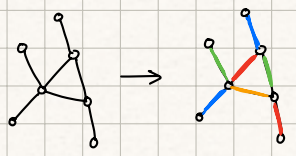
Part 1: Intro to Edge Coloring

29/8/25 Martín Costa

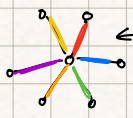


Input: Simple, undirected graph $G = (V, E)$.

Output: An edge coloring $\chi : E \to \mathcal{C}$ s.t. $\chi(e) \neq \chi(e')$ if e and e' share an endpoint

Remarks:

- Can assume that $\mathcal{C} = [k] := \{1, \dots, k\}$ for $k \in \mathbb{N}$
- Need $k \geq \Delta$, Δ = max-degree of G



← Need Δ different colors

Theorem [Vizing, '64]. Any graph can be $(\Delta+1)$-colored. $\quad \Delta \leq \chi'(G) \leq \Delta+1$

Theorem [Holyer, '81]. NP-Hard to determine if a graph is Δ-colorable.

→ Fast $(\Delta+1)$-coloring best we can hope for.

→ How fast can we compute such a coloring?

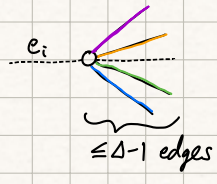
Vizing's Algorithm For Bipartite Graphs

Vizing-Bipartite(G):
1. Create an empty Δ-coloring χ of G
2. For each edge $e_1, \dots, e_m$:
   | Extend χ to e_i

We can find an $\{\alpha, \beta\}$-alternating path P starting at v

Extending χ:

At most $\Delta-1$ colors incident on u:

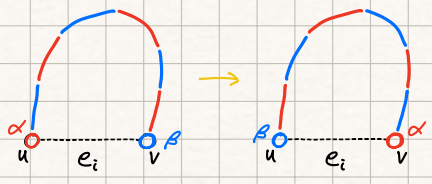


→ $\exists$ a 'missing' color $\alpha \in [\Delta]$ at u.

Flipping P changes which of α and β is missing at v

Crucial Observation:

If P ends at u, the algorithm fails:



← Cannot color e_i with α or β

Can't happen in bipartite graphs: forms odd cycle

Running Time

Time to extend the coloring to an edge = $O(\text{length of the alternating path}) = O(n)$.

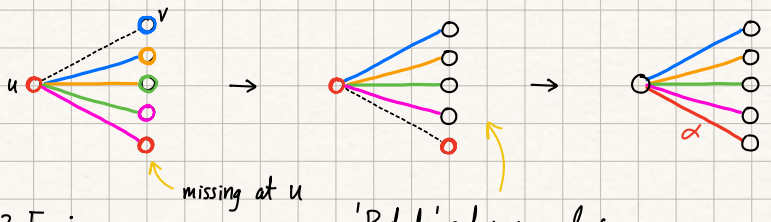
Total running time = $m \cdot O(n) = O(mn)$.

Vizing's Chains and Fans

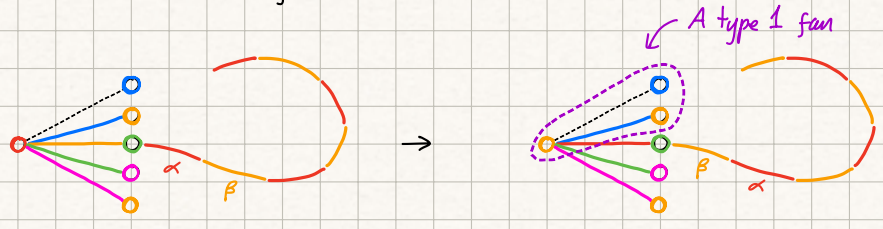
How can we extend a coloring to an edge in a general graph?

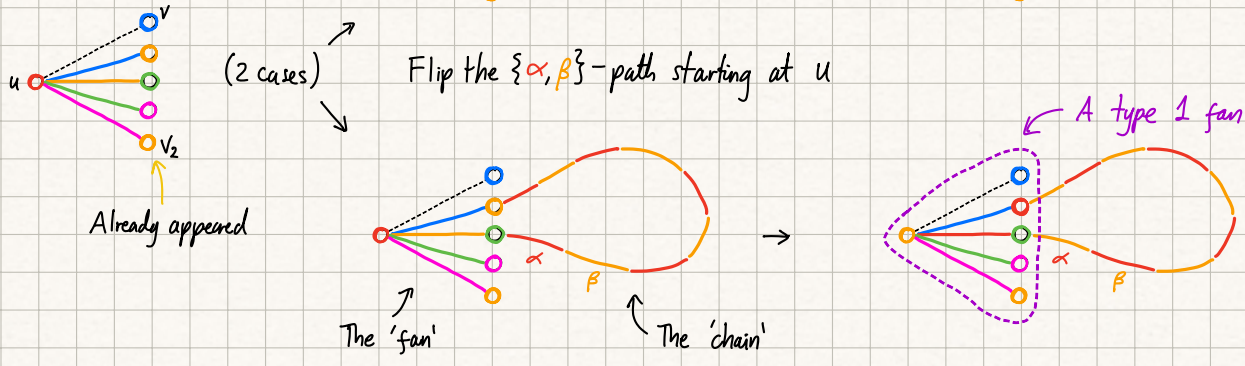
We can use Vizing fans and Vizing chains. Suppose we now have $\Delta+1$ colors.

Type 1 Fan:



Type 2 Fan:





Running Time

Time to extend the coloring to an edge $= O(\text{size of fan} + \text{size of chain}) = O(\Delta + n) = O(n)$.

Compute a $(\Delta+1)$-coloring in time $= m \cdot O(n) = O(mn)$.

Remarks:

- Need $\Delta+1$ colors to construct a fan.
- Vizing fans are complicated. Annoying to use algorithmically → often not too conceptually relevant, but makes things technical.
- For many algorithms, we can convey the main ideas by assuming bipartiteness.

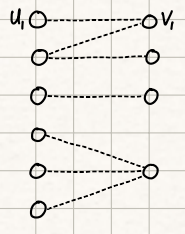
Sinnamon's $\tilde{O}(m\Delta)$ Time Algorithm

(Assume G is bipartite).

Random-Vizing(G):
1. Create an empty Δ-coloring χ of G
2. While there is an uncolored edge:
   - Sample a random uncolored edge e
   - Extend χ to e

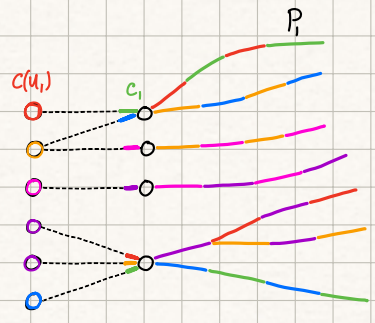
Theorem [Sinnamon, '19]. Random-Vizing runs in time $\tilde{O}(m\Delta)$.

Proof. Let $e_1=(u_1,v_1),\dots,e_\lambda=(u_\lambda,v_\lambda)$ be the uncolored edges.



1. Assign each u_i a missing color $c(u_i)$.
2. Assign each v_i a missig color c_i

Crucially: $v_i = v_j \Rightarrow c_i \neq c_j$.



P_i = the $\{c(u_i), c_i\}$-alternating path at v_i

Key Observation. All of these alternating paths are distinct.

The total length of all (maximal) APs is $O(m\Delta)$: Each colored edge appears in $\leq \Delta$ APs.

$\rightarrow$ λ APs + total length $O(m\Delta)$ $\Rightarrow$ expected length $O(m\Delta/\lambda)$.

$\rightarrow$ Total running time: $\sum_{\lambda=m}^{1} O(m\Delta/\lambda) = O(m\Delta \log n)$.

Remarks:

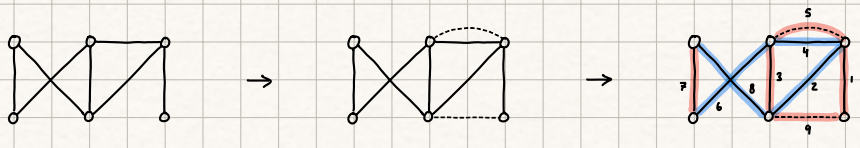
- Easy to extend to general graphs by using Vizing fans.
- Suffices to just randomly sample missing colors for APs.
- Different analysis gives $\tilde{O}(m\alpha)$, α = arboricity.
- These sorts of 'averaging arguments' are the core of <u>all</u> fast algorithms based on Vizing chains.

Euler Partitioning

How can we implement a divide-and-conquer approach?

Lemma. Given $G=(V,E)$ of max deg. Δ, can split into disjoint subgraphs $G_1=(V,E_1)$ and $G_2=(V,E_2)$ of max deg. $\lceil \Delta/2 \rceil$ in $O(m)$ time.

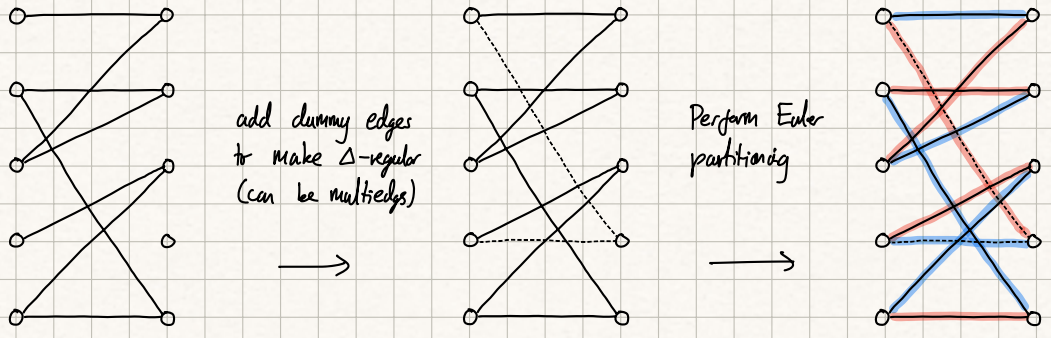
Idea: Iterate over an Eulerian tour and alternatively add edges to E_1 and E_2.



Application I: Bipartite Edge Coloring

Theorem [Cole, Ost, Schirra, '01]. Bipartite Δ-coloring in $O(m \log \Delta)$ time.

Idea: Suppose G has $\Delta = 2^k$ for $k \in \mathbb{N}$.



Since G is bipartite: $\Delta(G_1) = \Delta(G_2) = \frac{\Delta}{2}$

Since $\Delta = 2^k$, after $O(\log \Delta)$ steps, we have partitioned E into Δ matchings.

Time: $O(n\Delta) \cdot O(\log \Delta) = O(n\Delta \log \Delta)$.

- Can be extended to any Δ: If $\Delta \notin 2\mathbb{N}$, find a perfect matching in $O(m)$ time.
- Merge vertices u, v s.t. $\deg(u) + \deg(v) \leq \Delta \Rightarrow$ regularized graph has size $O(m)$ instead of $O(\Delta n)$.

Open Problem: Bipartite Δ-coloring in $O(m)$ time? (Even assuming $\Delta = 2^k$).

Application II: Reduction to Coloring a Matching

Suppose we want to compute a $(\Delta+1)$-coloring of G:

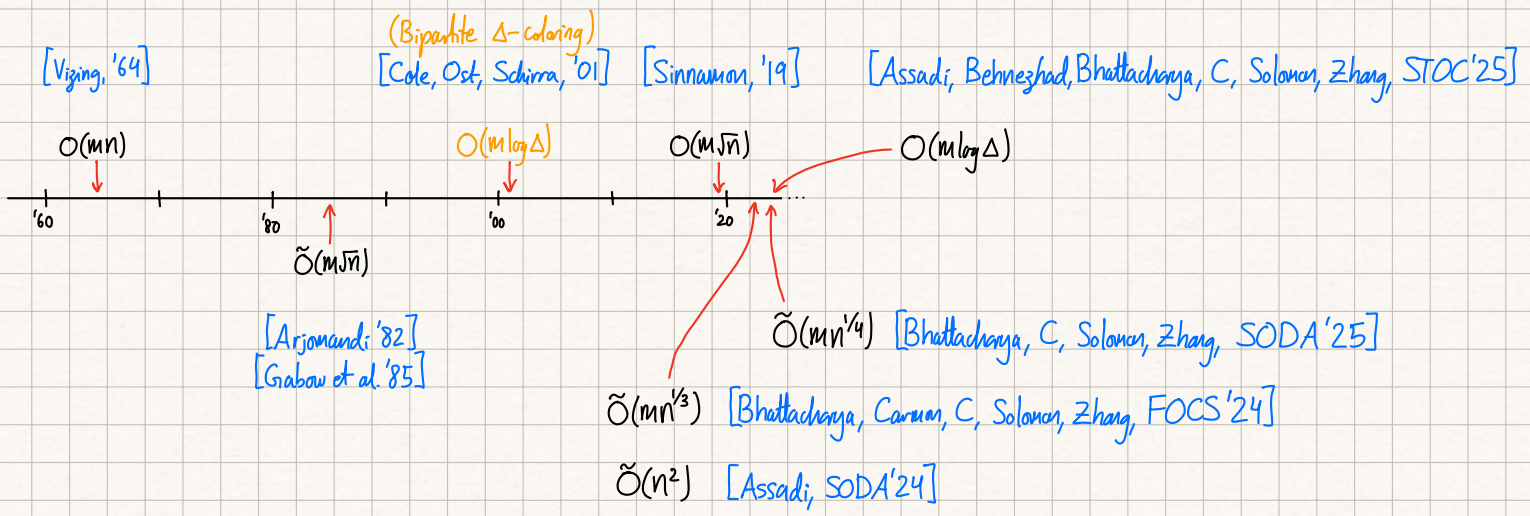
1. Split G into G_1 and G_2, s.t. $\Delta(G_1), \Delta(G_2) \leq \lceil \Delta/2 \rceil$.
2. Recursively compute a $(\Delta(G_i)+1)$-coloring χ_i of G_i.
3. Let $\chi \leftarrow \chi_1 \cup \chi_2$ be a $\Delta(G_1) + \Delta(G_2) + 2 \leq 2 \cdot \lceil \Delta/2 \rceil + 2 \leq \Delta + 3$ coloring of G.
4. Uncolor the 2 smallest color classes of χ: Leaves a $(\Delta+1)$-coloring with $O(m/\Delta)$ uncolored edges.
5. Extend χ to the uncolored edges.

Theorem [Grabow et al., '85]. $(\Delta+1)$-coloring in $\tilde{O}(m\sqrt{n})$ time.

Proof. Given a $(\Delta+1)$-coloring w/ $O(m/\Delta)$ uncolored edges, we can color them in $\tilde{O}(\min\{\Delta m, n \cdot m/\Delta\}) = \tilde{O}(m\sqrt{n})$ time. (Sinnamon: Δm; Vizing: $n \cdot m/\Delta$)

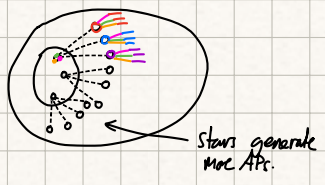
$\rightarrow T(m) = 2 \cdot T(m/2) + \tilde{O}(m\sqrt{n}) \quad \Rightarrow T(m) = \tilde{O}(m\sqrt{n})$.

Part 2: $(\Delta+1)$-Coloring in Near-Linear Time



$\tilde{O}(n^2)$ time alg: corollary of a $\tilde{O}(m)$ time alg. for $(\Delta+O(\log n))$-coloring. Based on random walks & peeling 'fair' matchings.

$\tilde{O}(mn^{1/3})$ time: based on creating 'stars' and getting more Vizing chains for less edges:



A bottleneck for these algos: Coloring the last few edges is very slow!

- $\tilde{O}(n^2)$ term comes from $\tilde{O}(m/\Delta)$ Vizing chains
- When stars become small, our algo also uses Vizing chains.

Observation: Color extension in $\tilde{O}(\Delta)$ time $\Rightarrow$ $(\Delta+1)$-coloring in $\tilde{O}(m)$ time.

Proof. By Euler partitioning: $T(m,\Delta) = 2\cdot T(m/2, \Delta/2) + O(m/\Delta)\cdot\tilde{O}(\Delta) = 2\cdot T(m/2,\Delta/2) + \tilde{O}(m)$. $\quad T(m,\Delta) = \tilde{O}(m)$.

$\tilde{O}(mn^{1/4})$ time algo: Color extension in $\tilde{O}(\Delta^2+\sqrt{\Delta n})$ time + [Assadi, SODA'25] + [BCCSZ, FOCS'24]

Open Problem. Color extension in $\tilde{O}(\Delta)$ time.

Many barriers to beating $\tilde{O}(\Delta^2)$ time color extension → Single color extension framework problematic.

Core idea of [ABBCSZ'25]: Single color extension → Batch color extension.

Roadmap:

1. A new near-linear time algorithm for bipartite graphs based on Vizing chains. (We ignore $\tilde{O}(1)$ factors → $O(m\log^5 n)$.
   - Introduces main new ideas
   - Only need Δ colors
   - Assume $M = \Theta(\Delta n)$ (simplify expressions).
2. Extension to general graphs.

Near-Linear Time Edge Coloring Bipartite Graphs

Goal: Extend a coloring to a matching of uncolored edges in $\tilde{O}(m)$ time.

A Useful Insight:

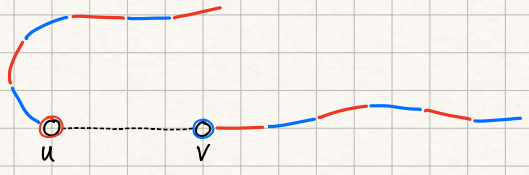
Suppose we have many vertex disjoint uncolored edges, missing α and β:

Their $\{\alpha,\beta\}$-APs have a combined length of $O(n)$ $\longrightarrow$ Can flip them all in $O(n)$ time.

Create 'many' uncolored edges with the same missing colors.

The Algorithm:

An edge (u,v) is of type $\{\alpha,\beta\}$ if $\alpha \in \text{miss}(u)$ and $\beta \in \text{miss}(v)$



The algorithm has 2 components:

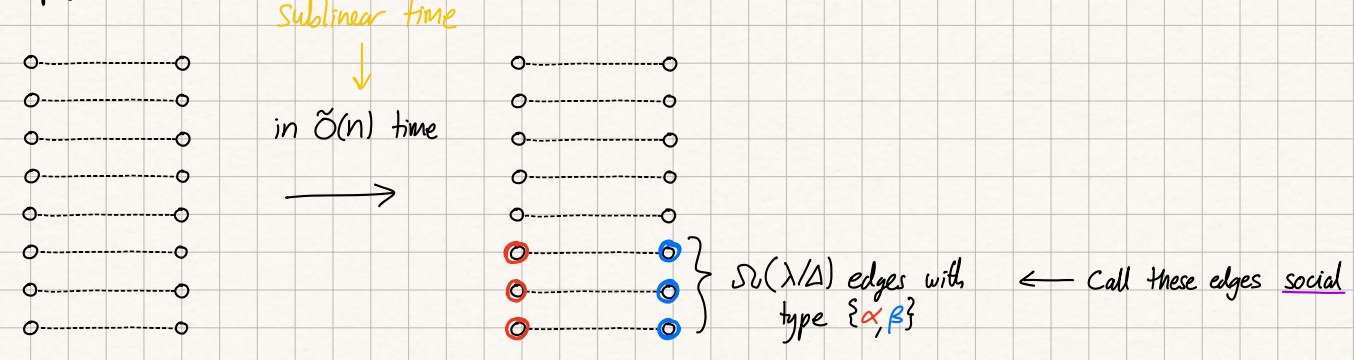
Component I: Type Amplification

- Picks a type $\{\alpha,\beta\}$, and makes many uncolored edges of this type.

Component II: Uniform Type Color Extension

- Takes many uncolored edges with type $\{\alpha,\beta\}$, and extends the coloring to them. ← Can be done in $O(n)$ time.

Type Amplification



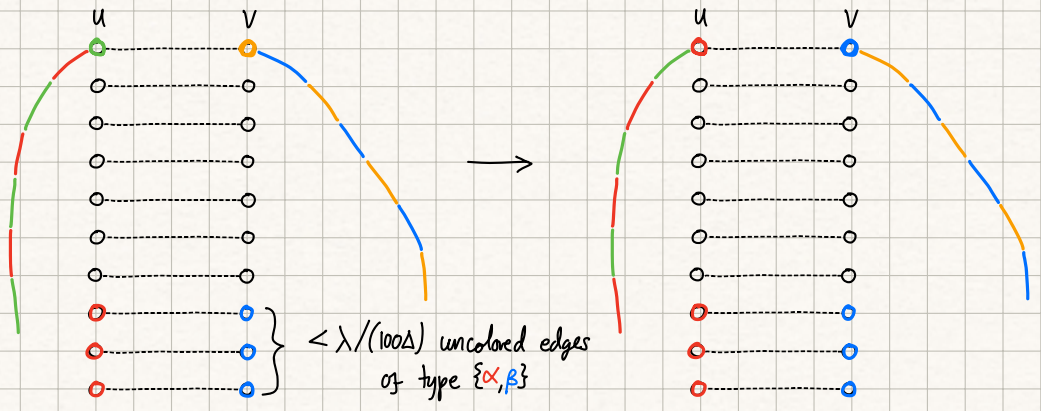
λ uncolored edges forming a matching

Subroutine does not change the set of uncolored edges $\longrightarrow$ Remain disjoint.

Type Amplification Algorithm:

While number of edges of type $\{\alpha,\beta\}$ is $< \lambda/(100\Delta)$:

1. Sample (u,v) u.a.r. of type $\{\gamma,\delta\}$.
2. P_u = $\{\alpha,\gamma\}$-AP at u
3. P_v = $\{\beta,\delta\}$-AP at v
4. Flip P_u and P_v

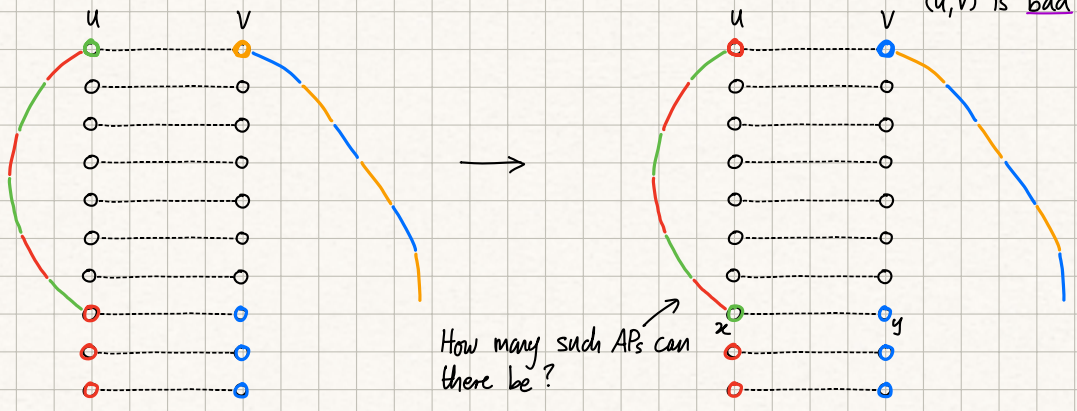


Observation. (u,v) now has type $\{\alpha,\beta\}$.

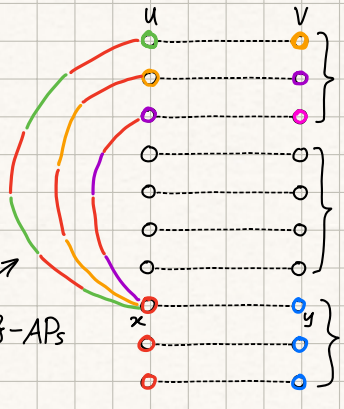
Key Lemma. w.p. ≥ 0.5, this does not 'damage' any other edge of type $\{\alpha,\beta\}$. → Number of edges with type $\{\alpha,\beta\}$ increases.

Proof.

(u,v) is bad → (x,y) is responsible.



Claim. Each social edge is responsible for at most 2Δ bad edges.



$\leq \frac{\lambda}{100\Delta} \cdot 2\Delta = \frac{\lambda}{50}$ bad edges.

At least $\lambda - \frac{\lambda}{100\Delta} - \frac{\lambda}{50} \geq \frac{\lambda}{2}$ edges are good (not social or bad)

Random sampling hits a good edge w.p. $\geq 1/2$.

$< \lambda/(100\Delta)$ uncolored edges of type $\{\alpha,\beta\}$

Running Time:

Running time of an iteration $= O(|P_u|+|P_v|)$

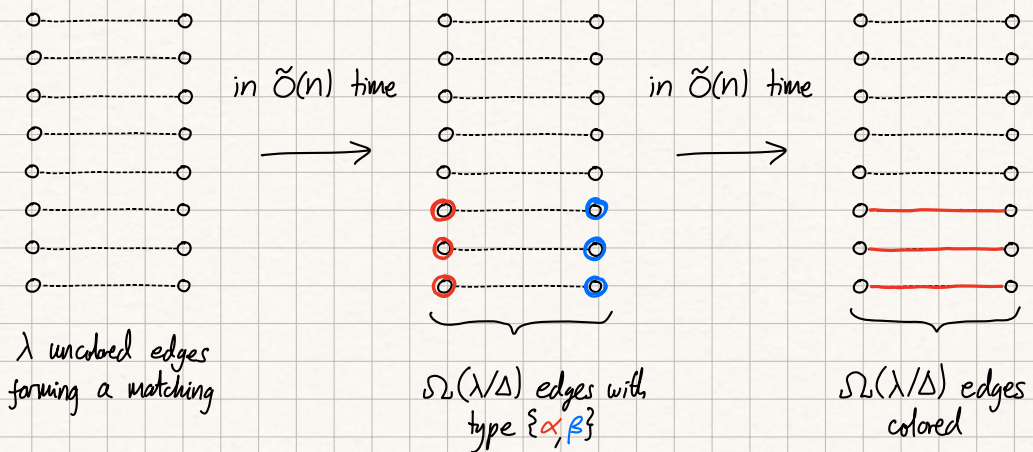
Total length of all $\{\alpha,*\}$ and $\{\beta,*\}$-APs is $O(\Delta n)$ ($O(\Delta)$ types, and all APs for a specific type have total length $O(n)$).

We sample a pair from a collection of λ pairs of distinct such paths. $\Rightarrow \mathbb{E}[|P_u|+|P_v|] = O(n\Delta/\lambda)$.

Repeat for $O(\lambda/\Delta)$ iterations $\Rightarrow$ Total running time $= O(\lambda/\Delta)\cdot O(n\Delta/\lambda) = O(n)$.

Lemma. Given λ uncolored edges forming a matching and a type $\{\alpha,\beta\}$: Make $\Omega(\lambda/\Delta)$ of them of type $\{\alpha,\beta\}$ in $O(n)$ time.

Putting Everything Together



In $O(n)$ time, we can extend χ to $\Omega(\lambda/\Delta)$ uncolored edges.

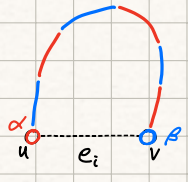
Repeating $\tilde{O}(\Delta)$ times colors all λ edges in $\tilde{O}(\Delta)\cdot O(n) = \tilde{O}(n\Delta) = \tilde{O}(m)$ time.

Remark:

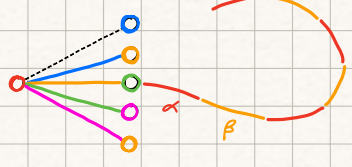
- The alternating paths are not Vizing chains – can shift uncolored edges with them.
- By choosing a type $\{\alpha,\beta\}$ of uncommon colors (that appear $O(m/\Delta)$ times): $O(n) \rightsquigarrow O(m/\Delta)$ bounds.

Extension to General Graphs

Recall the problem:



Thus, we need Vizing fans:

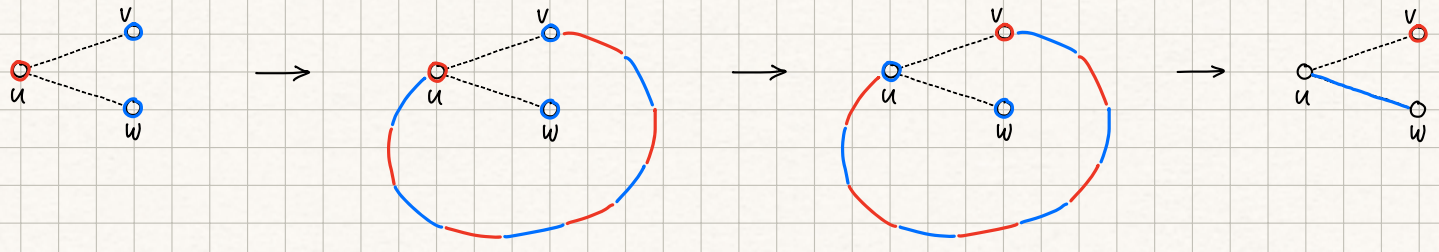


The Problem:
We have no control over the second color of the path!!!

It's very hard to control the colors of a Vizing chain → Cannot integrate them into previous framework → Need a new approach.

U-Fans [Gabow et al., '85]

A u-fan: 2 uncolored edges sharing an endpoint, with a common missing color at the other endpoints:



Crucially: Now we control both colors of the alternating path. Takeaway: u-fans in general graph ≃ uncolored edges in bipartite graphs.

Goal: Extend a coloring to $O(n)$ uncolored edges in $\tilde{O}(m)$ time. (By Euler partitioning)

Suppose we have λ uncolored edges.

Phase I: Color $\Omega(\lambda)$ edges via Vizing fans or create $\Omega(\lambda)$ u-fans

Phase II: Given $\Omega(\lambda)$ u-fans, color $\Omega(\lambda)$ edges

Component I: Type Amplification

- Picks a type $\{\alpha, \beta\}$, and makes many ~~uncolored edges~~ u-fans of this type.

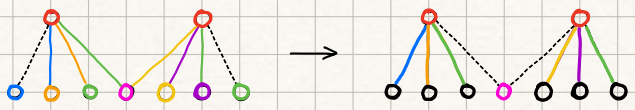
Component II: Uniform Type Color Extension

- Takes many ~~uncolored edges~~ u-fans with type $\{\alpha, \beta\}$, and extends the coloring to an $\Omega(1)$ fraction of them.

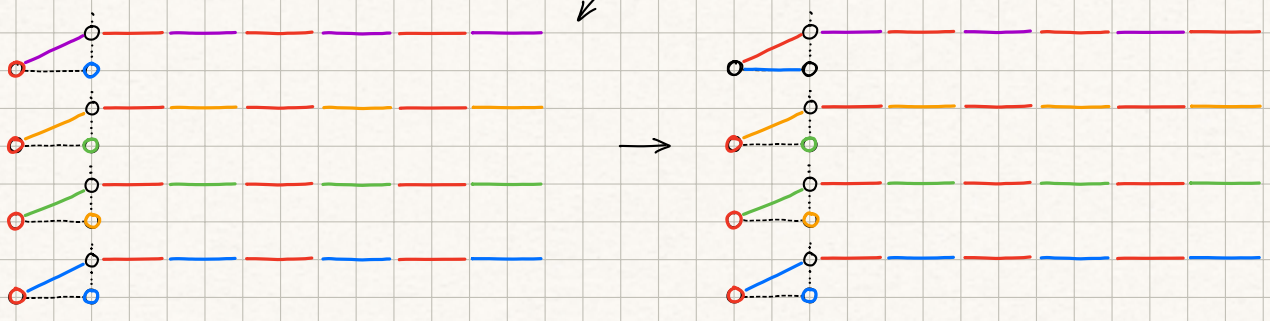
Creating U-Fans

Scan over each color α, and compute Vizing fans at the o·····o edges.

If they intersect: We can create a u-fan. → Assume they are disjoint

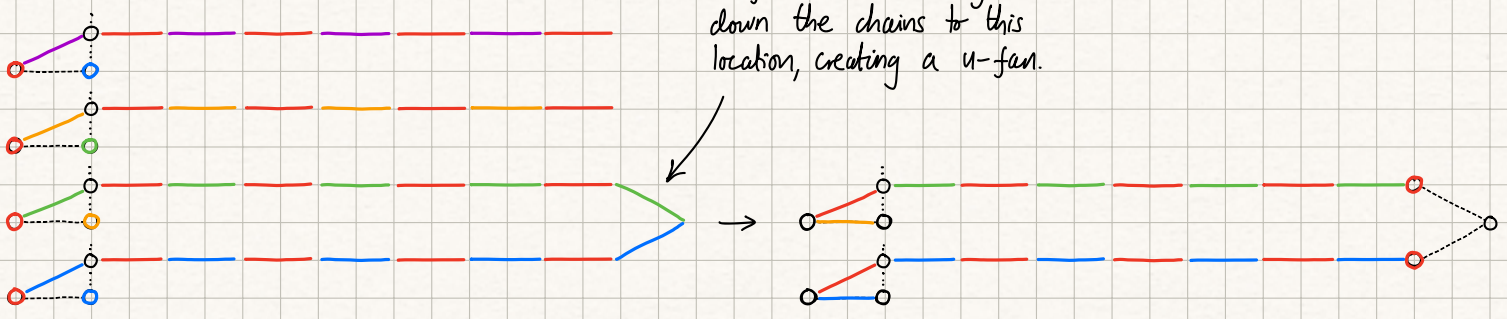


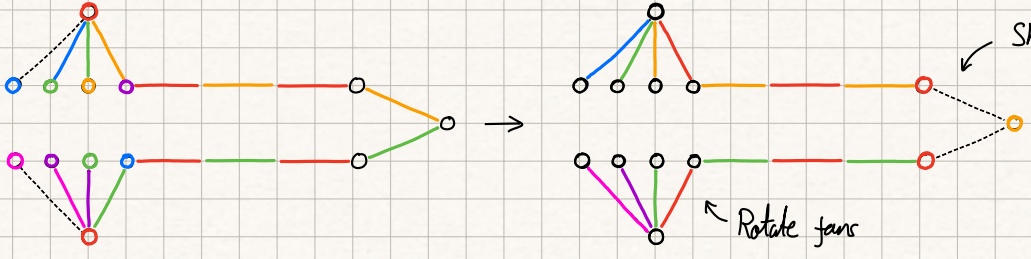
Traverse Vizing chains in parallel, 1 edge at a time. If Vizing chain ends, activate it → Colors the edge



Interesting case: Vizing chains intersect at a vertex.

Shift the uncolored edges down the chains to this location, creating a u-fan.

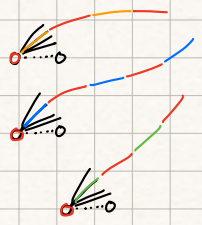




Running Time

Suppose we have μ o·····o edges. At any point in time: total length of all APs is $O(m_\alpha)$ ← Vertex disjoint

Since all APs have same length ± 1: When ℓ remain, they have length $O(m_\alpha/\ell)$.



Total running time $\sum_{\ell=\mu}^{1} O(m_\alpha/\ell) = \tilde{O}(m_\alpha)$.

Repeating for all colors α: $\sum_\alpha \tilde{O}(m_\alpha) = \tilde{O}(m)$.

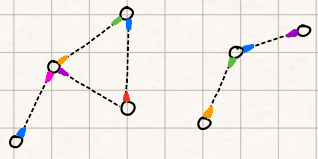
Lemma. λ uncolored edges $\Rightarrow$ Color $\Omega(\lambda)$ edges or create $\Omega(\lambda)$ u-fans in $O(m)$ time.

Remarks:

- Some details are missing: m_α dynamic, time to handle Vizing fans } minor details
- Activating a chain can 'damage' a u-fan
- u-fans are not vertex disjoint } Can be handled.

Separable Collection

Why disjointness important? Different uncolored edges give different alternating paths.



u touches k uncolored edges $\Rightarrow$ u has k missing colors. u can assign each one a unique missing color. Arguments easily extend.

Part III: Deterministic $(\Delta+1)$-Coloring in Almost-Linear Time

[ABBCSZ '25] needs random sampling to make edges social → Hit a 'good' edge w.p. $\geq \frac{1}{2}$.

This is the only part that uses randomness → Can we remove it?

Why is this interesting? All recent fast static algorithms rely on randomized sublinear-time algorithms!!

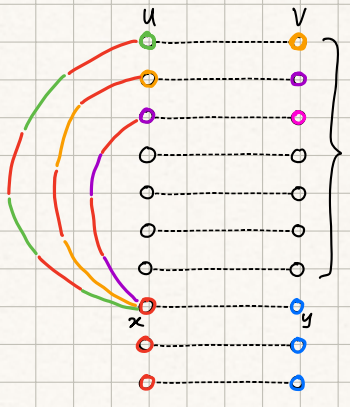
- [BCCSZ, FOCS '24]: Randomly extends coloring to edge in a star
- [Assadi, SODA'25]: $\tilde{O}(n)$ time random walk, applied Δ times to extract matchings.
- [BCSZ, SODA'25]: Randomly constructing 'multi-step Vizing chains'

Can we design an algorithm that does not use sublinear algorithms?

Theorem. Deterministic $(\Delta+1)$-Coloring in $m \cdot 2^{O(\sqrt{\log \Delta})} \cdot \log n = m^{1+o(1)}$ time.

Just like before: Suffices to consider a matching of uncolored edges in a bipartite graph (Deterministic construction of separable u-fans).

Barrier to Derandomization



Total length of APs $= O(n\Delta)$

No idea which edges are good

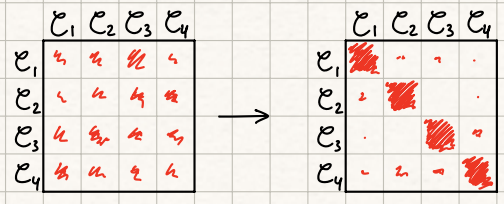
Time to make one more social edge $= O(n\Delta)$

Type Sparsification

Notation: $\chi : E \to C \cup \{\perp\}$. $A \times B = \{\{a,b\} \mid a \in A, b \in B\}$. e has type $A \times B$ if $\tau(e) \in A \times B$.

Theorem. Let $G=(V,E)$ be a graph and $\chi : E \to C \cup \{\perp\}$ a partial coloring, and U a matching of λ uncolored edges.

Given a parameter η, partition C into $\mathcal{C}_1, \dots, \mathcal{C}_\eta$ s.t. $|\mathcal{C}_i| = |C|/\eta$. In $O(m \cdot \text{poly}(\eta))$ deterministic time, modify χ so that $\Omega(\lambda)$ of the edges in U have a type in $\bigcup_{k=1}^{\eta} \mathcal{C}_k \times \mathcal{C}_k$.



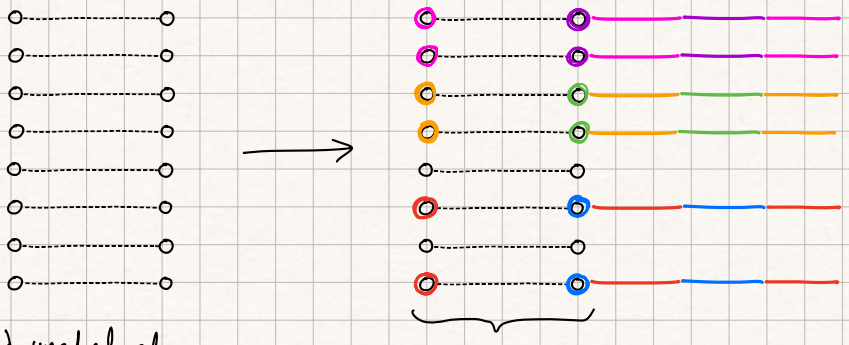
Type Sparsification → Fast Edge Coloring

Suppose $\Delta = 2 \cdot \eta^q$, $q \in \mathbb{N}$ (clean recursion). Apply type sparsification recursively:

- $C \to \mathcal{C}_1, \dots, \mathcal{C}_\eta$, $E_i = \chi^{-1}(\mathcal{C}_i) \cup \{e \in U \mid \tau(e) \in \mathcal{C}_i \times \mathcal{C}_i\}$, $G_i = (V, E_i)$, $\chi_i : E_i \to \mathcal{C}_i \cup \{\perp\}$.
- Recurse on $(G_1, \chi_1, \mathcal{C}_1), \dots, (G_\eta, \chi_\eta, \mathcal{C}_\eta)$.

Repeat until we have $\mathcal{C}_1, \dots, \mathcal{C}_{\Delta/2}$, s.t. $|\mathcal{C}_i| = 2$. Suppose $\eta = \Delta^{O(\varepsilon)}$, then recursion depth is $O(1/\varepsilon)$, thus $\lambda \cdot O(1)^{-O(1/\varepsilon)} = \lambda \cdot 2^{-O(1/\varepsilon)}$ edges from U survive to bottom of recursion.

Extend the coloring to $\lambda \cdot 2^{-O(1/\varepsilon)}$ edges in $O(m)$ time:



λ uncolored edges forming a matching

$O(\Delta)$ disjoint types

← All of these paths can be flipped simultaneously in $O(m)$ time.

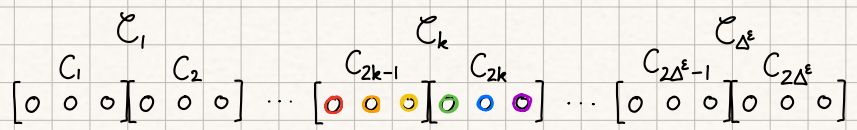
Total time to color $\lambda \cdot 2^{-O(1/\varepsilon)}$ edges: $m \cdot \Delta^{O(\varepsilon)} \cdot \frac{1}{\varepsilon}$

Total time to color all λ edges in U: $m \cdot \Delta^{O(\varepsilon)} \cdot \frac{1}{\varepsilon} \cdot 2^{O(1/\varepsilon)} \cdot \log n$.

Setting $\varepsilon = \Theta(1/\sqrt{\log \Delta})$: $m \cdot \Delta^{O\left(\frac{1}{\sqrt{\log \Delta}}\right)} \cdot \sqrt{\log \Delta} \cdot 2^{O(\sqrt{\log \Delta})} \cdot \log n = m \cdot 2^{O(\sqrt{\log \Delta})} \cdot \log n$.

Type Sparsification Algorithm

Let $\eta = \Delta^\varepsilon$. Partition C into $\mathcal{C}_1, \dots, \mathcal{C}_{\Delta^\varepsilon}$. For $k \in [\Delta^\varepsilon]$, split $\mathcal{C}_k$ into C_{2k-1} & C_{2k}.



Definition. $e \in U$ is *uniform* if $\tau(e) \in C_i \times C_i$ or *aligned* if $\tau(e) \in C_{2k-1} \times C_{2k}$.

Goal: Make $\Omega(\lambda)$ of the edges in U aligned or uniform.

Assumption: If $\Omega(\lambda)$ of the edges in U are uniform, we are done → Assume no edges are uniform.

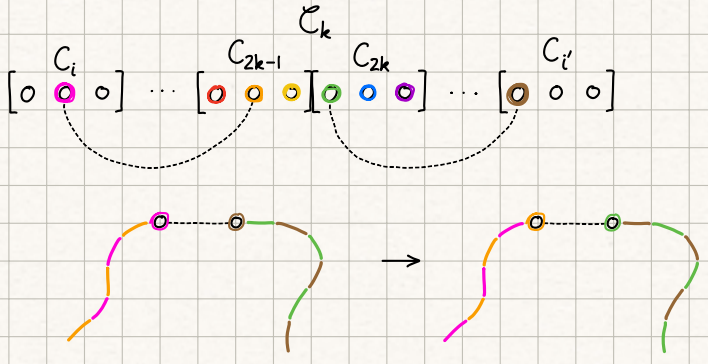
Changing the Type of an Edge

For each $i \in [2\Delta^\varepsilon]$, let $C_i = \{c_i(1), \dots, c_i(q)\}$, i.e. $c_i(j)$ is the j^{th} color in C_i.

Let e have type $C_i \times C_{i'}$. Suppose we want $\tau(e) \in C_{2k-1} \times C_{2k}$.

Let $\tau(e) = \{\overline{c_i(j)}, \overline{c_{i'}(j')}\}$.

Flip the $\{\overline{c_i(j)}, \overline{c_{2k-1}(j)}\}$ and $\{\overline{c_{i'}(j')}, \overline{c_{2k}(j')}\}$-APs at e.



Definition. An AP is *relevant* if it has type a $\{c_i(j), c_{i'}(j)\}$.

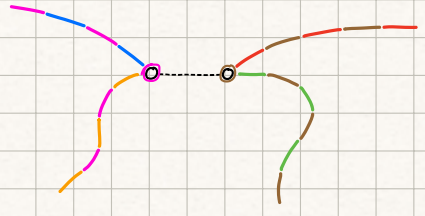
An AP is *k-relevant* if it has a color in $\mathcal{C}_k$.

The Randomized Algorithm

We have η rounds: we create edges of type $C_{2k-1} \times C_{2k}$ during round k.

$U \leftarrow$ set of uncolored edges
for $k = 1, \dots, \eta$:
  $U_k \leftarrow \emptyset$
  While $|U_k| < \lambda/(100\eta)$:
    Sample $e \sim U$ u.a.r.
    if e is *k-good*:
      Flip the *k-relevant* APs at e
      $U_k \leftarrow U_k + e$

[Say e is *k-good* if flipping the k-relevant APs at e does not *damage* any edge in $U_1 \cup \dots \cup U_k$ (and not yet processed).]



Lemma. $\Pr[e \text{ is good}] \geq 1/2$.

Proof. Each edge in $U_1 \cup \dots \cup U_{k-1}$ is responsible for ≤ 4 bad edges. (each endpoint touches 2 k-relevant APs)
Each edge in U_k is responsible for $\leq 4\eta$ bad edges. (each endpoint touches 2η k-relevant APs)

$$\Rightarrow \#\text{ bad edges} \leq 4 \cdot (|U_1| + \dots + |U_{k-1}|) + 4\eta \cdot |U_k| + \underbrace{(|U_1| + \dots + |U_k|)}_{\text{processed edges}} \leq 9\eta \cdot \lambda/(100\eta) < \lambda/2.$$

Running Time:

k-relevant APs have $O(\Delta/\eta) \cdot O(\eta) = O(\Delta)$ types.

Total length of all k-relevant APs $= O(n) \cdot O(\Delta) = O(n\Delta) = O(m)$ (assuming regularity)

During a round, sample one of λ edges u.a.r. and flip its k-relevant APs $\rightarrow O(m/\lambda)$ expected time.

Time of each round $= O(m/\lambda) \cdot O(\lambda/\eta) = O(m/\eta)$

Time of algo $= O(\eta) \cdot O(m/\eta) = O(m)$.

Derandomizing the k^{th} Round

Goal: Convert $\Omega(\Delta/\eta)$ edges to type $C_{2k-1} \times C_{2k}$, without damaging previously processed edges.

Lemma. There exists a popular meta-type $C_i \times C_{i'}$ s.t. $\exists\, \mathcal{U}_{i,i'} \subseteq \mathcal{U} \setminus (\mathcal{U}_1 \cup \cdots \cup \mathcal{U}_k)$ with $|\mathcal{U}_{i,i'}| \geq \Omega(\lambda/\eta^2)$ and all k-good with type $C_i \times C_{i'}$.

Furthermore, we can find such a meta-type in $O(m)$ time.

Proof. There are $\Omega(\lambda)$ k-good edges in $\mathcal{U}$, and each has one of $O(\eta^2)$ meta-types. By averaging, one is popular.

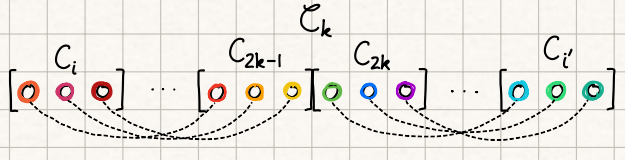
To find it, traverse all k-relevant APs starting from all processed edges, and identify all k-bad edges. Then scan over all of $\mathcal{U}$ and identify most common meta-type of k-good edges.

Total length of k-relevant APs $= O(m)$.

Popular Meta-Type

Given a popular meta-type $C_i \times C_{i'}$ and $\mathcal{U}_{i,i'}$, make all edges in $\mathcal{U}_{i,i'}$ of type $C_{2k-1} \times C_{2k}$ in $O(m)$ time:

Pair up $C_i \leftrightarrow C_{2k-1}$ and $C_{i'} \leftrightarrow C_{2k}$, flip disjoint APs, total length $= O(m)$.



Repeating $O(\eta)$ time, we create $\Theta(\lambda/\eta)$ edges of type $C_{2k-1} \times C_{2k}$.

Total time of round $= O(m\eta)$.

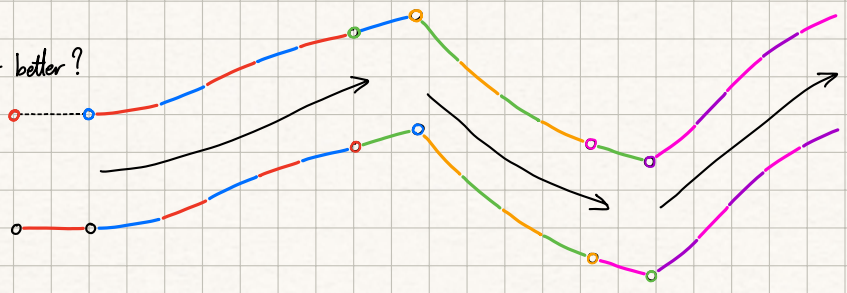
Running time of deterministic algorithm $= O(\eta) \cdot O(m\eta) = O(m\eta^2)$.

↑ Why we can't have uniform edges!

Part IV: Multi-Step Vizing Chains

Q: How fast can we extend a coloring to an edge?

With a Vizing chain: can be done in $O(n)$ time. Can we do better?



Multi-Step Vizing chain: Multiple Vizing chains glued together.

Many applications in static, dynamic, and distributed settings:

Theorem [Duan, He, Zhang, SODA'19]. Dynamic $(1+\varepsilon)\Delta$-coloring in $\tilde{O}(1/\varepsilon^2)$ update time, $\Delta = \Omega(\log^2 n/\varepsilon^2)$.

Theorem [Bernshteyn, J. Comb. Theory '22]. Deterministic LOCAL $(\Delta+1)$-coloring in $\tilde{O}(\text{poly}(\Delta))$ rounds.

Theorem [Christiansen, STOC'23]. Dynamic $(1+\varepsilon)\Delta$-coloring in $\tilde{O}(\text{poly}(1/\varepsilon))$ update time.

Theorem [Bernshteyn, Dhawan '24]. Static $(1+\varepsilon)\Delta$-coloring in $O(\log n \cdot \text{poly}(1/\varepsilon))$ rounds.

Line of work designing short multi-step Vizing chains:

- $\tilde{O}(1/\varepsilon^2)$ length chains, for $(1+\varepsilon)\Delta$-coloring $(\varepsilon = \tilde{\Omega}(1/\sqrt{\Delta}))$
- $\tilde{O}(\Delta^6)$ length chains, for $(\Delta+1)$-coloring

A lower bound:

Theorem [Chang, He, Li, Pettie, Uitto, SODA'18]. $\exists$ a graph G and $(1+\varepsilon)\Delta$-coloring χ s.t. extending χ requires changing the colors of $\Omega(\log(\varepsilon n)/\varepsilon)$ edges.

Our Focus: $\tilde{O}(\Delta^2 + \sqrt{\Delta n})$-length MSV for $(\Delta+1)$-coloring

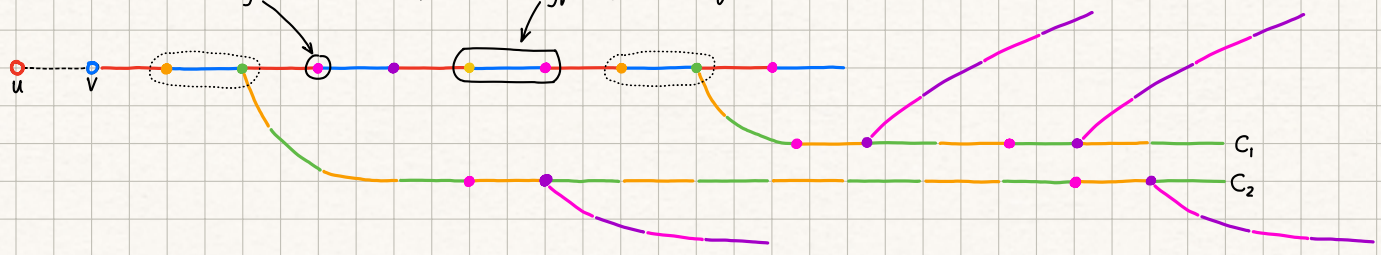
- An MSV construction that highlights key ideas of the constructions.
- Assume bipartiteness for simplicity. Extends to general graphs by adding Vizing fans + many technical details.
- Start with existential guarantees

A $\tilde{O}(\Delta^2)$-length MSV for $(\Delta + O(\log n))$-coloring

Consider an edge $e=(u,v)$ with Vizing chain C.

If $|C| = O(\Delta^2)$, we are done. → Assume $|C| \geq 8\Delta^2$ and consider the first $8\Delta^2$ edges.

Consider the colors missing at each vertex, and the type of each edge.



Claim. Since (1) $|C| \geq 8\Delta^2$ and (2) there are $\leq (\Delta + O(\log n))^2 \leq (2\Delta)^2 = 4\Delta^2$ types, some type appears twice.

→ Gives 2 different choices for where to start next chain. These paths are disjoint.

Repeat the Process:

If either $|C_1|$ or $|C_2|$ is $O(\Delta^2)$, we are done. → Assume $|C_1|, |C_2| \geq 8\Delta^2$.

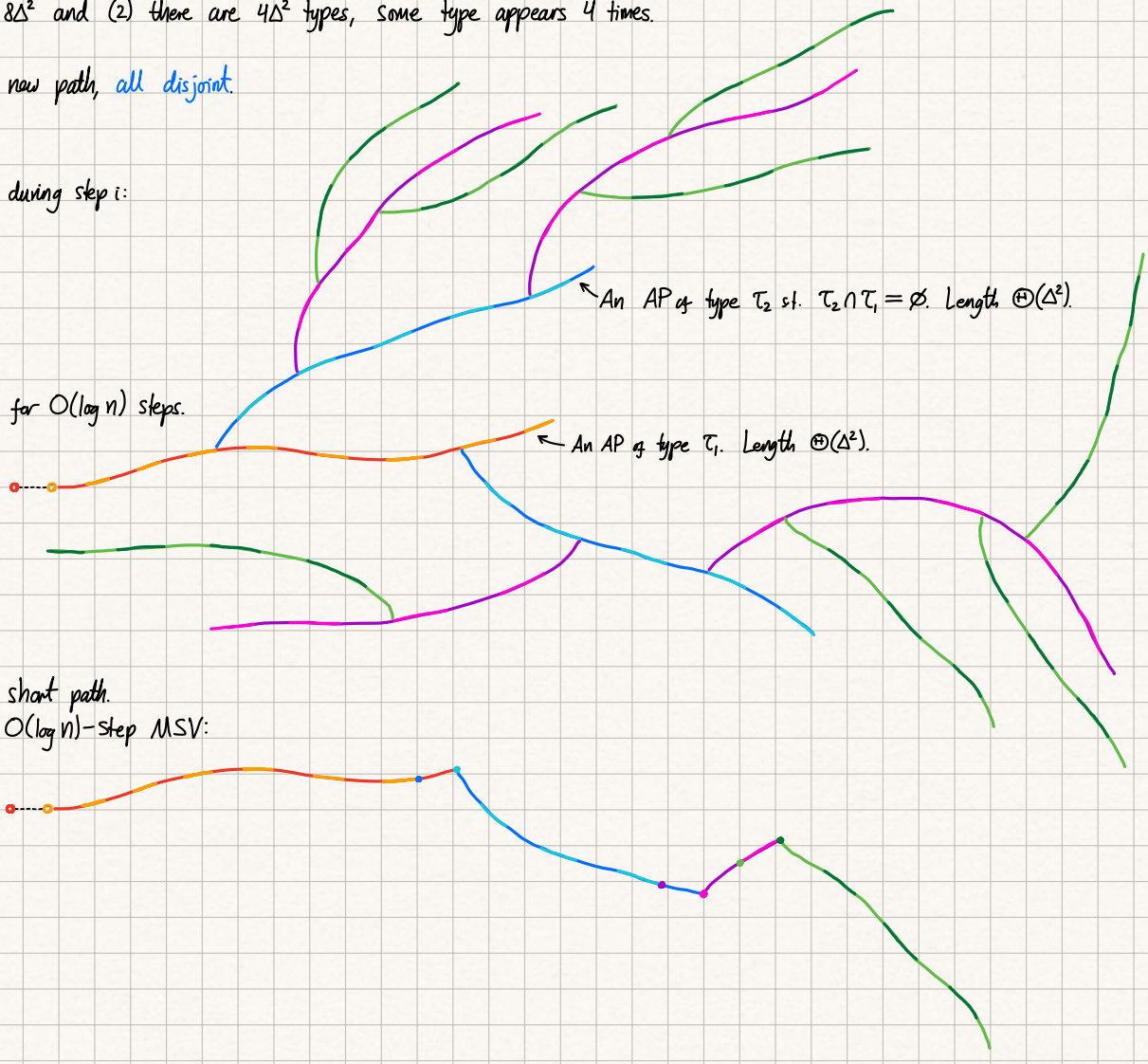
Consider missing colors on these paths. Since there are $O(\log n)$ extra colors, we only consider 'new' colors.

Claim. Since (1) $|C_1 \cup C_2| \geq 2 \cdot 8\Delta^2$ and (2) there are $4\Delta^2$ types, some type appears 4 times.

→ Leads to 4 choices for new path, all disjoint.

Consider the paths constructed during step i:
1. They are vertex disjoint.
2. There are 2^{i-1} of them
3. They have length $\Theta(\Delta^2)$.

→ Total length is $\Omega(2^i)$.
→ The process can only go on for $O(\log n)$ steps.



After $O(\log n)$ steps, find a short path.
→ Use this to construct a $O(\log n)$-step MSV:

→ $\exists$ a $\tilde{O}(\Delta^2)$-length MSV.

The Algorithm

Let C_1 be a Vizing chain at (u,v), $i \leftarrow 1$

While $|C_i| \geq \Omega(\Delta^2)$:
  Pick a random edge e_i in C_i and shift the uncolored edge to e_i
  $C_{i+1} \leftarrow$ A Vizing chain at e_i with new colors
  $i \leftarrow i+1$

Activate C_i

Lemma. This algorithm succeeds w.h.p.

Algorithmic guarantees almost match the existential ones.

Main Idea: (1) the random edge we pick corresponds to a type that appears multiple times w.p. $\Omega(1)$, (2) we pick between the different paths uniformly.

$\Delta + O(\log n) \rightarrow \Delta + 1$ Colors

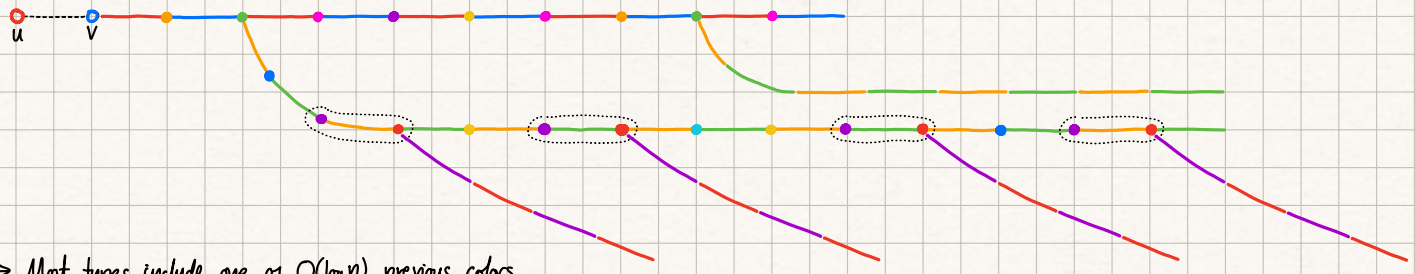
What if we don't have $O(\log n)$ extra colors?

If the paths of the chain overlap, the construction breaks $\rightarrow$ Flipping one path changes another.

Perform the same construction with paths of length $L = \Theta(\Delta^2 + \sqrt{\Delta n})$.

If most of the types in each path use new colors $\rightarrow$ still works.

Otherwise, we say that a path is contaminated.



$\rightarrow$ Most types include one of $O(\log n)$ previous colors.

$\rightarrow$ Some type appears $\Omega(L/(\Delta \log n))$ times.

Average length of $\{\bullet, \bullet^3$ paths $\leq n \cdot \tilde{O}(\Delta/L) = \tilde{O}(\sqrt{\Delta n}) \rightarrow$ one has length $\tilde{O}(\sqrt{\Delta n})$.

Open Question Can we extend a $(\Delta+1)$-coloring in $\tilde{O}(\Delta^{1.99})$ time?

Easier question: Extend a $(1+\varepsilon)\Delta$-coloring in $\tilde{O}(1/\varepsilon^{1.99})$ time, for large enough $\varepsilon > 0$?