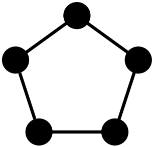
Distributed Recoloring

Marthe Bonamy, Nicolas Bousquet, Laurent Feuilloley, Marc Heinrich,
Paul Ouvrard, **Mikaël Rabie**, Jukka Suomela, Yara Uitto

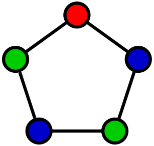
IRIF - Université Paris Cité

Thursday, March 17

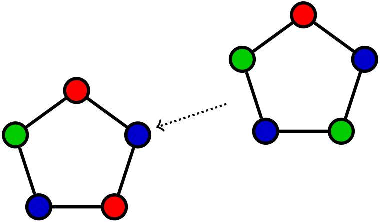
Recoloring



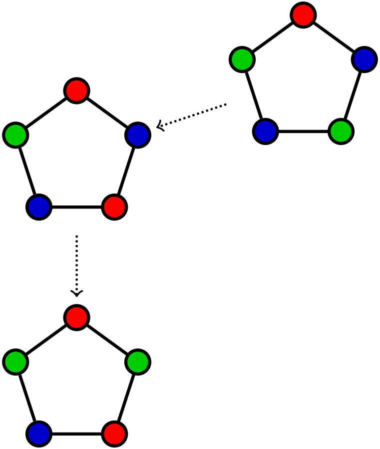
Recoloring



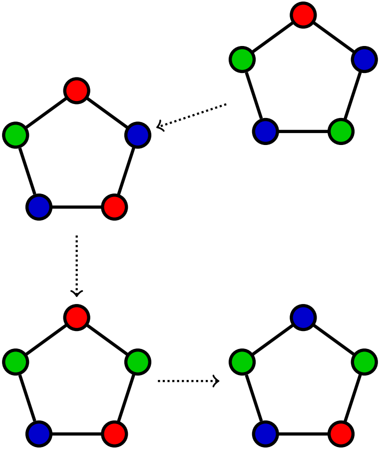
Recoloring



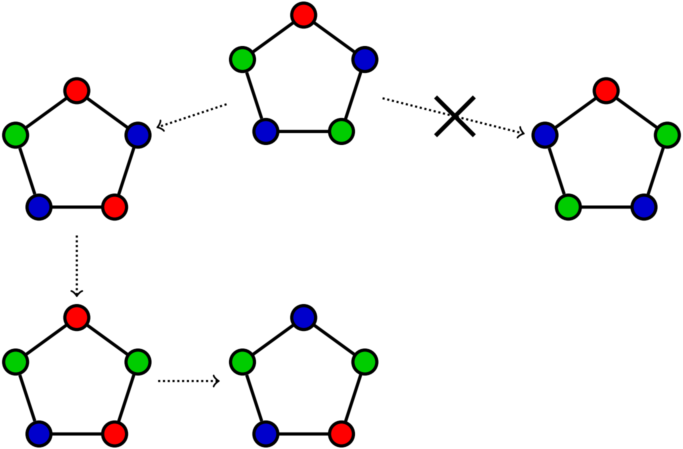
Recoloring



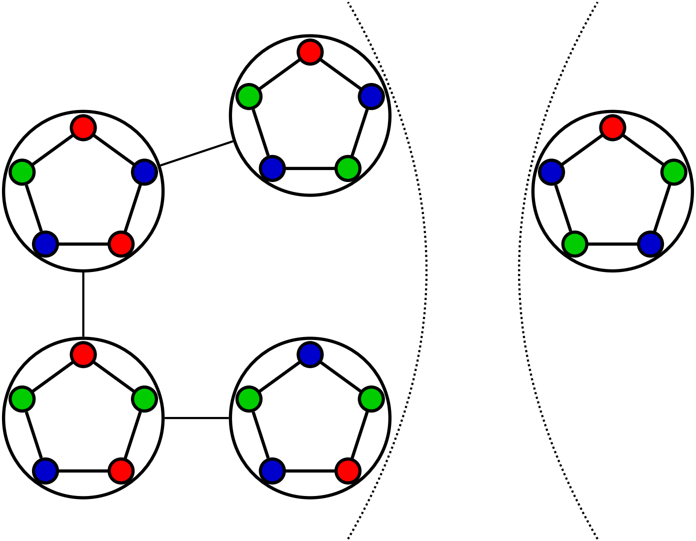
Recoloring



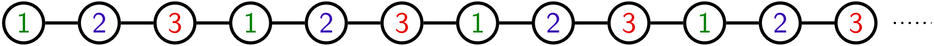
Recoloring



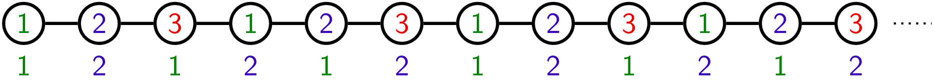
Recoloring



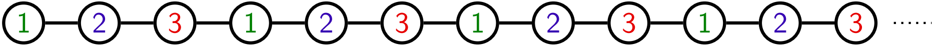
Recoloring a Path – 3 to 2 colors



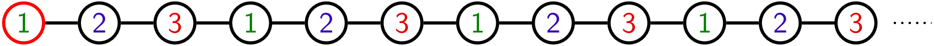
Recoloring a Path – 3 to 2 colors



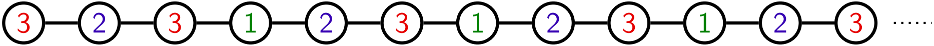
Recoloring a Path – 3 to 2 colors



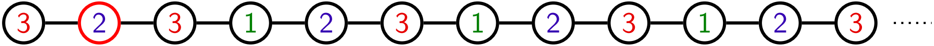
Recoloring a Path – 3 to 2 colors



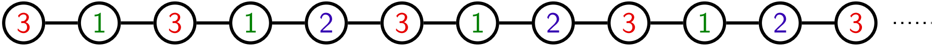
Recoloring a Path – 3 to 2 colors



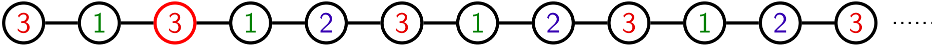
Recoloring a Path – 3 to 2 colors



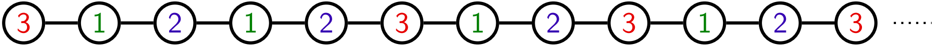
Recoloring a Path – 3 to 2 colors



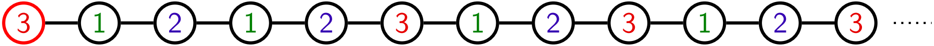
Recoloring a Path – 3 to 2 colors



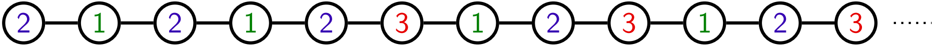
Recoloring a Path – 3 to 2 colors



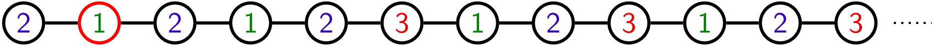
Recoloring a Path – 3 to 2 colors



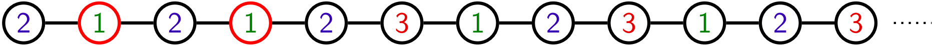
Recoloring a Path – 3 to 2 colors



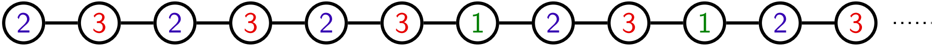
Recoloring a Path – 3 to 2 colors



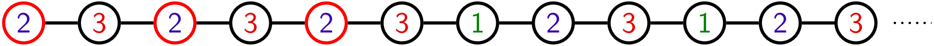
Recoloring a Path – 3 to 2 colors



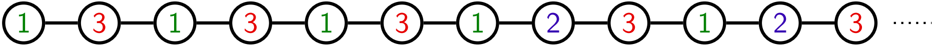
Recoloring a Path – 3 to 2 colors



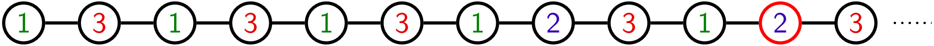
Recoloring a Path – 3 to 2 colors



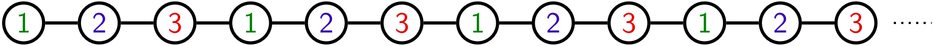
Recoloring a Path – 3 to 2 colors



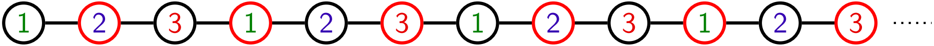
Recoloring a Path – 3 to 2 colors



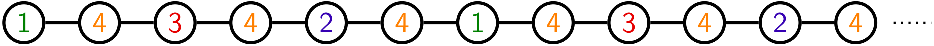
Recoloring a Path – 3 to 2 colors With an Extra Color



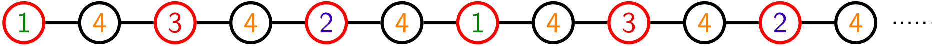
Recoloring a Path – 3 to 2 colors With an Extra Color



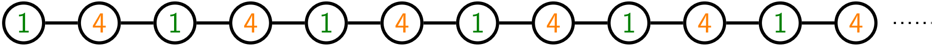
Recoloring a Path – 3 to 2 colors With an Extra Color



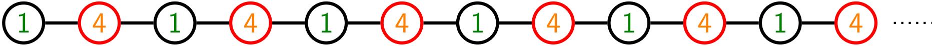
Recoloring a Path – 3 to 2 colors With an Extra Color



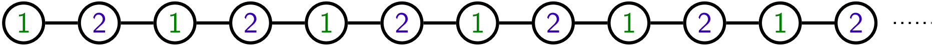
Recoloring a Path – 3 to 2 colors With an Extra Color



Recoloring a Path – 3 to 2 colors With an Extra Color



Recoloring a Path – 3 to 2 colors With an Extra Color

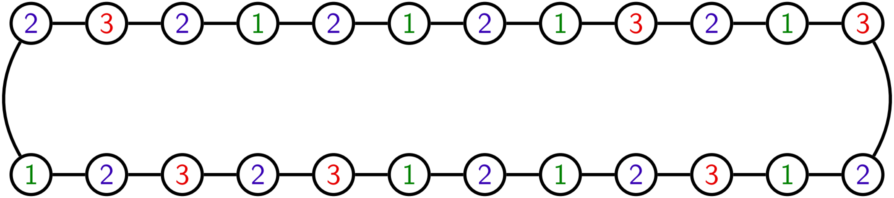


Problem Definition

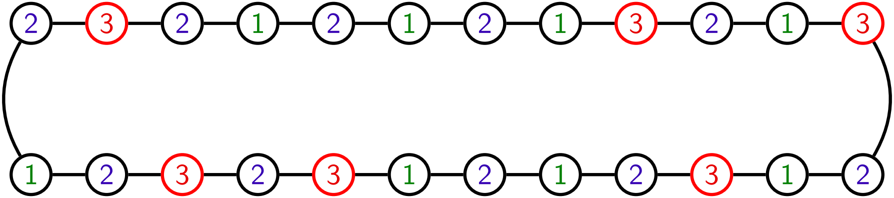
- Input :
  1. Graph G
  2. Two k colorings α and β
  3. c extra colors
- Output :
  1. Number r of communication rounds in LOCAL model
     $\Rightarrow$ Each node has knowledge of neighborhood of distance $\leq r$.
  2. Recoloring schedule of length l for each node.
     At each step, the reconfigured nodes are independent.
     $\Rightarrow$ Schedule locally checkable.

- Global Problem :
  Given a class of graphs, k and c, determine $r(n)$ and $l(n)$, n being the number of nodes.

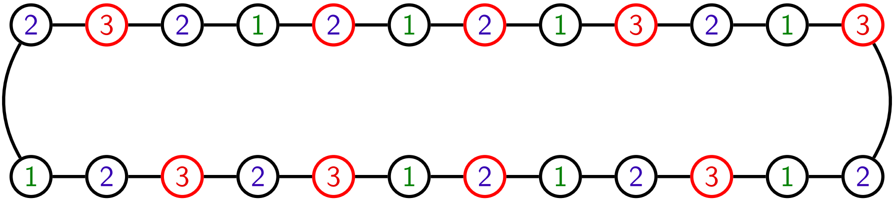
Distributed Recoloring of Cycles – 3+1



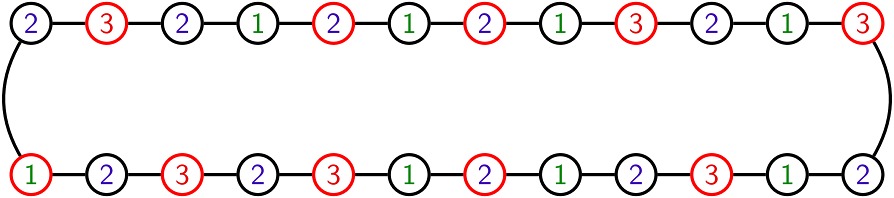
Distributed Recoloring of Cycles – 3+1



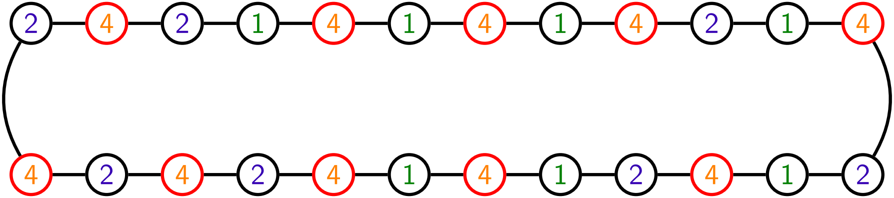
Distributed Recoloring of Cycles – 3+1



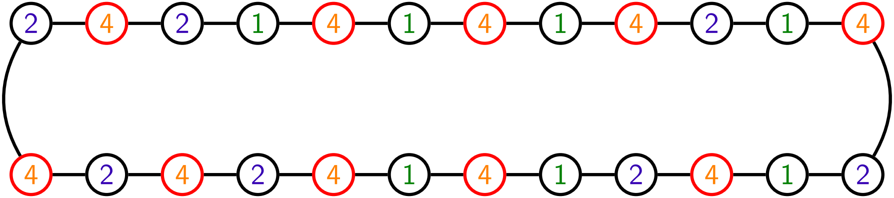
Distributed Recoloring of Cycles – 3+1



Distributed Recoloring of Cycles – 3+1



Distributed Recoloring of Cycles – 3+1

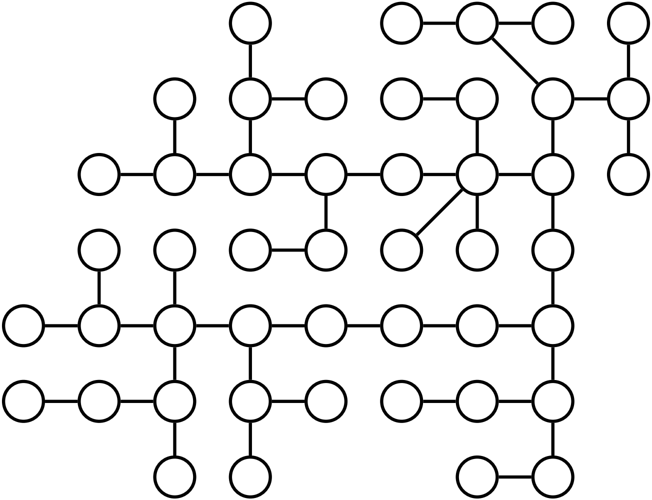


- $\mathcal{O}(1)$ communication rounds.
- $\mathcal{O}(1)$ recoloring schedule.

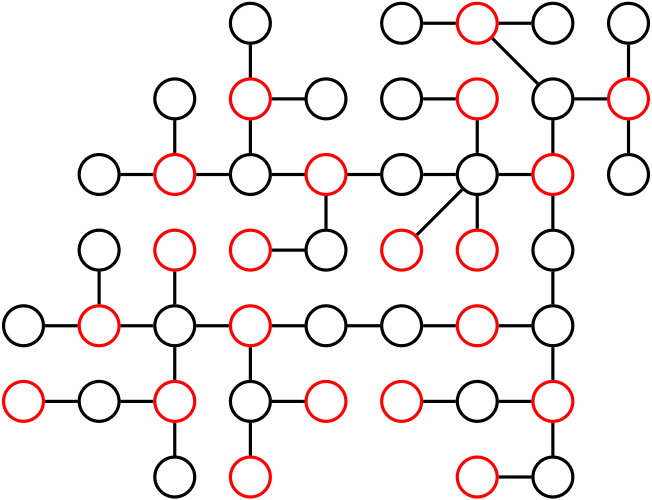
Tree Recoloring Results

input colors	extra colors	schedule length	communication rounds
2	0	∞	
2	1	$\mathcal{O}(1)$	0
3	0	$\Theta(n)$	$\Theta(n)$
3	1	$\mathcal{O}(1)$	$\mathcal{O}(\log n)$
3	2	$\mathcal{O}(1)$	0
4	0	$\Theta(\log n)$	$\Theta(\log n)$
4	1	$\mathcal{O}(1)$	$\mathcal{O}(\log n)$
4	2	$\mathcal{O}(1)$	?
4	3	$\mathcal{O}(1)$	0

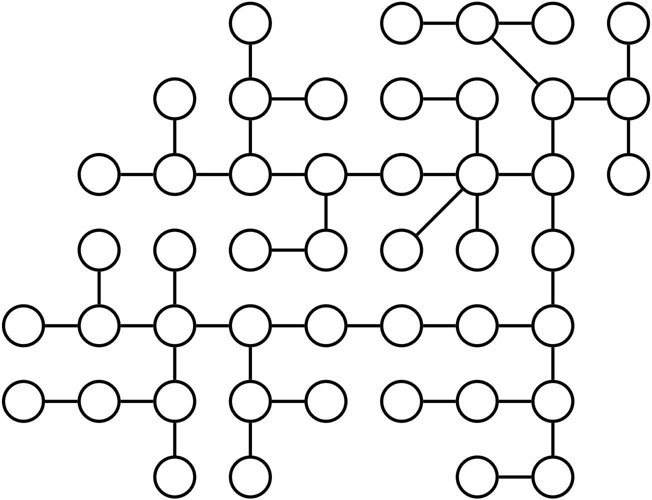
Tree Shattering



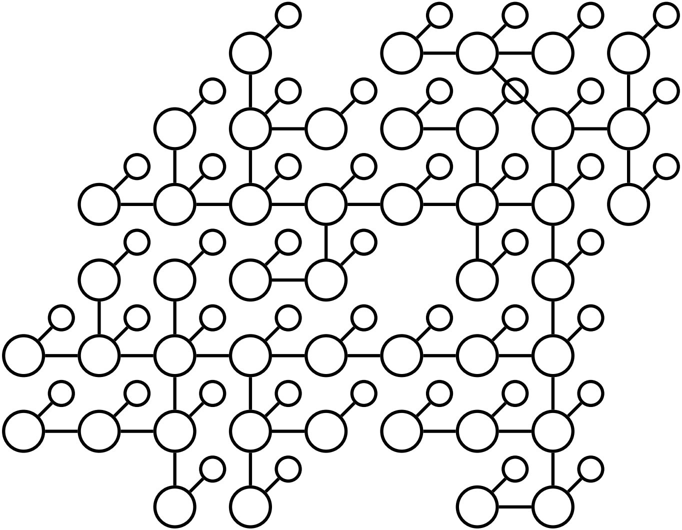
Tree Shattering



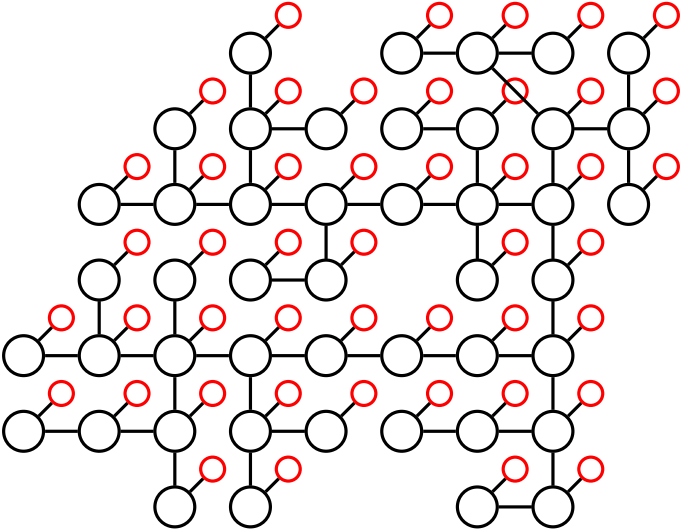
Tree Shattering



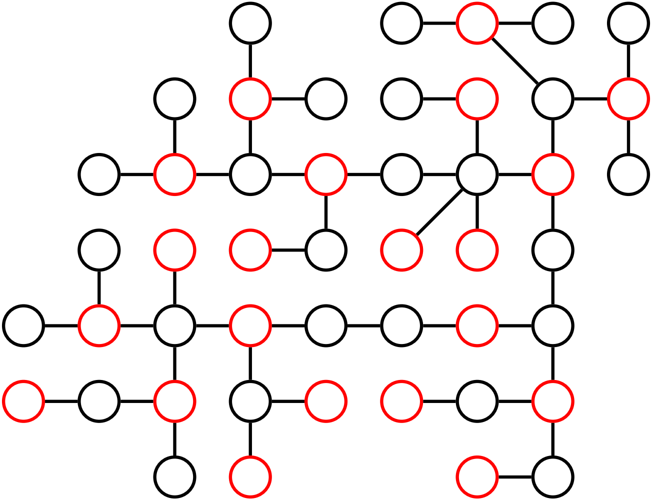
Tree Shattering



Tree Shattering



Tree Shattering



Toroidal Grids Recoloring Results

input colors	extra colors	schedule length	communication rounds
2	0	∞	
2	1	$\mathcal{O}(1)$	0
3	0	∞	
3	1	∞	
3	2	$\mathcal{O}(1)$	0
4	0	∞	
4	1	$\mathcal{O}(1)$	?
4	2	$\mathcal{O}(1)$	$\mathcal{O}(1)$
4	3	$\mathcal{O}(1)$	0
5	0	∞	
5	1	$\mathcal{O}(1)$	$\mathcal{O}(1)$
5	4	$\mathcal{O}(1)$	0
6	0	$\mathcal{O}(1)$	$\mathcal{O}(1)$

Toroidal Grids Recoloring Results

input colors	extra colors	schedule length	communication rounds
2	0	∞	
2	1	$\mathcal{O}(1)$	0
3	0	∞	
3	1	∞	
3	2	$\mathcal{O}(1)$	0
4	0	∞	
4	1	$\mathcal{O}(1)$	$\mathcal{O}(\log^* n)$ (2022)
4	2	$\mathcal{O}(1)$	$\mathcal{O}(1)$
4	3	$\mathcal{O}(1)$	0
5	0	∞	
5	1	$\mathcal{O}(1)$	$\mathcal{O}(1)$
5	4	$\mathcal{O}(1)$	0
6	0	$\mathcal{O}(1)$	$\mathcal{O}(1)$

Toroidal Grids Impossibility

3	2	1	2	1	2
1	3	2	1	2	1
2	1	3	2	1	2
1	2	1	3	2	1
2	1	2	1	3	2
1	2	1	2	1	3

$\Rightarrow$

1	2	1	2	1	3
2	1	2	1	3	2
1	2	1	3	2	1
2	1	3	2	1	2
1	3	2	1	2	1
3	2	1	2	1	2

Toroidal Grids Impossibility

3	4	1	4	1	4
4	3	4	1	4	1
2	4	3	4	1	4
4	2	4	3	4	1
2	4	2	4	3	4
4	2	4	2	4	3

$\Rightarrow$

1	2	1	2	1	3
2	1	2	1	3	2
1	2	1	3	2	1
2	1	3	2	1	2
1	3	2	1	2	1
3	2	1	2	1	2

Toroidal Grids Impossibility

3	2	1	2	1
1	3	2	1	2
2	1	3	2	1
1	2	1	3	2
2	1	2	1	3

$\not\Rightarrow$

2	1	2	1	3
1	2	1	3	2
2	1	3	2	1
1	3	2	1	2
3	2	1	2	1

Toroidal Grids Impossibility

3	2	1	2	1
1	3	2	1	2
2	1	3	2	1
1	2	1	3	2
2	1	2	1	3

$\not\Rightarrow$

2	1	2	1	3
1	2	1	3	2
2	1	3	2	1
1	3	2	1	2
3	2	1	2	1

2	3
3	1

1	3
3	2

	X	

$\Rightarrow$

	Y	

Recoloring Results

Recoloring Interval Graphs

Let G be an interval graph and α, β be two proper k-colorings of G. It is possible to find a schedule to transform α into β in the LOCAL model in $\mathcal{O}(\text{poly}(\Delta)\log^* n)$ rounds using at most :

- c additional colors, with $c = \omega - k + 4$, if $k \leq \omega + 2$, with a schedule of length poly(Δ),
- 1 additional color if $k \geq \omega + 3$, with a schedule of length poly(Δ),
- no additional color if $k \geq 2\omega$ with a schedule of exponential-in-Δ length.
- no additional color if $k \geq 4\omega$ with a schedule of length $\mathcal{O}(\omega\Delta)$.

Recoloring Results

Recoloring Chordal Graphs

Let G be a chordal graph and α, β be two proper k-colorings of G. It is possible to find a schedule of length $n^{\mathcal{O}(\log \Delta)}$ to transform α into β in $\mathcal{O}(\omega^2 \Delta^2 \log n)$ rounds in the LOCAL model using at most :

- c additional colors, with $c = \omega - k + 4$, if $k \leq \omega + 2$,
- 1 additional color if $k \geq \omega + 3$.

Coloring Results

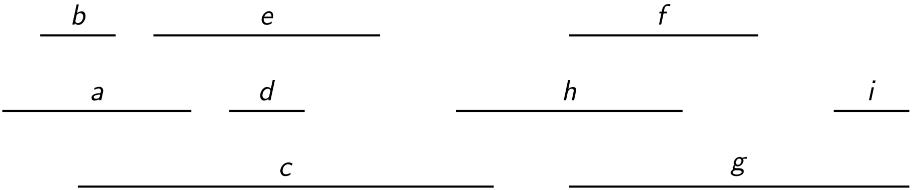
Coloring Interval Graphs

Interval graphs can be colored with $(\omega + 1)$-colors in $\mathcal{O}(\omega \log^* n)$ rounds in the LOCAL model.

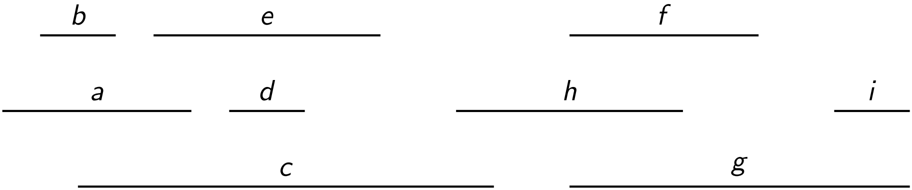
Coloring Chordal Graphs

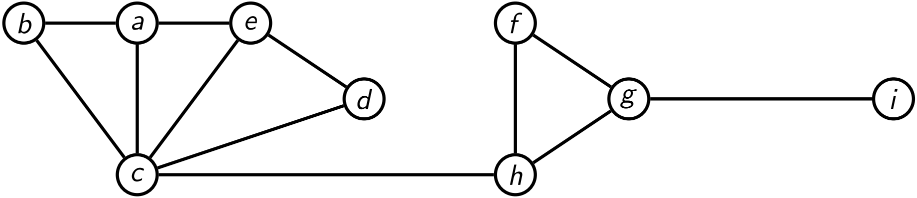
Chordal graphs can be colored with $(\omega + 1)$-colors in $\mathcal{O}(\omega \log n)$ rounds in the LOCAL model.

Interval Graphs

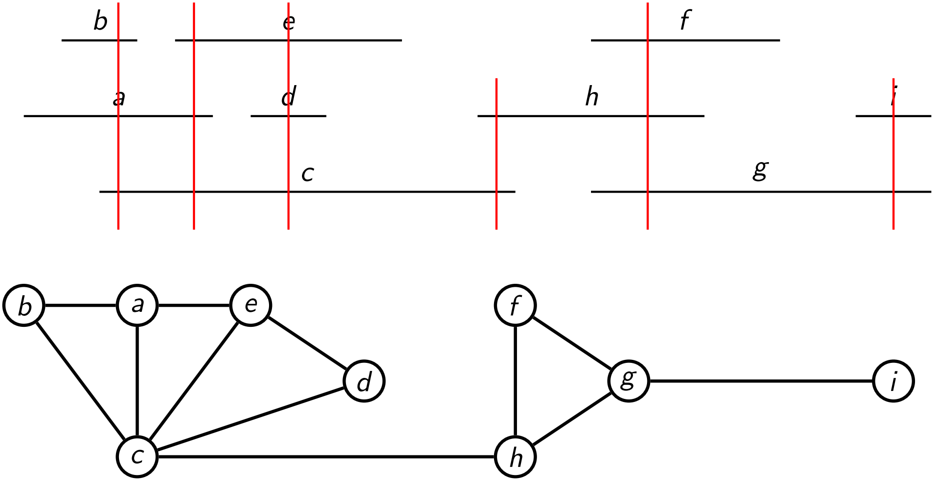


Interval Graphs

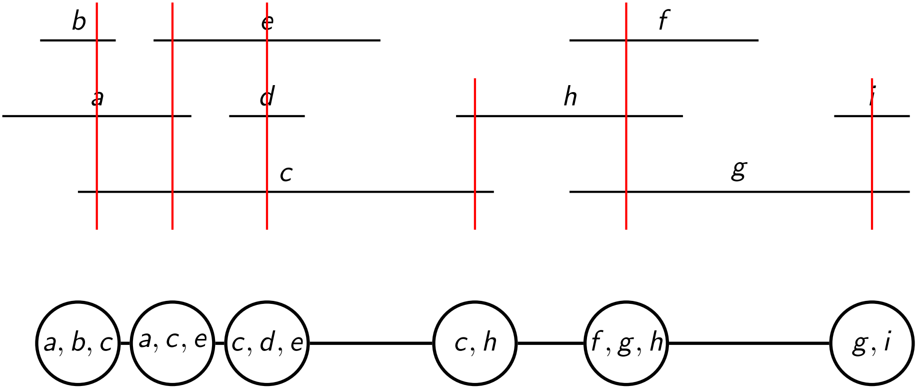




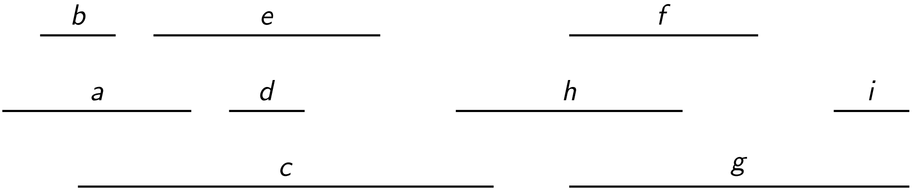
Interval Graphs



Interval Graphs



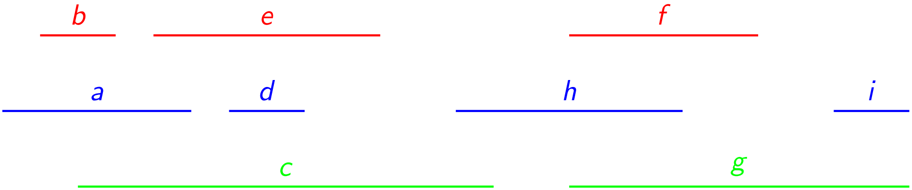
Interval Graphs



Properties of Interval Graphs :

- **Clique path** : maximal cliques form a path. Each node appears in consecutive cliques.
- **Coloring** : can always be colored with ω colors, ω being the size of largest clique.
- **Max Degree** : Δ can be arbitrarily large compared to ω.

Interval Graphs



Properties of Interval Graphs :

- **Clique path** : maximal cliques form a path. Each node appears in consecutive cliques.
- **Coloring** : can always be colored with ω colors, ω being the size of largest clique.
- **Max Degree** : Δ can be arbitrarily large compared to ω.

Roadmap

Goal : find a schedule from α to β

- Compute a canonical $\omega + 1$-coloring γ

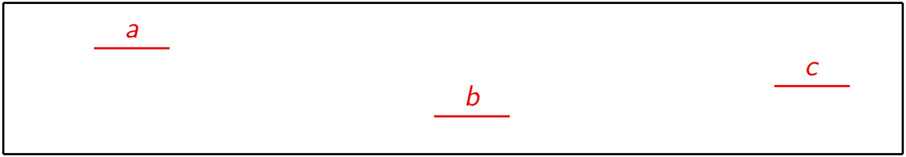
  $\Rightarrow$ **New goal :** Find a schedule from α to γ

- Reach a coloring γ' from α such that :
  - We use two extra colors
  - γ' and γ match on subintervals of length L at distance D

- Reach γ from γ' on each subinterval graph

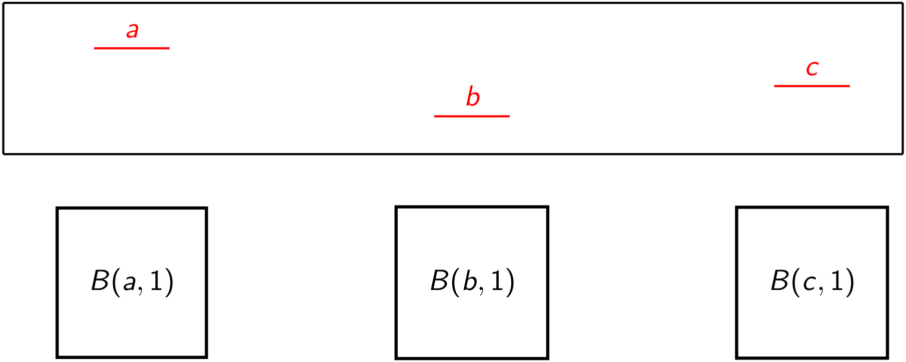
Boxes and Interboxes partition

- Compute a $(4, 5)$-ruling set S of G.
- For any $s \in S$, the **box** of s is $\{s\} \cup N(s) = B(s, 1)$
- The nodes that are in a path between s_1 and s_2, but not in a box, are in the **interbox** between s_1 and s_2.



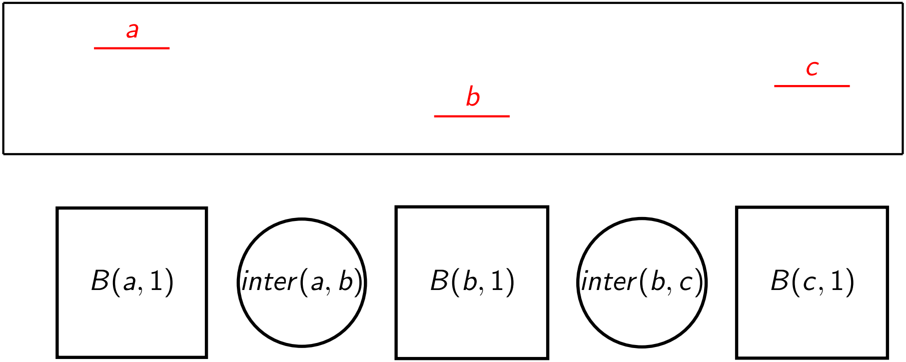
Boxes and Interboxes partition

- Compute a $(4,5)$-ruling set S of G.
- For any $s \in S$, the **box** of s is $\{s\} \cup N(s) = B(s,1)$
- The nodes that are in a path between s_1 and s_2, but not in a box, are in the **interbox** between s_1 and s_2.



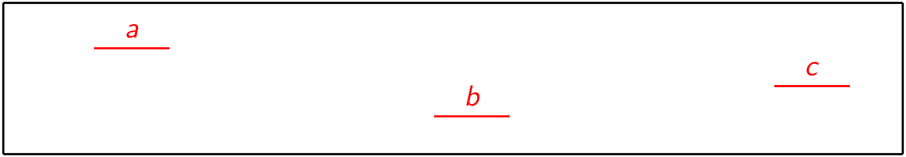
Boxes and Interboxes partition

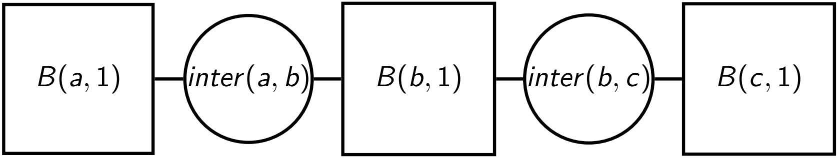
- Compute a $(4, 5)$-ruling set S of G.
- For any $s \in S$, the **box** of s is $\{s\} \cup N(s) = B(s, 1)$
- The nodes that are in a path between s_1 and s_2, but not in a box, are in the **interbox** between s_1 and s_2.



Boxes and Interboxes partition

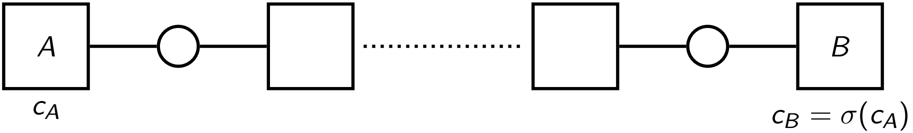
- Compute a $(4, 5)$-ruling set S of G.
- For any $s \in S$, the **box** of s is $\{s\} \cup N(s) = B(s, 1)$
- The nodes that are in a path between s_1 and s_2, but not in a box, are in the **interbox** between s_1 and s_2.





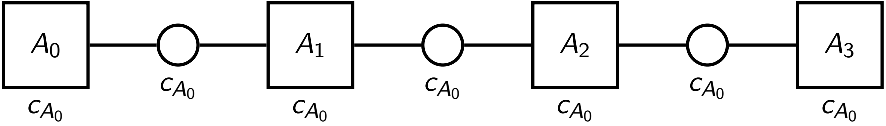
$\omega + 1$-coloring of the graph

- Compute a maximal independent set I at distance 3ω of S.
- Compute a coloring of the boxes of I.
- For two consecutive boxes A and B of I, c_B is a permutation of c_A.
- Perform up to ω inversions to reach c_B from c_A.



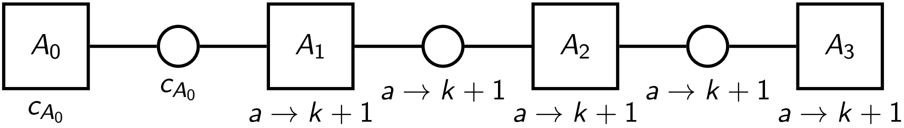
$\omega + 1$-coloring of the graph

- Compute a maximal independent set I at distance 3ω of S.
- Compute a coloring of the boxes of I.
- For two consecutive boxes A and B of I, c_B is a permutation of c_A.
- Perform up to ω inversions to reach c_B from c_A.



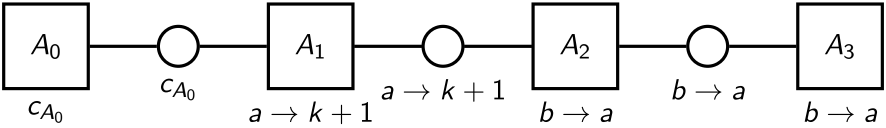
$\omega + 1$-coloring of the graph

- Compute a maximal independent set I at distance 3ω of S.
- Compute a coloring of the boxes of I.
- For two consecutive boxes A and B of I, c_B is a permutation of c_A.
- Perform up to ω inversions to reach c_B from c_A.



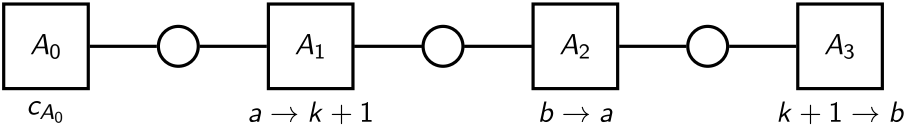
$\omega + 1$-coloring of the graph

- Compute a maximal independent set I at distance 3ω of S.
- Compute a coloring of the boxes of I.
- For two consecutive boxes A and B of I, c_B is a permutation of c_A.
- Perform up to ω inversions to reach c_B from c_A.



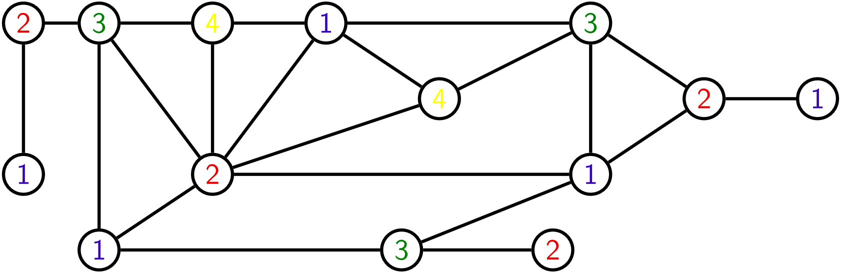
$\omega + 1$-coloring of the graph

- Compute a maximal independent set I at distance 3ω of S.
- Compute a coloring of the boxes of I.
- For two consecutive boxes A and B of I, c_B is a permutation of c_A.
- Perform up to ω inversions to reach c_B from c_A.



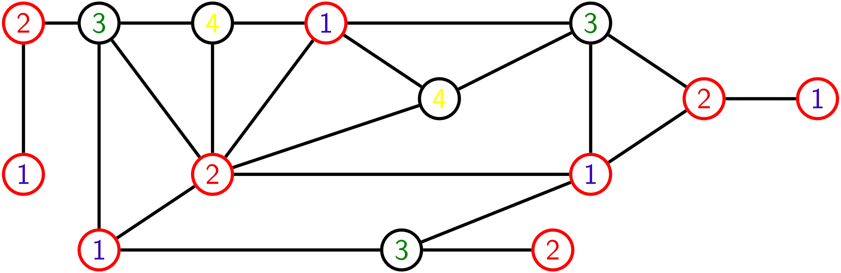
Kempe Recoloring

- Select two colors *a* and *b*.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.



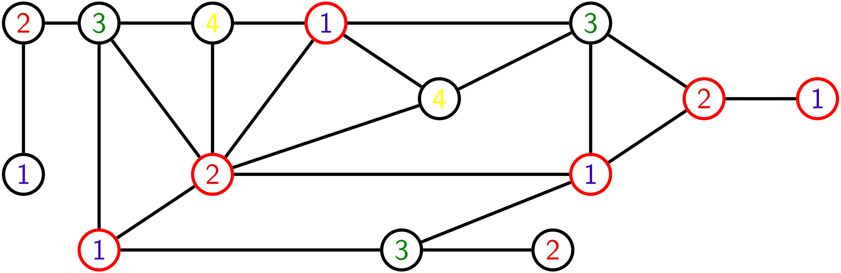
Kempe Recoloring

- Select two colors *a* and *b*.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.



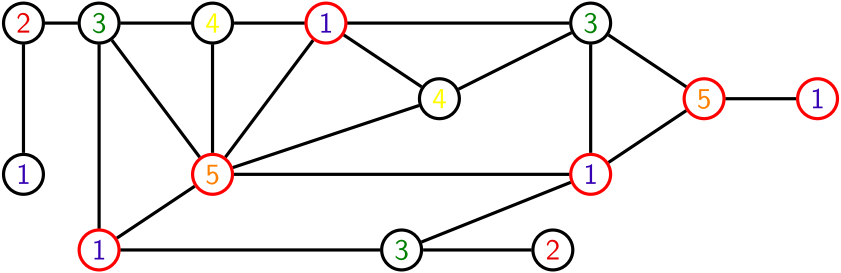
Kempe Recoloring

- Select two colors a and b.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.



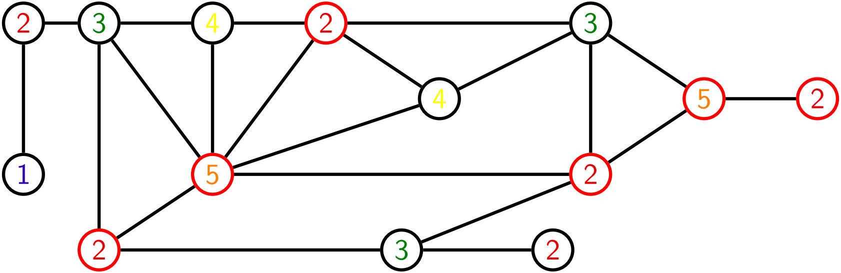
Kempe Recoloring

- Select two colors *a* and *b*.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.



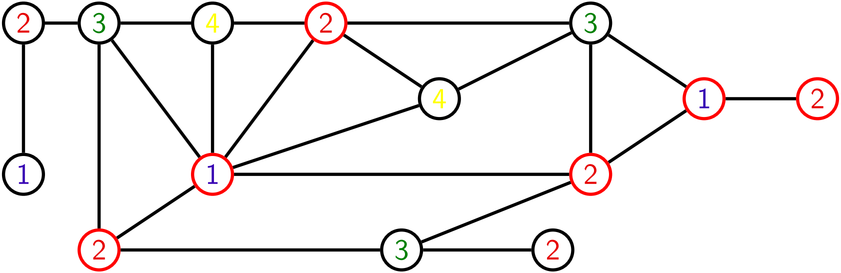
Kempe Recoloring

- Select two colors a and b.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.



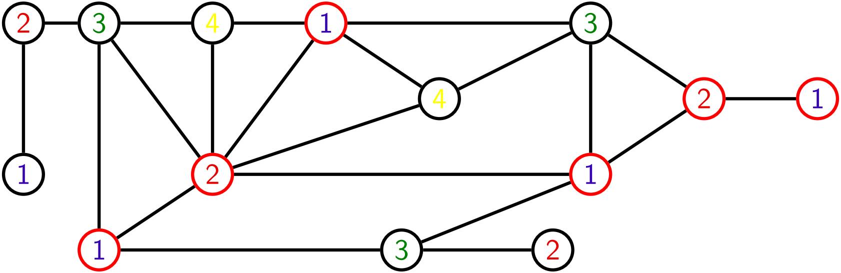
Kempe Recoloring

- Select two colors *a* and *b*.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.



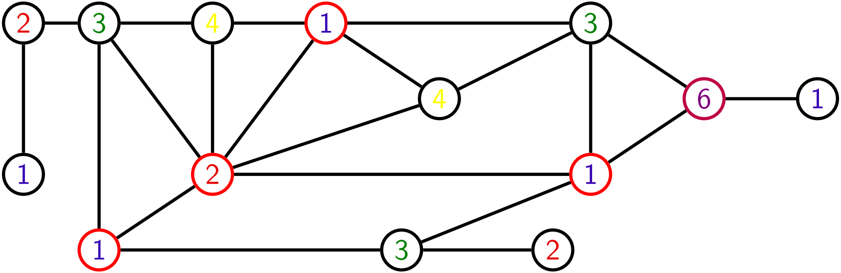
Local Kempe Recoloring

- Select two colors *a* and *b*.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.



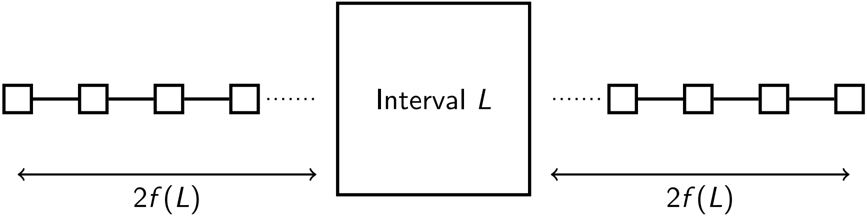
Local Kempe Recoloring

- Select two colors *a* and *b*.
- Select a connected component of nodes of those colors.
- With an extra color, switch the color of those nodes.
- With one more extra color, bound the component



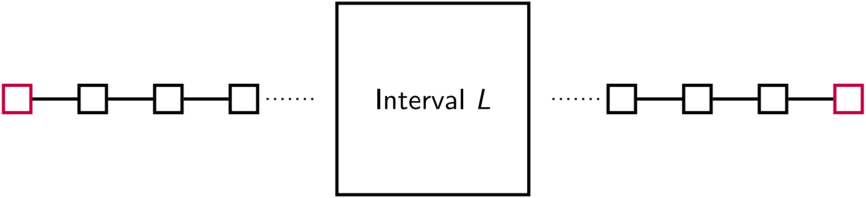
Sequencing Kempe recolorings

- To recolor a subinterval of length $L \Rightarrow f(L)$ Kempe recolorings.
- Need $2f(L)$ blocks on both ends of the interval.
- Iterate Kempe recolorings by adding bounds with the extra color.



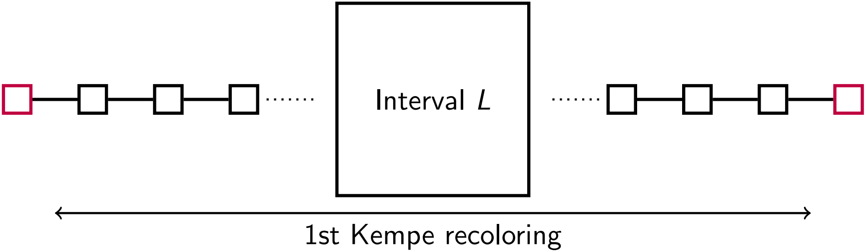
Sequencing Kempe recolorings

- To recolor a subinterval of length $L \Rightarrow f(L)$ Kempe recolorings.
- Need $2f(L)$ blocks on both ends of the interval.
- Iterate Kempe recolorings by adding bounds with the extra color.



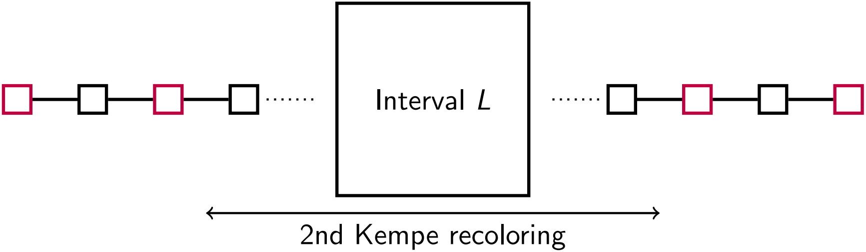
Sequencing Kempe recolorings

- To recolor a subinterval of length $L \Rightarrow f(L)$ Kempe recolorings.
- Need $2f(L)$ blocks on both ends of the interval.
- Iterate Kempe recolorings by adding bounds with the extra color.



Sequencing Kempe recolorings

- To recolor a subinterval of length $L \Rightarrow f(L)$ Kempe recolorings.
- Need $2f(L)$ blocks on both ends of the interval.
- Iterate Kempe recolorings by adding bounds with the extra color.



Completing the coloring

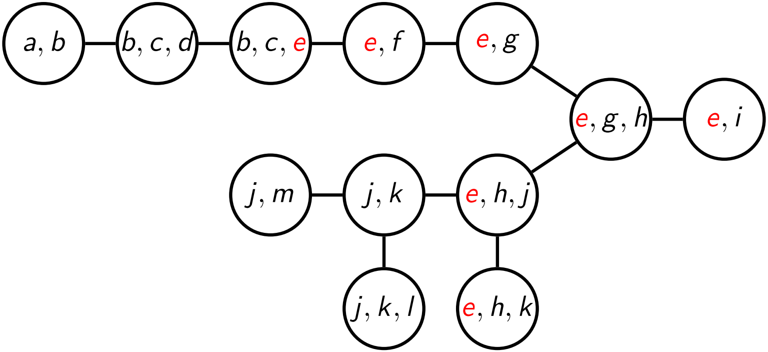
We have an alternation of intervals corresponding to γ and intervals $k+1$-colored.
We use as a blackbox the corresponding algorithm :

Bartier and Bousquet, ESA 2019

Let G be an interval graph with clique path $\mathcal{P}$ (given with an ordering). Let γ', γ be two colorings. Let k' be the number of colors in γ and $k \geq k'+3$. Let Y be a set of consecutive cliques of G such that γ' corresponds to γ on its borders of length at least L. Let C be the first clique of Y. Then we can obtain a coloring γ'' from γ such that :

- Only vertices that belong to vertices in Y are recolored.
- No vertex of C is recolored. In particular $\gamma''_C = \gamma_C$.
- The coloring γ'' restricted to the N cliques Z starting in the clique after C correspond to γ.
- The total length of the schedule is poly($|Y|, k$).

Clique Tree



Properties of Chordal Graphs :

- **Clique tree** : partition into cliques forming a tree. Each node forms a subtree.
- **Coloring** : can always be colored with ω colors, ω being the size of largest clique.

Idea of the Generalization

- Rake and Compress :
  - At each step, remove leafs and long paths.
  - Level of a node : step when it is fully removed.
  - After $\mathcal{O}(\log n)$ steps, everything is removed.
- Build the schedule from higher level to smaller.
- For long paths, act as interval graphs.
- For leafs (small diameter), compute optimal recoloring schedule.
- To go from level i to $i-1$, at each step of level $\geq i$,
  first recolor level $i-1$ nodes to avoid conflicts.

Recoloring Chordal Graphs

Let G be a chordal graph and α, β be two proper k-colorings of G. It is possible to find a schedule of length $n^{\mathcal{O}(\log \Delta)}$ to transform α into β in $\mathcal{O}(\omega^2 \Delta^2 \log n)$ rounds in the LOCAL model.

Main Results

Centralized Result

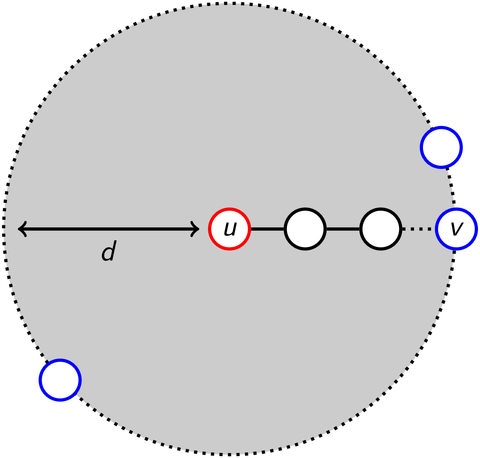
Let G be a connected graph with $\Delta \geq 3$ and α, β be two non-frozen k-colorings of G with $k \geq \Delta + 1$. Then we can transform α into β with a sequence of at most $\mathcal{O}(\Delta^{c\Delta} n)$ single vertex recolorings, where c is a constant.

Distributed Result

Let G be a graph with $\Delta \geq 3$. Let α, β be two $\Delta + 1$-colorings of G which are r-locally non-frozen. There exists three constants c, c', c'' such that we can transform α into β with a parallel schedule of length at most $\mathcal{O}(\Delta^{c\Delta + c' r})$ in $\mathcal{O}(\Delta^{c''} + \log^2 n \cdot \log^2 \Delta)$ rounds in the LOCAL model.

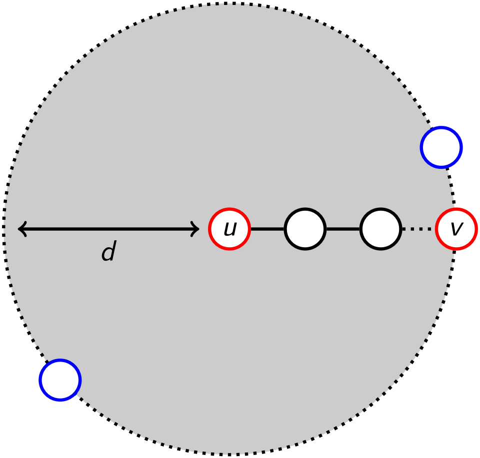
Unfreezing the Border of a Ball

Let u be a node an unfrozen node, and v a frozen node at distance d. There exists a schedule to unfreeze v in $2d$ rounds that changes color of nodes in $B(u, d-1)$.



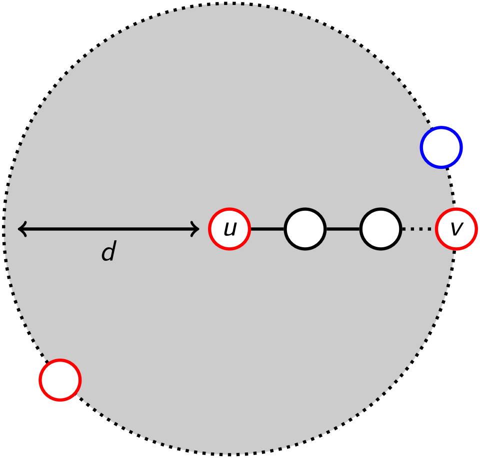
Unfreezing the Border of a Ball

Let u be a node an unfrozen node, and v a frozen node at distance d. There exists a schedule to unfreeze v in $2d$ rounds that changes color of nodes in $B(u, d-1)$.



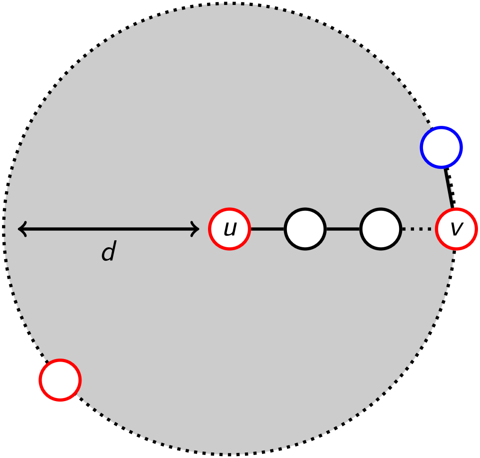
Unfreezing the Border of a Ball

Let u be a node an unfrozen node, and v a frozen node at distance d. There exists a schedule to unfreeze v in $2d$ rounds that changes color of nodes in $B(u, d-1)$.



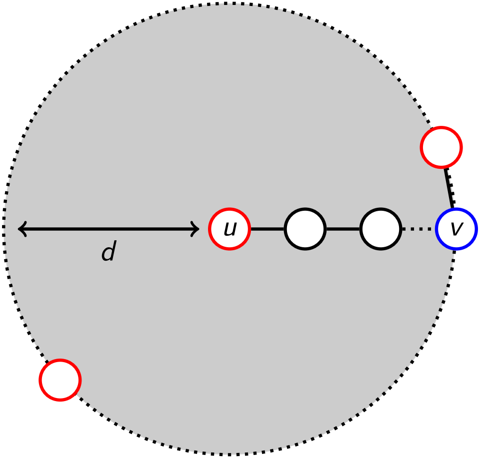
Unfreezing the Border of a Ball

Let u be a node an unfrozen node, and v a frozen node at distance d. There exists a schedule to unfreeze v in $2d$ rounds that changes color of nodes in $B(u, d-1)$.



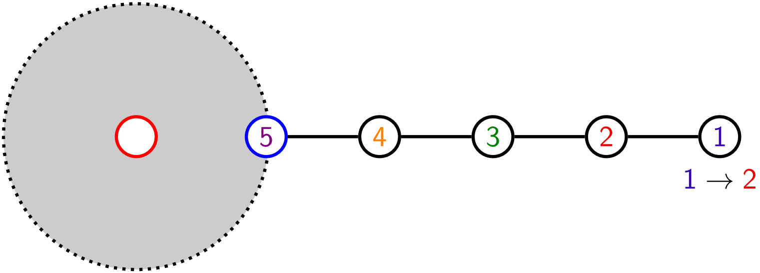
Unfreezing the Border of a Ball

Let u be a node an unfrozen node, and v a frozen node at distance d. There exists a schedule to unfreeze v in $2d$ rounds that changes color of nodes in $B(u, d-1)$.



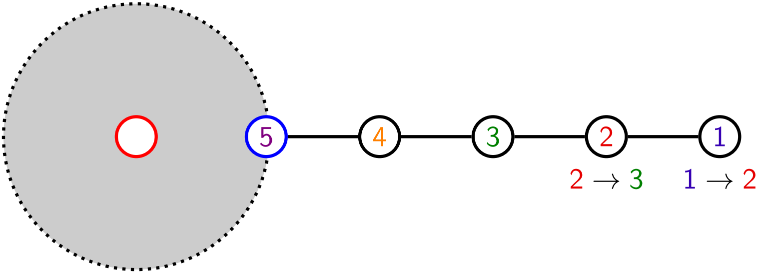
Outside of the Balls

- Compute a maximal independent set I at distance $2d$
- Consider the graph without the balls $B(u, d)$ for $u \in I$
- Recolor from the farthest nodes to the closest nodes to those balls



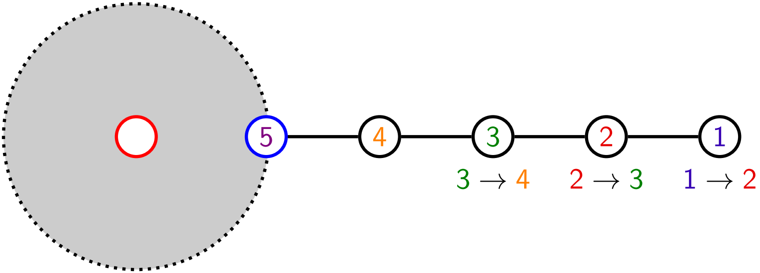
Outside of the Balls

- Compute a maximal independent set I at distance $2d$
- Consider the graph without the balls $B(u, d)$ for $u \in I$
- Recolor from the farthest nodes to the closest nodes to those balls



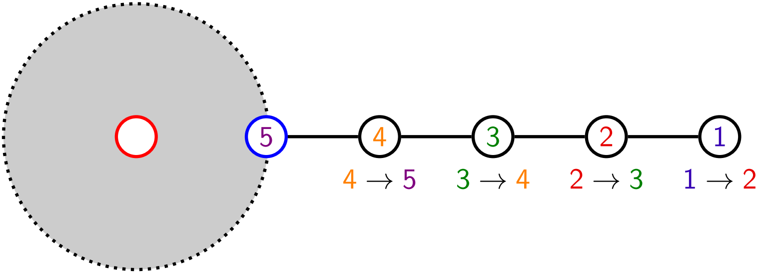
Outside of the Balls

- Compute a maximal independent set I at distance $2d$
- Consider the graph without the balls $B(u, d)$ for $u \in I$
- Recolor from the farthest nodes to the closest nodes to those balls



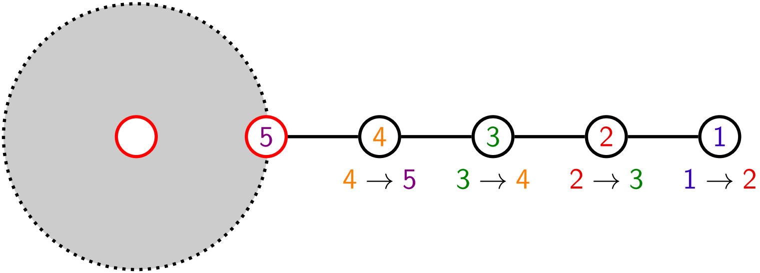
Outside of the Balls

- Compute a maximal independent set I at distance $2d$
- Consider the graph without the balls $B(u, d)$ for $u \in I$
- Recolor from the farthest nodes to the closest nodes to those balls



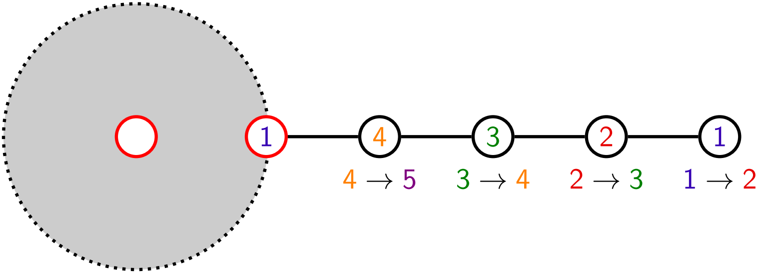
Outside of the Balls

- Compute a maximal independent set I at distance $2d$
- Consider the graph without the balls $B(u, d)$ for $u \in I$
- Recolor from the farthest nodes to the closest nodes to those balls



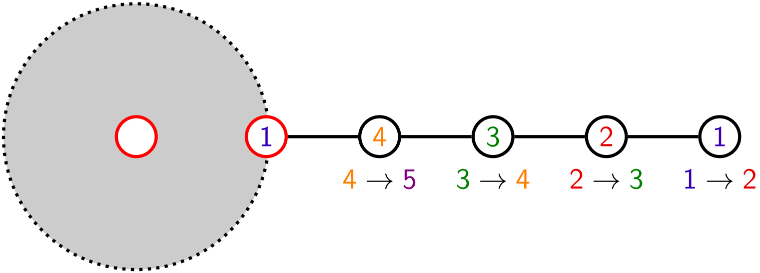
Outside of the Balls

- Compute a maximal independent set I at distance $2d$
- Consider the graph without the balls $B(u, d)$ for $u \in I$
- Recolor from the farthest nodes to the closest nodes to those balls



Outside of the Balls

- Compute a maximal independent set I at distance $2d$
- Consider the graph without the balls $B(u, d)$ for $u \in I$
- Recolor from the farthest nodes to the closest nodes to those balls



- A $\Delta^{O(d)}$ length schedule exists to recolor those nodes
- A schedule exists to recolor those balls (up to some extra nodes)

T H A N K

Y O U