Coloring in Graph Streams via Deterministic and Adversarially Robust Algorithms

Sepehr Assadi | Amit Chakrabarti | Prantar Ghosh | *Manuel Stoeckl*

Link to paper

S.A at Rutgers, supported in part by a NSF CAREER Grant CCF-2047061, a Google Research gift, and a Fulcrum award from the Rutgers Research Council. A.C. and MS. at Dartmouth. P.G. at DIMACS, work done in part at Dartmouth. This work was supported in part by NSF under awards CCF-1907738 and CCF-2006589.

To what extent is randomization necessary for Δ-based graph coloring?

Deterministic multi-pass coloring

Multi-pass streaming algorithm:

- Input is a sequence of elements
- In each pass, algorithm processes elements one by one
- Algorithm has limited working space $\ll$ input size

$f(\Delta)$-graph coloring task:

- Stream consists of the edges of a graph with n vertices and maximum degree Δ
- Output a vertex coloring using $f(\Delta)$ colors

Immediate prior work

- [ACS22]: Cannot $O(\text{poly}(\Delta))$-color a graph in 1 pass using $\tilde{O}(n)$ space with deterministic algorithm
- [ACS22] $O(\Delta)$ coloring in $O(\log \Delta)$ passes
- [GK21] $\Delta + 1$ coloring in distributed models
- Many papers with 1-pass randomized algorithms [ACK19, AA20, ACS22, HKNT22, AKM22] or in distributed models [BKM20, Kuh20, HKNT22]

Our results

- *A deterministic streaming algorithm for $(\Delta+1)$-coloring using $O(\log \Delta \log\log \Delta)$ passes and $O\left(n(\log n)^2\right)$ space*
- Inspired by techniques of [GK21] and [ACS22]
- This generalizes to degree+1 list coloring, if the vertex color lists are provided in a certain way

Multi-pass streaming algorithm, details

- Use $O(\log \Delta)$ epochs: in each epoch, fix colors for a constant fraction of the uncolored vertices.
- First epoch: *randomly* assigning colors in $[\Delta+1]$ to each vertex, gives $O(n)$ conflicting edges $\implies$ have set of $\Omega(n)$ non-conflicting vertices
- Derandomization: use method of conditional expectations to propose vertex colors with $O(n)$ conflicting edges

Partially committed coloring (PCC):

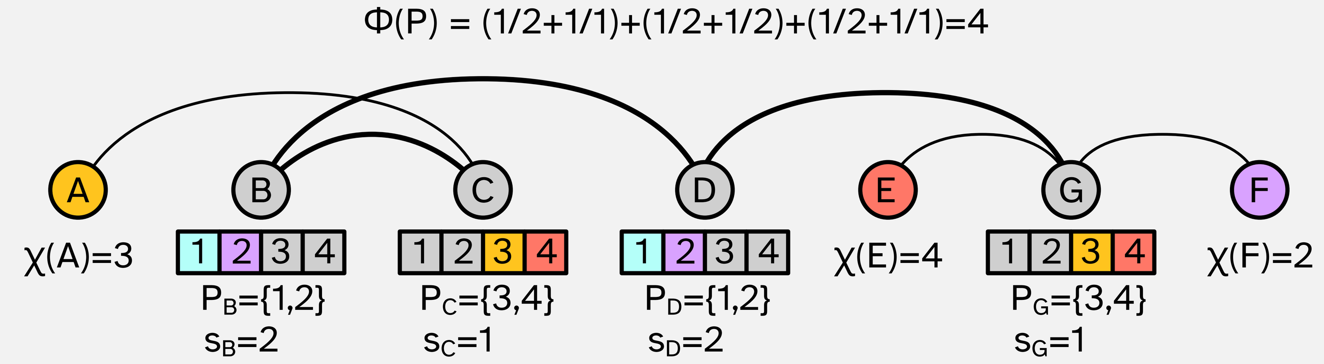
- Constrains a color assignment
- Say U is set of vertices whose color was not fixed
- If vertex $x \notin U$, its color is $\chi(x)$; if $x \in U$, it has a subset of colors P_x from a partition of $[\Delta+1]$
- Bound on number of conflicting edges of PCC $\mathcal{P} = (P_x)_{x \in V}$ is:

$$\Phi(\mathcal{P}) = \sum_{\{x,y\} \in G[U]} 1_{P_x = P_y} \left(\frac{1}{s_x} + \frac{1}{s_y} \right)$$

where $s_x = |P_x| - |y \in N(x) \setminus U : \chi(y) \in P_x|$.

- s_x is $\geq$ the "slack"[HKNT22] of a vertex
- Initial PCC: have $P_x = [\Delta+1]$
- Once $|P_x| = 1$, have a *proposed* color for x

Φ(P) = (1/2+1/1)+(1/2+1/2)+(1/2+1/1)=4



Refining PCC: repeat to reduce $\sum_{x \in V} |P_x|$

- Pass 1: Compute $s_{x,i}$ for every uncolored vertex x
- From current PCC $\mathcal{P}$, construct family $\mathcal{F}$ of $\tilde{O}(n^2)$ "refined" PCCs where $\text{avg}_{Q \in \mathcal{F}} \Phi(Q) \leq \Phi(\mathcal{P})$
- Passes 2-3: Find a $Q \in \mathcal{F}$ where $\Phi(Q) \leq \Phi(\mathcal{P})$
- Set $\mathcal{P} \leftarrow Q$, and repeat until all $|P_x| = 1$

Adversarially robust graph coloring

Streaming algorithm:

- Input is a sequence of elements $e_1, \ldots, e_m$
- "tracking" algorithm: emits output o_i after every input e_i, and is wrong if any output is

Static setting:

- For all valid input streams, $\Pr[\text{output wrong}] \leq \delta$

Adversarial setting[BJWY20]:

- An *adversary* adaptively produces an input sequence $e_1, \ldots, e_n$, where it chooses input e_i based on the algorithm outputs $o_1, \ldots, o_{i-1}$ so far
- "adversarially robust": for all adversaries making valid input streams, $\Pr[\text{output wrong}] \leq \delta$

$f(\Delta)$ -graph coloring task

- Receive sequence of edges in a simple graph on n vertices of max degree Δ
- Output a vertex coloring of the edges so far using $f(\Delta)$ colors

Immediate prior work

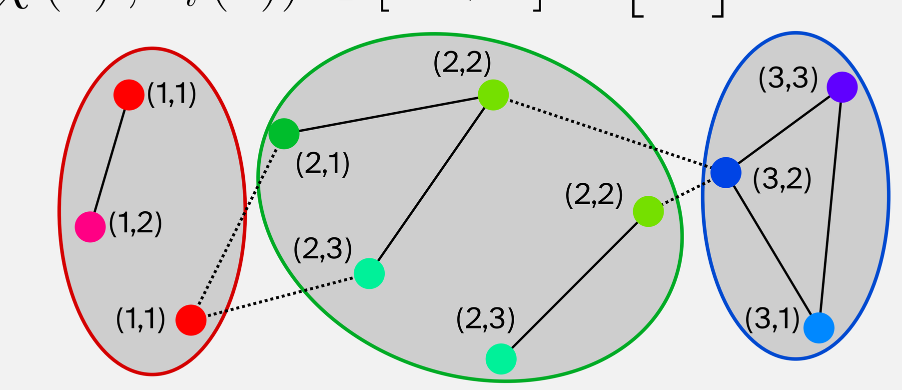
- [CGS22]: adversarially robust algorithm for $O(\Delta^3)$-coloring using $\tilde{O}(n)$ space, $\tilde{O}(n\Delta)$ random bits
- [CGS22]: adversarially robust $O(\Delta^2)$ coloring algorithms need $\Omega(n)$ space
- Many papers for static setting [BG18, ACK19, AA20, ACS22, HKNT22]

Our results

- *An adversarially robust, $O(\Delta^{5/2})$-coloring algorithm using $\tilde{O}(n)$ space and oracle access to $\tilde{O}(n\Delta)$ random bits*
- *An adversarially robust, $O(\Delta^3)$-coloring algorithm using $\tilde{O}(n)$ space.*

An $O(\Delta^3)$ color $\tilde{O}(n)$ space algorithm

- Divide stream into "epochs" of n edges each
- For each epoch i, present output of independent sub-sketch:
  - Pick random $h_i : [n] \to [\Delta^2]$
  - *Before* epoch i, record edges $\{x, y\}$ with $h_i(x) = h_i(y)$ into set D_i
  - During epoch i, record edges $\{x, y\}$ with $h_i(x) = h_i(y)$ into set B_i
  - Compute $(\Delta+1)$-coloring χ of edge set $D_i \cup B_i$
  - For each vertex v, output $(\chi(v), h_i(v)) \in [\Delta+1] \times [\Delta^2]$



- Discard sets when no longer needed
- $|D_i| = O(n/\Delta)$ w.h.p. because adversary doesn't see h_i until epoch i; and $|B_i| \leq n$

O$(\Delta^{5/2})$ coloring algorithm with $\tilde{O}(n)$ space

- Different handling for *fast vertices* (those which receive $\geq \sqrt{\Delta}$ edges per epoch of n edges)
- Remaining *slow vertices* form bounded degree graphs $D_i \cup B_i \implies$ only $O(\Delta^{5/2})$ colors used

Fast vertices:

- Partition vertices by degree into $\sqrt{\Delta}$ *levels* $\left[1, \sqrt{\Delta}\right], \left[\sqrt{\Delta}+1, 2\sqrt{\Delta}\right], \ldots \left[\Delta - \sqrt{\Delta}+1, \Delta\right]$
- Color each *level* like an epoch of the example alg.
  - Edges may cross levels, and levels coexist, unlike epochs
  - Have $\sqrt{\Delta}$ levels using $\Delta^{3/2} \times O\left(\sqrt{\Delta}\right)$ colors each