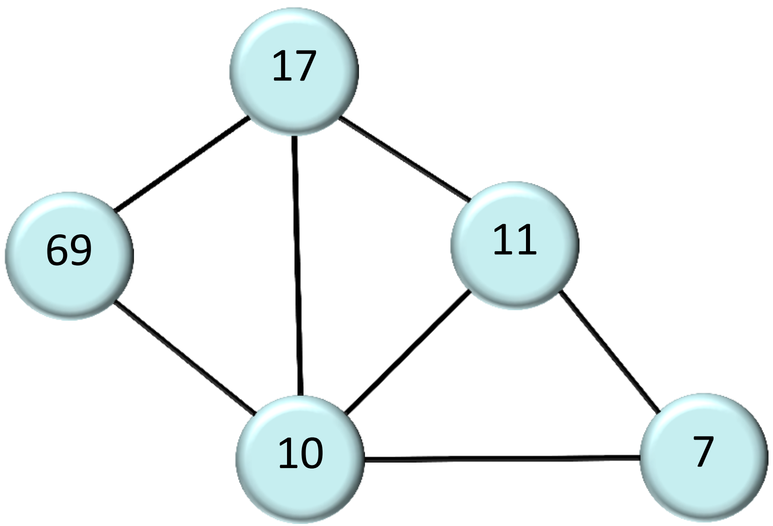
Pair of rows connected neither left nor right? Communication complexity:
Transmit $\Theta(n^2)$ information over $O(n)$ edges → $\Omega(n)$ time!

[Frischknecht, Holzer, W, 2012]

What about a “local” task?

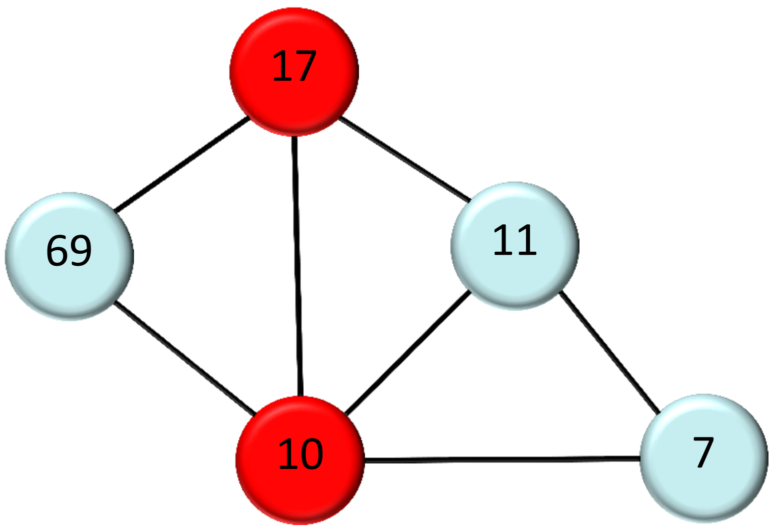
Example: Minimum Vertex Cover (MVC)

- Given a network with n nodes, nodes have unique IDs.
- Find a Minimum Vertex Cover (MVC)
  - a minimum set of nodes such that all edges are adjacent to node in MVC



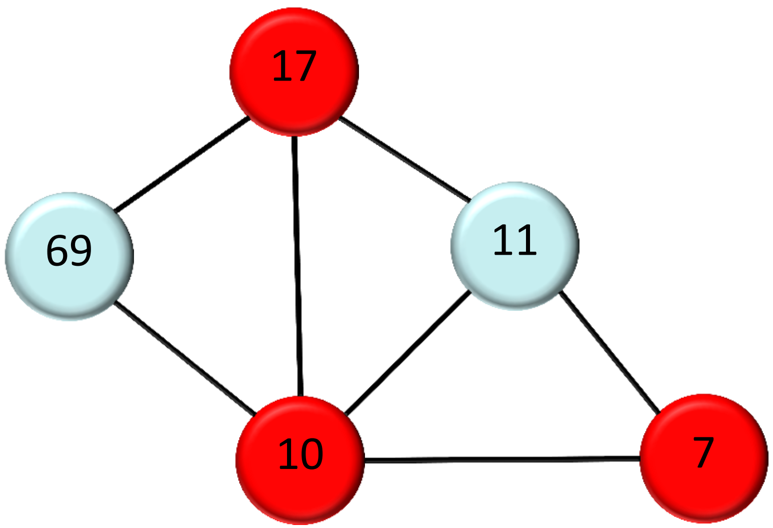
Example: Minimum Vertex Cover (MVC)

- Given a network with *n* nodes, nodes have unique IDs.
- Find a Minimum Vertex Cover (MVC)
  - a minimum set of nodes such that all edges are adjacent to node in MVC



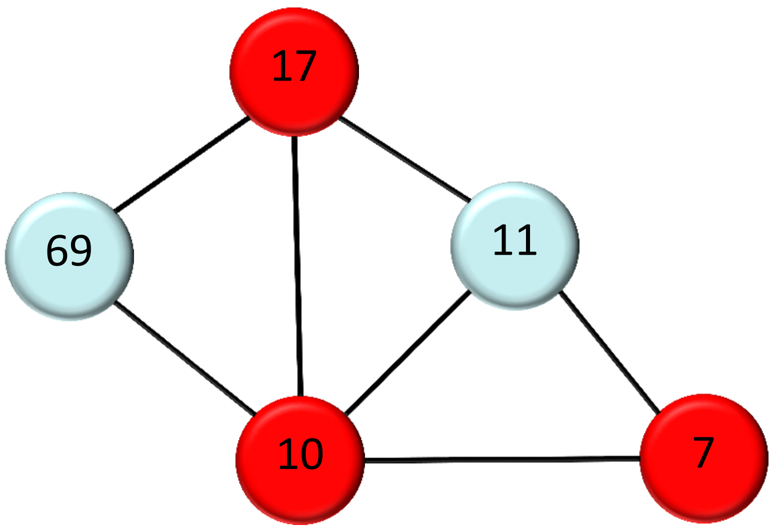
Example: Minimum Vertex Cover (MVC)

- Given a network with n nodes, nodes have unique IDs.
- Find a Minimum Vertex Cover (MVC)
  - a minimum set of nodes such that all edges are adjacent to node in MVC



On MVC

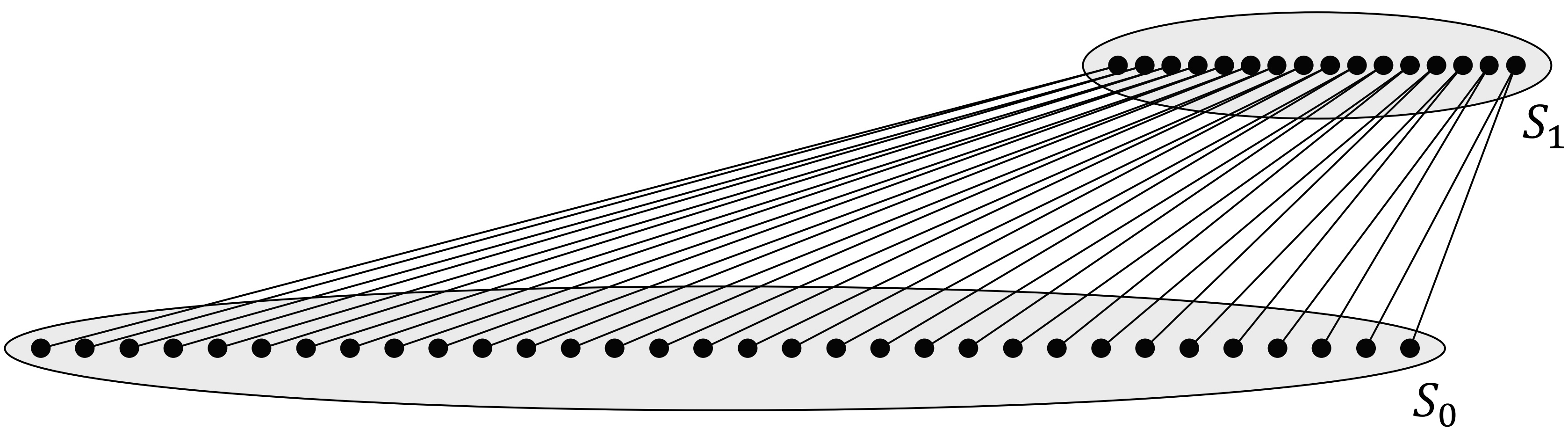
- Find an MVC that is “close” to minimum (approximation)
- Trade-off between time complexity and approximation ratio



- MVC: Various simple (non-distributed) 2-approximations exist!
- What about distributed algorithms?!?

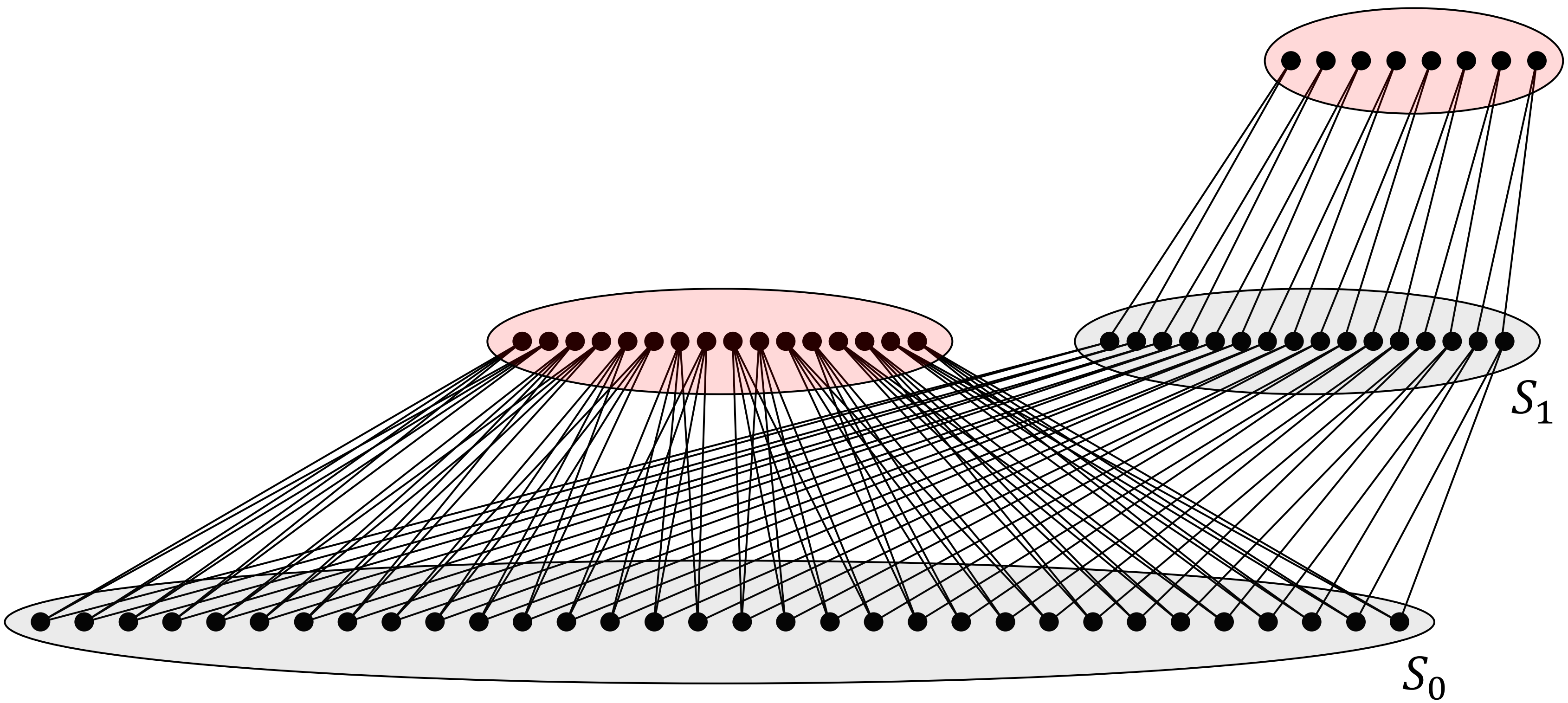
Finding the MVC (by Distributed Algorithm)

- Given the following bipartite graph with $|S_0| = \delta\ |S_1|$
- The MVC is just all the nodes in S_1
- Distributed Algorithm…



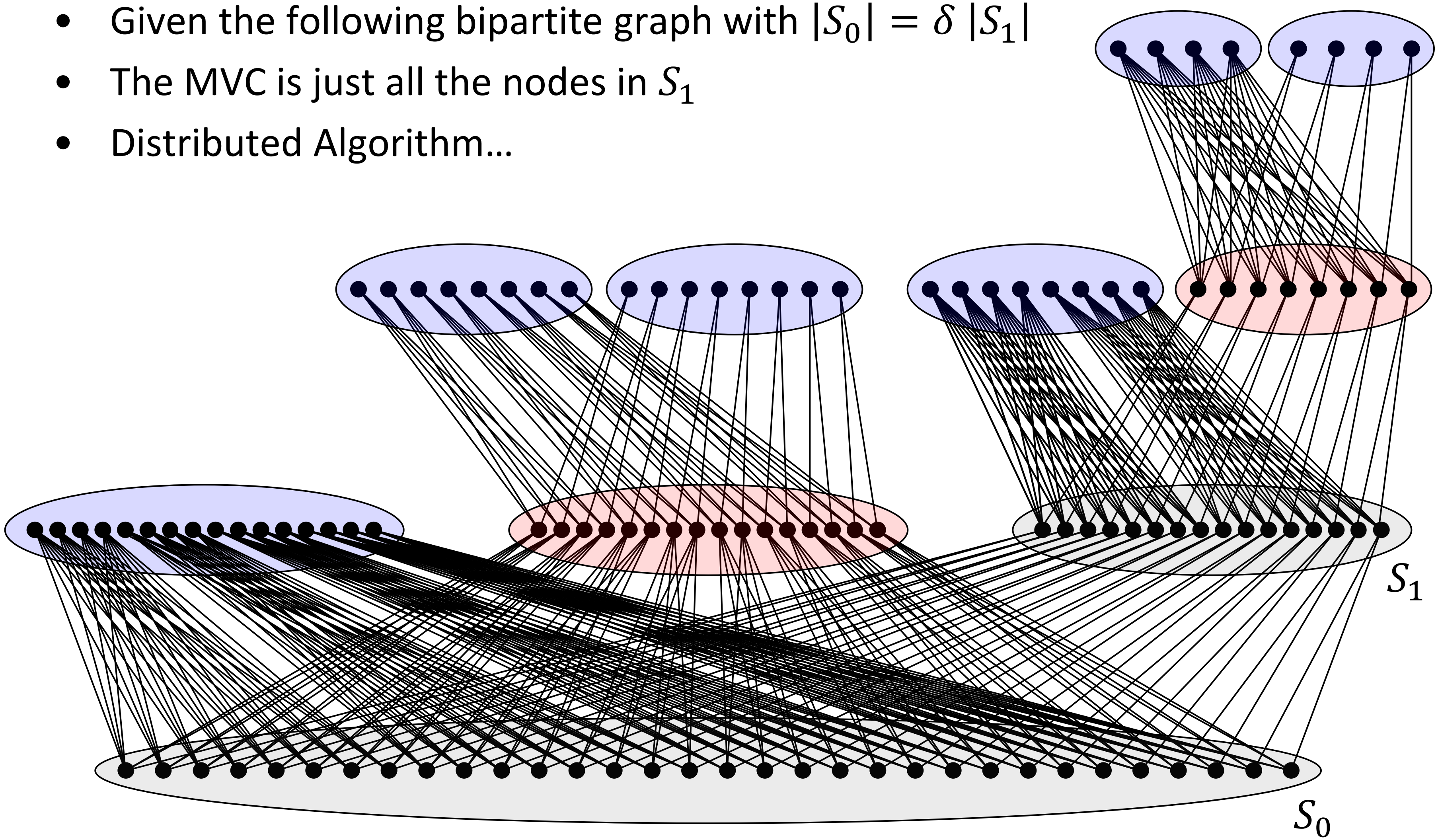
Finding the MVC (by Distributed Algorithm)

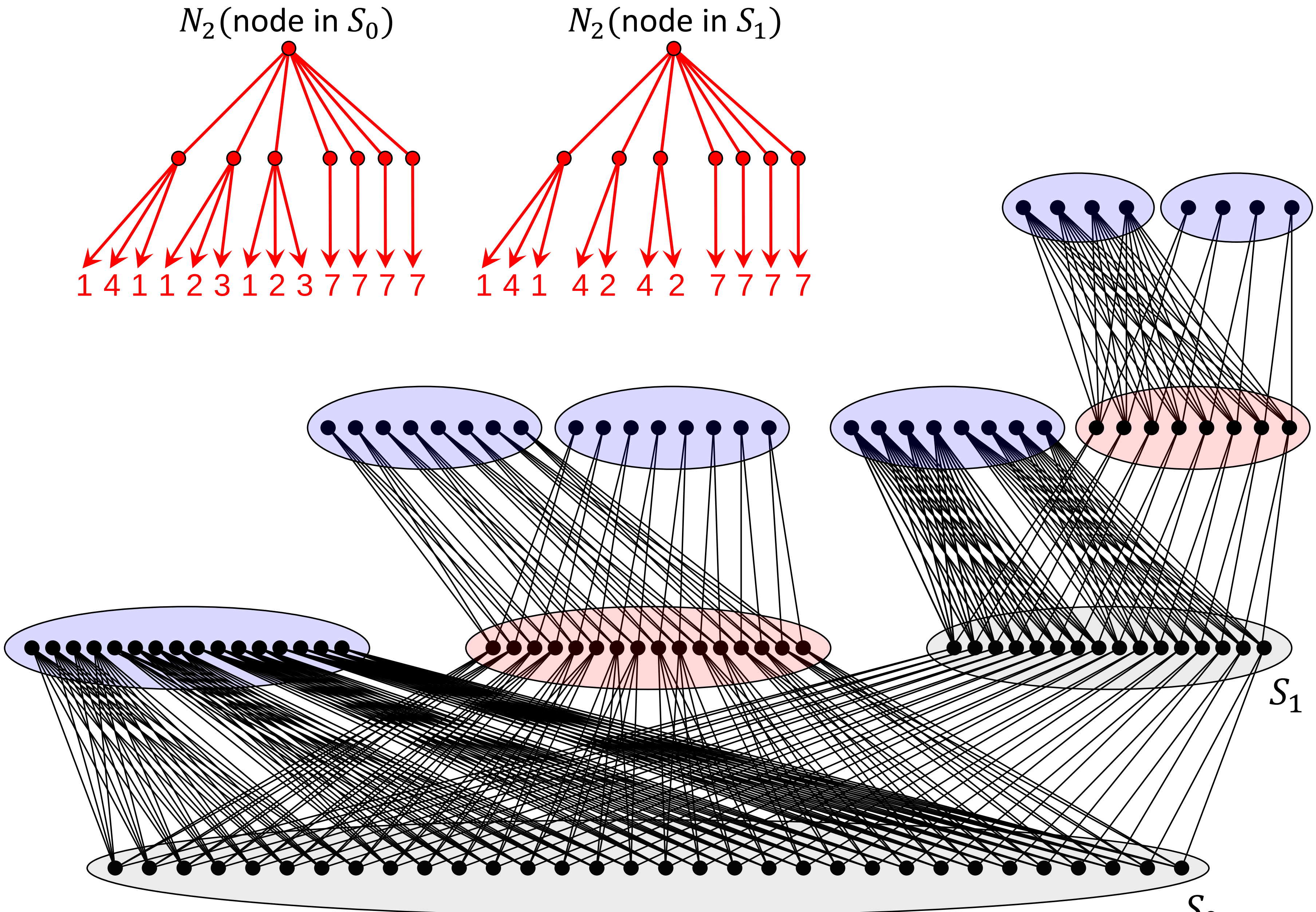
- Given the following bipartite graph with $|S_0| = \delta\ |S_1|$
- The MVC is just all the nodes in S_1
- Distributed Algorithm…



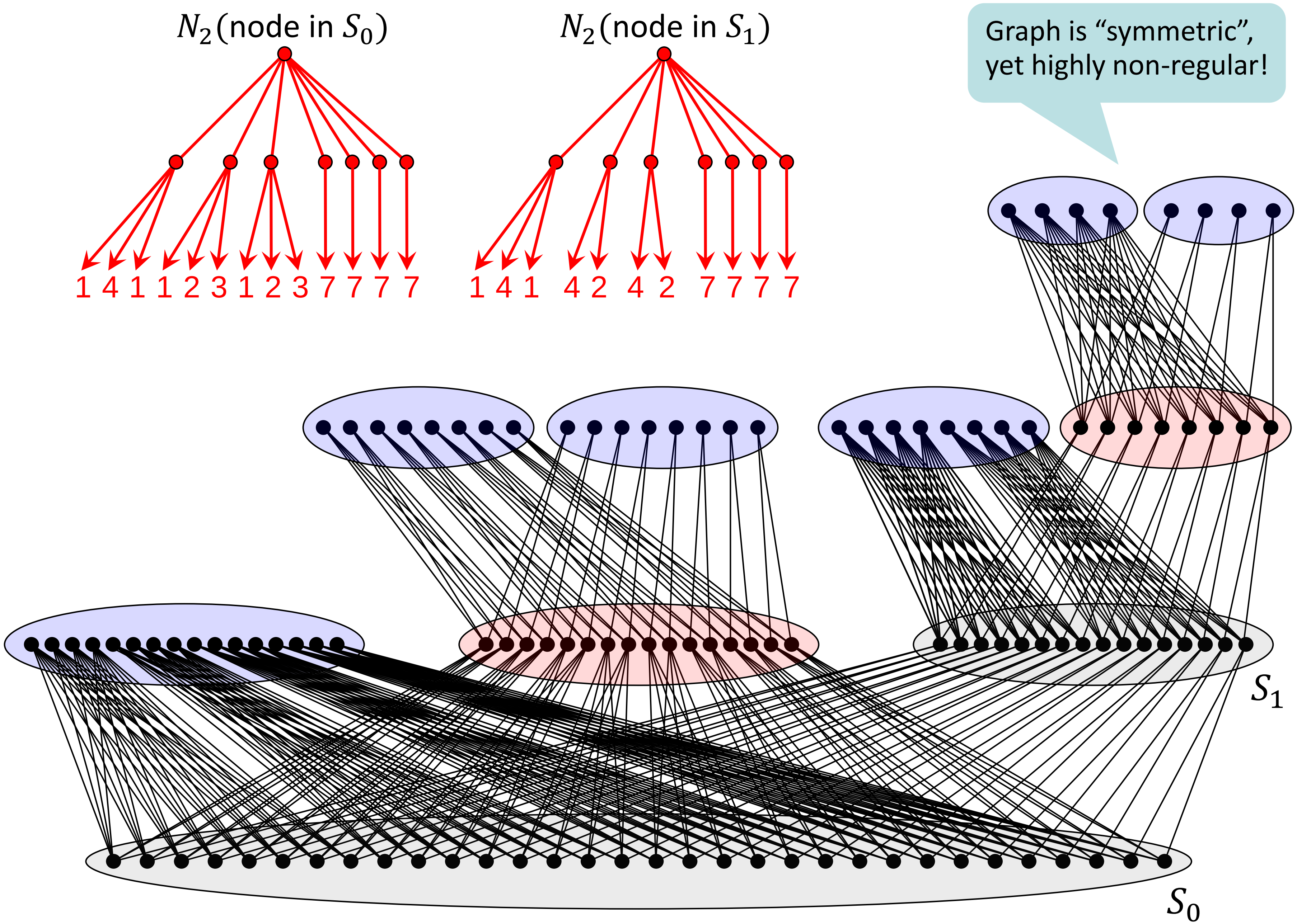
Finding the MVC (by Distributed Algorithm)

- Given the following bipartite graph with $|S_0| = \delta\ |S_1|$
- The MVC is just all the nodes in S_1
- Distributed Algorithm…





N_2(node in S_0)
1 4 1 1 2 3 1 2 3 7 7 7 7
N_2(node in S_1)
1 4 1 4 2 4 2 7 7 7 7
S_1
S_0



N_2(node in S_0)
1 4 1 1 2 3 1 2 3 7 7 7 7
N_2(node in S_1)
1 4 1 4 2 4 2 7 7 7 7
Graph is “symmetric”, yet highly non-regular!
S_1
S_0

Lower Bound: Results

- We can show that for $\epsilon > 0$, in t time, the approximation ratio is at least

$$\Omega\left(n^{\frac{1/4-\varepsilon}{t^2}}\right) \quad and \quad \Omega\left(\Delta^{\frac{1-\varepsilon}{t+1}}\right)$$

- Constant approximation needs at least $\Omega(\log\Delta)$ and $\Omega(\sqrt{\log n})$ time.
- Polylog approximation $\Omega(\log\Delta/\log\log\Delta)$ and $\Omega(\sqrt{\log n/\log\log n})$.

[Kuhn, Moscibroda, W, journal version in submission]

Lower Bound: Results

- We can show that for $\epsilon > 0$, in t time, the approximation ratio is at least

$$\Omega\left(n^{\frac{1/4-\varepsilon}{t^2}}\right) \quad and \quad \Omega\left(\Delta^{\frac{1-\varepsilon}{t+1}}\right)$$

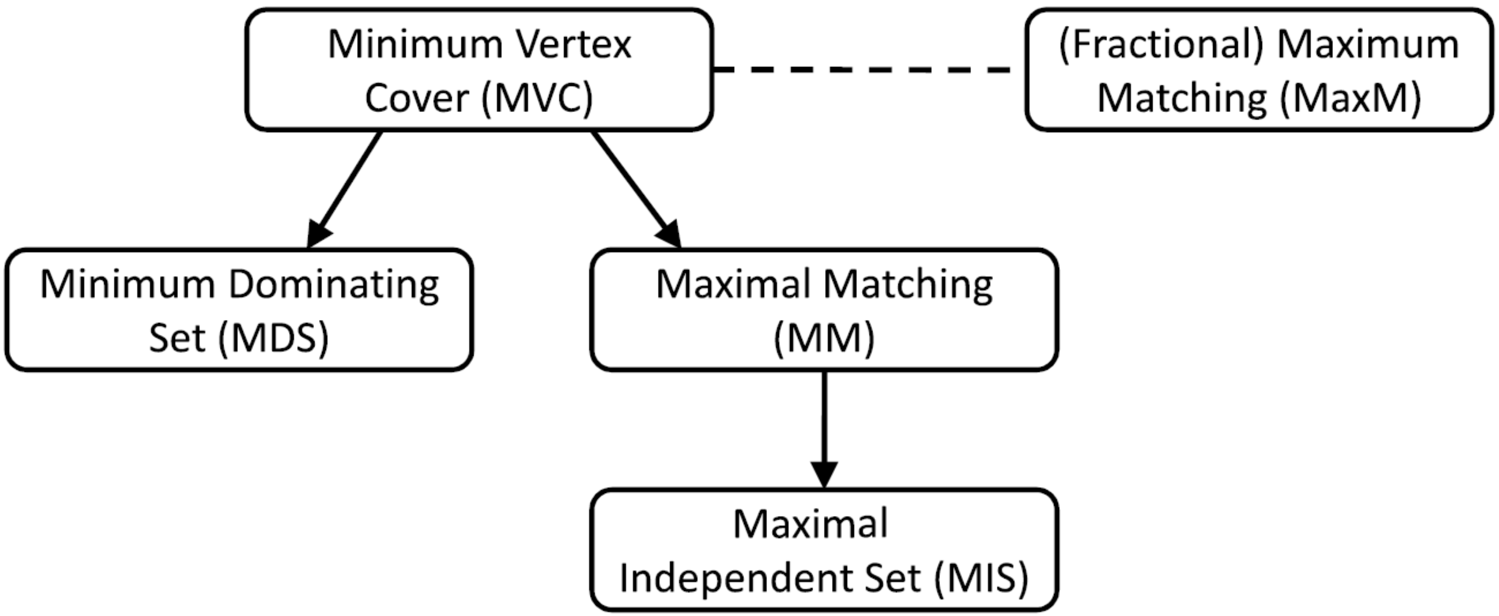
tight for MVC

- Constant approximation needs at least $\Omega(\log\Delta)$ and $\Omega(\sqrt{\log n})$ time.
- Polylog approximation $\Omega(\log\Delta/\log\log\Delta)$ and $\Omega(\sqrt{\log n/\log\log n})$.

[Kuhn, Moscibroda, W, journal version in submission]

Lower Bound: Reductions

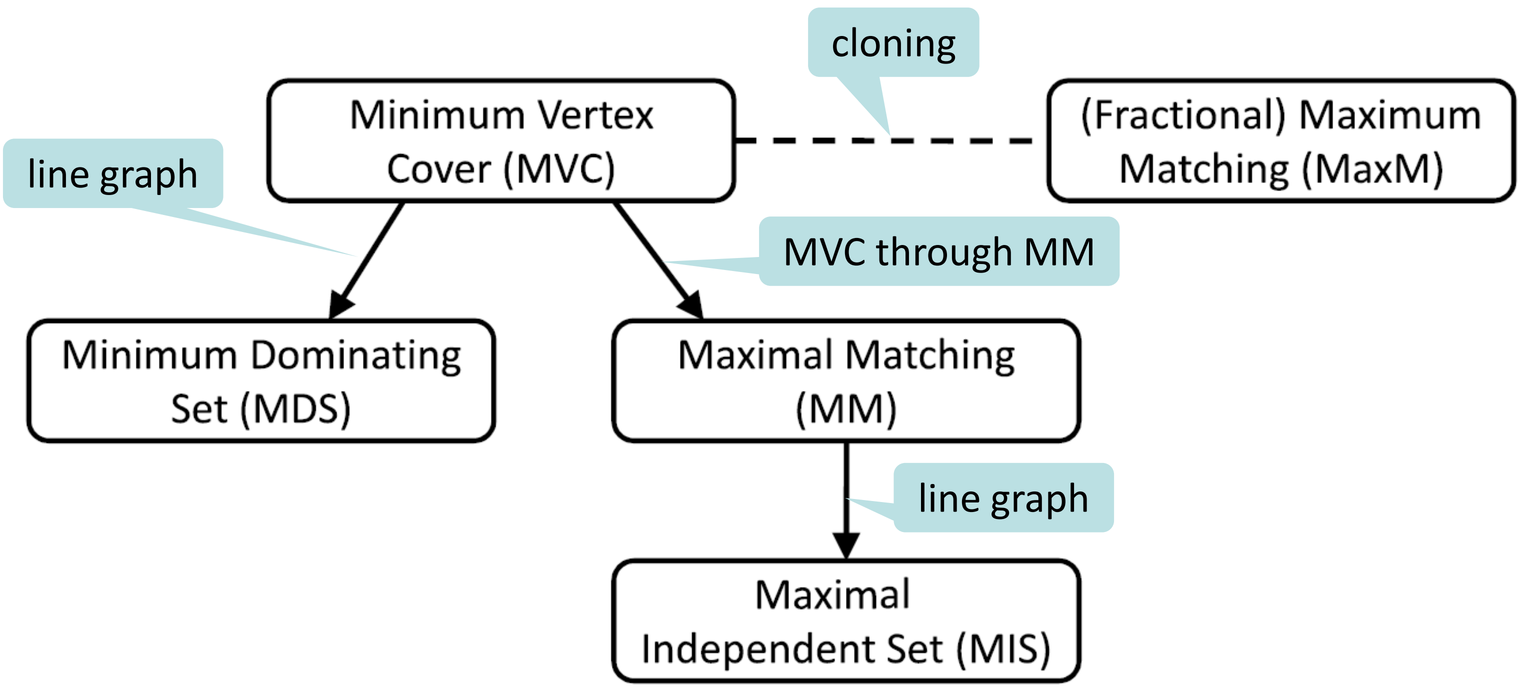
- Many "local looking" problems need non-trivial *t*, in other words, the bounds $\Omega(\log \Delta)$ and $\Omega(\sqrt{\log n})$ hold for a variety of classic problems.



[Kuhn, Moscibroda, W, journal version in submission]

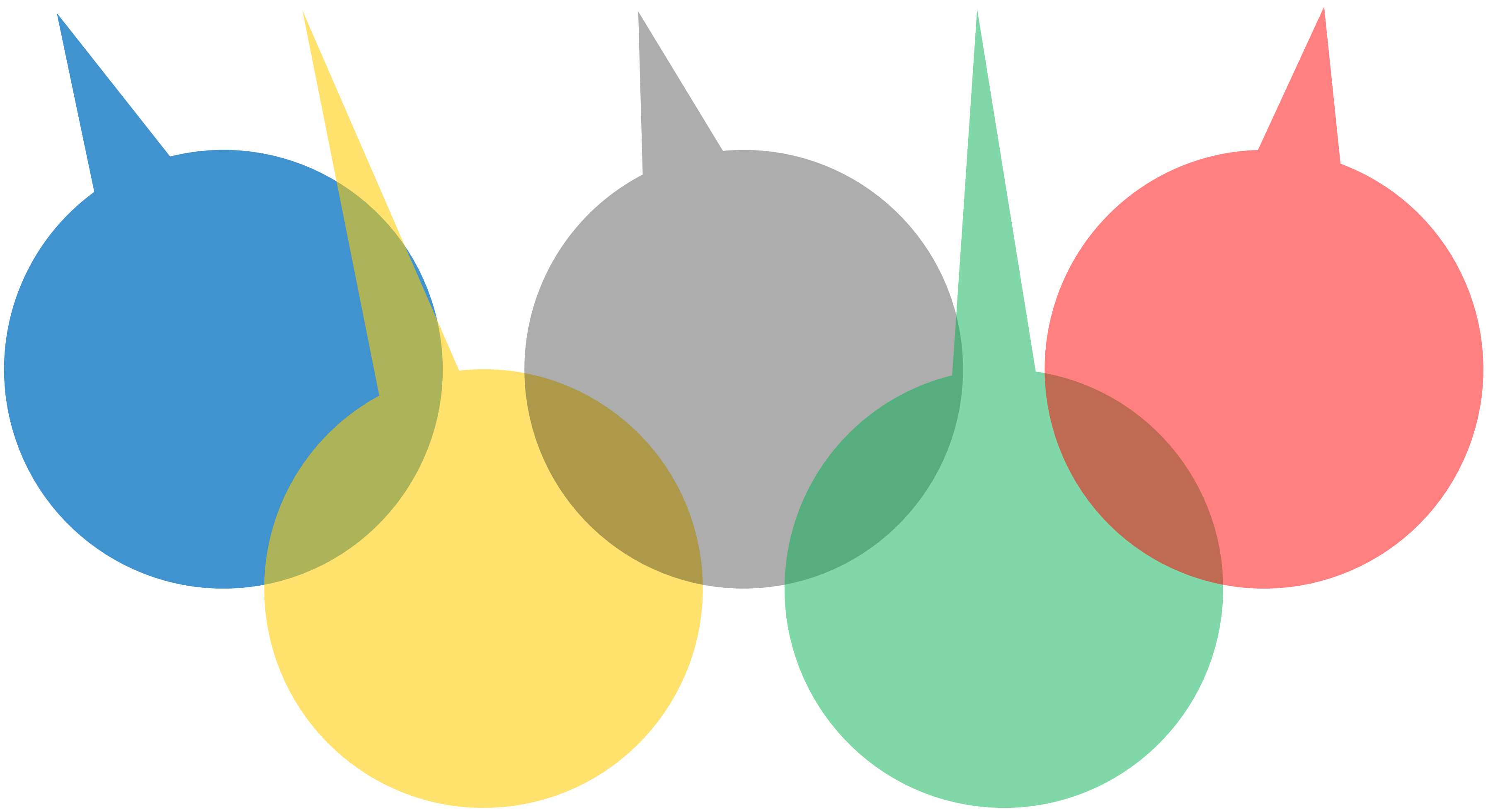
Lower Bound: Reductions

- Many "local looking" problems need non-trivial t, in other words, the bounds $\Omega(\log \Delta)$ and $\Omega(\sqrt{\log n})$ hold for a variety of classic problems.

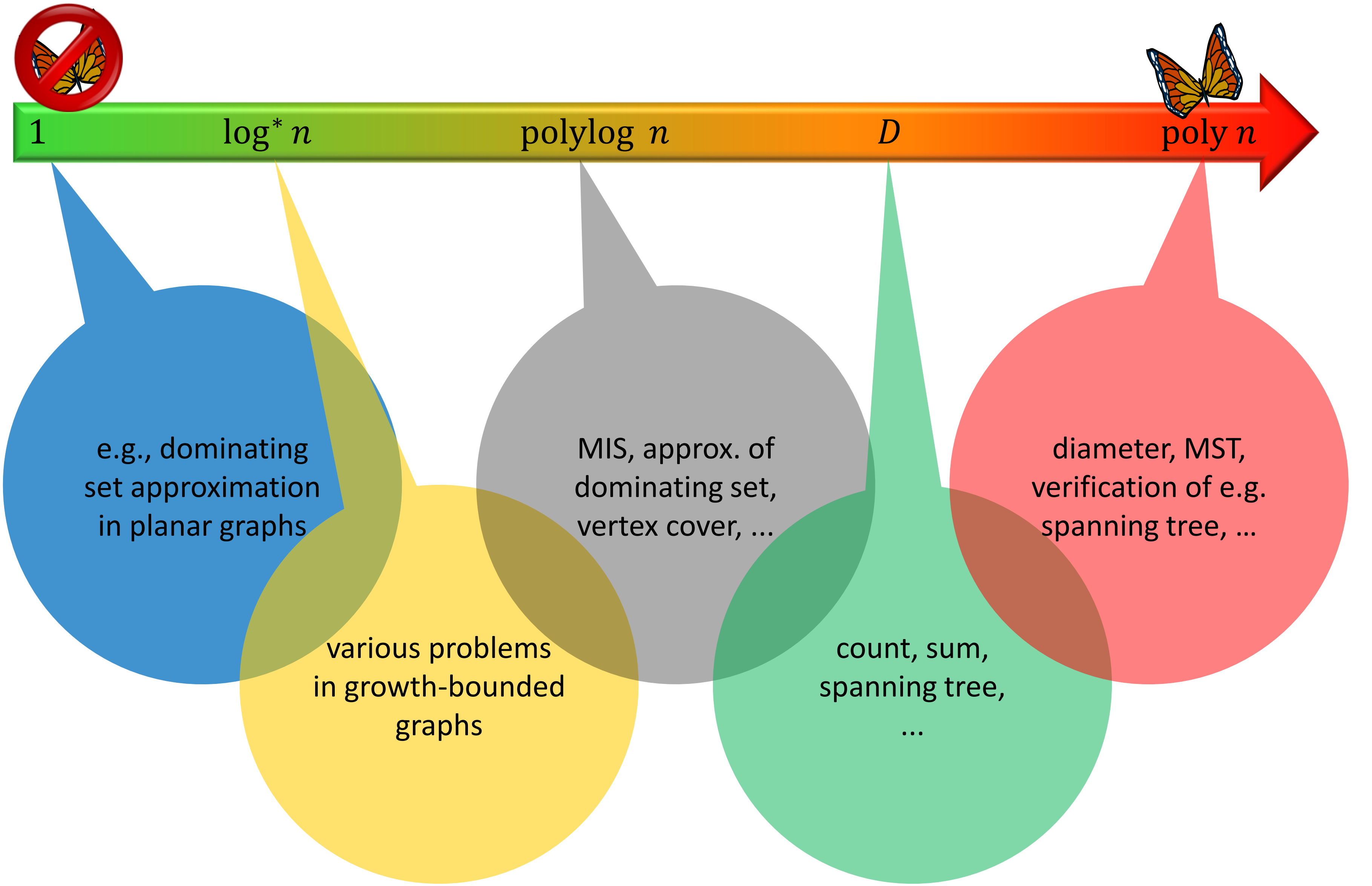


[Kuhn, Moscibroda, W, journal version in submission]

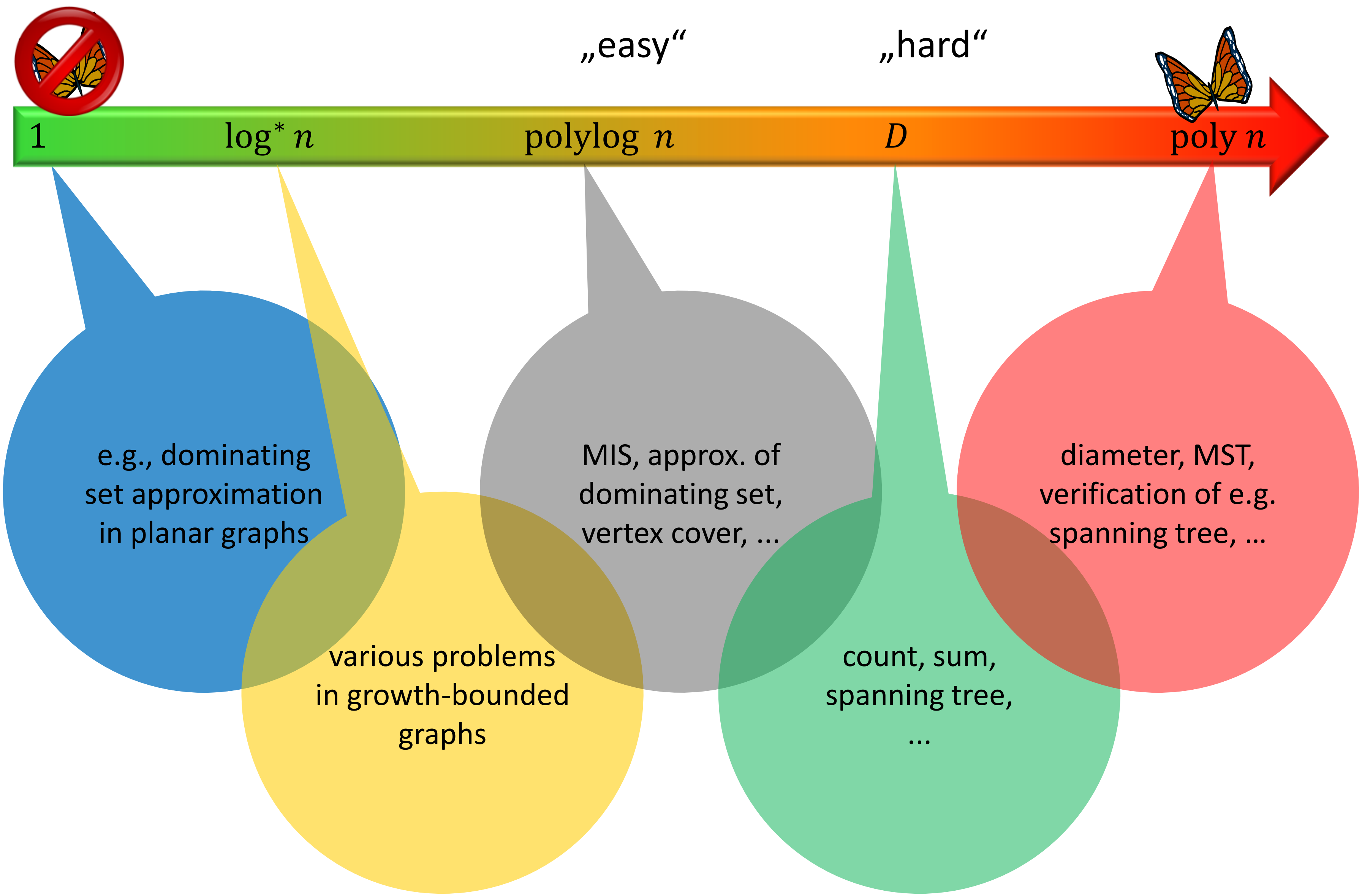
Olympics!



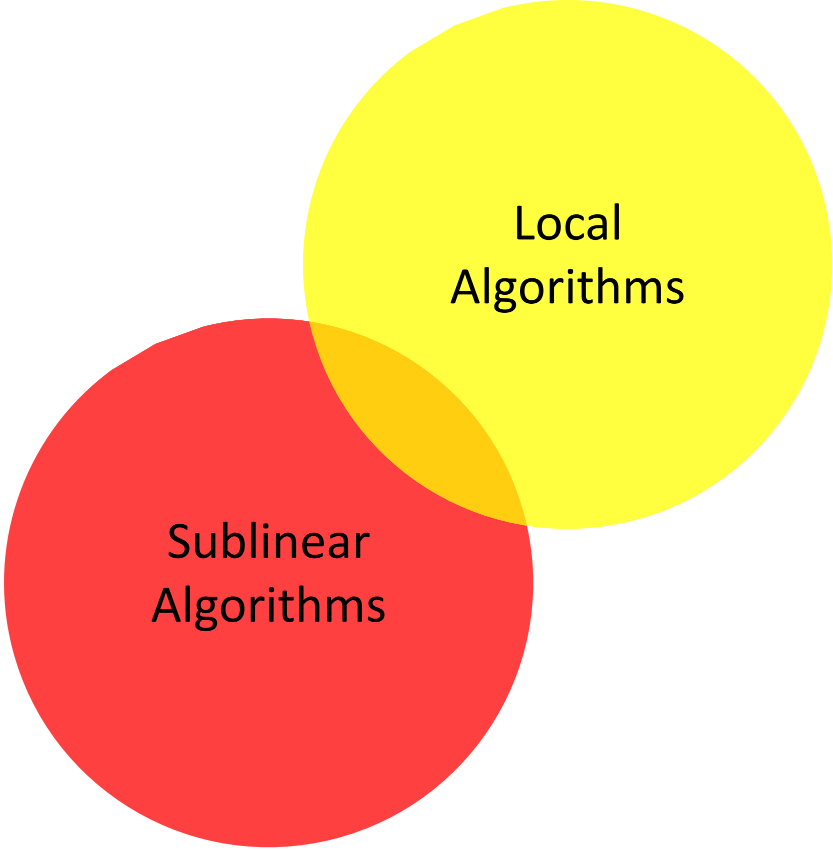
Distributed Complexity Classification



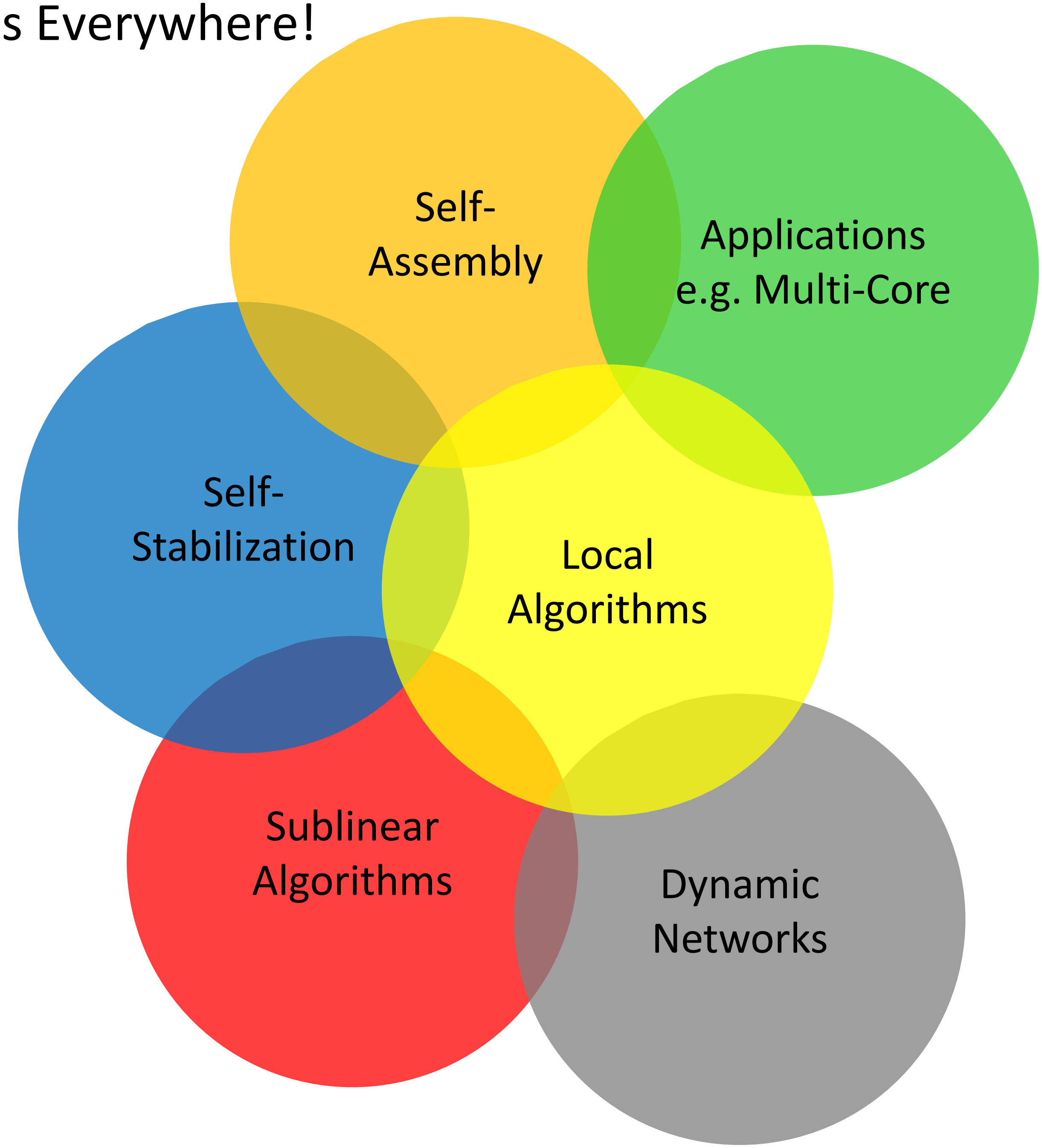
Distributed Complexity Classification



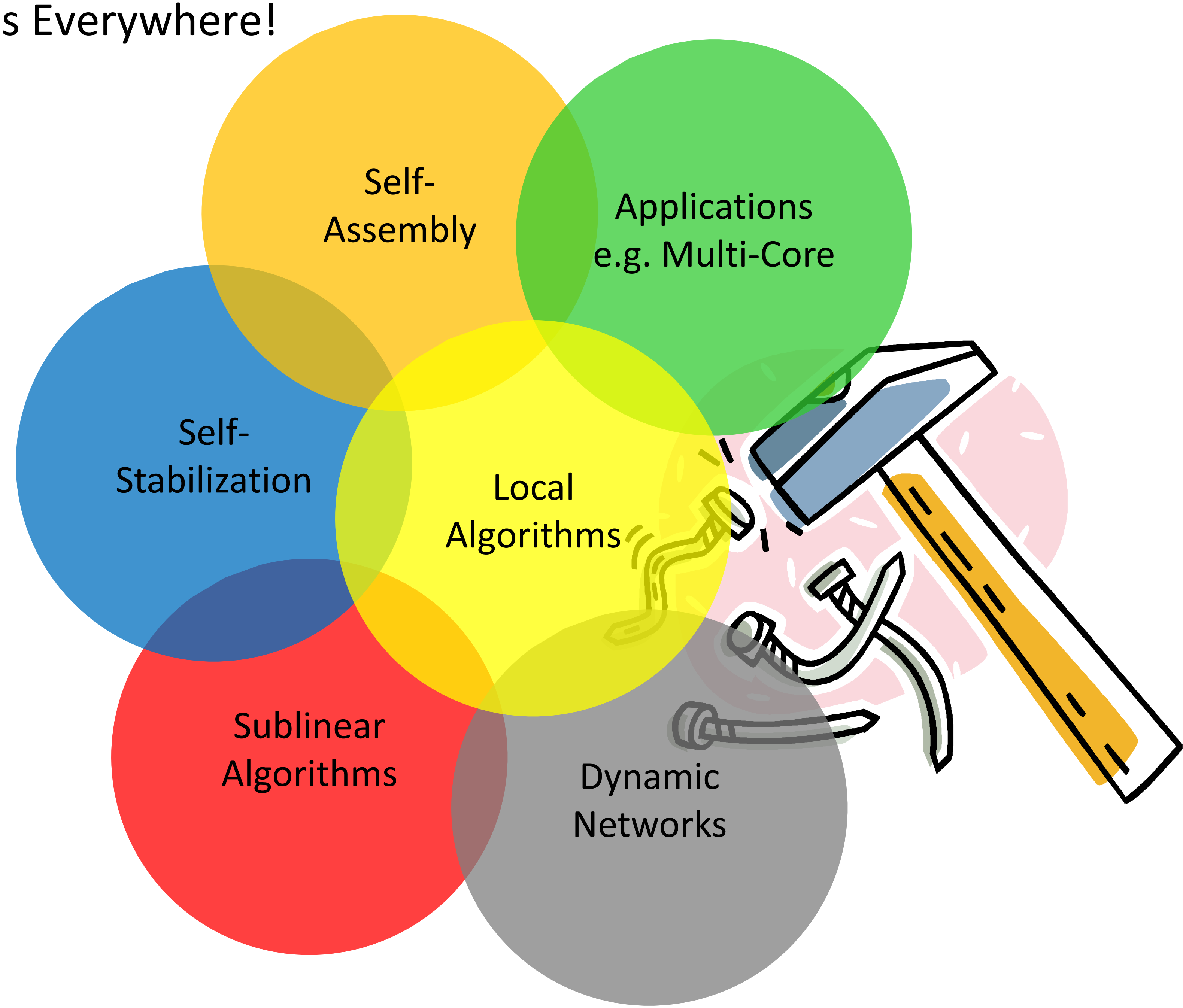
Locality

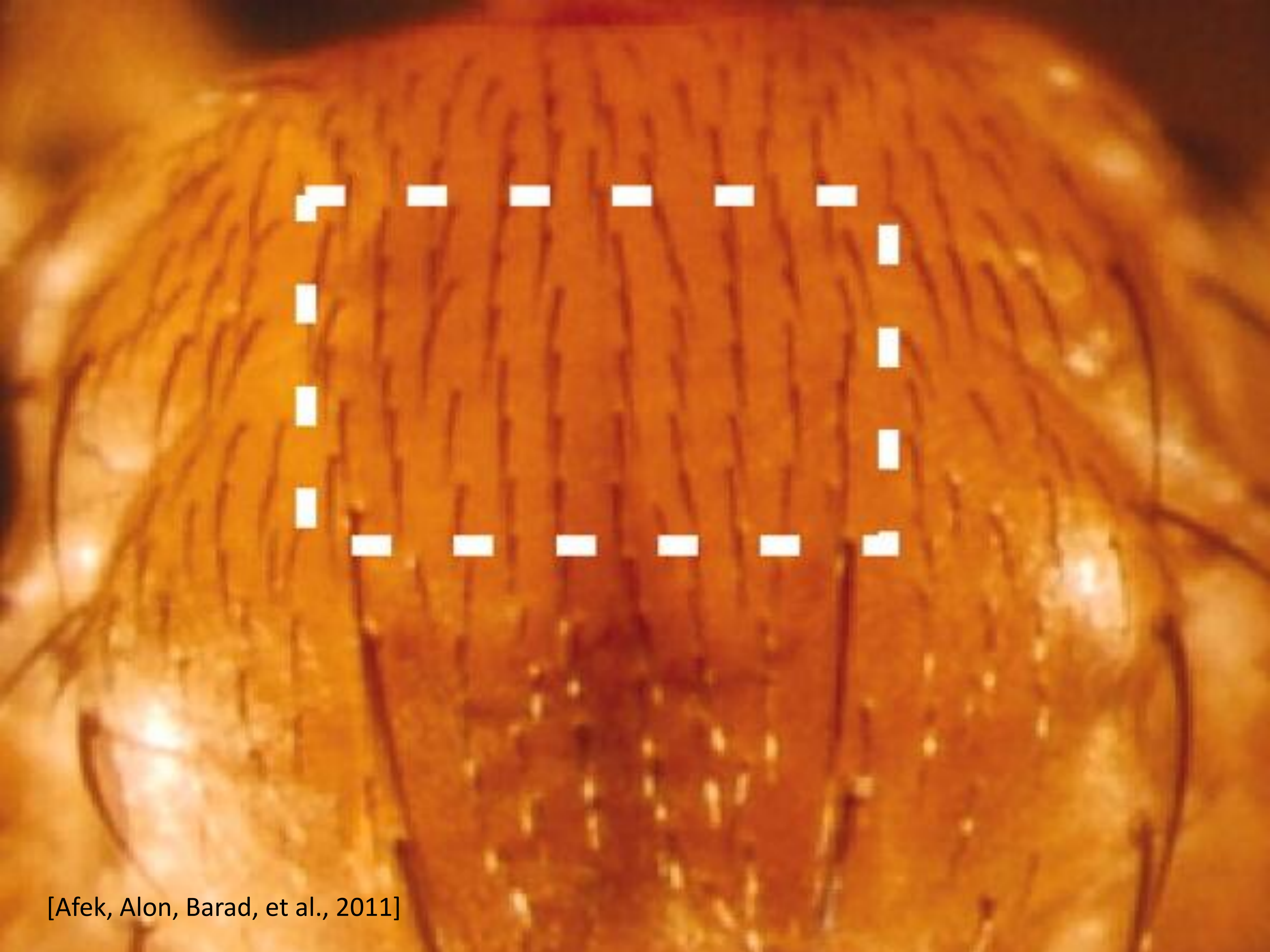


Locality is Everywhere!

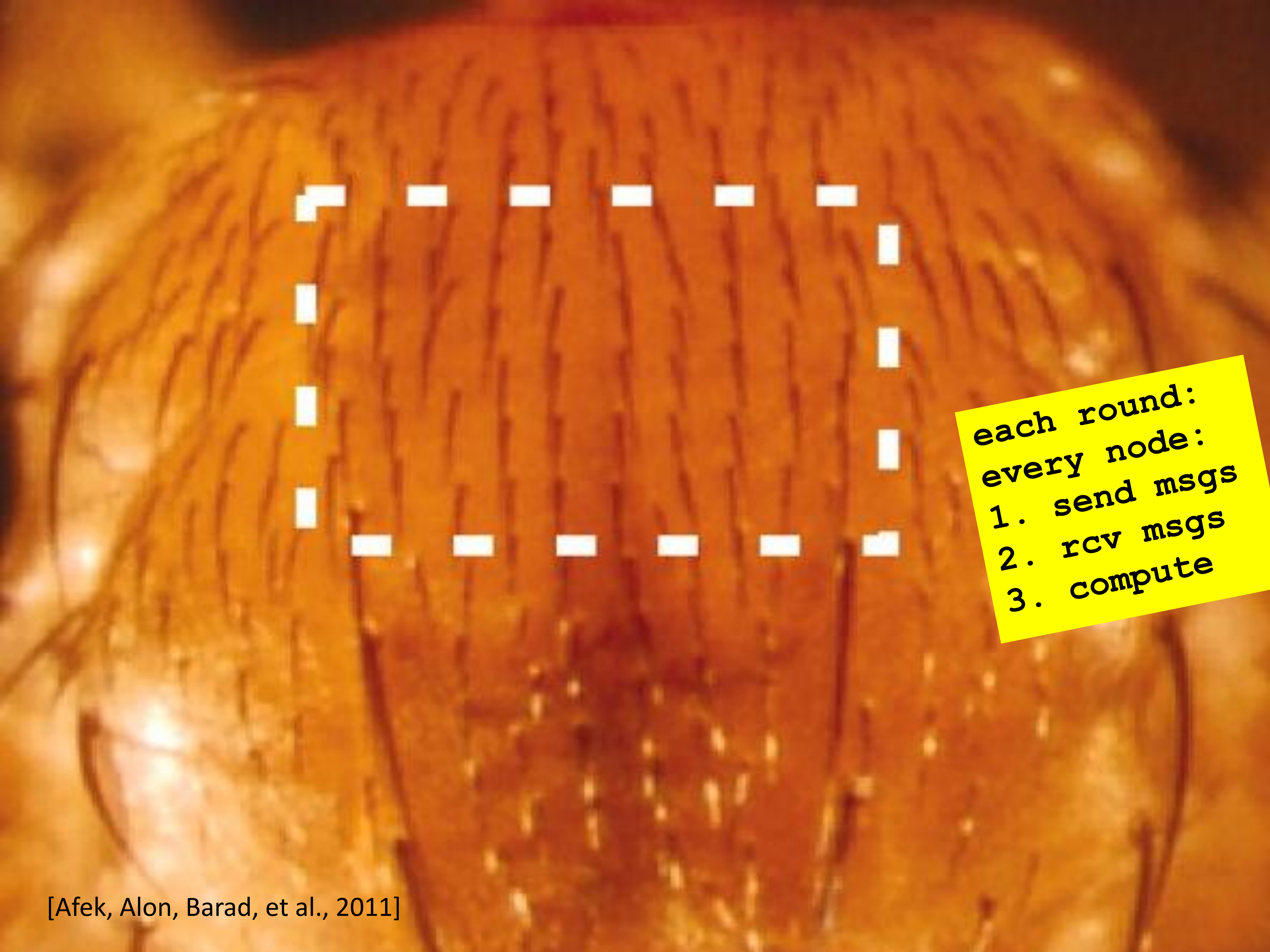


Locality is Everywhere!





[Afek, Alon, Barad, et al., 2011]



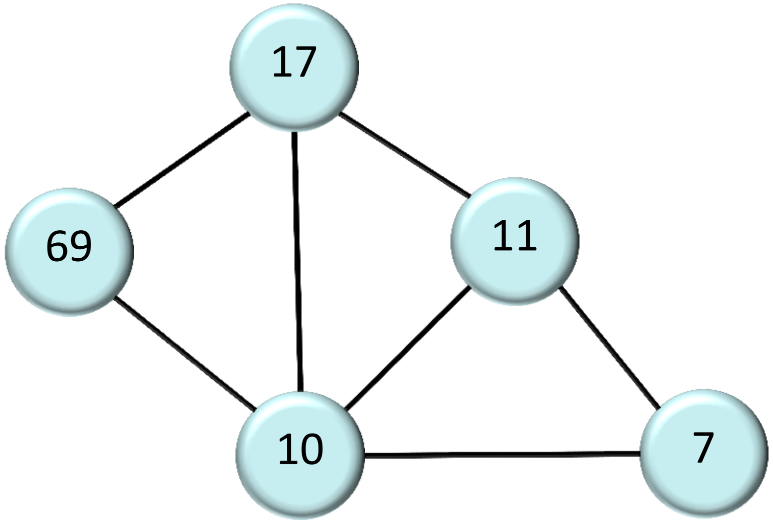
each round:
every node:
1. send msgs
2. rcv msgs
3. compute

[Afek, Alon, Barad, et al., 2011]


each round:
every node:
1. send msgs
2. rcv msgs
3. compute

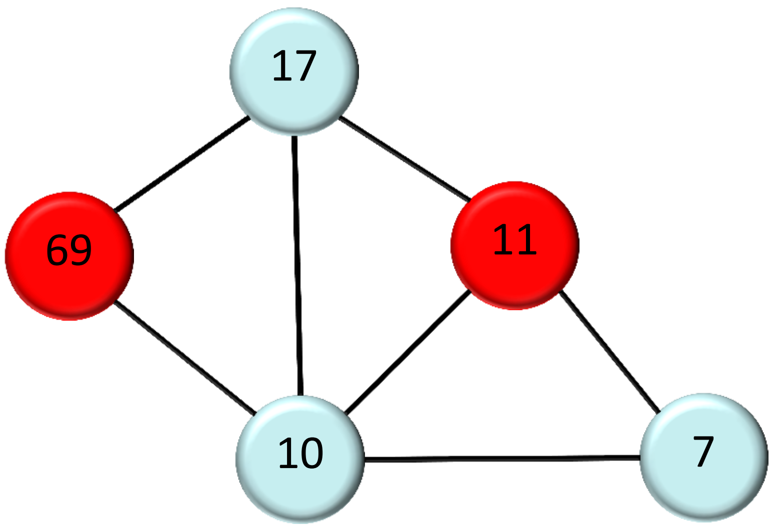
Maximal Independent Set (MIS)

- Given a network with *n* nodes, nodes have unique IDs.
- Find a Maximal Independent Set (MIS)
  - a non-extendable set of pair-wise non-adjacent nodes



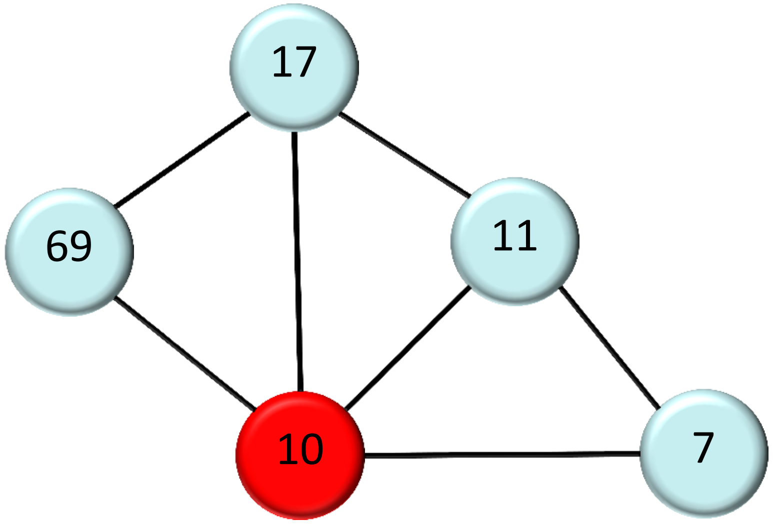
Maximal Independent Set (MIS)

- Given a network with *n* nodes, nodes have unique IDs.
- Find a Maximal Independent Set (MIS)
  - a non-extendable set of pair-wise non-adjacent nodes



Maximal Independent Set (MIS)

- Given a network with *n* nodes, nodes have unique IDs.
- Find a Maximal Independent Set (MIS)
  - a non-extendable set of pair-wise non-adjacent nodes



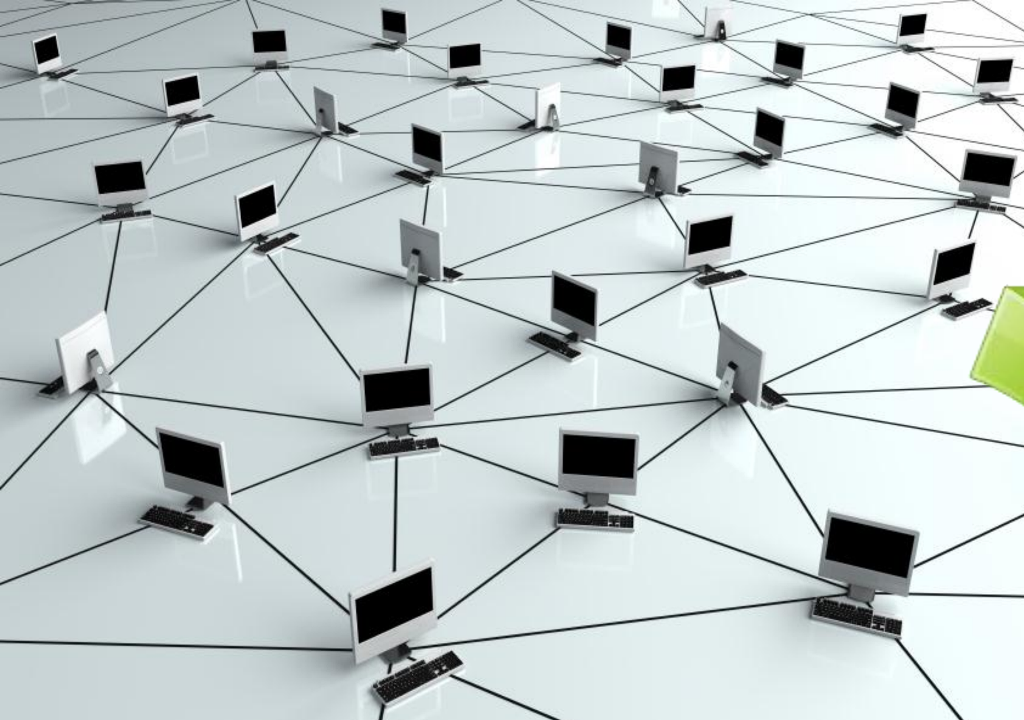
```
given: id, degree
synchronized while (true) {
    p = 1 /(2*degree);
    if (random value between 0 and 1 < p) {
        transmit "(degree, id)";
        …
```

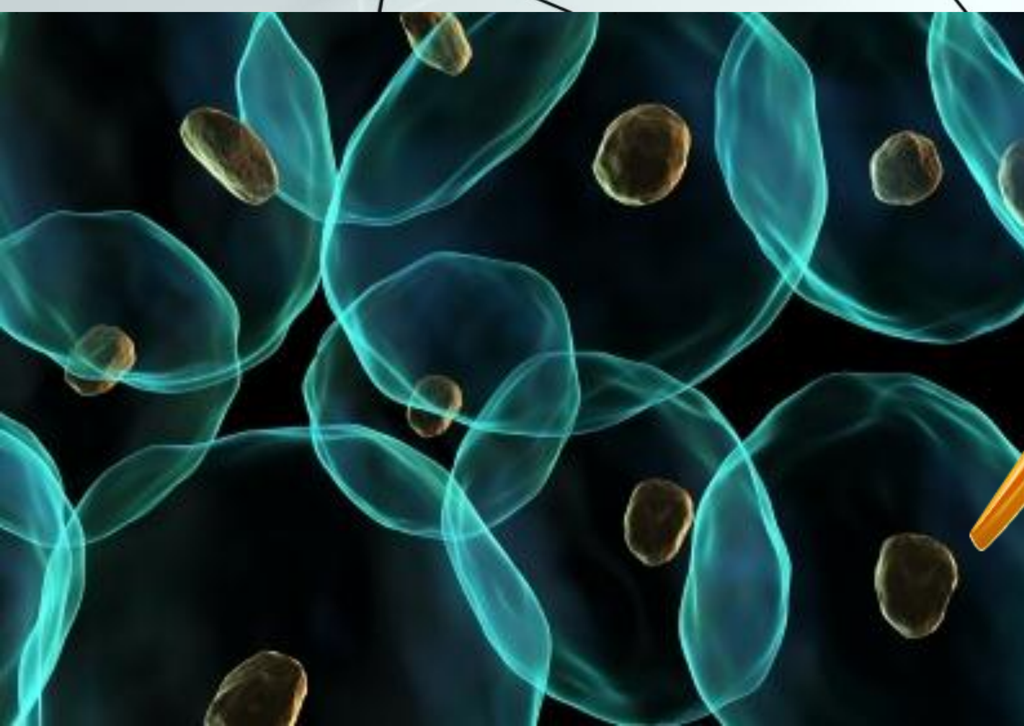
```
given: id, degree
synchronized while (true) {
    p = 1 /(2*degree);
    if (random value between 0 and 1 < p) {
        transmit "(degree, id)";
        …
```

?!

≪

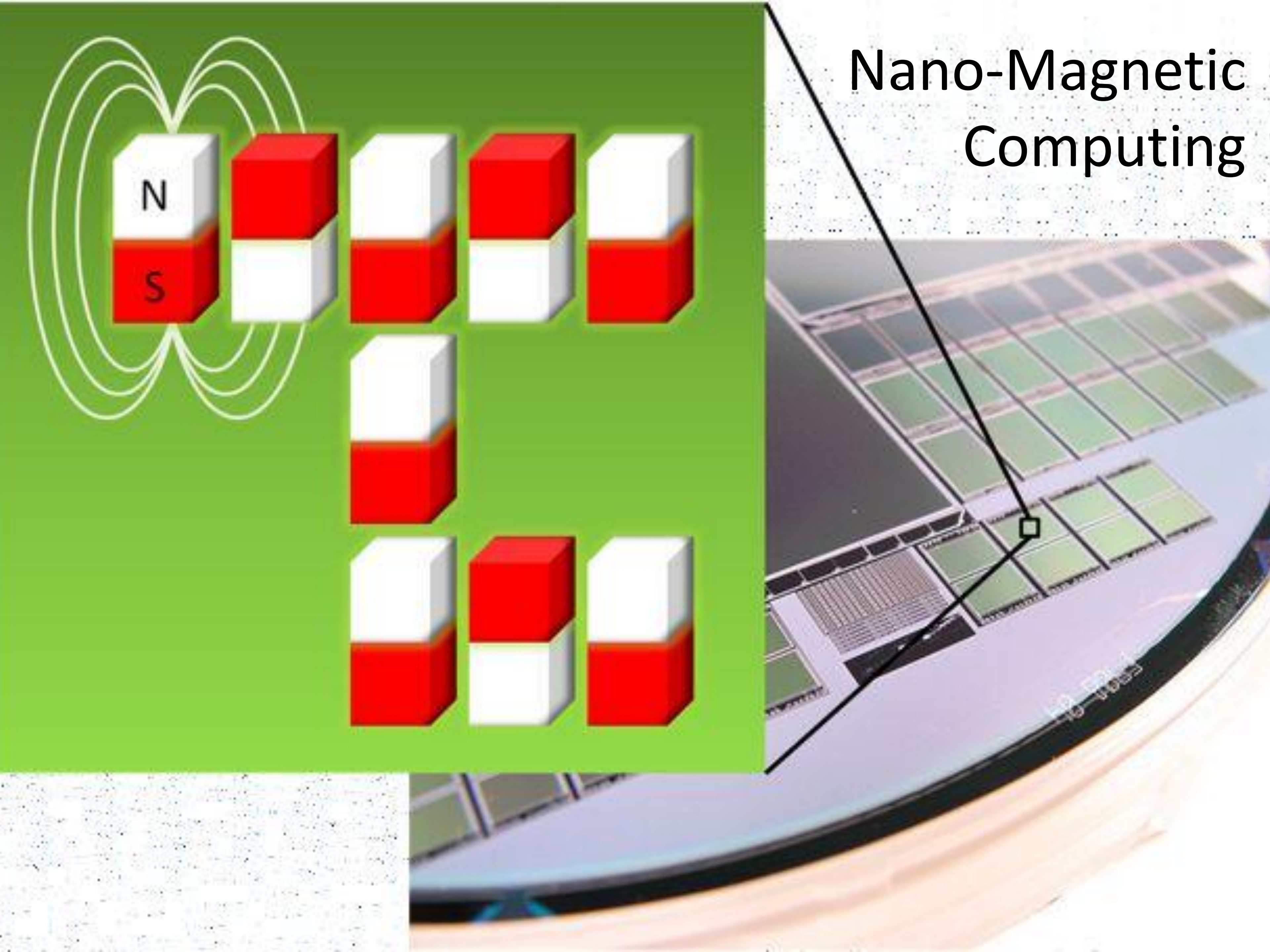
Distributed Computing
Without Computing!





each round:
every node:
1. send msgs
2. rcv msgs
3. compute

Nano-Magnetic Computing



Stone Age Distributed Computing

nFSM: networked Finite State Machine

- Every node is the same finite state machine, e.g. no IDs
- Apart from their state, nodes cannot store anything
- Nodes know nothing about the network, including e.g. their degree
- Nodes cannot explicitly send messages to selected neighbors, i.e. nodes can only implicitly communicate by changing their state
- Operation is asynchronous
- Randomized next state okay, as long as constant number
- Nodes cannot compute, e.g. cannot count

One, Two, Many Principle

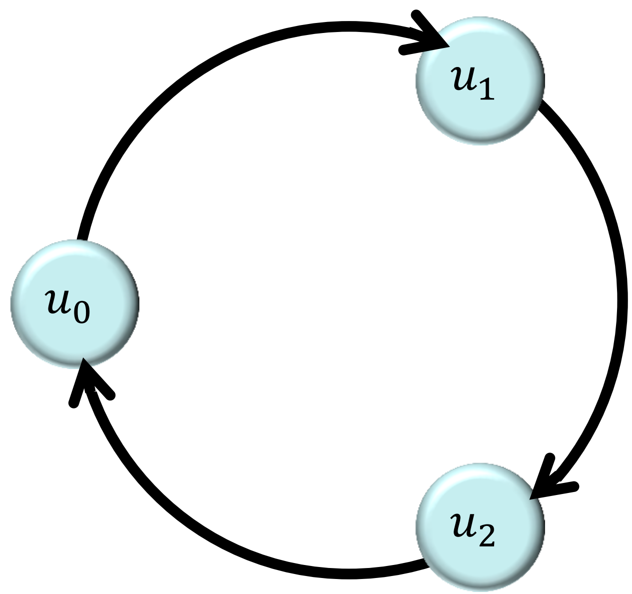
Piraha Walpiri

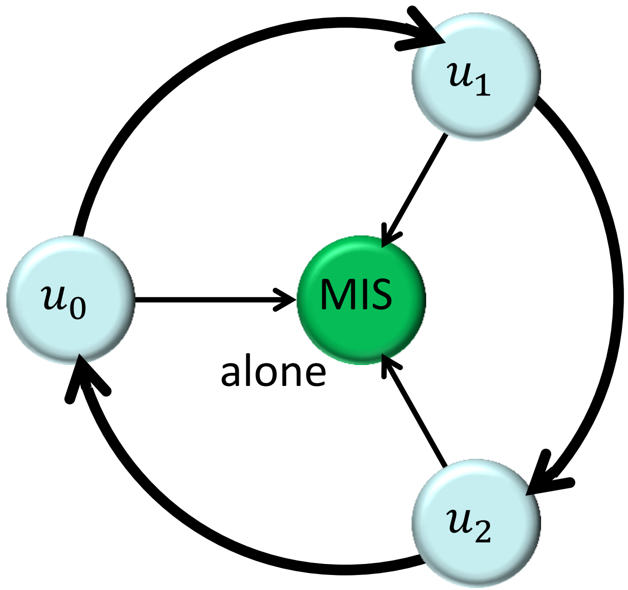
One, Two, Many Principle

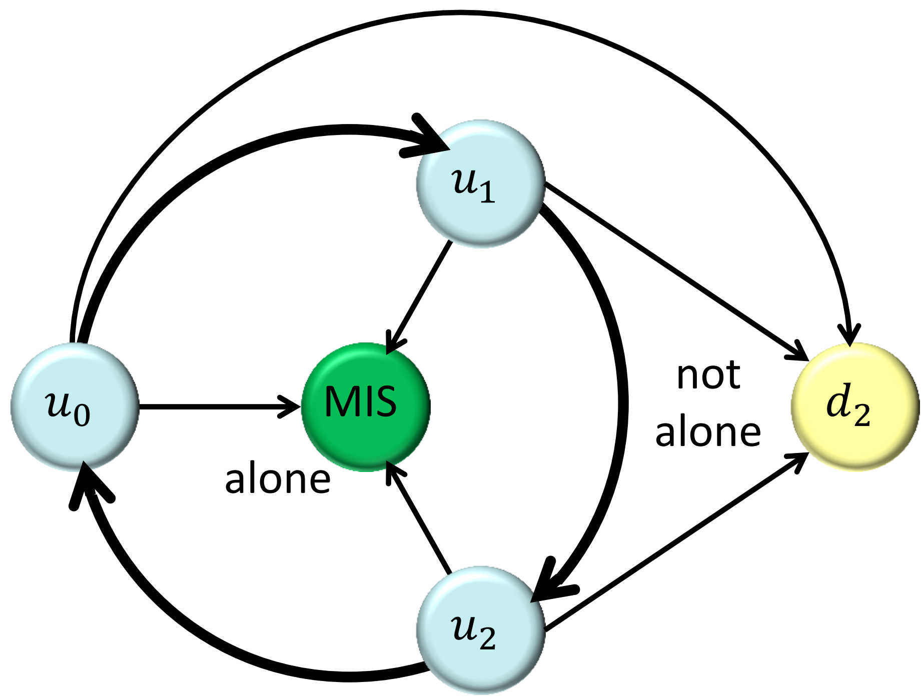
- Not okay
  - while (k < log n) {
  - At least half of neighbors in state s?
  - More neighbors in state s than in state t?

- Okay
  - No neighbor in state s?
  - Some neighbor in state s?
  - At most two neighbors in state s?

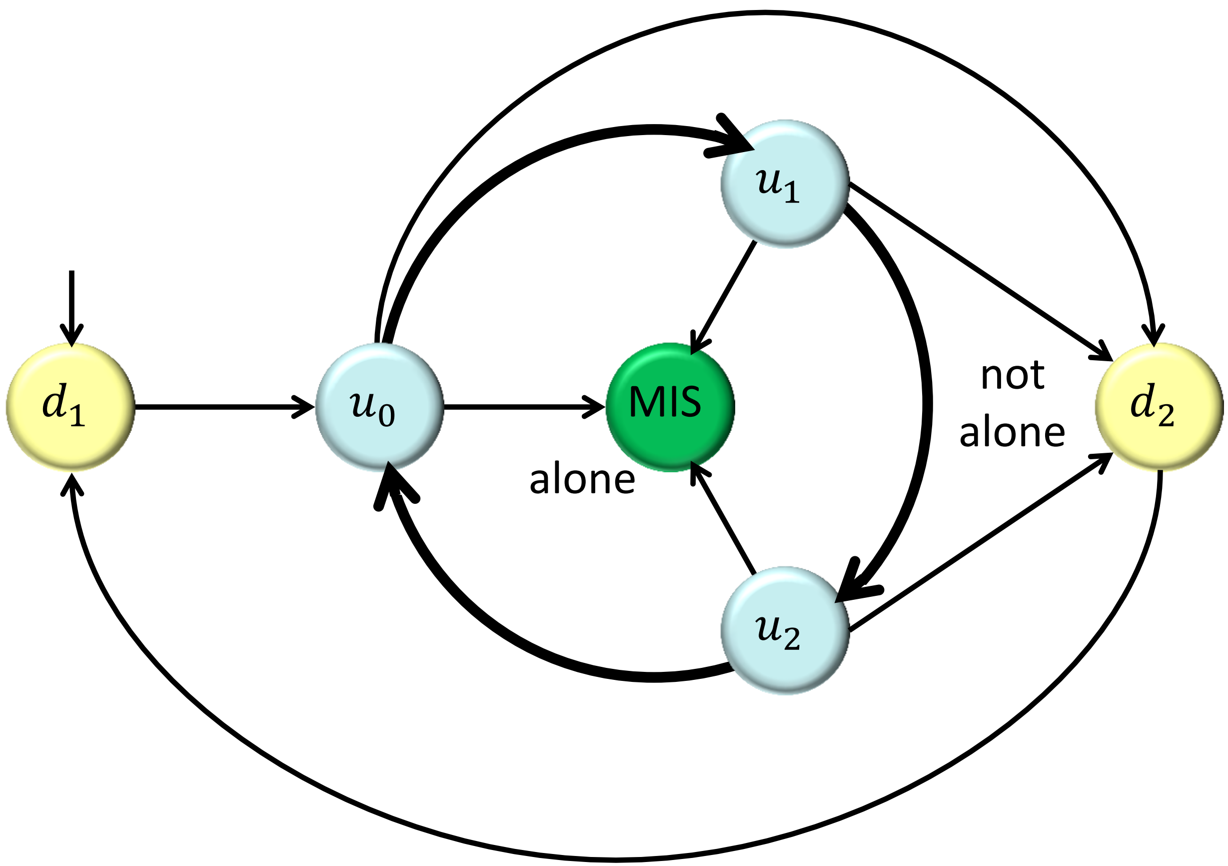
PRIMITIVE CULTURES DEVELOP SESAME STREET.


u_1
u_0
u_2

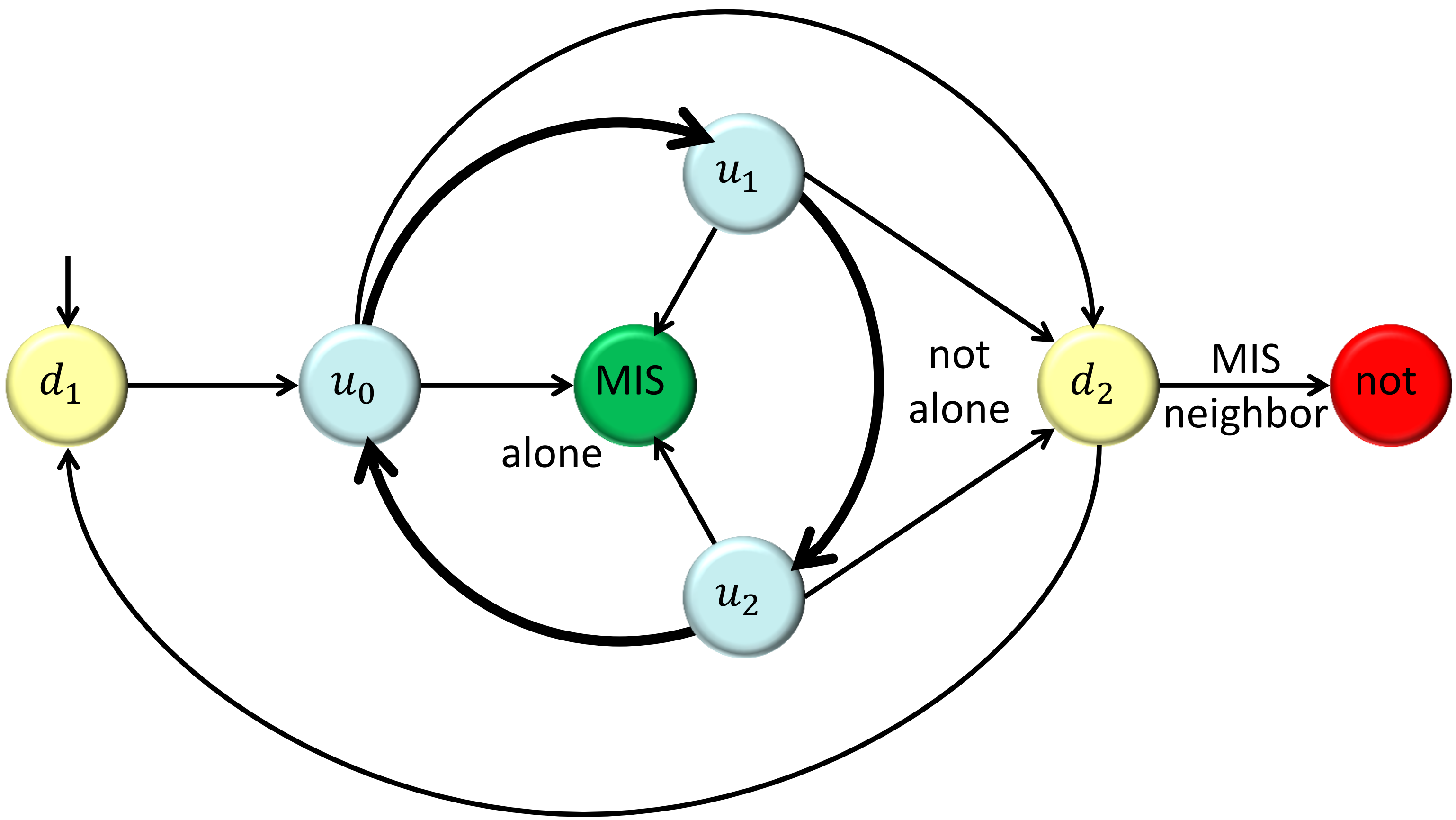

u_1
u_0
MIS
alone
u_2



u_1
u_0
MIS
alone
not alone
d_2
u_2

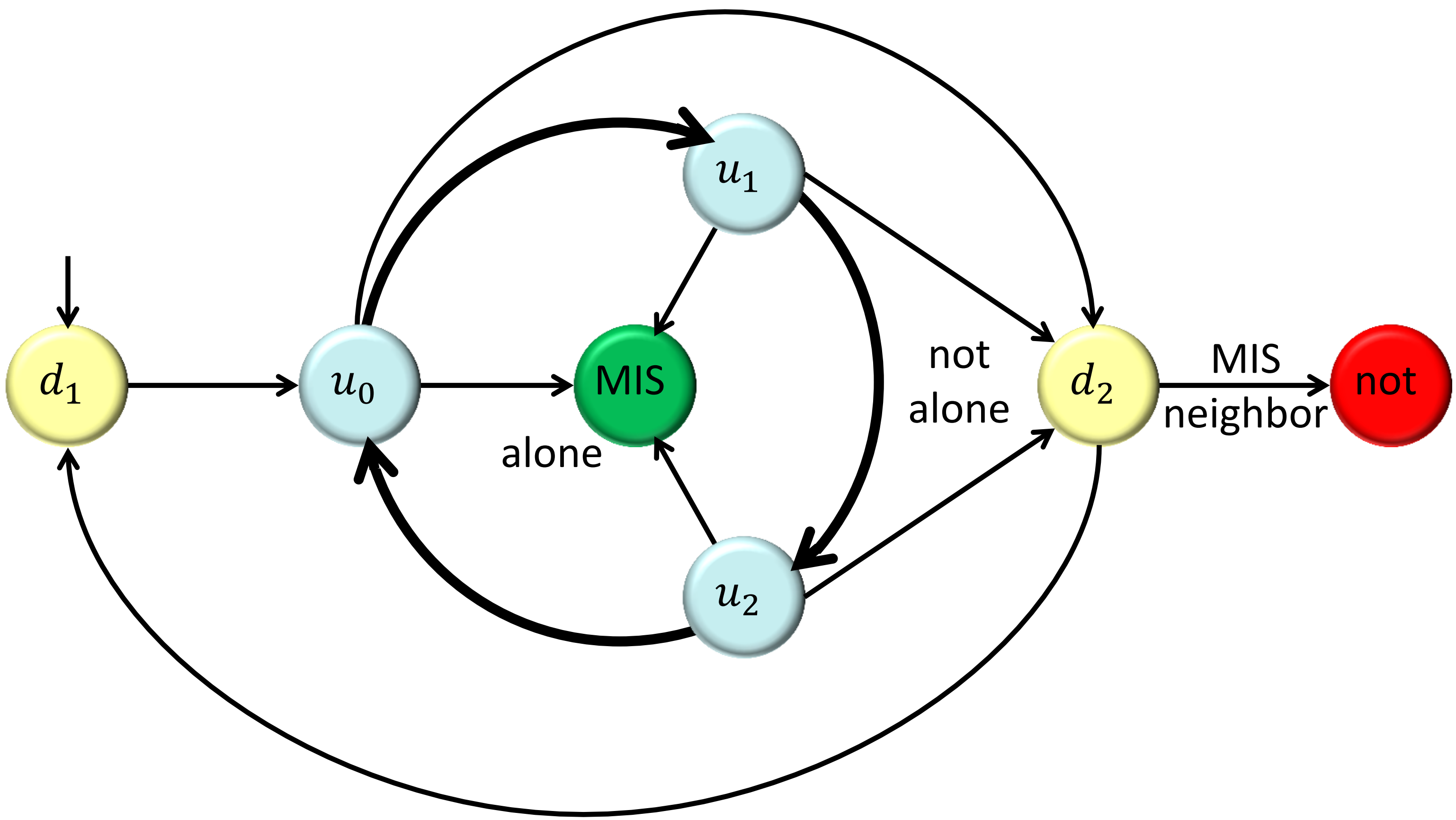


d_1
u_0
MIS
alone
u_1
u_2
not
alone
d_2



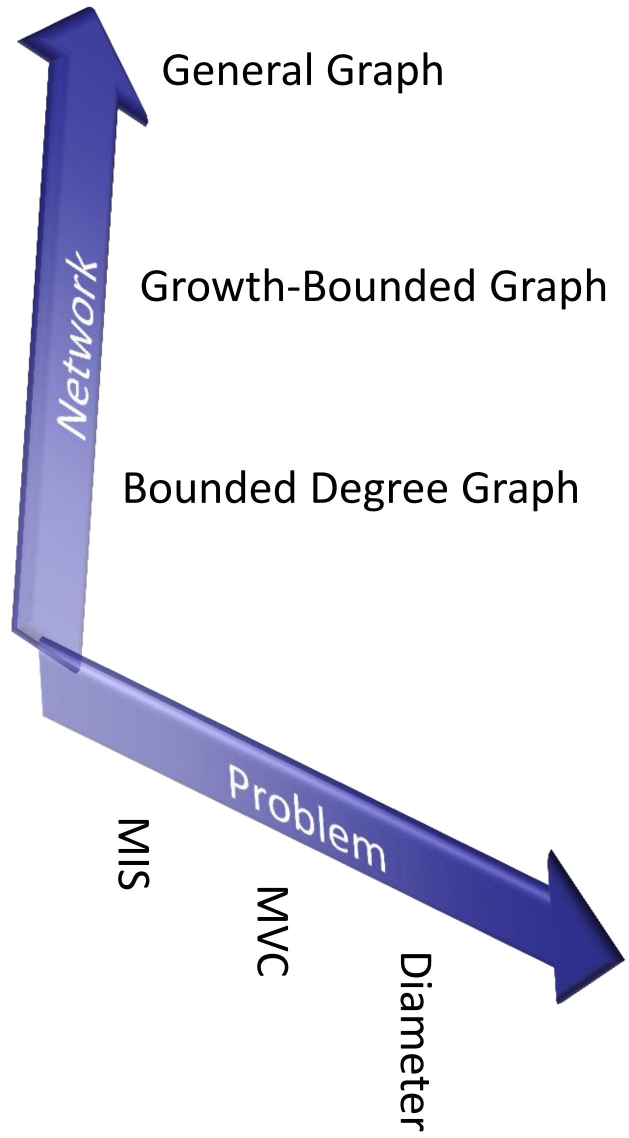
d_1
u_0
MIS
alone
u_1
u_2
not
alone
d_2
MIS
neighbor
not

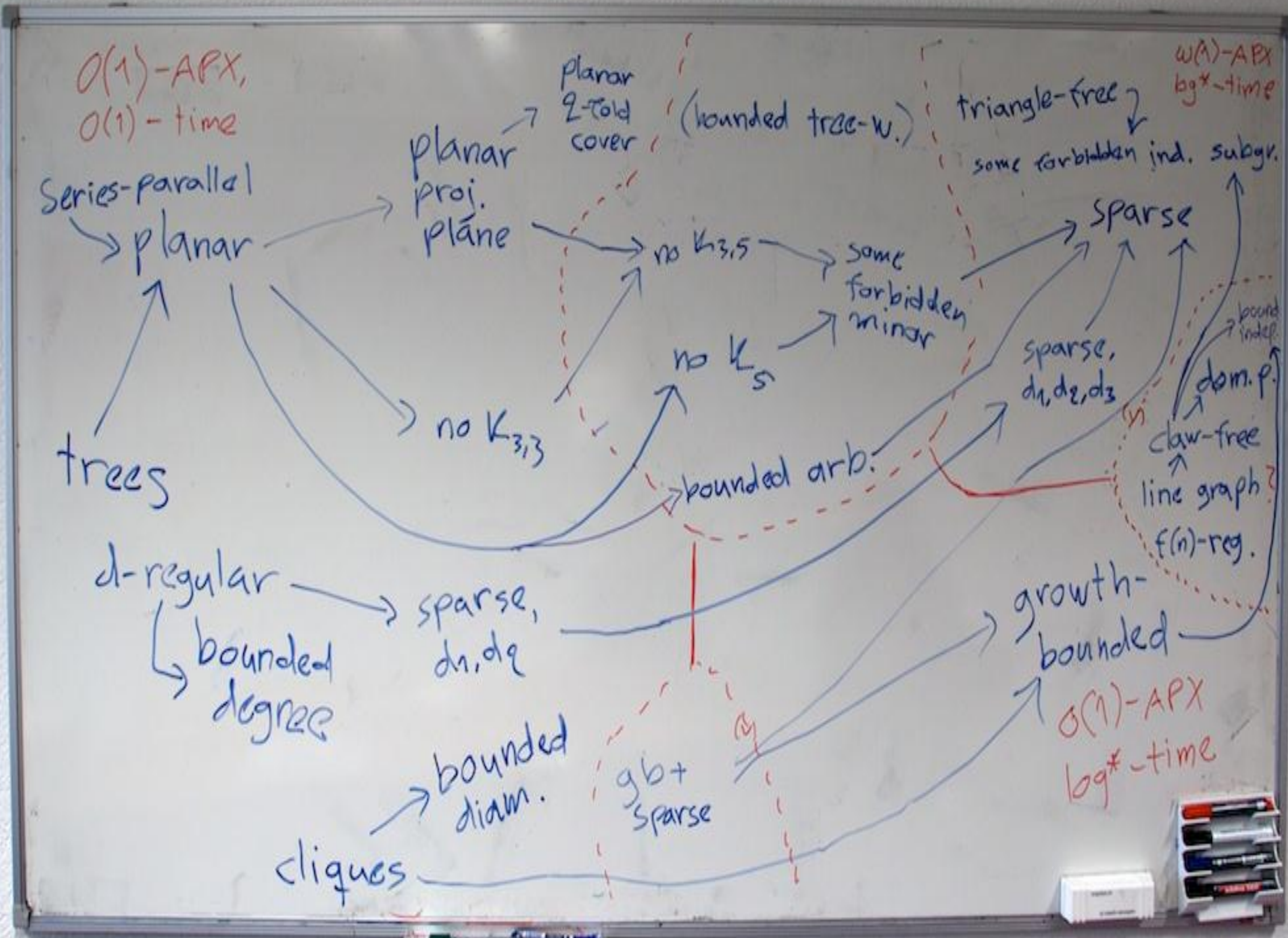
nFSM solves MIS whp in time $O(\log^2 n)$



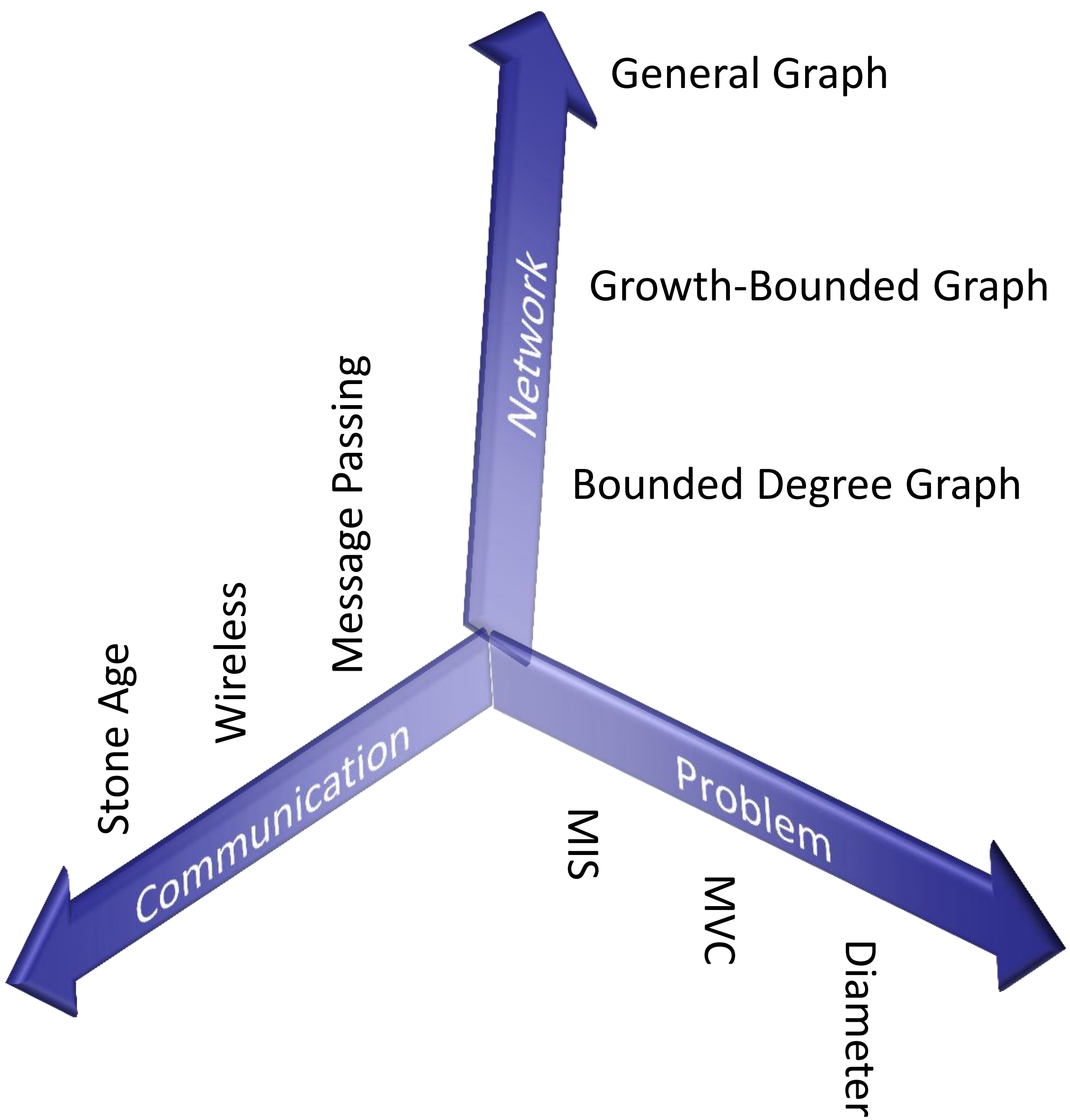
[Emek, Smula, W, in submission]

Overview

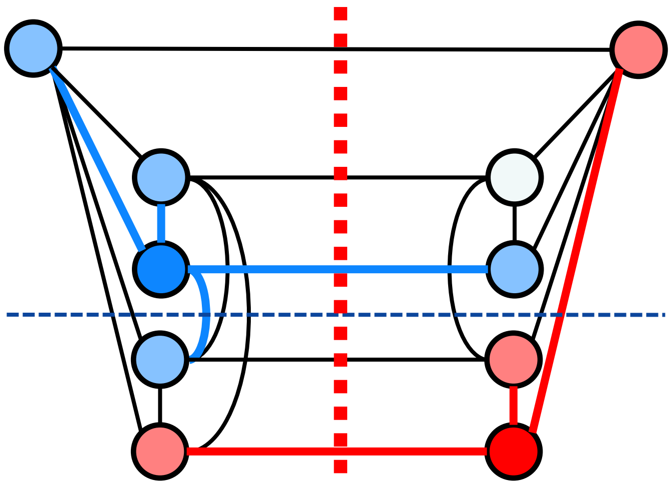

General Graph
Growth-Bounded Graph
Bounded Degree Graph
Network
Problem
MIS
MVC
Diameter

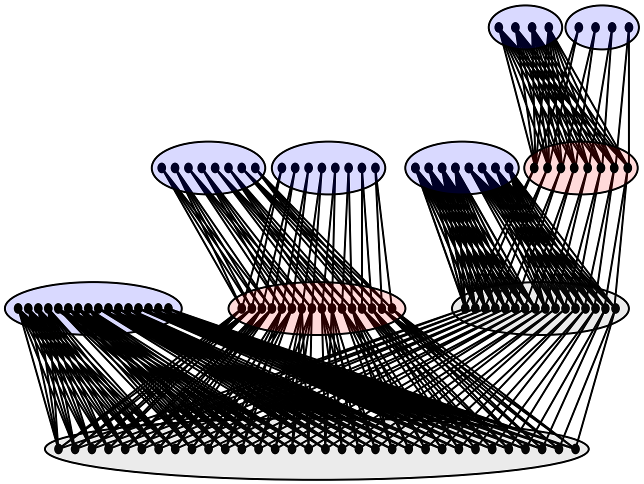

O(1)-APX,
O(1)-time
planar
2-fold
cover
(bounded tree-w.)
triangle-free
ω(1)-APX
log*-time
planar
proj.
plane
Series-parallel
planar
some forbidden ind. subgr.
sparse
no K3,5
some
forbidden
minor
no K5
no K3,3
trees
sparse,
d1,d2,d3
dom. p.
claw-free
line graph?
f(n)-reg.
bounded arb.
d-regular
sparse,
d1,d2
bounded
degree
growth-
bounded
O(1)-APX
log*-time
bounded
diam.
gb+
sparse
cliques

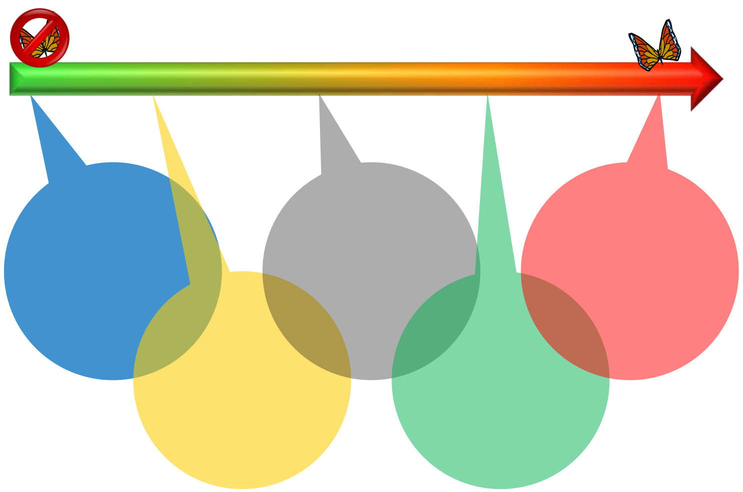
Overview


General Graph
Growth-Bounded Graph
Bounded Degree Graph
Network
Message Passing
Wireless
Stone Age
Communication
Problem
MIS
MVC
Diameter

Summary







Thank You!

Questions & Comments?

Thanks to my co-authors
Yuval Emek
Silvio Frischknecht
Stephan Holzer
Fabian Kuhn
Thomas Moscibroda
Jasmin Smula

www.disco.ethz.ch