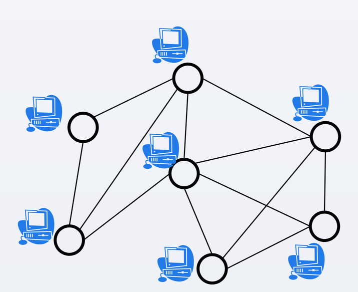
Distributed Algorithms for the Lovász Local Lemma and Graph Coloring

Kai-Min Chung[1], Seth Pettie[2], Hsin-Hao Su[2]

1. Cornell University 2. University of Michigan and MADALGO

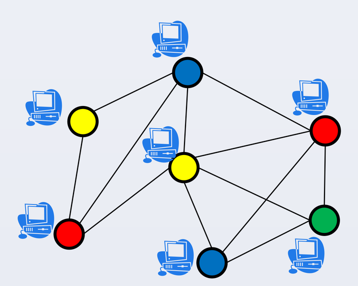
Distributed Model



Each round, each processor:

1. Receives messages from its neighbors
2. Perform some computation
3. Send messages to its neighbors

Minimize #rounds to compute a function



e.g. vertex coloring

Goal: each processor output a color such that adjacent processors receive different colors

Lovász Local Lemma (LLL)

Bad events: $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$

A_i depends on the variables in $\mathrm{vbl}(A_i) \subseteq \mathcal{P} = \{P_1, P_2, \ldots, P_m\}$

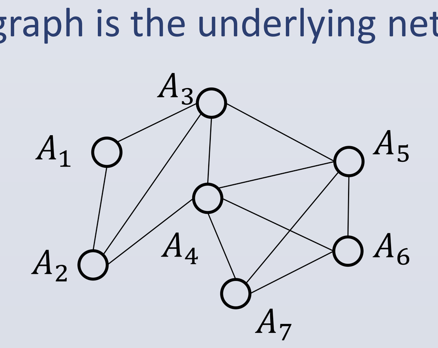
If

1. A_i shares variables with at most d other events
2. $\Pr(A_i) \le p$
3. $ep(d+1) < 1$

then $\Pr(\,no\ A_i\ occurs) > 0$

LLL in Distributed Model

Each processor is associated with a bad event.
The dependency graph is the underlying network.



Goal: Processors computes and agrees on $\mathcal{P}$ such that no bad events occur

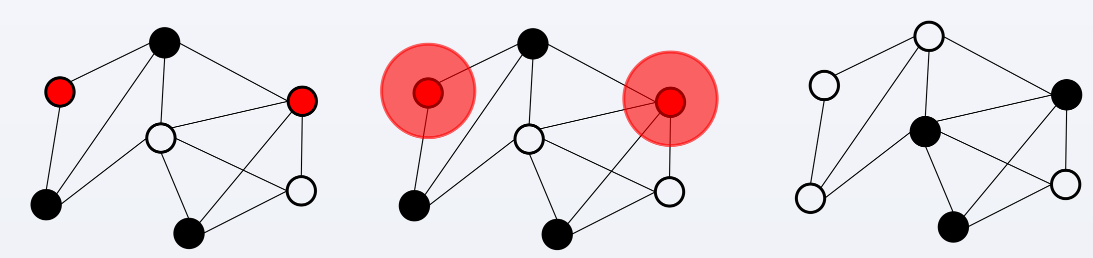
Moser and Tardos' Resampling Algorithm [MT10]

● $\mathcal{F}$: set of bad nodes

Get an initial sample for each $P_i \in \mathcal{P}$

Repeat the following for $O(\log_{1/ep(d+1)} n)$ times:

1. Compute a maximal independent set (MIS) $\mathcal{I}$ in the graph induced by $\mathcal{F}$
2. Resample every variable in $\bigcup_{A \in \mathcal{I}} \mathrm{vbl}(A)$



Total rounds: $O(\mathrm{MIS}(n,d) \cdot \log_{1/ep(d+1)} n)$

$\mathrm{MIS}(n,d)$:		
[Luby86]	$O(\log n)$	
[Kuhn09,BE09]	$O(d + \log^* n)$	larger message complexity
[BEPS12]	$O(\log d \cdot \sqrt{\log n})$	
Lower Bound[KMW10]	$\Omega(\min\{\sqrt{\log n}, \log d\})$	

Our Algorithm (I)

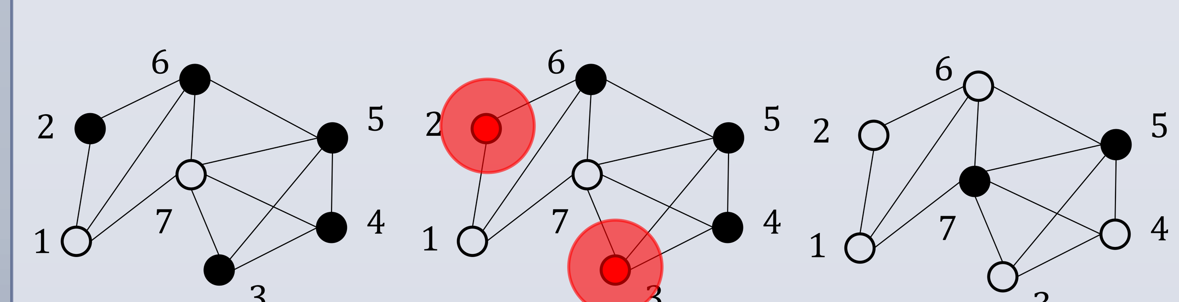
Needs stronger LLL condition: $epd^2 < 1$

Running time: $O(\log_{1/epd^2} n)$

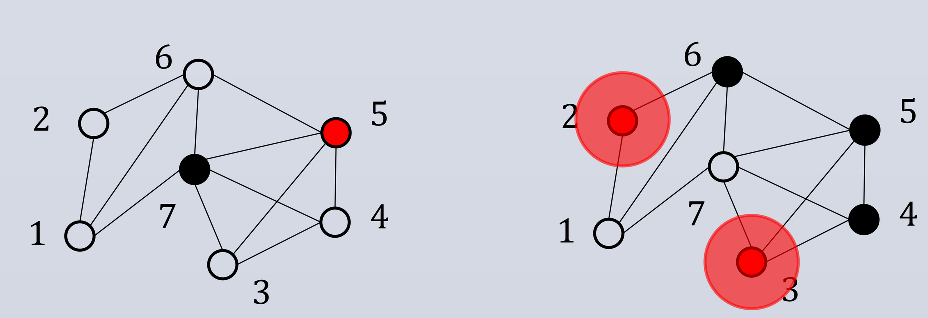
Get an initial sample for each $P_i \in \mathcal{P}$

Assume each node has an unique ID, repeat the following for $O(\log_{1/epd^2} n)$ times:

1. Let $\mathcal{I} = \{A \in F \mid A$ has the smallest ID among the neighbors of A that are in $\mathcal{F}\}$
2. Resample every variable in $\bigcup_{A \in \mathcal{I}} \mathrm{vbl}(A)$

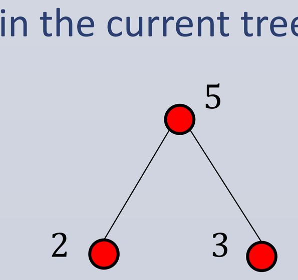


Key observation: At round $i > 1$, if A is resampled, then an event in the 2-neighborhood of A must have been resampled at round $i-1$



Round i Round $i-1$

2-witness tree: In the reverse order (break ties arbitrarily for events sampled in the same rounds), attach each resampled event to the deepest node in the current tree with distance ≤ 2



From the key observation, if there exists bad events after k rounds of resampling, then there is a 2-witness tree of size at least k

By modifying the Galton-Walton process from MT to generate 2-witness trees, we get

$$\Pr(\exists 2\text{–witness tree of size} \ge k) \le n(epd^2)^k$$

Set $k = O(\log_{1/epd^2} n)$

Our Algorithm (II)

Under the same LLL condition with MT: $ep(d+1) < 1$

Running time: $O(\log^2 d \cdot \log_{1/ep(d+1)} n)$

Replace step 1. by computing the weak MIS $\mathcal{I}$ such that each vertex belongs to the neighborhood of $\mathcal{I}$ with probability at least $1 - \Omega(1/d)$

A weak-MIS can be computed in $O(\log^2 d)$ rounds

Randomness from weak MIS

If there exists bad events after k rounds of weak MIS resampling, there exists a witness tree of size $k/2$ with probability at least $1/(d+1)^{k/2}$

Randomness from Resampling

$\Pr(\exists\text{witness tree of size} \ge k/2) \le n(ep(d+1))^{k/2}$

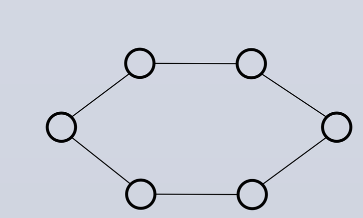
Conclusion: No bad events happens after $O(\max(\log_{d+1} n, \log_{1/ep(d+1)} n))$ rounds w.h.p.

This term dominates, because if $d+1 < 1/ep(d+1)$, then Algorithm(I) is applicable

Lower Bound

[Linial92] $\Omega(\log^* n)$ lower bound on $O(1)$-coloring a ring

Reduce coloring a ring to constructive LLL



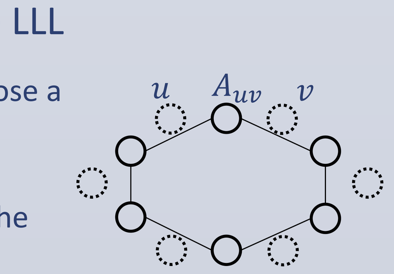
10-coloring

Each vertex u choose a color uniformly at random

A_{uv}: u and v has the same color

$\Pr(A_{uv}) \le 1/10$

$ep(d+1) = \frac{3e}{10} < 1$



Dependency graph

Applications: Distributed Graph Coloring

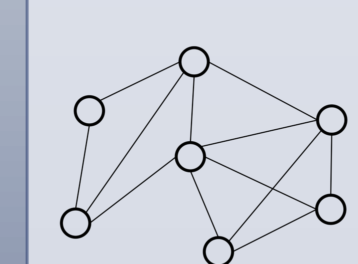
- **Frugal Coloring**:
  A β-frugal coloring is one in which each color appears at most β times in the neighborhood of any vertex. We gave algorithms for obtaining
  1. $O(\log^2\Delta / \log\log\Delta)$-frugal, $(\Delta+1)$ coloring in $O(\log n)$ rounds.
     [PS08] $O(\log\Delta \cdot \log n / \log\log n)$-frugal, $(\Delta+1)$ coloring in $O(\log n)$ rounds.
  2. β-frugal, $O(\Delta^{1+1/\beta})$ coloring in $O(\log n \cdot \log^2\Delta)$ rounds.
     [HMR 97] proved the existence of the coloring.

- **Girth 4 and 5:**
  1. $(4+\epsilon)\Delta/\log\Delta$ coloring triangle-free graphs in $O(\log n)$ rounds.
     [PS13] gave an algorithm that runs in $O(\log^{1+o(1)} n)$ rounds.
  2. $(1+\epsilon)\Delta/\log\Delta$ coloring girth-5 graphs in $O(\log n)$ rounds.
     [PS13] gave an algorithm that runs in $O(\log^{1+o(1)} n)$ rounds.

- **Edge Coloring:**
  $(1+\epsilon)\Delta$ edge-coloring in $O(\log n)$ rounds
  [DGP97] $(1+\epsilon)\Delta$ edge-coloring in $O(\log n)$ rounds for $\Delta \gg \log n$

- **List Coloring:**
  Every vertex has a list of $(1+\epsilon)D$ colors such that each color appears in at most D lists in the neighborhood of any vertex. We gave an algorithm to obtain such a coloring in $O\left(\log D + \log_D n + \frac{\log\log D}{\sqrt{D}} \cdot \log n\right) = O(\log n)$ rounds
  [RS02] proved the existence of the coloring

- **Defective Coloring:**
  A k-defective coloring is one in which a vertex may share its color up to k neighbors. For any $k = \Omega(\log\Delta)$, we gave an algorithm to obtain a k-defective $O(\Delta/k)$-coloring in $O\left(\frac{\log n}{k}\right)$ rounds.
  [BE] $O(\log n)$-defective $O(\Delta/\log n)$-coloring in O(1) rounds



Each vertex u choose a color uniformly at random

A_u: More than k neighbors having same color with u

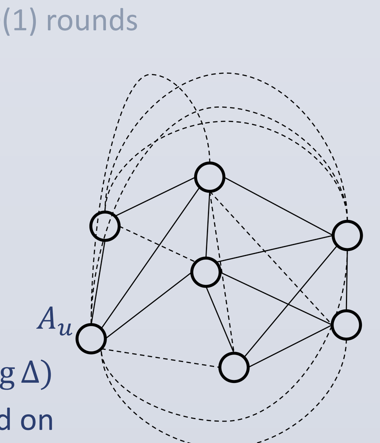
$d \le \Delta^2$

Chernoff Bound:
$\Pr(A_u) \le e^{-k/6}$

$epd^2 = e^{-\Omega(k)}$ for $k = \Omega(\log\Delta)$

Algorithm(I) can be simulated on the dependency graph with $O(1)$ overhead

Total rounds: $O(\log_{1/epd^2} n) = O((\log n)/k)$ rounds



Dependency graph

References

[BE] L. Barenboim and M. Elkin. Distributed Graph Coloring. *Manuscript*.

[BE10] L. Barenboim and M. Elkin. Distributed $(\Delta+1)$-coloring in Linear (in Δ) time. *STOC '09*, 111-120.

[BEPS12] L. Barenboim, M. Elkin, S. Pettie, J. Schneider. The Locality of Distributed Symmetry Breaking. *FOCS '12*, 321-330.

[DGP97] D. Dubhashi, D. Grable, and A. Panconesi. Near-Optimal, Distributed Edge Colouring via the Nibble Method. *Theor. Comput. Sci.*, 203(2):225-251.

[HMR97] H. Hind, M. Molloy, and B. Reed. Colouring a Graph Frugally. *Combinatorica*, 17(4):469-482.

[Kuhn09] Fabian Kuhn. Weak Graph Colorings: Distributed Algorithms and Applications. *SPAA '09*, 138-144.

[KMW10] F. Kuhn, T. Moscibroda, and R. Wattenhofer. Local Computation: Lower and Upper Bounds. *CoRR*, abs/1011.5470.

[Linial92] N. Linial. Locality in Distributed Graph Algorithms. *SIAM J. Comput.*, 21(1):193-201

[Luby86] M. Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM J. Comput.*, 15(4): 1036-1053.

[MT10] Robin A. Moser and Gábor Tardos. A Constructive of the General Lovász Local Lemma. *J. ACM*, 57(2):11:1-11:15.

[PS08] S. Pemmaraju and A. Srinivasan. The Randomized Coloring Procedure with Symmetry-Breaking. *ICALP '08*, 306-319.

[PS13] S. Pettie, H. Su. Fast Distributed Coloring Algorithms for Triangle-Free Graphs. *ICALP '13*, 687-699.

[RS02] B Reed and B. Sudakov. Asymptotically the List Colouring Constants are 1. *J. Combin. Theroy, Series B*, 86(1):27-37

www.PosterPresentations.com