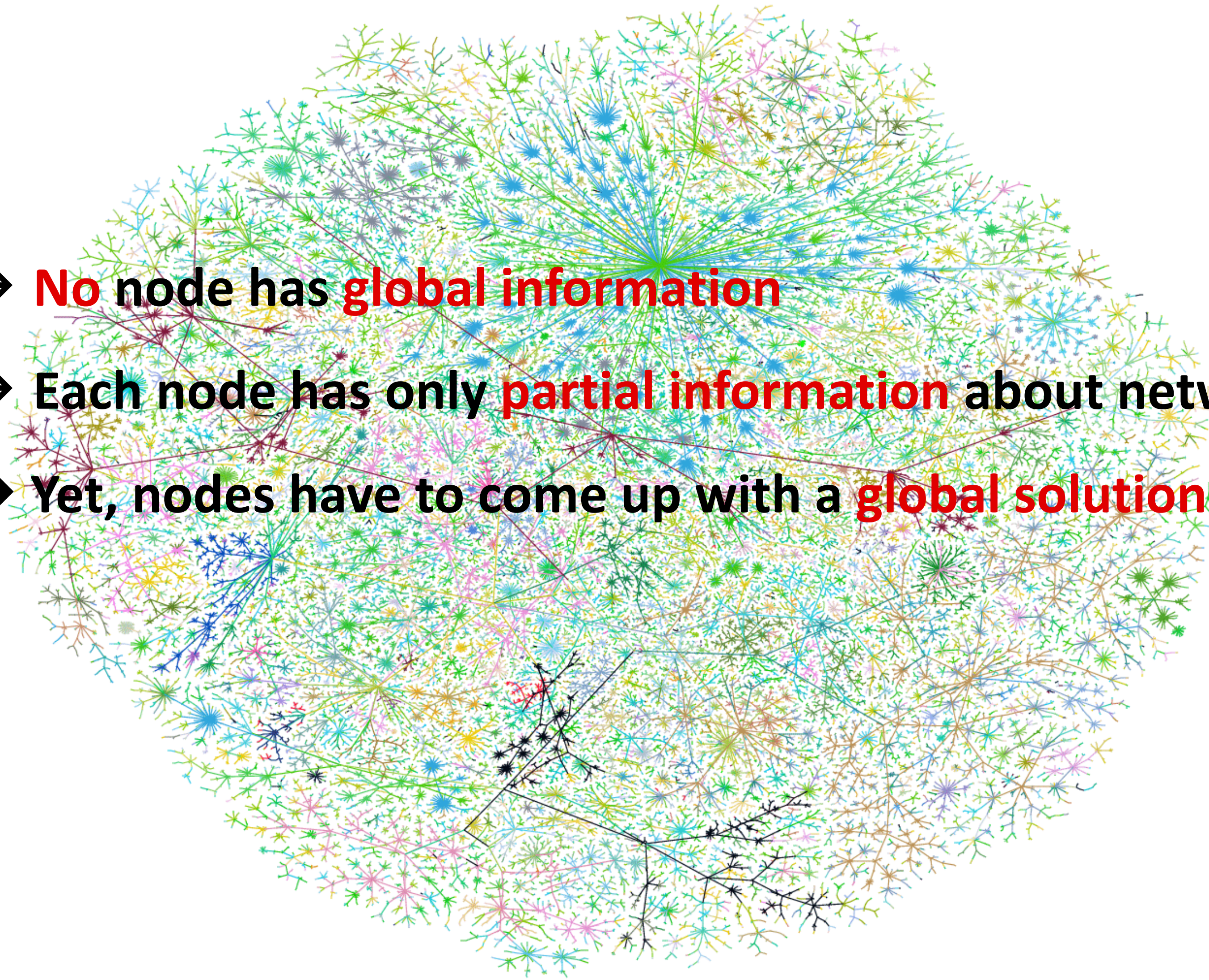
Global Solutions Based on Local Information

Fabian Kuhn

University of Freiburg, Germany

Distr. Computations in Large Networks



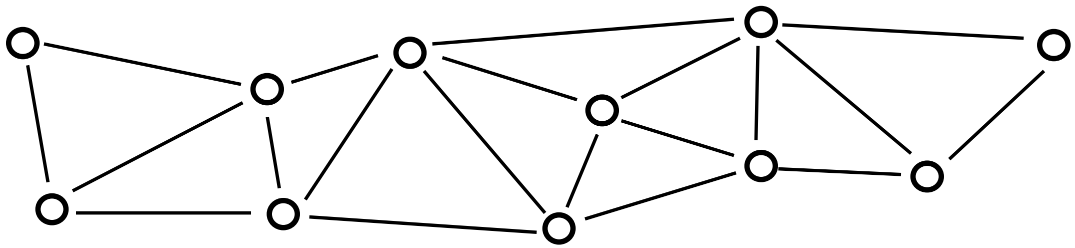
→ **No** node has **global information**

→ Each node has only **partial information** about network

→ Yet, nodes have to come up with a **global solution**!

General Problem

- Given: Network = Graph G



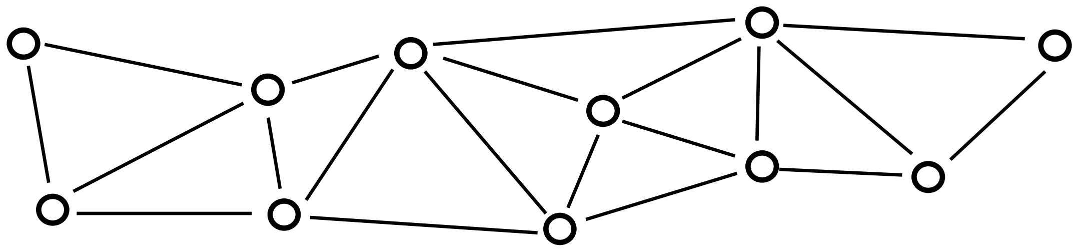
- Large, (possibly dynamic) network:
  Global view of whole network is not possible

- Computations have to be done at the nodes of the network

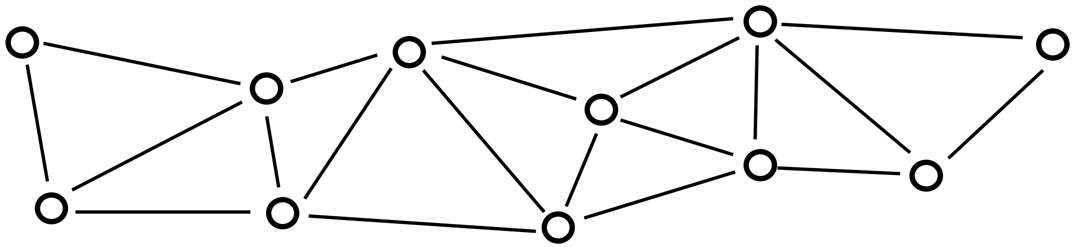
- Goal: Solve some given graph-theoretic problem on G by a distributed algorithm

Specific Problems

- Maximal Independent Set (MIS)
  Independent set $S \subseteq V$, s.t. $\forall v \in V \setminus S$, some neighbor of v is in S.



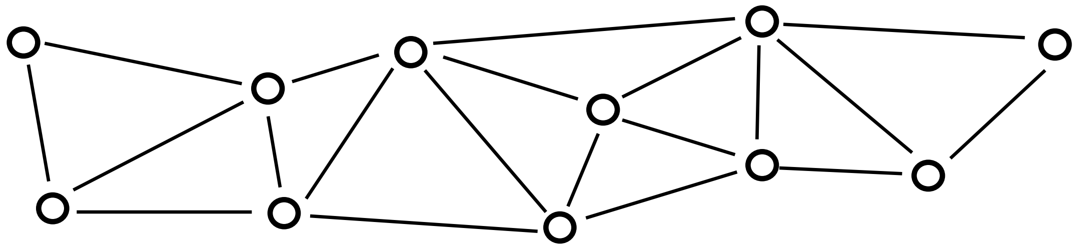
- Vertex Coloring:
  Properly color nodes with few colors



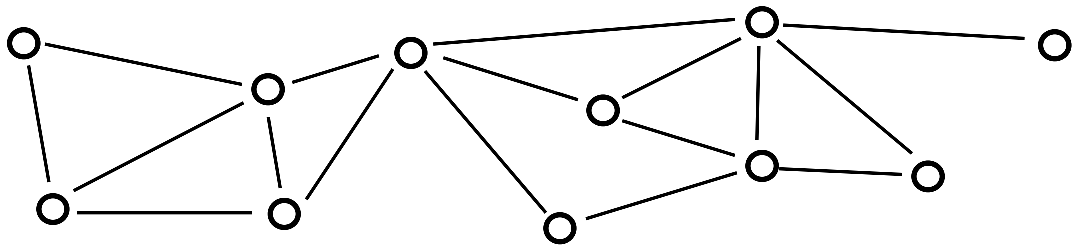
- Or more general graph labelings

Optimization Problems

- Minimum Dominating Set (MDS)
  Minimum $S \subseteq V$, s.t. $\forall v \in V$: $v \in S$ or v has at least one neighbor in S



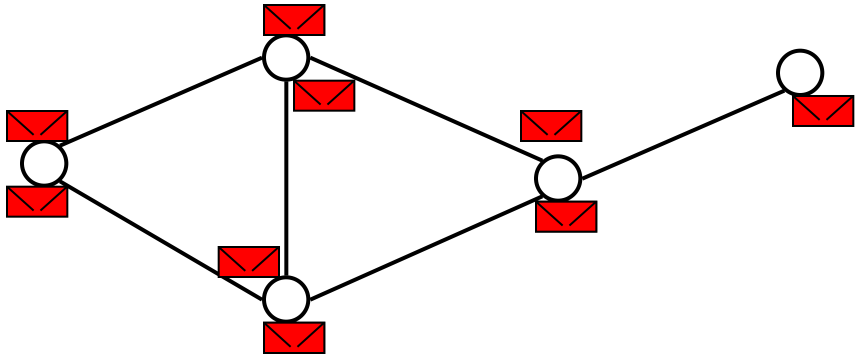
- Minimum Vertex Cover
  Minimum $S \subseteq V$, s.t. $\forall \{u, v\} \in E$, $S \cap \{u, v\} \neq \emptyset$



Communication Model

- Synchronous message passing model
- Network = graph (nodes: devic… comm. links)
- Node have unique IDs
- Time is divided into rounds:

$\mathcal{LOCAL}$ model:
Message size & local resources unbounded



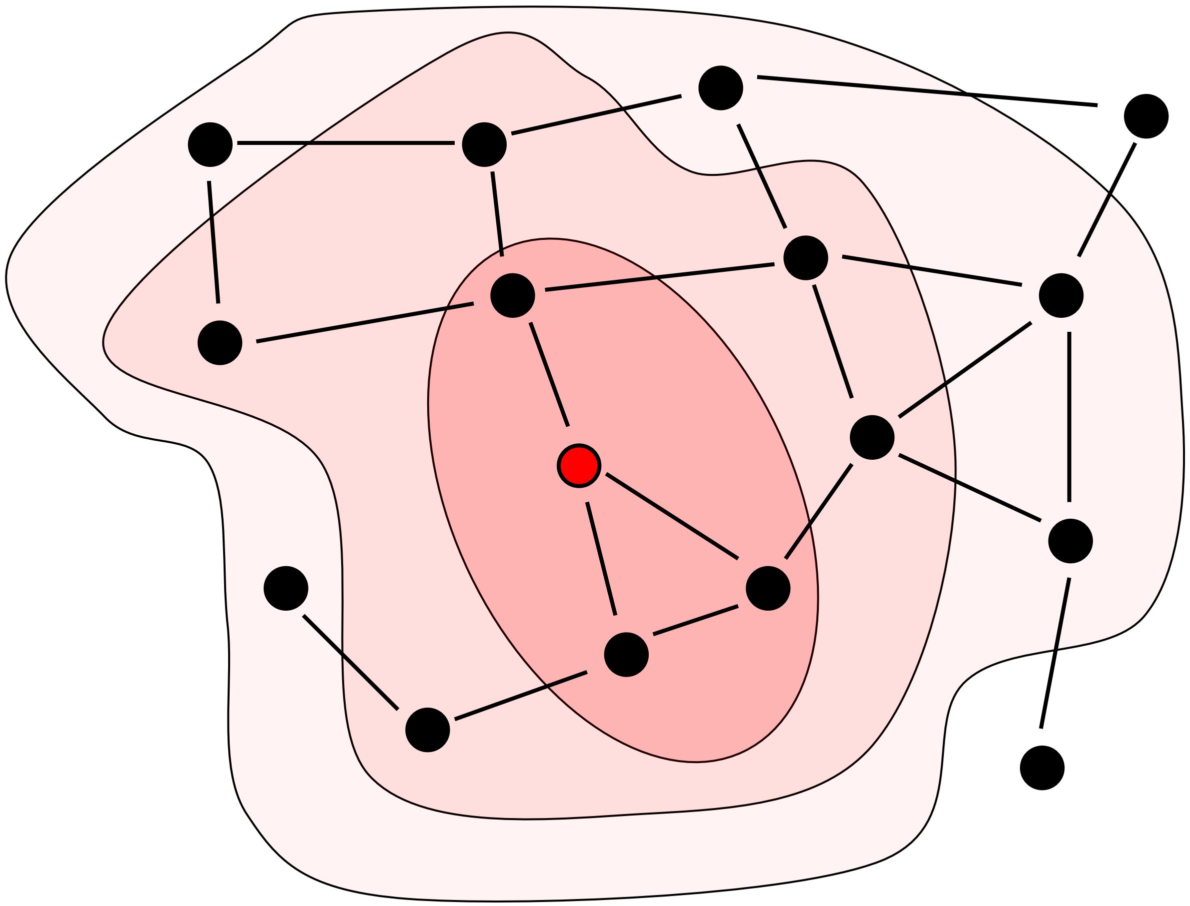
Each node sends message to each of its neighbors

time complexity = number of rounds

Locality

- What does this have to do with locality?

3 rounds



Locality

Observation:

- In r rounds of communication (time r), every node can collect information about r-neighborhood

- No bound on message size:

  every node can learn its
  complete r-neighborhood in r rounds

  - and nothing more...

Distributed Alg.: Alternative View

- For **r rounds**, all nodes communicate their complete states to all neighbors

- After r rounds, nodes **know r-neighborhood** in G

- Compute **output** (e.g. in or not in MIS/DS) based on this information (without additional communication)

- **Randomized algorithms:**
  Nodes choose sufficiently many random bits at the beginning

Local Algorithms

- **General question:**
  - What can be computed locally?
  - What can be computed in r rounds?

- **Local algorithm:**
  - Strict: time complexity independent of global parameters (n: # of nodes, D: diameter, Δ: largest degree (?))
  - Less strict: time complexity almost indep. of global param. (e.g., $\mathrm{polylog}(n, \Delta)$, $o(D)$, …)

Outline

1) **Overview over existing work**

2) Example: minimum dominating set

3) Open problems / directions

Goal: Make it interactive …

… please ask / interrupt!

Classic Results

[Linial; FOCS ‘87, SICOMP ‘92]:

- First paper that explicitly discusses locality
- Major results on distributed coloring:
  - 3-coloring ring deterministically: $\Omega(\log^* n)$ rounds (randomized lower bound in [Naor ‘91])
  - $O(\Delta^2)$-coloring of arbitrary graphs: $O(\log^* n)$ rounds (Δ: largest degree of the network)

[Naor, Stockmeyer; STOC ’93, SICOMP ‘95]:

- Some labelings can be computed in const. time
- Labeling problems: Const. round algorithms can be derandomized

Classic Results

MIS, maximal matching in $O(\log n)$ time:

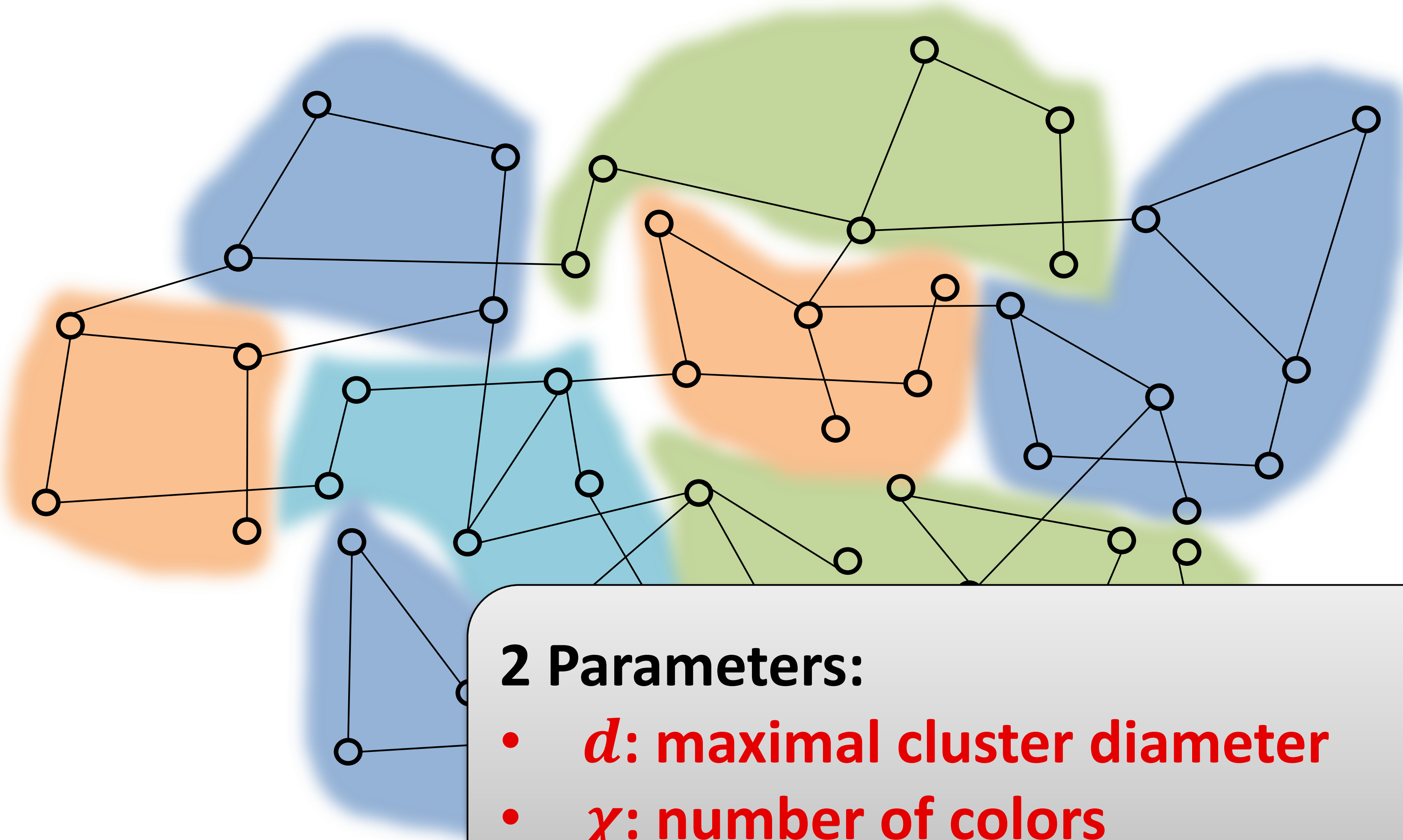
- [Luby; ‘86], [Israeli,Itai; ‘86], [Alon,Babai,Itai; ‘86]
- Algorithms described as PRAM algorithm

By reduction, same is true for $(\Delta + 1)$-coloring:

- [Linial ’92]

Network Decomposition

- Decomposition of network into colored clusters



2 Parameters:

- d: maximal cluster diameter
- χ: number of colors

Network Decomposition

Introduced in [Awerbuch,Goldberg,Luby,Plotkin ’89]

- deterministic algorithm
- $\#\text{rounds} = d = \chi = 2^{O\left(\sqrt{\log n \log\log n}\right)}$
  - Note: $\text{polylog}(n) = 2^{O(\log\log n)}$, $n^{\Theta(1)} = 2^{\Theta(\log n)}$

Improvement by [Panconesi,Srinivasan ‘95]

- Det. alg.: $\#\text{rounds} = d = \chi = 2^{O\left(\sqrt{\log n}\right)}$

Randomized algorithm [Linial,Saks ‘93]

- $\#\text{rounds} = d = \chi = O(\log n)$

Weak → strong decompositions [Awerbuch et al. ‘96]

Using Network Decompositions

Network decompositions give a generic technique:

1. Compute decomposition
2. Iterate through the colors
3. For each color, solve partial solutions on clusters in parallel (clusters of same color are not adjacent)

Example:

- Gives simple deterministic MIS / coloring algorithms with time complexity $2^{O(\sqrt{\log n})}$

Local Approximation

Minimum dominating set:

- [Jia,Rajaraman,Suel '02]:
  - $O(\log \Delta)$-approximation in $O(\log n \log \Delta)$ rounds

- [Kuhn,Wattenhofer '03],[K.,Moscibroda,W. '06]:
  - $O(\Delta^{1/\sqrt{r}} \log \Delta)$-approximation in $O(r)$ rounds
    $O(\log \Delta)$-approximation in $O(\log^2 \Delta)$ rounds
  - $O(n^{1/r} \log \Delta)$-approximation in $O(r)$ rounds
    $O(\log \Delta)$-approximation in $O(\log n)$ rounds

- Similar, stronger bounds hold for min. vertex cover, max. matching

Lower Bounds

[Kuhn,Moscibroda,Wattenhofer ’04] + journal subm.

- In r rounds, min. (fractional) dom. set, min. vertex cover, max. matching cannot be approx. better than

$$\min\left\{\Omega\left(\Delta^{(1-\varepsilon)/r}\right), \Omega\left(n^{\left(1/4-\varepsilon\right)/r^2}\right)\right\}$$

- Constant approximation requires time

$$\min\left\{\Omega(\log \Delta), \Omega\left(\sqrt{\log n}\right)\right\}$$

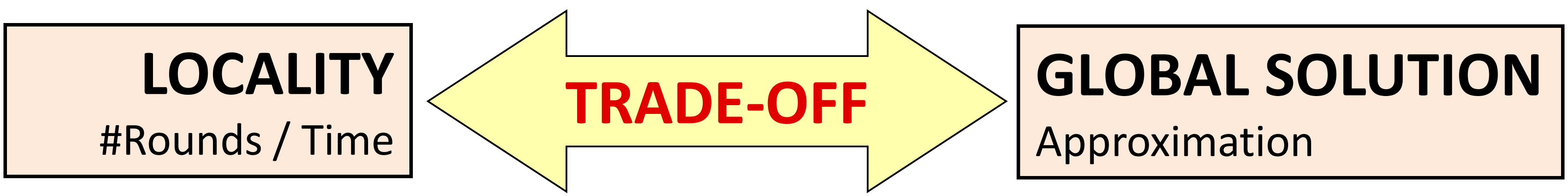
- Slightly weaker bounds for polylog. approximations
- Same lower bound holds by reduction also for MIS and maximal matching

The Price of Locality

- How well can a given optimization problem be approximated if we are only allowed to communicate for r rounds?

- Alternatively: How good can the approximation be if the decision for every node has to be based on its r-neighborhood

→ what is the price of being restricted to locality r?



Lower Bounds

- [Göös,Hirvonen,Suomela ‘12]:
  - Tight approximability lower bounds for constant time min. edge dominating set algorithms
- [Hirvonen,Suomela ’12]:
  - Tight time lower bound for maximal matching in anonymous, k-edge-colored graphs:

$$\boldsymbol{\Omega(\Delta + \log^* k)}$$

- [Göös,Suomela **DISC ‘12**]:
  - Approximation scheme for vertex cover in bipartite graphs requires $\boldsymbol{\Omega(\log n)}$ rounds

Strictly Local Approximation

- Many tight results for bounded degree graphs and strictly local algorithms
  - Max-min LPs:
    [Floréen,Hassinen,Kaski,Suomela ’08],[F.,Kaasinen,K.,S. ‘09]
  - Vertex cover:
    [Åstrand et al. ’09], [Åstrand,Suomela ’10]
  - Edge dominating sets:
    [Suomela ‘10], [Göös,Hirvonen,Suomela ‘12]
  - Fractional coloring (arbitrary graphs):
    [Kuhn ’09], [Hasemann,Hirvonen,Rybicki,Suomela ’12]

Distributed Coloring: Recent Progress

- Deterministic algorithms:
  - Arboricity [Barenboim,Elkin '08]
  - Defective coloring [Barenboim,Elkin '09], [Kuhn '09]
  - Combination of ideas lead to surprising new results:
  - [Barenboim,Elkin '10]: $\Delta^{1+o(1)}$ colors in polylog(n) time
    $O_\epsilon(\Delta)$ colors in $O(\Delta^\epsilon \log n)$ time
  - [Barenboim,Elkin '11]: slightly better results for edge col.

- Randomized algorithms:
  - [Schneider,Wattenh. '10], [Barenb.,Elkin,Pettie,Schneider '12]:
    $(\Delta+1)$-coloring in time $O\left(2^{O(\sqrt{\log\log n})}+\log\Delta\right)$
    MIS in time $O\left(\sqrt{\log n}\log\Delta\right)$

Distributed Decision

- Distributed decision problem:
  - Distributed input vector $\boldsymbol{x}$ (each node gets a part of the input)
  - Language $\mathcal{L}$
  - Yes-instance ($\boldsymbol{x} \in \mathcal{L}$): all nodes have to output yes
  - No-instance ($\boldsymbol{x} \notin \mathcal{L}$): at least one node has to output no

- Introduced in [Fraigniaud,Korman,Peleg '11]
  - Defines complexity classes $\mathrm{LD}(t)$, $\mathrm{NLD}(t)$, $\mathrm{BPLD}(t, p, q)$
  - Whether randomization helps depends on the error bounds
  - $\mathrm{LD}(t) \notin \mathrm{NLD}(t)$

Distributed Decision

- Additional work:
  - [Fraigniaud,Halldorson,Korman ’12]: impact of unique identifiers
  - [Fraigniaud,Korman,Parter,Peleg; **DISC 12**]: more on randomization
- Related problem studied by [Göös,Suomela ‘11]
  - Strictly (non-det.) local algorithms (i.e., $t = O(1)$)
  - Paper studies proof complexity
- **Apologies for all the related work I missed…**

Outline

1) Overview over existing work

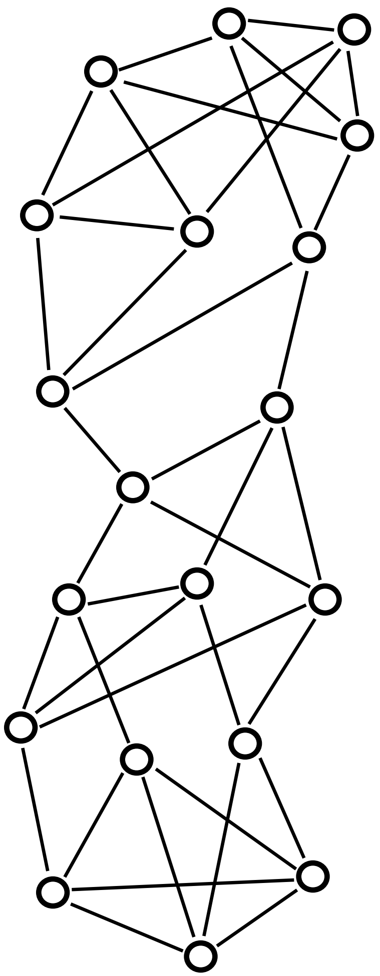
2) **Example: minimum dominating set**

3) Open problems / directions

Regular Graphs

Use randomization to break symmetries

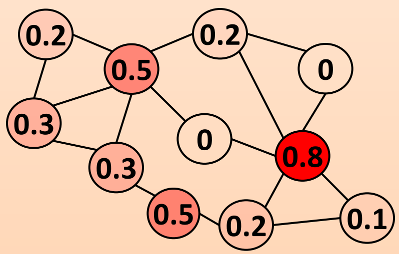
1. All nodes have degree d, start with empty set
2. Add each node with probability ${}^{\ln(d+1)}/_{d+1}$
   – Exp. number of nodes: ${}^{n\cdot\ln(d+1)}/_{d+1}$
3. Some nodes are not covered
   – Simple calc.: Prob. that not covered $< {}^{1}/_{d+1}$
   – Exp. number of uncovered nodes $< {}^{n}/_{d+1}$
   – Add all uncovered nodes to dominating set
4. Dominating set of exp. size ${}^{(\mathbf{1}+\mathbf{ln}(\boldsymbol{d}+\mathbf{1}))\boldsymbol{n}}/_{\boldsymbol{d}+\mathbf{1}}$
   – Each node covers $\leq d+1$ nodes
   – Opt. solution $\geq {}^{n}/_{d+1}$



General Networks

Compute probability for each node:

(fractional dominating set)



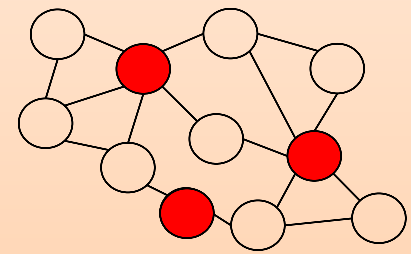
Linear Program:

$$\begin{aligned} \text{min} \quad & \sum_{i=1}^{n} x_i \\ \text{subject to} \quad & N \cdot \underline{x} \geq \underline{1} \\ & \underline{x} \geq \underline{0} \end{aligned}$$

Intuition from greedy algorithm:
high degree → larger probability

Randomized Rounding:

(fractional → integer)



Same algorithm as
for regular graphs
(use computed probablities)

increases approximation
ratio by factor $1 + \ln(\Delta + 1)$

Solving the Linear Program

LP Approximation Algorithm for Primal Node $v_i^{(p)}$:

```
x_i := 0;
for e_p := k_p − 2 to −f − 1 by −1 do
   for 1 to h do
      (* γ_i := (c_max / c_i) Σ_j a_ji r_j *)
      for e_d := k_d − 1 to 0 by −1 do
         γ̃_i := (c_max / c_i) Σ_j a_ji r̃_j;
         if γ̃_i ≥ 1/Γ_p^{e_p/k_p} then
            x_i^+ := 1/Γ_d^{e_d/k_d}; x_i := x_i + x_i^+;
         fi;
         send x_i^+, γ̃_i to dual neighbors;
         ...
         receive r̃_j from dual neighbors
         ...
      receive r_j from dual neighbors
   od
od;
x_i := x_i / min_{j∈N_i^(p)} Σ_ℓ a_jℓ x_ℓ
```

LP Approximation Algorithm for Dual Node $v_i^{(d)}$:

```
y_i := y_i^+ := w_i := f_i := 0; r_i := 1;
for e_p := k_p − 2 to −f − 1 by −1 do
   for 1 to h do
      r̃_i := r_i;
      for e_d := k_d − 1 to 0 by −1 do
         ...
         receive x_j^+, γ̃_j from ...
         y_i^+ := y_i^+ ... Σ_j a_i ...
         w_i^+ := Σ_j a_ij x_j^+;
         ... w_i + w_i^+; f_i := ...
         if w_i ≥ 1 then r̃_i := ...
         send r̃_i to primal nei...
      od;
      increase_duals();
      send r_i to primal neighb...
   od
od;
y_i := y_i / max_{j∈N_i^(d)} (1/c_j) Σ_ℓ a ...
```

procedure increase_duals():

```
if w_i ≥ 1 then
   if f_i ≥ f then
      y_i := y_i + y_i^+; y_i^+ := 0;
      r_i := 0; w_i := 0
   else if w_i ≥ 2 then
      y_i := y_i + y_i^+; y_i^+ := 0;
      r_i := r_i / Γ_p^{⌊w_i⌋/k_p}
   else
      λ := max{Γ_d^{1/k_d}, Γ_p^{1/k_p}};
      y_i := y_i + min{y_i^+, r_i λ/Γ_p^{e_p/k_p}};
      y_i^+ := y_i^+ − min{y_i^+, r_i λ/Γ_p^{e_p/k_p}};
      r_i := r_i / Γ_p^{1/k_p}
   fi;
   w_i := w_i − ⌊w_i⌋
fi
```

Uses ideas from greedy MDS alg.

$\Delta^{1/\sqrt{r}}$ – approx. in O(r) rounds

Solving Linear Program

- Solution based on network decomposition

- In $O(\log n)$ rounds, rand. alg. from [Linial,Saks '93] gives
  - Set of non-adjacent $O(\log n)$-diameter clusters
  - Every node is in some cluster with const. probability

- Algorithm idea:
  - Compute $O(\log n)$ such cluster sets (in parallel)
  - W.h.p., each node is in $\Theta(\log n)$ clusters
  - Solve local LP optimally for each cluster in $O(\log n)$ rounds
  - Linear combination of all local solutions gives constant approximation for the global solution in $O(\log n)$ rounds

Lower Bound: Intuition

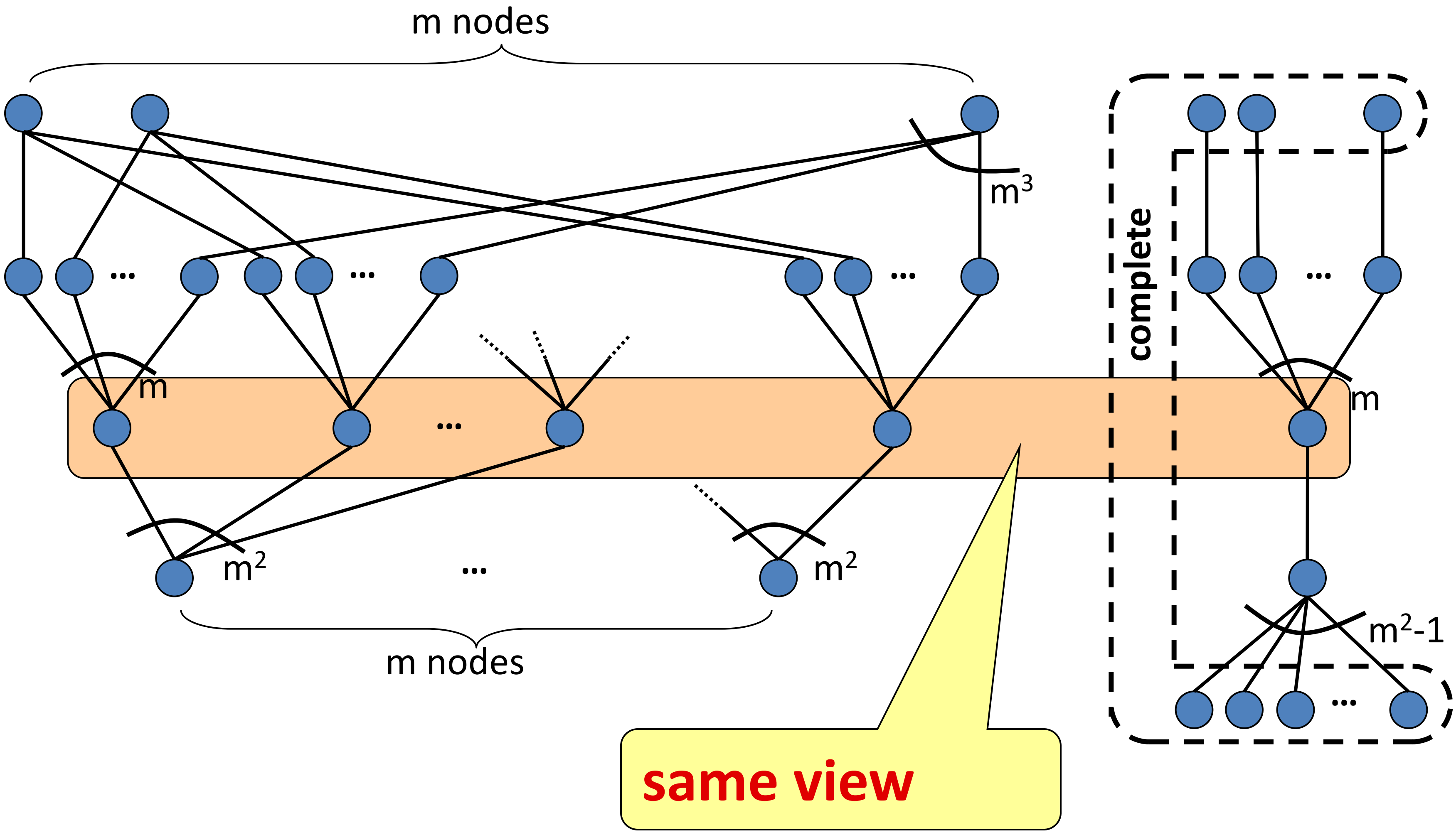
- How to prove a lower bound?

- Let's look at case $r = 2$ to get some intuition

- After 1 round, nodes know their neighbors

- After 2 rounds, nodes know the neighbors of their neighbors

Two-Round Lower Bound



Indistinguishability

- If we ignore node IDs:
  Node with same view have to make the same decision

- Assume random node ID assignment with IDs from {1,…,N}
- If nodes u and v see same topology up to distance 2 (r):
  - Every possible ID assignment is equally probable
  - Probability to see a particular ID assignment equal for u and v
  - u and v make the same decision with the same probability p

- Deterministic algorithms: ∃ node assignment for which solution is at least as bad as expected value with random IDs

- Randomized algorithms: Same bound using Yao's principle

Approximation Ratio Lower Bound

m nodes

Number of nodes:
$n=\Theta(m^2)$

Red node joins:
$|DS_{OPT}| = 2$

Red node
$|DS_{OPT}| \geq m$

If red node j
Expected approx. rat

$$\alpha \geq (1-p) \cdot \frac{m}{2} \in \Omega((1-p) \cdot \sqrt{n})$$

ber of nodes:

des join w. prob. p,
cted approx. ratio α:

m²-1

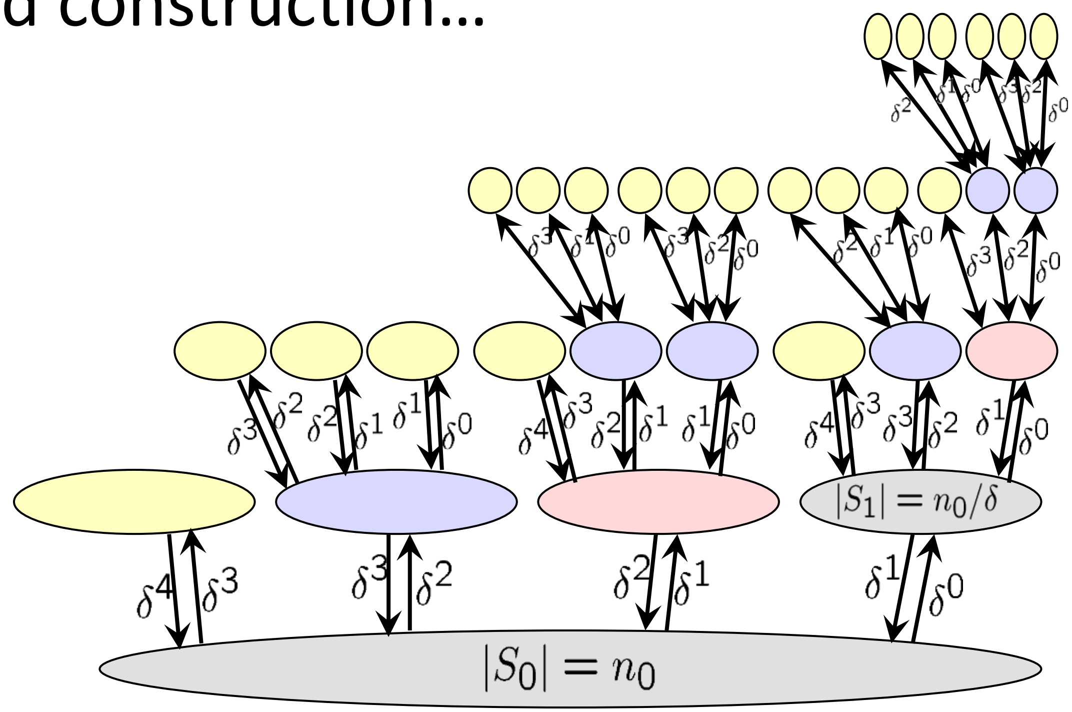
$$\alpha \geq \frac{pm^3}{2m} \in \Omega(p\sqrt{n})$$

Approximation ratio of any 2-round minimum dominating set algorithm is

$$\Omega\left(\sqrt{n}\right)$$

General Case

- We use vertex cover instead of dominating set
- And a more involved construction…



Results, Dominating Set

[Kuhn,Moscibroda,Wattenhofer '06]:

- $O(\Delta^{1/r}\log\Delta)$-approximation in $O(r^2)$ rounds
  $O(\log\Delta)$-approximation in $O(\log^2\Delta)$ rounds
- $O(n^{1/r}\log\Delta)$-approximation in $O(r)$ rounds
  $O(\log\Delta)$-approximation in $O(\log n)$ rounds

[Kuhn,Moscibroda,Wattenhofer '04]:

- In r rounds, approximation ratio is at least

$$\min\left\{\Omega\left(\Delta^{(1-\varepsilon)/r}\right), \Omega\left(n^{(1/4-\varepsilon)/r^2}\right)\right\}$$

- Time to get $O(\log\Delta)$-approximation:

$$\min\left\{\Omega\left(\frac{\log\Delta}{\log\log\Delta}\right), \Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)\right\}$$

Outline

1) Overview over existing work

2) Example: minimum dominating set

3) **Open problems / directions**

Goal: Make it interactive …

… please ask / interrupt!

Deterministic Local Algorithms

- Best deterministic algorithm for many problems has time complexity $2^{O(\sqrt{\log n})}$
- For example:
  - MIS
  - $(\Delta + 1)$-coloring
  - $(\mathrm{poly}\log n, \mathrm{poly}\log n)$-decomposition
  - Dominating set approximation
  - Dominating set rounding
  - Approximation scheme for recut?
- All these problems have $\mathrm{poly}\log n$ randomized sol.!

Long-Standing Open Problem

- **Is there really an exponential gap between deterministic and randomized solutions?**
  - We haven't found any faster det. algorithms for >20 years, so maybe?

- Or more positively:

  Can deterministic algorithms in the $\mathcal{LOCAL}$ model be efficiently derandomized?
  - Recent progress on deterministic, distributed coloring might suggest this?

Symmetry Breaking

Hard part seems to be to break symmetry…

Example: Distributed approximation

- Distributed LP algorithms can be derandomized:
  - Assumption: algorithm always computes feasible solution
  - Output value of a node of an r-round randomized alg.: function of inputs/rand. bits/topology of r-neighborhood
  - Possible to compute expectation of output value (deterministically)
  - Expected output values give feasible solution for LP
  - Approximation = expected approximation of rand. alg.

- Makes LP relaxation an attractive approach for distr. alg.

Cost of Symmetry Breaking?

- Randomization is a natural strategy to break symm.
- Is it necessary to do it efficiently?
- What is the cost of randomized symmetry breaking?
  - The $\Omega(\sqrt{\log n})$ lower bounds from [KMW '04] are about approximation and not about breaking symmetry
  - MIS lower bound merely a corollary
  - Lower bound does not seem to apply to coloring
  - $(\Delta + 1)$-coloring can be approximated very efficiently!

Distributed Complexity Theory

- Certainly a very interesting direction…
- Very promising work on local decision
- What about more standard distributed computations
  - In the sequential world, decision problems capture most of what we want to understand
  - This does not seem to be the case in the distributed context

Beyond the $\mathcal{LOCAL}$ model

- What if we cannot send arbitrarily large messages?

- Many efficient local algorithms are based on techniques like network decompositions
  - Pretty brute-force approach
  - Alg. often communication and computation intensive
  - Simpler, slower (but still very local) algorithms might exist, e.g., dominating set $O(\log n)$ vs. $O(\log^2 n)$
  - Can we prove lower bounds?
    e.g., by applying techniques from communication complexity…

Dynamic Networks

- Major practical motivation to study locality:
  fault tolerance, robustness in case network changes
- Effect of fault or change can be fixed locally!
  - But only if no other changes happen in the meantime…
- What happens if the network is really dynamic?
  - Can we still use the same techniques?
  - What problems can still be solved locally?
  - What is the additional cost?

Questions ?
Comments?