Distributed Edge Coloring in Time
Quasi-Polylogarithmic in Delta

Alkida Balliu, Fabian Kuhn, Dennis Olivetti
University of Freiburg, Germany
LOCAL model

- Undirected simple graph $G = (V, E)$ of $n$ nodes and maximum degree $\Delta$
- Each node has a unique ID
- Synchronous message passing model
- Unbounded computation
- Unbounded bandwidth
- Focus on locality: time = number of rounds = distance
LOCAL model: symmetry breaking
Four classical problems

- Maximal independent set
- (Δ + 1)-vertex coloring
- Maximal matching
- (2Δ - 1)-edge coloring
Four classical problems

These problems can be solved in $\text{poly}(\log n)$ rounds [Rozhon, Ghaffari ’20]

- Maximal independent set
- $(\Delta + 1)$-vertex coloring
- Maximal matching
- $(2\Delta - 1)$-edge coloring

But, how local are these problems?
Four classical problems: locality

Maximal independent set

$O(\Delta + \log^* n)$  
[Barenboim, Elkin, Kuhn '09]

$\Omega(\min\{\Delta, \log n/\log \log n\})$  
[Balliu et al. '19]

$\Omega(\log^* n)$  
[Linial '87]
## Four classical problems: locality

<table>
<thead>
<tr>
<th>Problem</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximal independent set</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n / \log \log n})$</td>
</tr>
<tr>
<td>Maximal matching</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n / \log \log n})$</td>
</tr>
</tbody>
</table>

- [Barenboim, Elkin, Kuhn '09]  
- [Balliu et al. '19]  
- [Linial '87]  
- [Panconesi, Rizzi '01]  
- [Balliu et al. '19]  
- [Linial '87]
## Four classical problems: locality

<table>
<thead>
<tr>
<th>Problem</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
<th>Refs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximal independent set</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n/\log \log n})$</td>
<td>[Balliu et al. '19]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Linial '87]</td>
</tr>
<tr>
<td>Maximal matching</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n/\log \log n})$</td>
<td>[Balliu et al. '19]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Linial '87]</td>
</tr>
<tr>
<td>$(\Delta + 1)$-vertex coloring</td>
<td>$\widetilde{O}(\sqrt{\Delta} + \log^* n)$</td>
<td>$\Omega(\log^* n)$</td>
<td>[Panconesi, Rizzi '01]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Barenboim, Elkin, Kuhn '09] [FHK '16][BEG '18][MT '20]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[Linial '87]</td>
</tr>
</tbody>
</table>
## Four classical problems: locality

<table>
<thead>
<tr>
<th>Problem</th>
<th>Time Complexity</th>
<th>Lower Bound</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximal independent set</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n/\log \log n})$</td>
<td>[Barenboim, Elkin, Kuhn '09] [Balliu et al. '19] [Linial '87]</td>
</tr>
<tr>
<td>Maximal matching</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n/\log \log n})$</td>
<td>[Panconesi, Rizzi '01] [Balliu et al. '19] [Linial '87]</td>
</tr>
<tr>
<td>$(\Delta + 1)$-vertex coloring</td>
<td>$\tilde{O}(\sqrt{\Delta} + \log^* n)$</td>
<td>$\Omega(\log^* n)$</td>
<td>[FHK '16][BEG '18][MT '20] [Linial '87]</td>
</tr>
<tr>
<td>$(2\Delta - 1)$-edge coloring</td>
<td>$2^{O(\sqrt{\log \Delta})} + O(\log^* n)$</td>
<td>$\Omega(\log^* n)$</td>
<td>[Kuhn '20] [Linial '87]</td>
</tr>
</tbody>
</table>
## Four classical problems: locality

<table>
<thead>
<tr>
<th>Problem</th>
<th>Upper Bound</th>
<th>Lower Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximal independent set</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n / \log \log n})$</td>
</tr>
<tr>
<td></td>
<td>[Barenboim, Elkin, Kuhn '09]</td>
<td>[Balliu et al. '19]</td>
</tr>
<tr>
<td>Maximal matching</td>
<td>$O(\Delta + \log^* n)$</td>
<td>$\Omega(\min{\Delta, \log n / \log \log n})$</td>
</tr>
<tr>
<td></td>
<td>[Panconesi, Rizzi '01]</td>
<td>[Balliu et al. '19]</td>
</tr>
<tr>
<td>$(\Delta + 1)$-vertex coloring</td>
<td>$\tilde{O}(\sqrt{\Delta} + \log^* n)$</td>
<td>$\Omega(\log^* n)$</td>
</tr>
<tr>
<td></td>
<td>[FHK '16][BEG '18][MT '20]</td>
<td>[Linial '87]</td>
</tr>
<tr>
<td>$(2\Delta - 1)$-edge coloring</td>
<td>$2^{O(\sqrt{\log \Delta})} + O(\log^* n)$</td>
<td>$\Omega(\log^* n)$</td>
</tr>
<tr>
<td></td>
<td>[Kuhn '20]</td>
<td>[Linial '87]</td>
</tr>
</tbody>
</table>
Edge coloring: state of the art

- \((2\Delta - 1)\)-edge coloring (achieved through \((\Delta + 1)\)-vertex coloring):
  
- \(O(\Delta + \log^* n)\) [Barenboim, Elkin ’09], [Kuhn ’09]

- \(O(\Delta^{3/4} + \log^* n)\) [Barenboim ’15]

- \(\tilde{O}(\sqrt{\Delta} + \log^* n)\) [Fraigniaud, Heinrich, Kosowski ’16] [Barenboim, Elkin, Goldenberg ’18] [Maus, Tonoyan ’20]

- \(O(\Delta)\)-edge coloring: \(O(\Delta^\varepsilon + \log^* n)\) [Barenboim, Elkin ’10]

- \((2\Delta - 1)\)-edge coloring in \(2^{O(\sqrt{\log \Delta})} + O(\log^* n)\) [Kuhn ’20]
Our result

\[(2\Delta - 1)\text{-edge coloring}\]

\[2^{O(\log^2 \log \Delta)} + O(\log^* n)\]

[Linial '87]

\[\Omega(\log^* n)\]

[this paper]
Our result

(deg(e) + 1)-list edge coloring can be solved in time quasi-polylogarithmic in $\Delta$

$(2\Delta - 1)$-edge coloring $2^{O(\log^2 \log \Delta)} + O(\log^* n) \quad \Omega(\log^* n)$

[this paper] [Linial ’87]
List edge coloring

Color palette:
List edge coloring

Color palette: 🟥🟨🟧🟩🟦
List edge coloring

$(\deg(e) + 1)$-list edge coloring: lists of at least $\deg(e) + 1$ colors
List edge coloring

- $(2\Delta - 1)$-edge coloring:
  - lists of $2\Delta - 1$ colors
  - all lists the same
Let's try the following (non list coloring based) algorithm:

• Start from a graph of **maximum degree** $\Delta$

• **2-color** the edges such that the graph induced by each color has **maximum degree** $\Delta/2$

• **Recurse** on each subgraph
Why list coloring
Why list coloring
Why list coloring
Why list coloring

- Start from a graph of maximum degree $\Delta$
- 2-color the edges such that the graph induced by each color has maximum degree $\Delta/2$
- Recurse on each subgraph

Too hard
Why list coloring

- Start from a graph of maximum degree $\Delta$
- c-color the edges such that the graph induced by each color has maximum degree $O(\Delta/c)$
- Recurse on each subgraph

We need too many colors
Why list coloring

- **Without lists**, by using a recursive algorithm, we have to *early commit* on color subspaces.

- **With lists**, we can color a subgraph and then *recurse* on the remaining uncolored subgraph.
(deg(e) + 1)-list edge coloring in time
\((\log \Delta)^{O(\log \log \Delta)} + O(\log^* n)\)

High level ideas
Definitions

• **Goal**: \((\text{deg}(e) + 1)\)-list edge coloring

• **Relaxed version**: \((\beta \times \text{deg}(e) + 1)\)-list edge coloring
Definitions

- **Goal**: $(\text{deg}(e) + 1)$-list edge coloring
- **Relaxed version**: $(\beta \times \text{deg}(e) + 1)$-list edge coloring

slack 1

slack $\beta$
Definitions

• **Goal**: $(\deg(e) + 1)$-list edge coloring

• **Relaxed version**: $(\beta \times \deg(e) + 1)$-list edge coloring
Definitions

- **Goal**: \( (\text{deg}(e) + 1) \)-list edge coloring
- **Relaxed version**: \( (\beta \times \text{deg}(e) + 1) \)-list edge coloring

\[ T(\beta, C) = \text{time required to solve a list coloring instance with a palette of size } C \text{ and slack } \beta \]
High level idea

\[ T(1, C) \leq \beta^2 \cdot \log \Delta \cdot T(\beta, C) \]

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]
High level idea

\[ T(1, C) \leq \beta^2 \cdot \log \Delta \cdot T(\beta, C) \]
\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

\[ T(1, \Delta) \leq \text{polylog } \Delta \cdot T(1, \sqrt{\Delta}) \]
\[ T(1, \Delta) \leq (\log \Delta)^{O(\log \log \Delta)} \cdot T(1, O(1)) = (\log \Delta)^{O(\log \log \Delta)} \]
High level idea

\[
T(1, C) \leq \beta^2 \cdot \log \Delta \cdot T(\beta, C)
\]

\[
T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta / \text{polylog } p, C/p)
\]

\[
T(1, \Delta) \leq \text{polylog } \Delta \cdot T(1, \sqrt{\Delta})
\]

\[
T(1, \Delta) \leq (\log \Delta)^{O(\log \log \Delta)} \cdot T(1, O(1)) = (\log \Delta)^{O(\log \log \Delta)}
\]
Increasing the slack

• Suppose we can solve “fast” a list edge coloring with slack $\beta$
Increasing the slack

• Suppose we can solve “fast” a list edge coloring with slack $\beta$
• Reduce the degree by computing a defective edge coloring
Increasing the slack

• Suppose we can solve “fast” a list edge coloring with slack $\beta$

• Reduce the degree by computing a defective edge coloring

• Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

• Suppose we can solve “fast” a list edge coloring with slack $\beta$

• Reduce the degree by computing a defective edge coloring

• Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

• Suppose we can solve “fast” a list edge coloring with slack $\beta$

• Reduce the degree by computing a defective edge coloring

• Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

• Suppose we can solve “fast” a list edge coloring with slack $\beta$

• Reduce the degree by computing a defective edge coloring

• Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

• Suppose we can solve “fast” a list edge coloring with slack $\beta$

• Reduce the degree by computing a defective edge coloring

• Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

- Suppose we can solve “fast” a list edge coloring with slack $\beta$
- Reduce the degree by computing a defective edge coloring
- Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

- Suppose we can solve “fast” a list edge coloring with slack $\beta$
- Reduce the degree by computing a defective edge coloring
- Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

- Suppose we can solve “fast” a list edge coloring with slack $\beta$
- Reduce the degree by computing a defective edge coloring
- Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

- Suppose we can solve “fast” a list edge coloring with slack $\beta$
- Reduce the degree by computing a defective edge coloring
- Solve many instances of relaxed list edge coloring sequentially (by going through color classes)
Increasing the slack

- Compute a $\deg(e)/(2\beta)$-defective edge coloring $g(e)$ with $O(\beta^2)$ colors
- Iterate through each color class $i$. Edges of color $i$ do the following:
  - Remove from the list the colors $c(e')$ already used by the neighbors
  - If the list has size larger than $\deg(e)/2$ stay active
  - Apply the algorithm for slack $\beta$ on active edges, obtain $c(e)$
- Recurse on uncolored edges
Increasing the slack

\[ T(1, C) \leq T(\text{defective-coloring}) + \text{nr\_color\_classes} \cdot T(\text{large slack}) + T(\text{recursion}) \]

\[ T(1, C) \leq \beta^2 \cdot \log \Delta \cdot T(\beta, C) \]
High level idea

\[ T(1, C) \leq \beta^2 \cdot \log \Delta \cdot T(\beta, C) \]

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/polylog p, C/p) \]

\[ T(1, \Delta) \leq \text{polylog} \Delta \cdot T(1, \sqrt{\Delta}) \]

\[ T(1, \Delta) \leq (\log \Delta)^{O(\log \log \Delta)} \cdot T(1, O(1)) = (\log \Delta)^{O(\log \log \Delta)} \]
Relaxed list edge coloring
Relaxed list edge coloring
Relaxed list edge coloring
Relaxed list edge coloring
Relaxed list edge coloring

$$T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/polylog p, C/p)$$
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/polylog p, C/p) \]

- Split the color space into many independent subspaces
- Assign a subspace to each edge
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
- Assign a subspace to each edge
- Independently recurse on each graph induced by edges with the same assigned subspace
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
- Assign a subspace to each edge
- Independently recurse on each graph induced by edges with the same assigned subspace
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
- Assign a subspace to each edge
- Independently recurse on each graph induced by edges with the same assigned subspace
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
- Assign a subspace to each edge
- Independently recurse on each graph induced by edges with the same assigned subspace
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
- Assign a subspace to each edge
- Independently recurse on each graph induced by edges with the same assigned subspace
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
- **Assign a subspace to each edge**
- Independently recurse on each graph induced by edges with the same assigned subspace
Subspace assignment
"There are many subspaces that are large enough"
Subspace assignment

$\exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq |L_e| / (k H_p)$
Subspace assignment
Subspace assignment

\( \exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq |L_e| / (k H_p) \)
Subspace assignment

\[ \exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq \frac{|L_e|}{k H_p} \]
Subspace assignment

\[ \exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq |L_e| / (k H_p) \]

Simple case:
Subspace assignment

\[ \exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq |L_e| / (k H_p) \]

Simple case:
- \( k \) is the same for all edges
Subspace assignment

\[ \exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq |L_e| / (k H_p) \]

Simple case:
- \( k \) is the same for all edges

Goal:
Subspace assignment

\( \exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq |L_e| / (k H_p) \)

Simple case:
- \( k \) is the same for all edges

Goal:
- assign a list to each edge such that "few" neighboring edges have the same list
Subspace assignment
Subspace assignment

How:
Subspace assignment

How:
• Transform this problem into a list coloring instance
Subspace assignment

How:
• Transform this problem into a list coloring instance
Subspace assignment

How:
• Transform this problem into a list coloring instance
• Modify the graph such that the edge degree is at most $k-1$
Subspace assignment

How:
• Transform this problem into a list coloring instance
• Modify the graph such that the edge degree is at most $k-1$
Subspace assignment
Subspace assignment

\[ \exists k \text{ s.t. there are at least } k \text{ lists } C_i \text{ satisfying } |C_i \cap L_e| \geq |L_e| / (k H_p) \]

Different edges may have different \( k \). Solution:
• Split edges in **buckets** with "similar" values of \( k \)
• Solve each bucket **sequentially** as in the simple case

\[
\begin{align*}
  k & = & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & 17 & \ldots
\end{align*}
\]
∃k s.t. there are at least k lists $C_i$ satisfying $|C_i \cap L_e| \geq |L_e| / (k H_p)$

Different edges may have different $k$. Solution:

- Split edges in buckets with "similar" values of $k$
- Solve each bucket sequentially as in the simple case

$k = 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10 \ 11 \ 12 \ 13 \ 14 \ 15 \ 16 \ 17 \ ...$
Subspace assignment

\[
\begin{align*}
\{1, 3, 8, 9, 14\} & \rightarrow 3 \\
\{3, 4\} & \rightarrow 3 \\
\{1, 3, 8, 9, 14\} & \\
\{2, 3, 9, 10, 14\} & 
\end{align*}
\]
Subspace assignment

{1,3,8,9,14} → 3

{1,3,8,9,14}

{2,3,9,10,14}
Subspace assignment

Solution:

- Make some edges inactive, based on their current defect
- More recursion!
Relaxed list edge coloring

\[ T(\beta, C) \leq \log p \cdot T(1, p) + T(\beta/\text{polylog } p, C/p) \]

- Split the color space into many independent subspaces
- Assign a subspace to each edge
- Independently recurse on each graph induced by edges with the same assigned subspace
Putting things together

• Express \((\text{degree}(e) + 1)\)-list edge coloring as a function of relaxed list edge coloring
  ‣ Create many instances with smaller degree
  ‣ Handle instances sequentially
  ‣ Recurse

• Express relaxed list edge coloring as a function of smaller list edge colouring instances
  ‣ Split the color space in many parts
  ‣ Assign subspaces by solving many new list coloring instances
  ‣ Recurse
Open questions: CONGEST model

• Can we adapt this algorithm to work in the CONGEST model?
  ‣ If we assign colors to some edges, we have to remove those colors from the lists of their neighboring edges
  ‣ Nodes incident on the same edge need to agree on the new list
  ‣ Valid colors are the intersection of colors that are good for each side
Open questions: upper bounds

• We can solve \((2\Delta - 1)\)-edge coloring in \(\text{quasi-polylog}(\Delta) + O(\log^* n)\)
  
  ‣ Can we solve it in \(\text{polylog}(\Delta) + O(\log^* n)\) rounds?

• Can we solve vertex coloring in \(\text{subpoly}(\Delta)\)?
  
  ‣ Can we solve \(O(\Delta/c)\)-defective \(c\)-coloring fast?
Open questions: lower bounds

• Can we prove a non-trivial lower bound for solving \((2\Delta - 1)\)-edge coloring?
  ‣ Can we show that it cannot be solved in \(o(\log \Delta) + O(\log^* n)\)?