Online Edge Coloring: Sharp Thresholds

Joakim Blikstad

CWI

Ola Svensson

EPFL

Radu Vintan

EPFL

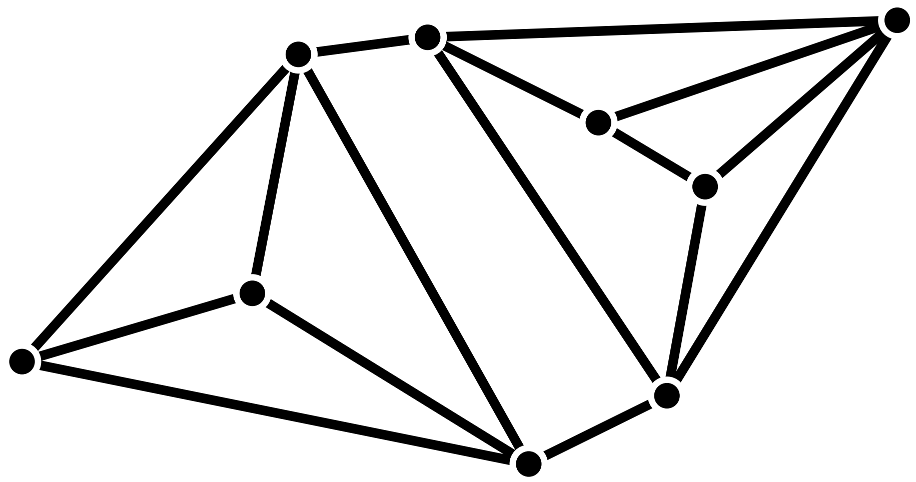
David Wajc

TECHNION Israel Institute of Technology

FOCS 2025

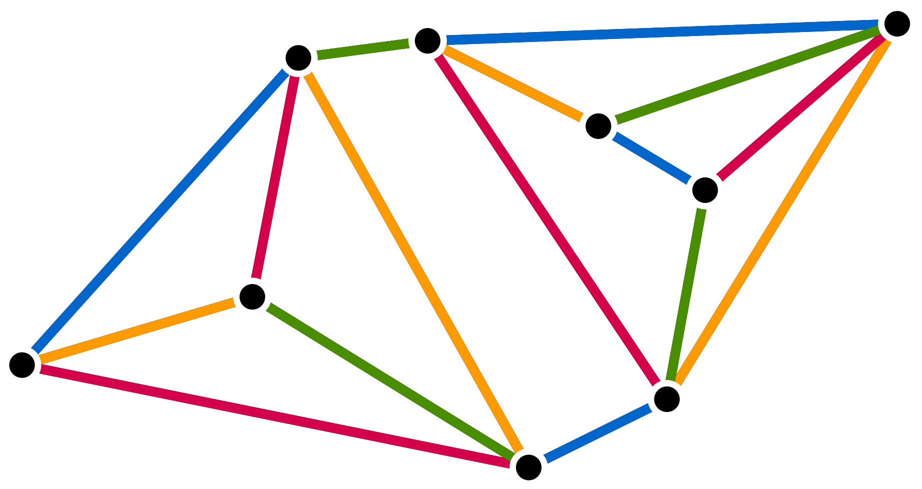
Edge Coloring

Given: Graph $G = (V, E)$
Goal: Color *edges* with few colors
Constraint: No two incident edges get the same color



Edge Coloring

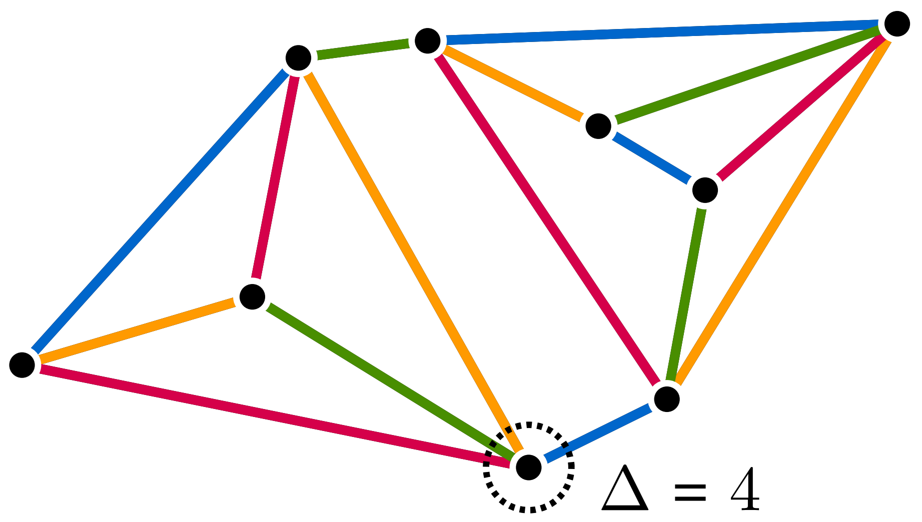
Given: Graph $G = (V, E)$
Goal: Color *edges* with few colors
Constraint: No two incident edges get the same color



4 colors?
Optimal?

Edge Coloring

Given: Graph $G = (V, E)$
Goal: Color *edges* with few colors
Constraint: No two incident edges get the same color



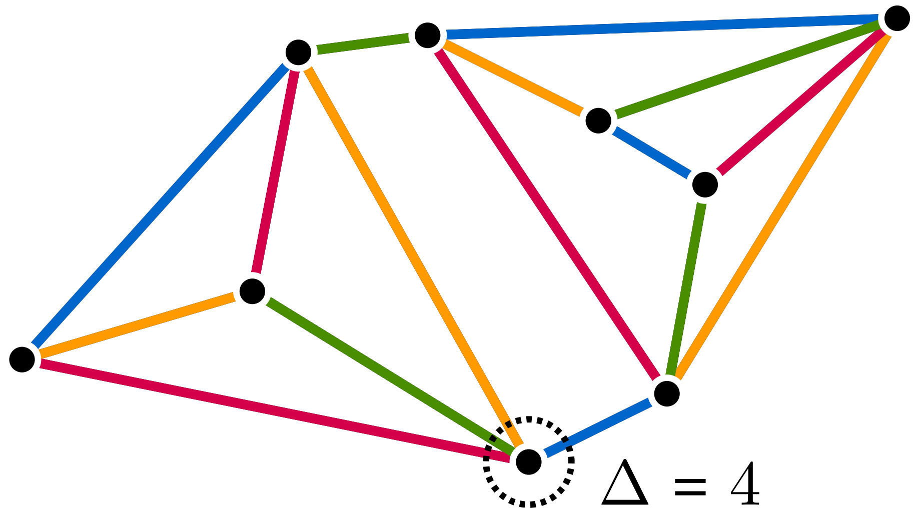
4 colors?
Optimal?

$\Delta := \max_{v \in V} \deg(v)$

Claim: $\#\text{Colors} \geq \Delta$

Edge Coloring

Given: Graph $G = (V, E)$
Goal: Color *edges* with few colors
Constraint: No two incident edges get the same color



$\Delta = 4$

4 colors?
Optimal?

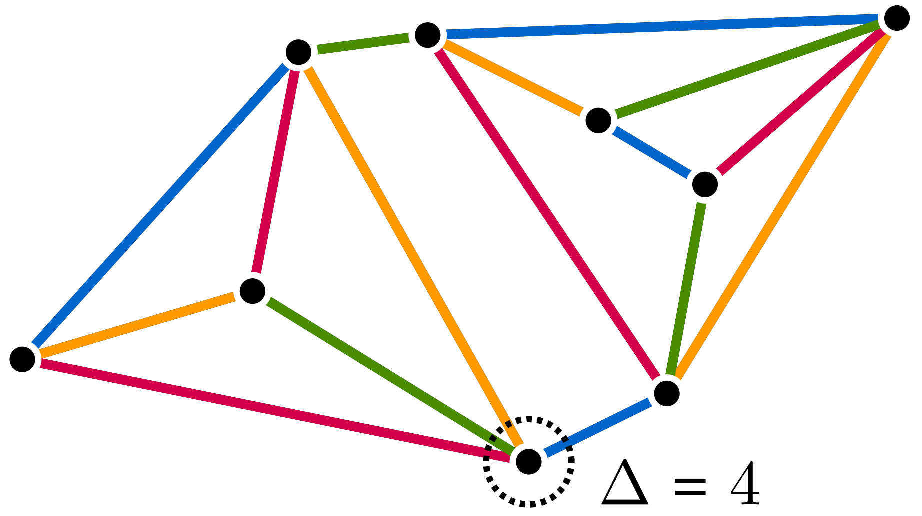
$\Delta := \max_{v \in V} \deg(v)$

Claim: $\#\text{Colors} \geq \Delta$

Theorem: $\#\text{Colors} \leq \Delta + 1$ [Vizing 1964]

Edge Coloring

Given: Graph $G = (V, E)$
Goal: Color *edges* with few colors
Constraint: No two incident edges get the same color



4 colors?
Optimal?

$\Delta := \max_{v \in V} \deg(v)$

Claim: $\#\text{Colors} \geq \Delta$

Theorem: $\#\text{Colors} \leq \Delta + 1$ [Vizing 1964]

Answer = Δ or $(\Delta + 1)$
NP-complete deciding which
$O(|E| \log |E|)$ time compute $\Delta + 1$ [ABBCSZ'25]

Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.

Online Edge Coloring

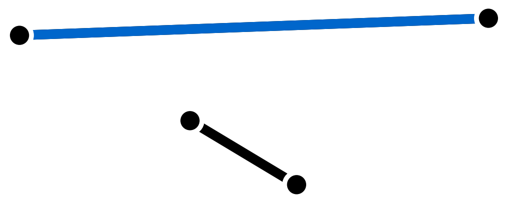
Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.

Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.

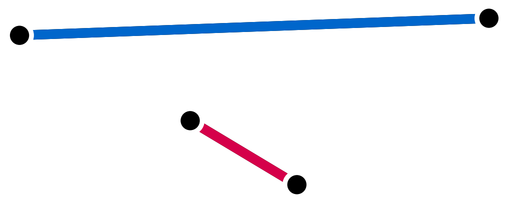
Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.



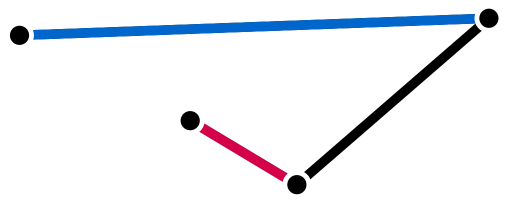
Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.



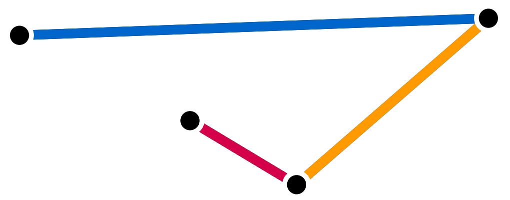
Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.



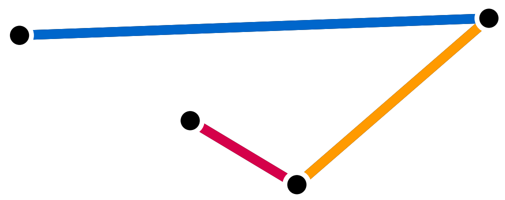
Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.



Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.

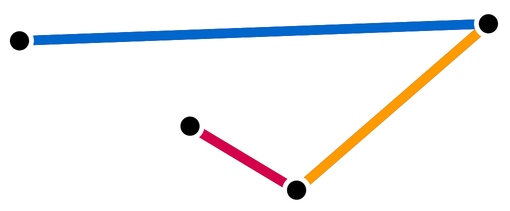


Two-Player Game:

- Adversary (reveals edges)
- Online Algorithm (colors edges)

Online Edge Coloring

Online: Graph revealed over time: edge-by-edge. Max-degree Δ known.
Task: Color edge *irrevocably* when it is revealed.



Two-Player Game:

- Adversary (reveals edges)
- Online Algorithm (colors edges)

How many colors do we need? Still $\approx \Delta$?

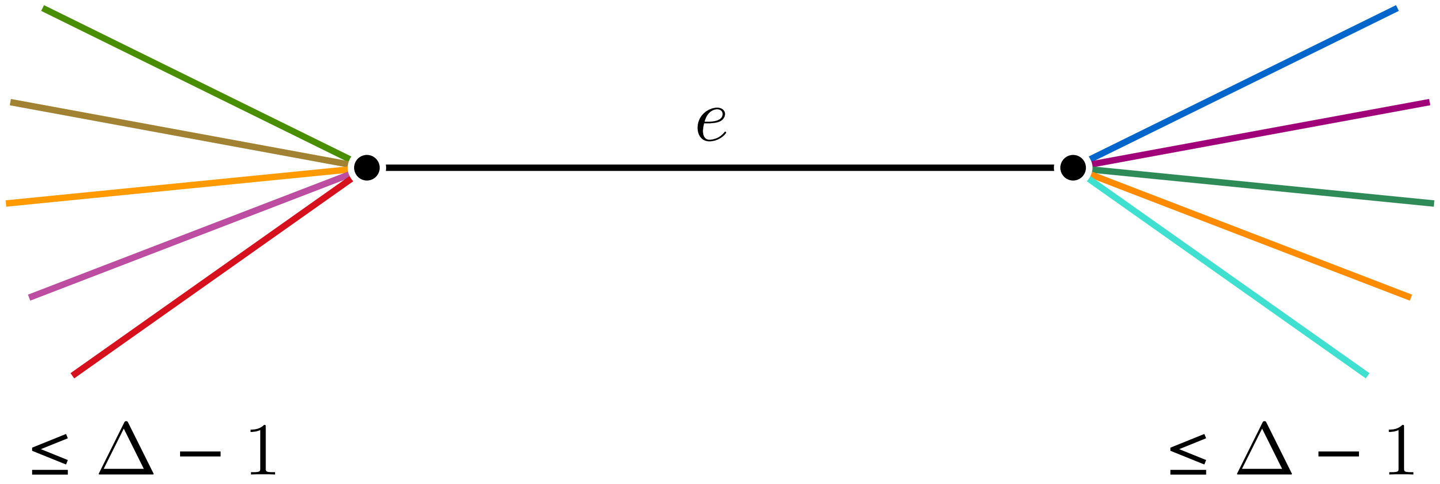
Warm-up: Greedy Algorithm

Greedy: Color edge with "lowest" avaliable color.

Colors $= \{1, 2, 3, \ldots\}$

Warm-up: Greedy Algorithm

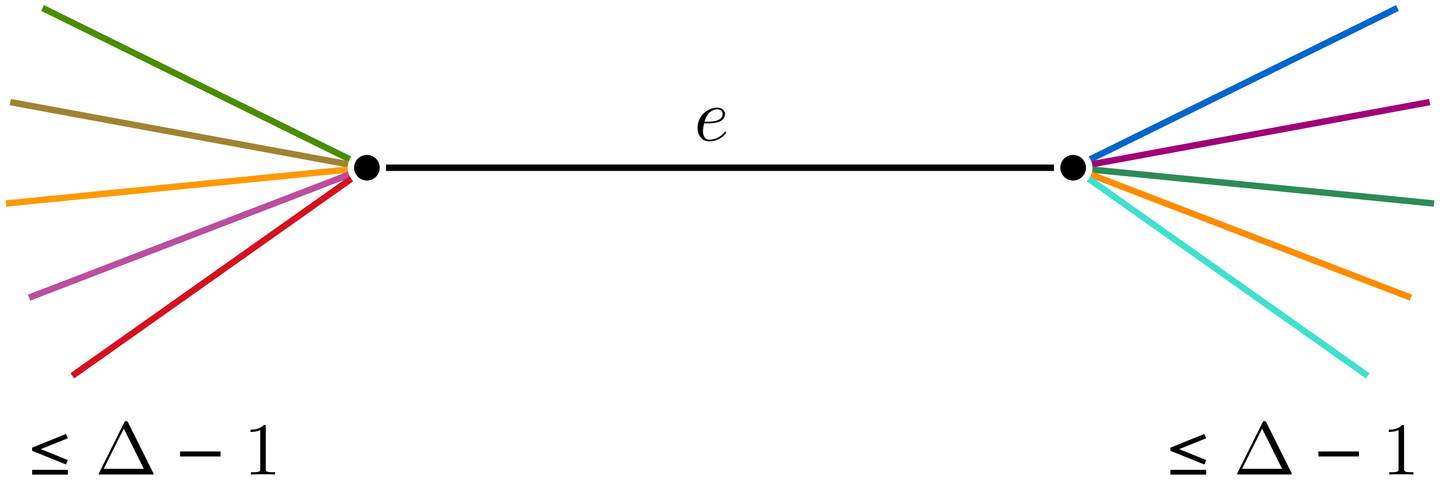
Greedy: Color edge with “lowest” avaliable color.
Colors $= \{1, 2, 3, \ldots\}$



Warm-up: Greedy Algorithm

Greedy: Color edge with “lowest” avaliable color.

Colors $= \{1, 2, 3, \ldots\}$

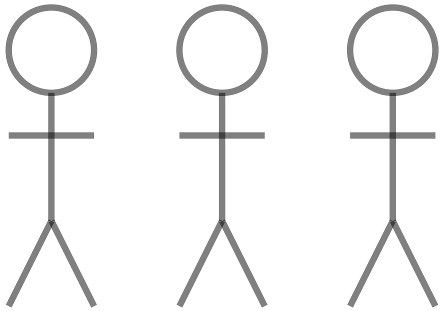


Claim: $\leq 2(\Delta - 1)$ blocked colors

Claim: Greedy uses $\leq 2\Delta - 1$ colors

Brief (Fictional?) History

1992: **Can we beat greedy?**

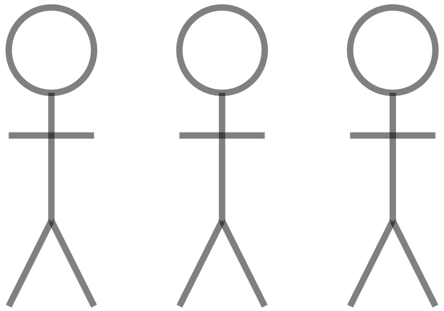


[Bar-Noy/Motwani/Naor]

Brief (Fictional?) History

1992:

Can we beat greedy?



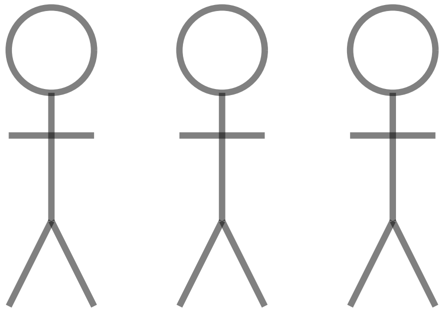
[Bar-Noy/Motwani/Naor]

"The Greedy Algorithm is Optimal for Online Edge Coloring"

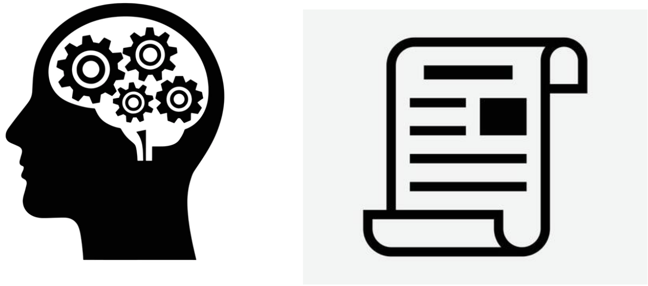
Brief (Fictional?) History

1992:

Can we beat greedy?



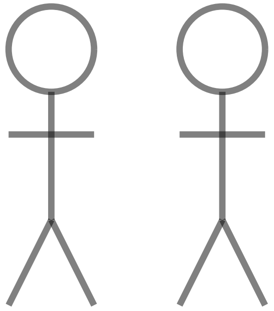
[Bar-Noy/Motwani/Naor]



"The Greedy Algorithm is Optimal for Online Edge Coloring"

2021:

Is it though. . . ?

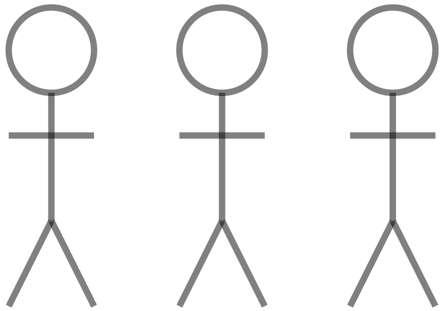


[Saberi/Wajc]

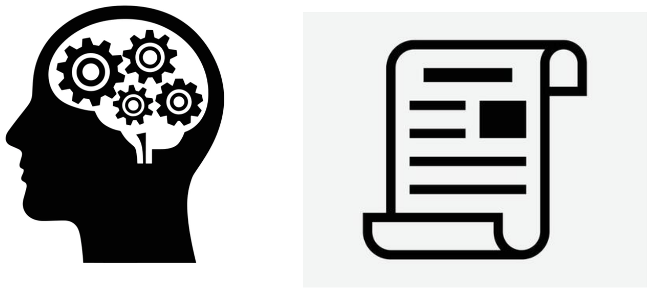
Brief (Fictional?) History

1992:

Can we beat greedy?



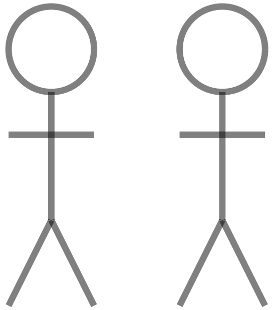
[Bar-Noy/Motwani/Naor]



"The Greedy Algorithm is Optimal for Online Edge Coloring"

2021:

Is it though. . . ?



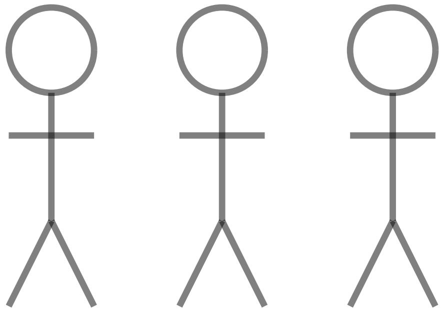
[Saberi/Wajc]

"The Greedy Algorithm is **Not** Optimal for Online Edge Coloring"

Brief (Fictional?) History

1992:

Can we beat greedy?



[Bar-Noy/Motwani/Naor]

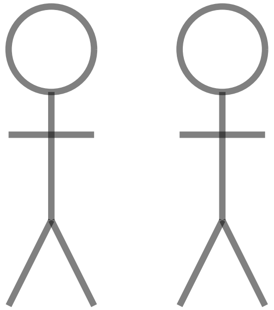
"The Greedy Algorithm is Optimal for Online Edge Coloring"

* when $\Delta \leq \log n$ (deterministic)

when $\Delta \leq \sqrt{\log n}$ (randomized)

2021:

Is it though. . . ?



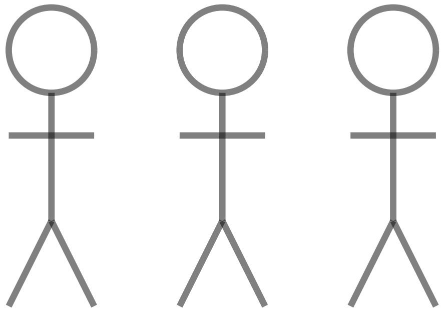
[Saberi/Wajc]

"The Greedy Algorithm is **Not** Optimal for Online Edge Coloring"

Brief (Fictional?) History

1992:

Can we beat greedy?



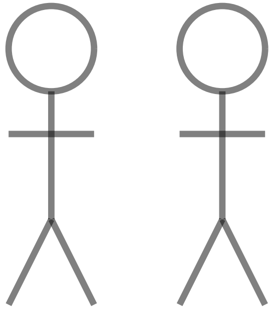
[Bar-Noy/Motwani/Naor]

"The Greedy Algorithm is Optimal for Online Edge Coloring"

* when $\Delta \leq \log n$ (deterministic)
when $\Delta \leq \sqrt{\log n}$ (randomized)

2021:

Is it though. . . ?



[Saberi/Wajc]

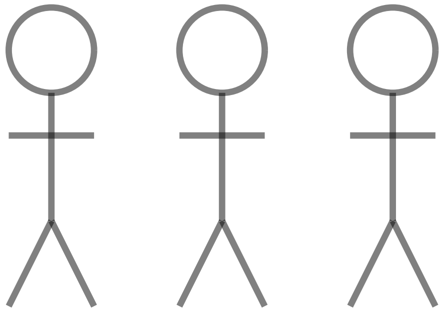
"The Greedy Algorithm is **Not** Optimal for Online Edge Coloring"

* when $\Delta \gg \log n$ (randomized)
under vertex-arrivals

Brief (Fictional?) History

1992:

Can we beat greedy?



[Bar-Noy/Motwani/Naor]

"The Greedy Algorithm is Optimal for Online Edge Coloring"

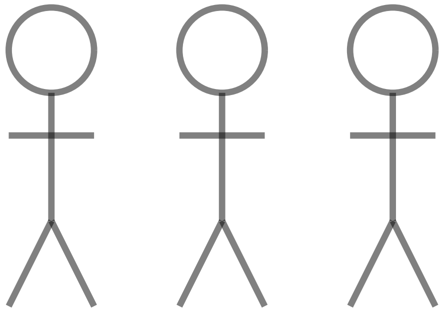
* when $\Delta \le \log n$ (deterministic)

when $\Delta \le \sqrt{\log n}$ (randomized)

Conjecture:
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Brief (Fictional?) History

1992: **Can we beat greedy?**

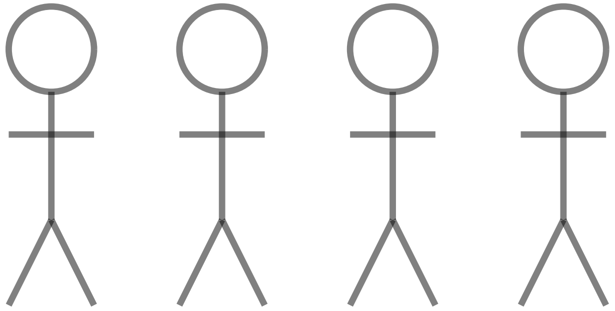


[Bar-Noy/Motwani/Naor]

"The Greedy Algorithm is Optimal for Online Edge Coloring"
* when $\Delta \leq \log n$ (deterministic)
when $\Delta \leq \sqrt{\log n}$ (randomized)

Conjecture:
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

2024:

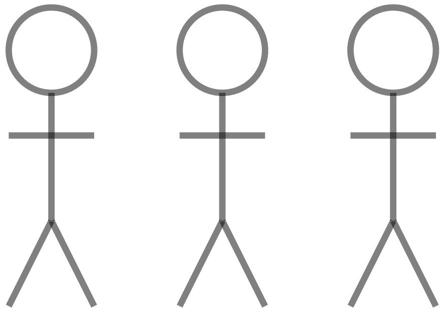


[**B.**/Svensson/Vintan/Wajc]

Brief (Fictional?) History

1992:

Can we beat greedy?



[Bar-Noy/Motwani/Naor]

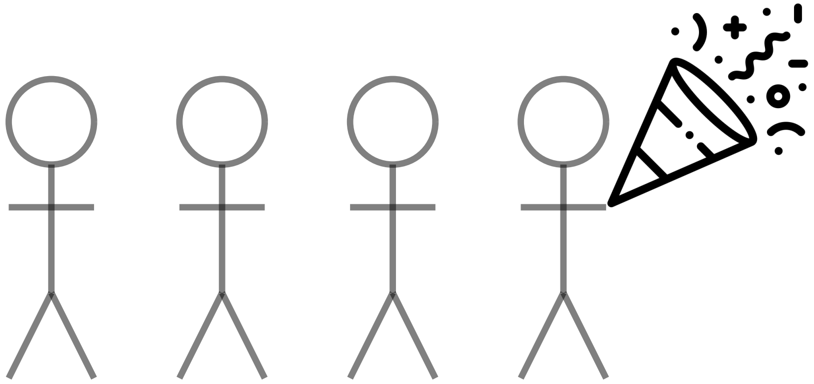
"The Greedy Algorithm is Optimal for Online Edge Coloring"

* when $\Delta \leq \log n$ (deterministic)

when $\Delta \leq \sqrt{\log n}$ (randomized)

~~Conjecture:~~ Theorem:
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

2024:



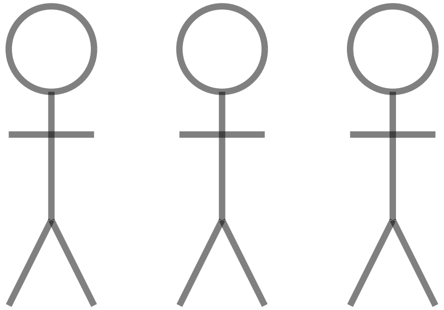
[**B.**/Svensson/Vintan/Wajc]

"Online Edge Coloring is (Nearly) as Easy as Offline"

The End?

Deterministic?

1992: **Can we beat greedy?**



[Bar-Noy/Motwani/Naor]

"The Greedy Algorithm is Optimal for Online Edge Coloring"

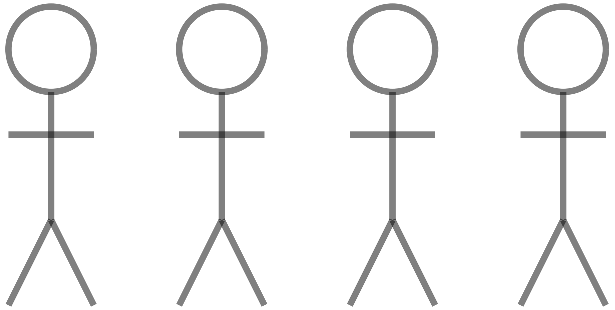
* when $\Delta \leq \log n$ (deterministic)

when $\Delta \leq \sqrt{\log n}$ (randomized)

~~**Conjecture:**~~ **Theorem:**

There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

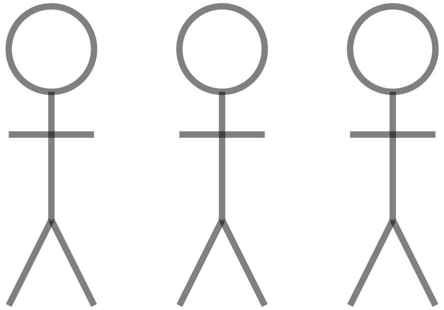
What about Deterministic Algos?



[**B.**/Svensson/Vintan/Wajc]

Deterministic?

1992: **Can we beat greedy?**



[Bar-Noy/Motwani/Naor]

"... less plausible for the deterministic case ..."

"The Greedy Algorithm is Optimal for Online Edge Coloring"

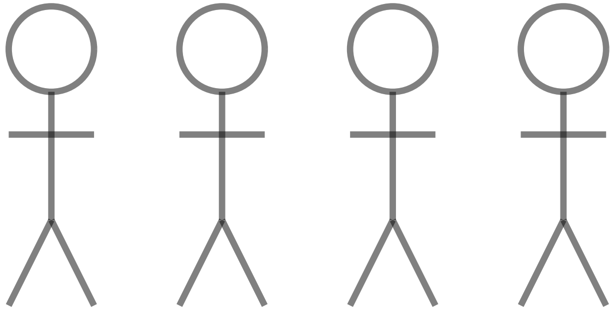
* when $\Delta \leq \log n$ (deterministic)

when $\Delta \leq \sqrt{\log n}$ (randomized)

~~**Conjecture:**~~ **Theorem:**

There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

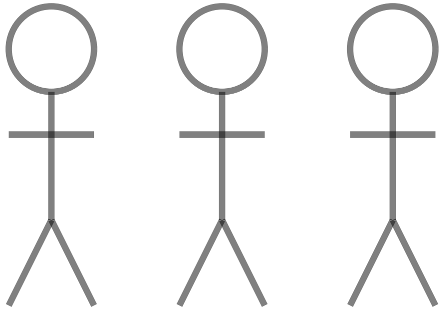
What about Deterministic Algos?



[**B.**/Svensson/Vintan/Wajc]

Deterministic?

1992: **Can we beat greedy?**



[Bar-Noy/Motwani/Naor]

"…less plausible for the deterministic case …"

"The Greedy Algorithm is Optimal for Online Edge Coloring"

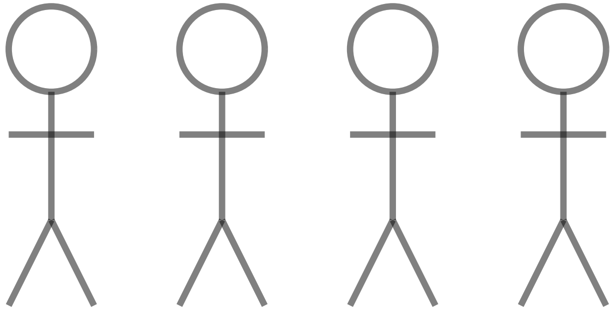
* when $\Delta \le \log n$ (deterministic)

when $\Delta \le \sqrt{\log n}$ (randomized)

~~Conjecture:~~ Theorem:

There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

What about Deterministic Algos?

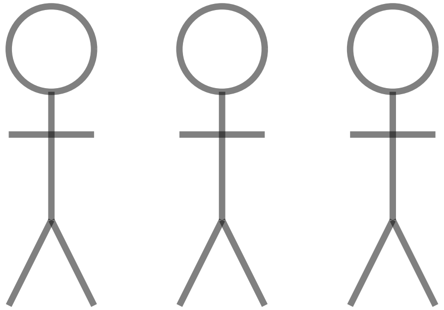


[**B.**/Svensson/Vintan/Wajc]

Can we improve the Lower Bounds?

Deterministic?

1992: **Can we beat greedy?**



[Bar-Noy/Motwani/Naor]

"…less plausible for the deterministic case …"

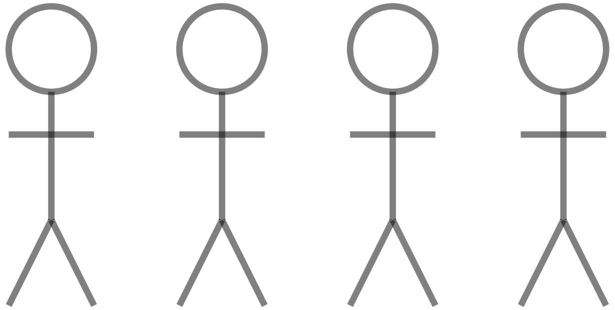
"The Greedy Algorithm is Optimal for Online Edge Coloring"

* when $\Delta \leq \log n$ (deterministic)

when $\Delta \leq \sqrt{\log n}$ (randomized)

~~Conjecture:~~ Theorem:
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

What about Deterministic Algos?

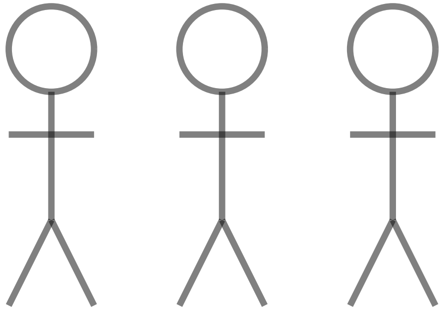


[**B.**/Svensson/Vintan/Wajc]

Can we improve the Lower Bounds?

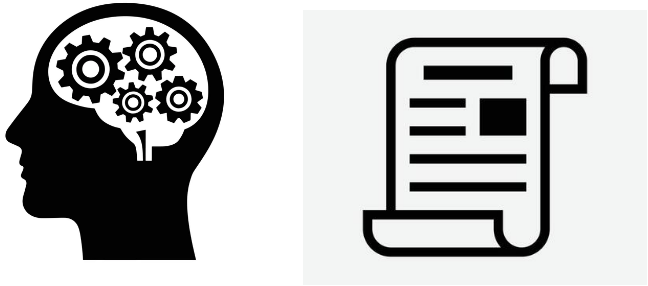
Deterministic?

1992: **Can we beat greedy?**



[Bar-Noy/Motwani/Naor]

"…less plausible for the deterministic case …"



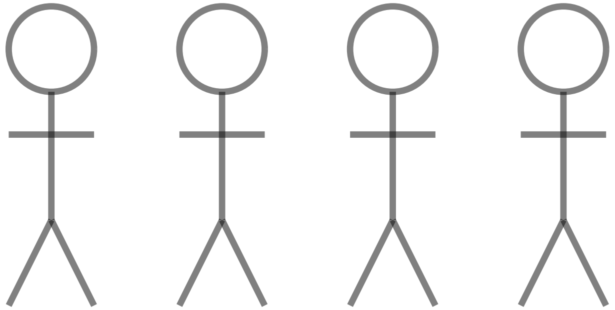
"The Greedy Algorithm is Optimal for Online Edge Coloring"

* when $\Delta \le \log n$ (deterministic)

when $\Delta \le \sqrt{\log n}$ (randomized)

~~Conjecture:~~ Theorem:
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

What about Deterministic Algos?

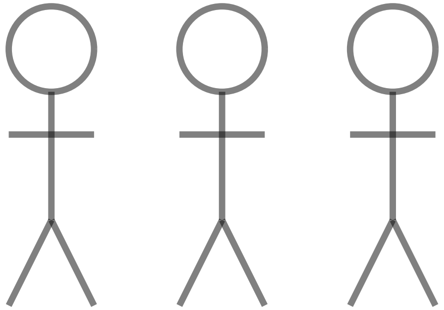


[**B.**/Svensson/Vintan/Wajc]

Can we improve the Lower Bounds?

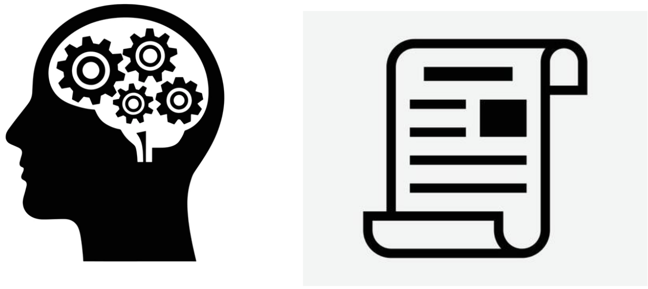
Deterministic?

1992: **Can we beat greedy?**



[Bar-Noy/Motwani/Naor]

"...less plausible for the deterministic case ..."

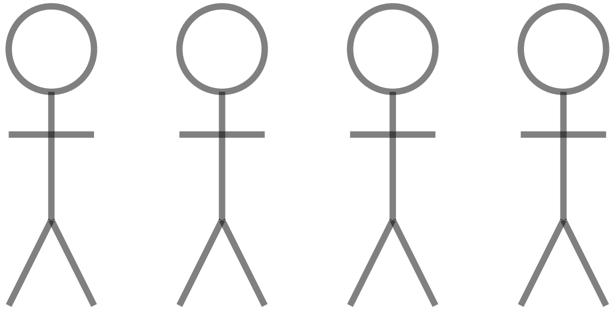


"The Greedy Algorithm is Optimal for Online Edge Coloring"

* when $\Delta \leq \log n$ (deterministic)
  when $\Delta \leq \sqrt{\log n}$ (randomized)

~~Conjecture:~~ Theorem:
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

What about Deterministic Algos?



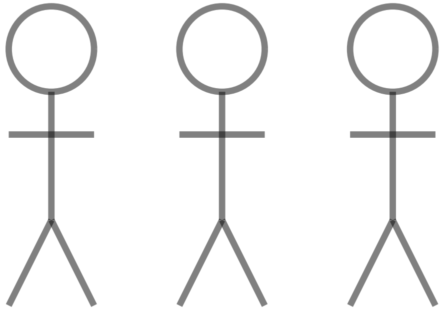
[**B.**/Svensson/Vintan/Wajc]

Can we improve the Lower Bounds?

Design Determinstic Algo?

Deterministic?

1992: **Can we beat greedy?**



[Bar-Noy/Motwani/Naor]

"...less plausible for the deterministic case ..."

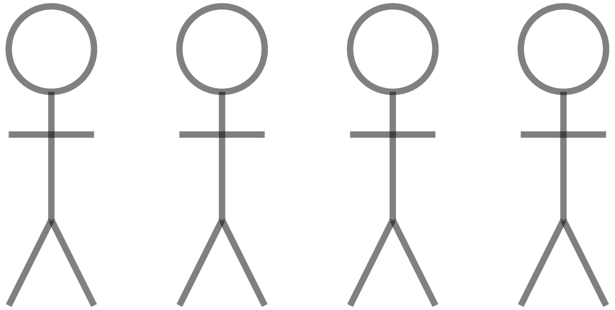
"The Greedy Algorithm is Optimal for Online Edge Coloring"

* when $\Delta \leq \log n$ (deterministic)

when $\Delta \leq \sqrt{\log n}$ (randomized)

~~Conjecture:~~ Theorem:
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

What about Deterministic Algos?



[**B.**/Svensson/Vintan/Wajc]

Can we improve the Lower Bounds?

Design Determinstic Algo?

SODA 2025: Two partial results

$\approx 1.58\Delta$ colors, vertex-arrivals [**B**SVW]

$\approx \Delta$ colors, when $\Delta = \Theta(n)$ [DGS]

This Paper!

Conjecture: [Bar-Noy/Motwani/Naor 1992]

There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Go beyond in two ways:

Theorem: [This Paper]

Online deterministic $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Online randomized $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.

This Paper!

Conjecture: [Bar-Noy/Motwani/Naor 1992]

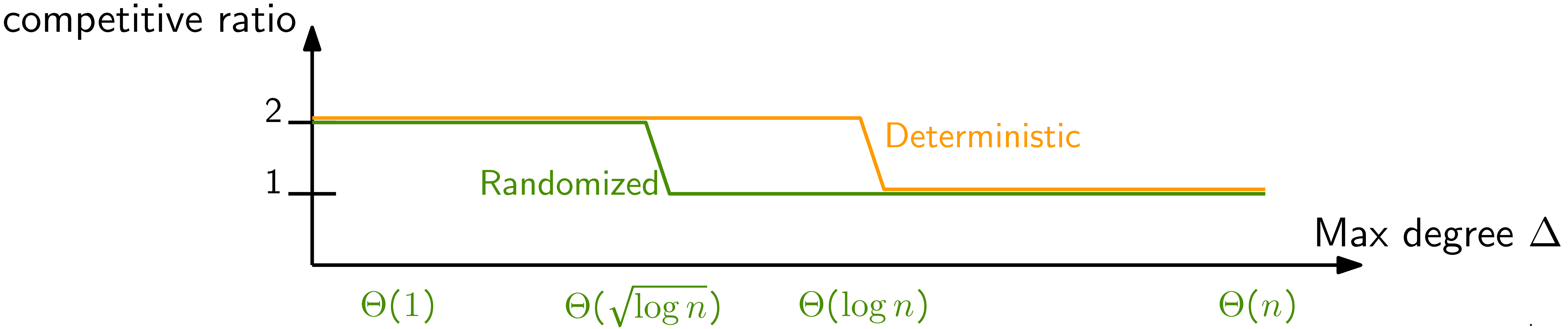
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Go beyond in two ways:

Theorem: [This Paper]

Online deterministic $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Online randomized $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.



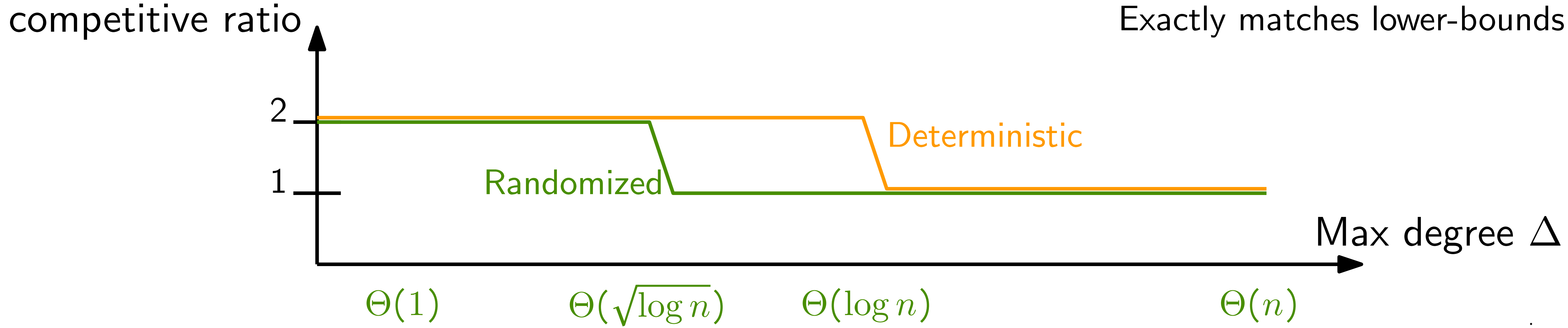
This Paper!

Conjecture: [Bar-Noy/Motwani/Naor 1992]
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Go beyond in two ways:

Theorem: [This Paper]
Online deterministic $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.
Online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.

Exactly matches lower-bounds



This Paper!

Conjecture: [Bar-Noy/Motwani/Naor 1992]
There is an online randomized $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Go beyond in two ways:

Theorem: [This Paper]
Online deterministic $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.
Online randomized $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.

Exactly matches lower-bounds

Our algo: $\Delta + O(\Delta^{15/16} \log^{1/16}(n))$ colors

Lowerbound: $\Delta + \Omega(\log n + \sqrt{\Delta})$

Open: close gap

The Other Player: Adversaries

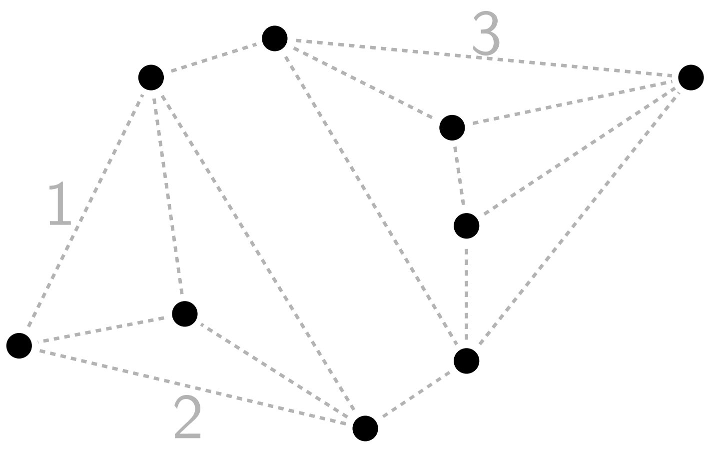
Oblivious

Adaptive

The Other Player: Adversaries

Oblivious

Fixes graph and
arrival order in advance

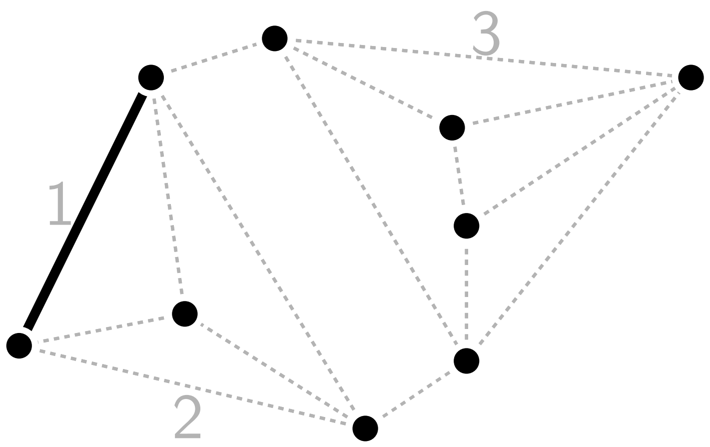


Adaptive

The Other Player: Adversaries

Oblivious

Fixes graph and
arrival order in advance

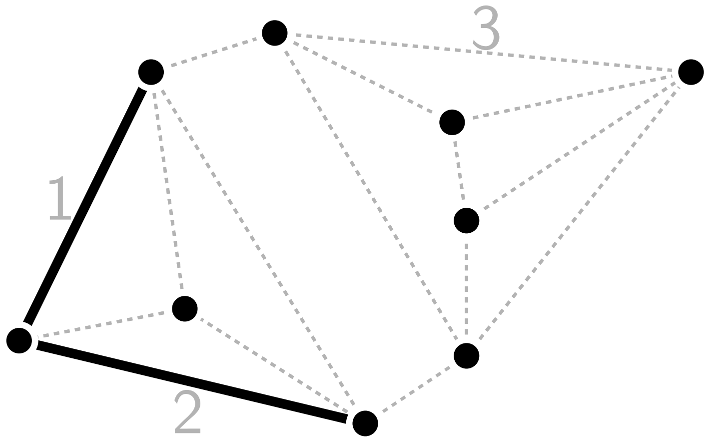


Adaptive

The Other Player: Adversaries

Oblivious

Fixes graph and
arrival order in advance

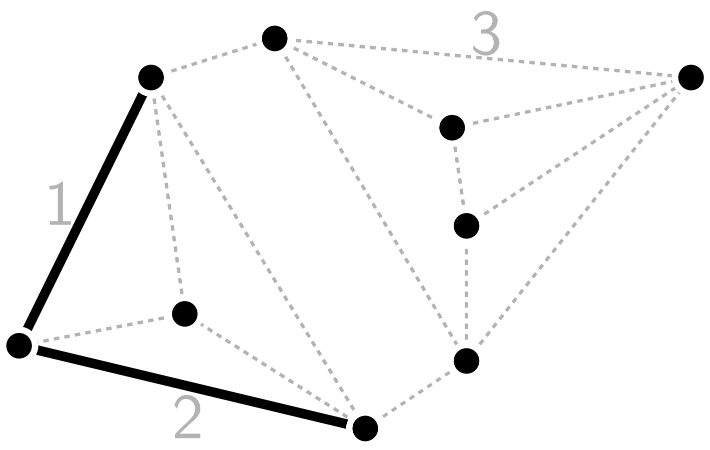


Adaptive

The Other Player: Adversaries

Oblivious

Fixes graph and
arrival order in advance



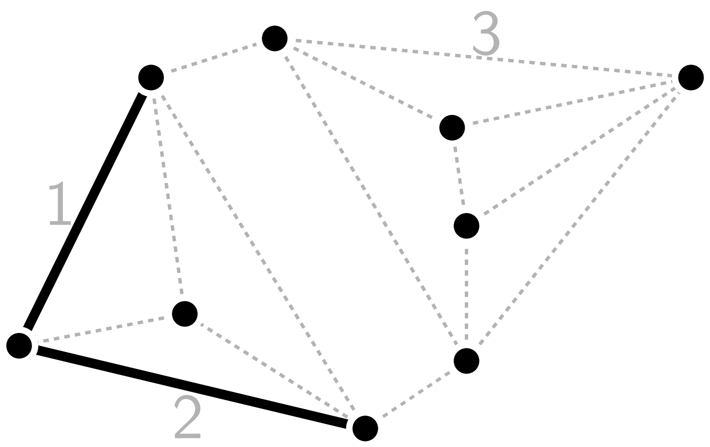
Adaptive

Generates graph adaptively based
on algorithms decisions/randomness

The Other Player: Adversaries

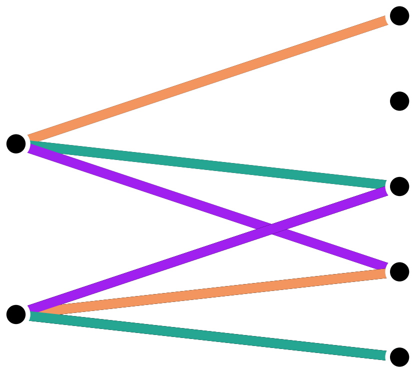
Oblivious

Fixes graph and
arrival order in advance



Adaptive

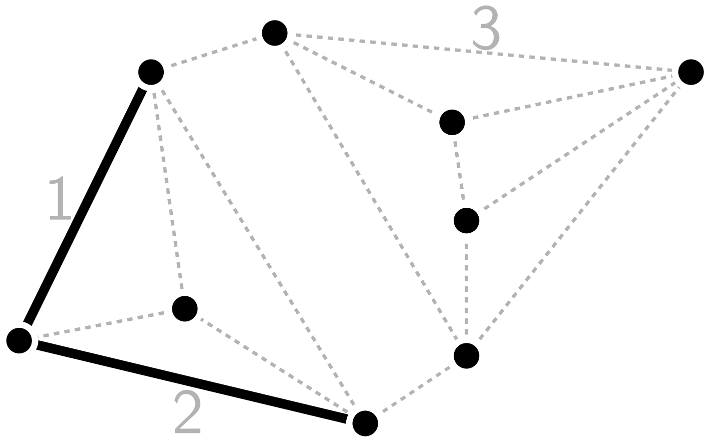
Generates graph adaptively based
on algorithms decisions/randomness



The Other Player: Adversaries

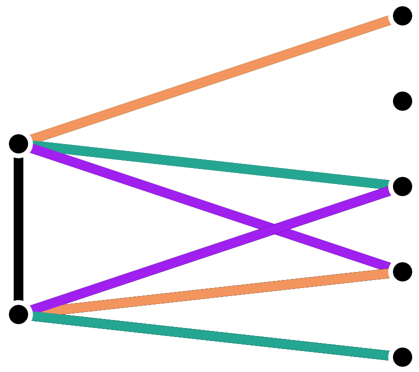
Oblivious

Fixes graph and
arrival order in advance



Adaptive

Generates graph adaptively based
on algorithms decisions/randomness

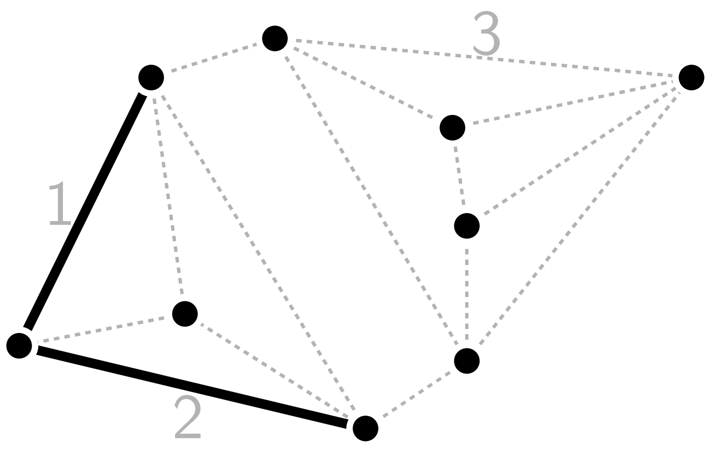


"I will connect to two vertices
where purple is taken"

The Other Player: Adversaries

Oblivious

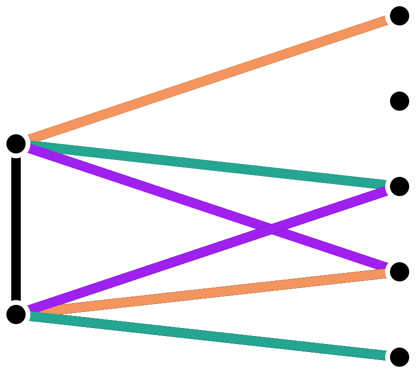
Fixes graph and
arrival order in advance



one (unknown) future

Adaptive

Generates graph adaptively based
on algorithms decisions/randomness



“I will connect to two vertices
where purple is taken”

many ($\gg n^{\Delta}$) possible futures

The Other Player: Adversaries

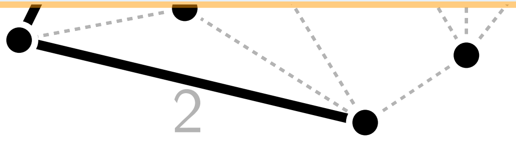
Oblivious

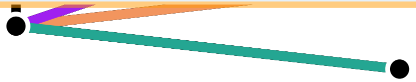
Fixes graph and
arrival order in advance

Adaptive

Generates graph adaptively based
on algorithms decisions/randomness

Randomness does not help against Adaptive adversary!





“I will connect to two vertices
where purple is taken”

one (unknown) future

many ($\gg n^{\Delta}$) possible futures

The Other Player: Adversaries

Oblivious

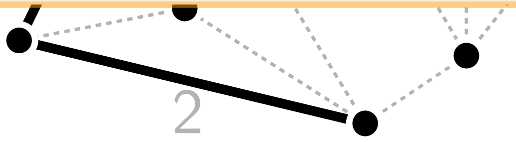
Fixes graph and
arrival order in advance

Adaptive

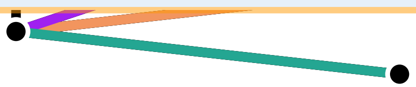
Generates graph adaptively based
on algorithms decisions/randomness

Randomness does not help against Adaptive adversary!

$\exists$ *randomized* online algorithm against *adaptive* adversary
$\implies \exists$ *determinstic* online algorithm



one (unknown) future



"I will connect to two vertices
where purple is taken"

many ($\gg n^{\Delta}$) possible futures

Deterministic

=

Randomized Against Adaptive Adversary

Algorithm & Analysis

ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Algorithm & Analysis

ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

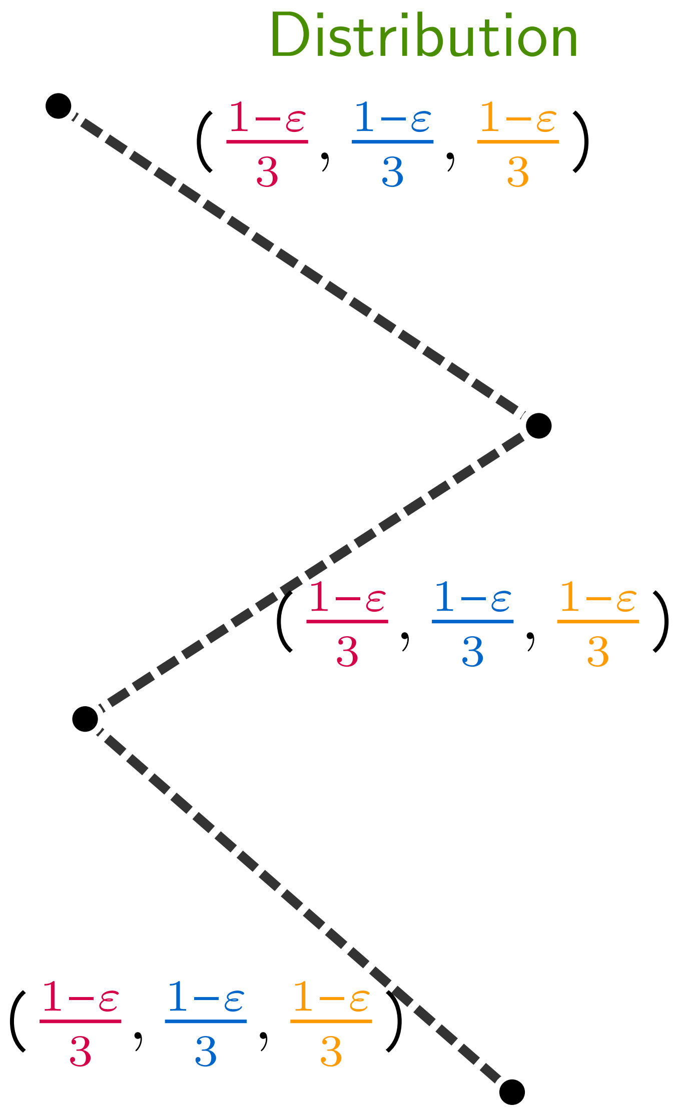
When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

Algorithm & Analysis



ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

Algorithm & Analysis

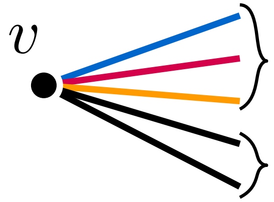
ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Goal: $\Pr[\text{we assign } e \text{ color from main palette}] = 1 - \varepsilon$

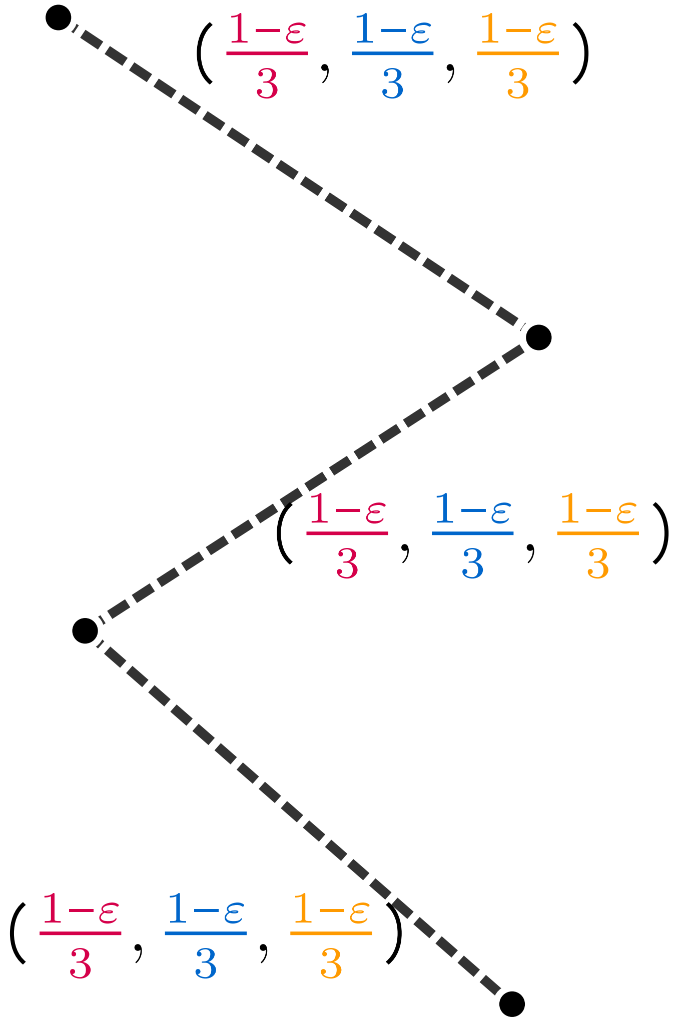


main palette

$\approx \varepsilon\Delta$ $\leftarrow$ emergency palette

"Forward to greedy"

Algorithm & Analysis



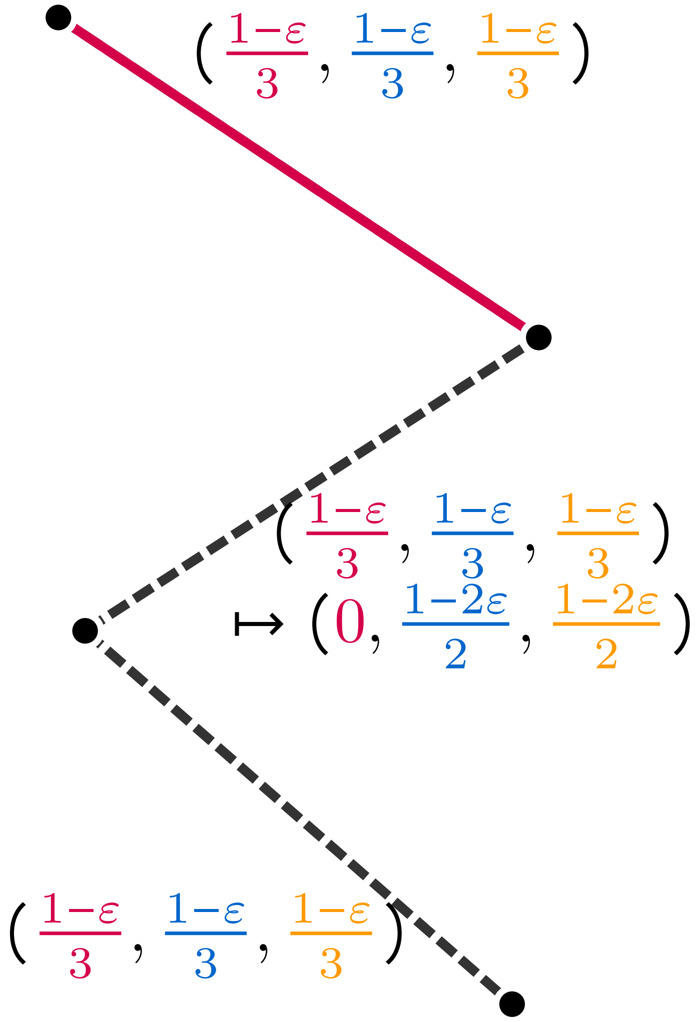
ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Algorithm & Analysis



ALGO: (simplified)

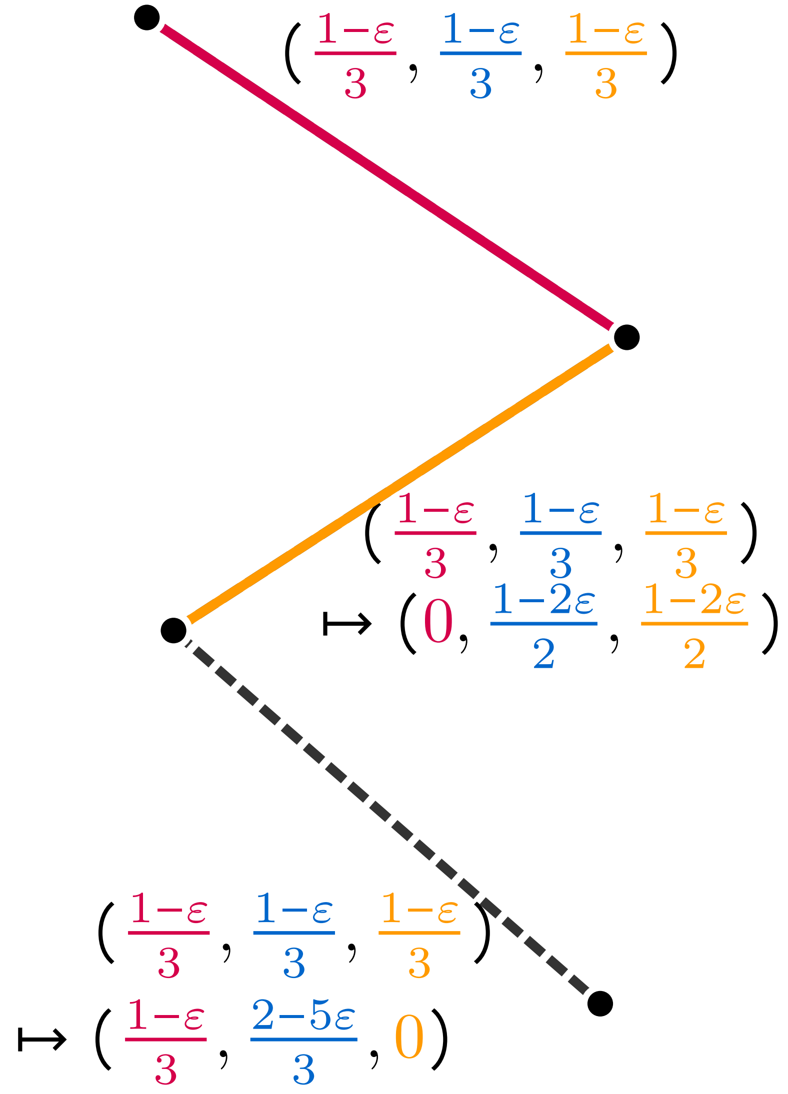
Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

"Bayesian Update"

Algorithm & Analysis



ALGO: (simplified)

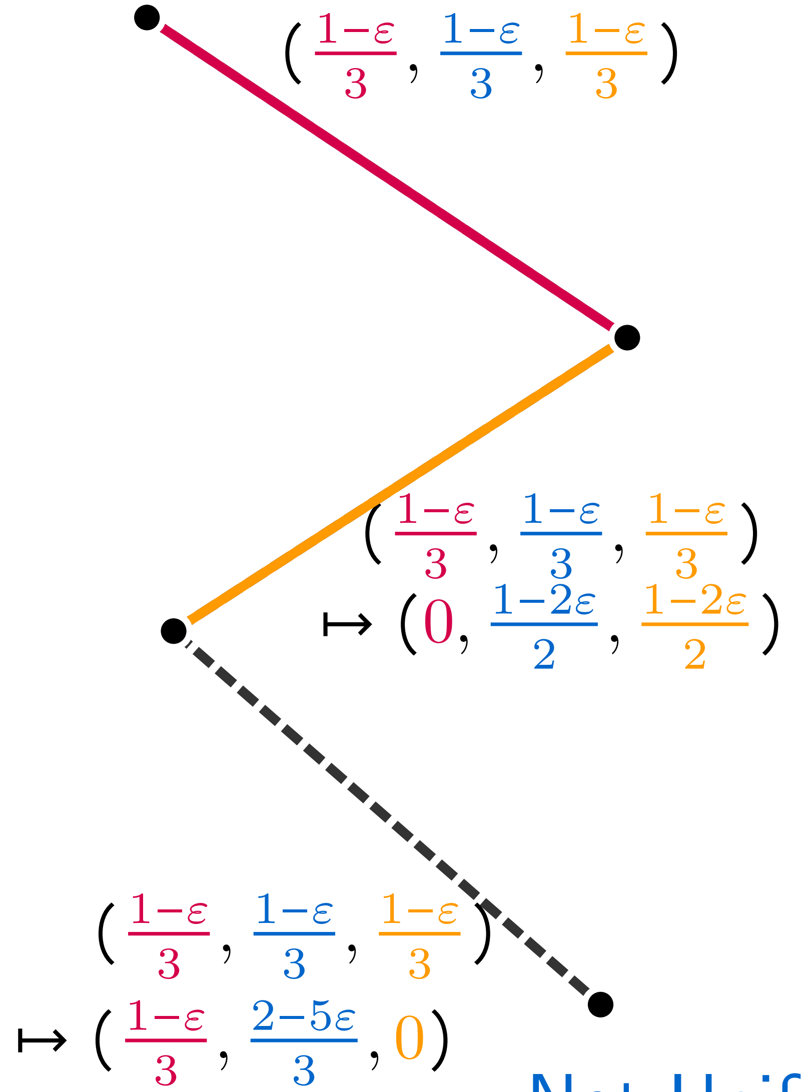
Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

"Bayesian Update"

Algorithm & Analysis



ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

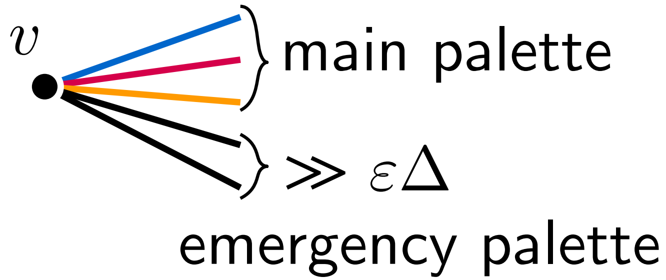
- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P_{f,c}^{\text{new}} \leftarrow 0$
  - Update $P_{f,k}^{\text{new}} \leftarrow P_{f,k}^{\text{old}} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

"Bayesian Update"

Not Uniform! Depends on execution

Algorithm & Analysis

- Fail if $\sum_c P_{e,c} > 1$
- Fail if $\sum_c P_{e,c} < 1 - 2\varepsilon$



ALGO: (simplified)

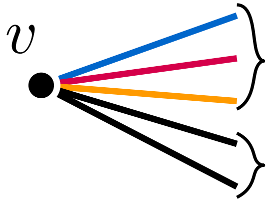
Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Algorithm & Analysis

- Fail if $\sum_c P_{e,c} > 1$
- Fail if $\sum_c P_{e,c} < 1 - 2\varepsilon$



At time $t = 0$:

$\forall e \in E, \quad \sum_c P_{e,c}^{(0)} = 1 - \varepsilon$

ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

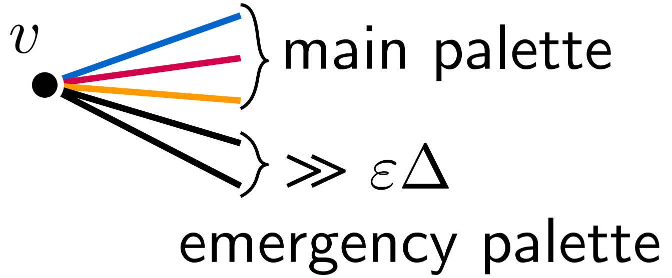
When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P_{f,c}^{\text{new}} \leftarrow 0$
  - Update $P_{f,k}^{\text{new}} \leftarrow P_{f,k}^{\text{old}} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Goal: Show $\sum_c P_{e,c}^{(t)} \in [1 - 2\varepsilon, 1]$ for all times t, and edges e

Algorithm & Analysis

- Fail if $\sum_c P_{e,c} > 1$
- Fail if $\sum_c P_{e,c} < 1 - 2\varepsilon$



At time $t = 0$:

$\forall e \in E, \quad \sum_c P_{e,c}^{(0)} = 1 - \varepsilon$

ALGO: (simplified)

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$

For potential future incident edges f:

Set $P_{f,c}^{\text{new}} \leftarrow 0$

Update $P_{f,k}^{\text{new}} \leftarrow P_{f,k}^{\text{old}} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Goal: Show $\sum_c P_{e,c}^{(t)} \in [1 - 2\varepsilon, 1]$ for all times t, and edges e

Martingale $\sum_c P_{e,c}^{(0)}$

Analysis: Continuation + Adversaries

ALGO:

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Analysis: Continuation + Adversaries

ALGO:

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

"Exercise": 😇 oblivious adversary and $\Delta \geq 10^4 \log n$

Analysis: Continuation + Adversaries

ALGO:

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

"Exercise":  oblivious adversary and $\Delta \geq 10^4 \log n$

$\Pr[\text{edge } e \text{ bad}] \leq [\text{Azuma's}] \leq e^{-\Delta} \ll \frac{1}{n^{100}}$

$\Longrightarrow$ simultaneously none of the $\leq n^2$ edges are bad, with high probability

Analysis: Continuation + Adversaries

ALGO:

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

$\Pr[\text{bad event}] = e^{-\Delta}$

$\Delta \approx \sqrt{\log n}$

one (unknown) future

$\Delta \approx \log n$

$n^{\Delta} \leq e^{\Delta^2}$ futures

"Exercise": oblivious adversary and $\Delta \geq 10^4 \log n$

$\Pr[\text{edge } e \text{ bad}] \leq [\text{Azuma's}] \leq e^{-\Delta} \ll \frac{1}{n^{100}}$

$\Longrightarrow$ simultaneously none of the $\leq n^2$ edges are bad, with high probability

Analysis: Continuation + Adversaries

ALGO:

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for $e \in E$ and $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

$\Pr[\text{bad event}] = \cancel{e^{-\Delta}}$ $e^{-\Delta^2}$

$\Delta \approx \sqrt{\log n}$ 😇
one (unknown) future

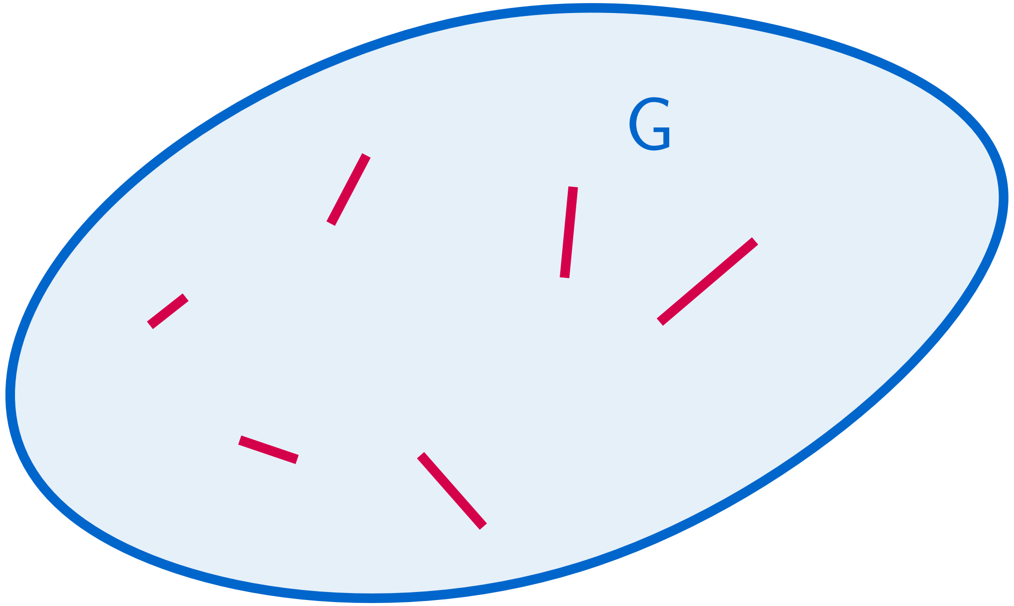
$\Delta \approx \log n$ 😈
$n^{\Delta} \le e^{\Delta^2}$ futures

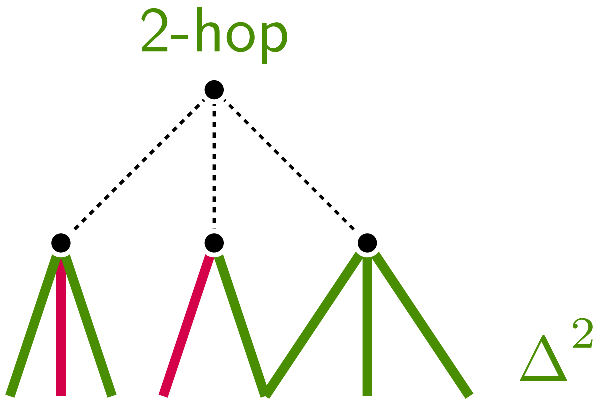
"Exercise": 😇 oblivious adversary and $\Delta \geq 10^4 \log n$

$\Pr[\text{edge } e \text{ bad}] \le [\text{Azuma's}] \le e^{-\Delta} \ll \frac{1}{n^{100}}$

$\Longrightarrow$ simultaneously none of the $\le n^2$ edges are bad, with high probability

Analysis: Continuation + Adversaries





$\varepsilon\Delta^2$ “bad” edges

$\Pr[\text{bad event}] = \cancel{e^{-\Delta}}$

$e^{-\Delta^2}$

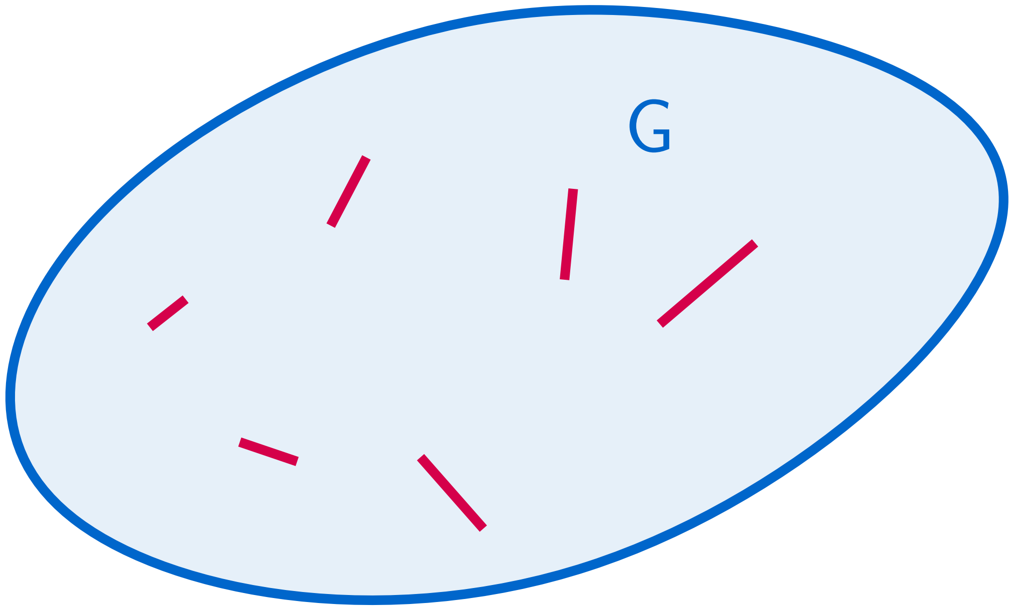
$\Delta \approx \sqrt{\log n}$ 😇
one (unknown) future

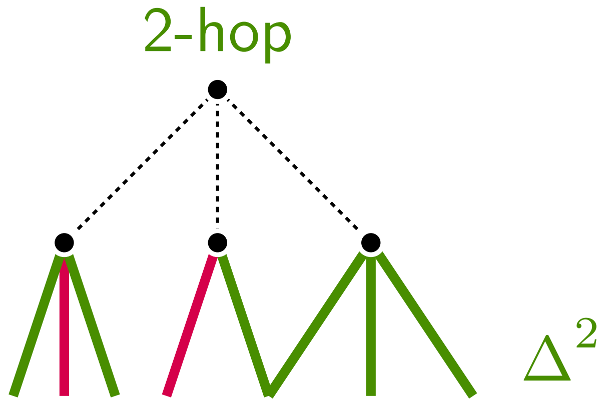
$\Delta \approx \log n$ 😈
$n^{\Delta} \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out

Theorem:
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious 😇
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive 😈

Analysis: Continuation + Adversaries





$\varepsilon\Delta^2$ "bad" edges

Pr[bad event] = $\cancel{e^{-\Delta}}$ $e^{-\Delta^2}$

$\Delta \approx \sqrt{\log n}$
one (unknown) future

$\Delta \approx \log n$
$n^{\Delta} \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out
Add special-case handling for various bad events

Theorem:
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive

Analysis: Continuation + Adversaries

Synonyms for "bad" events

bad color

$\Pr[\text{bad event}] = \cancel{e^{-\Delta}}$

$e^{-\Delta^2}$

$\Delta \approx \sqrt{\log n}$ 😇

one (unknown) future

$\Delta \approx \log n$ 😈

$n^{\Delta} \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out

Add special-case handling for various bad events

Theorem:

There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious 😇

There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive 😈

Analysis: Continuation + Adversaries

Synonyms for "bad" events

bad color

bad vertex

$\Pr[\text{bad event}] = \cancel{e^{-\Delta}}$

$e^{-\Delta^2}$

$\Delta \approx \sqrt{\log n}$ 😇
one (unknown) future

$\Delta \approx \log n$ 😈
$n^{\Delta} \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out
Add special-case handling for various bad events

Theorem:
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious 😇
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive 😈

Analysis: Continuation + Adversaries

Synonyms for "bad" events

bad color

bad vertex

dangerous vertex

$\Pr[\text{bad event}] = \cancel{e^{-\Delta}}$

$e^{-\Delta^2}$

$\Delta \approx \sqrt{\log n}$ 😇
one (unknown) future

$\Delta \approx \log n$ 😈
$n^{\Delta} \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out
Add special-case handling for various bad events

Theorem:
There is an $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious 😇
There is an $(1+o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive 😈

Analysis: Continuation + Adversaries

Synonyms for "bad" events

bad color

hot vertex

bad vertex

dangerous vertex

$\Pr[\text{bad event}] = \cancel{e^{-\Delta}}$

$e^{-\Delta^2}$

$\Delta \approx \sqrt{\log n}$ 😇
one (unknown) future

$\Delta \approx \log n$ 😈
$n^\Delta \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out
Add special-case handling for various bad events

Theorem:
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious 😇
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive 😈

Analysis: Continuation + Adversaries

Synonyms for "bad" events

bad color

hot vertex

bad vertex

annoying edge

dangerous vertex

$\Pr[\text{bad event}] = \cancel{e^{-\Delta}}$ $e^{-\Delta^2}$

$\Delta \approx \sqrt{\log n}$ 😇
one (unknown) future

$\Delta \approx \log n$ 😈
$n^{\Delta} \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out
Add special-case handling for various bad events

Theorem:
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious 😇
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive 😈

Analysis: Continuation + Adversaries

Synonyms for "bad" events

bad color

bad vertex

dangerous vertex

hot vertex

annoying edge

unlucky edge

Pr[bad event] = $\cancel{e^{-\Delta}}$ $e^{-\Delta^2}$

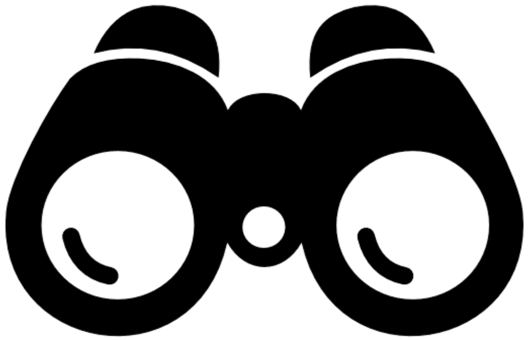
$\Delta \approx \sqrt{\log n}$ 😇
one (unknown) future

$\Delta \approx \log n$ 😈
$n^{\Delta} \leq e^{\Delta^2}$ futures

Main Idea to Fix: Bad events do happen, but they are spread out
Add special-case handling for various bad events

Theorem:
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$ against oblivious 😇
There is an $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$ against adaptive 😈

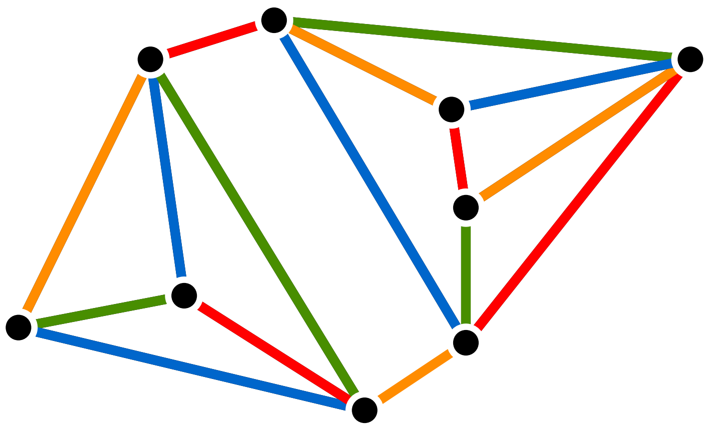
Summary and Future Thoughts



Summary

Greedy $2\Delta - 1$ coloring optimal when Δ small

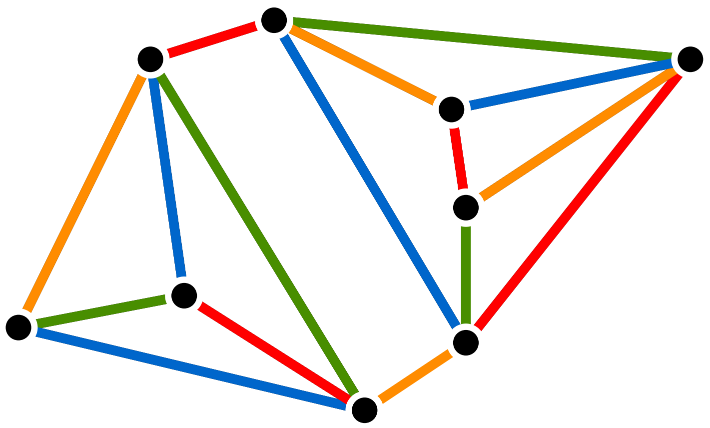
Conjecture: [Bar-Noy/Motwani/Naor 1992]
There is an online randomized $\approx \Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.



Summary

Greedy $2\Delta - 1$ coloring optimal when Δ small

Conjecture: [Bar-Noy/Motwani/Naor 1992] **Theorem:** [BSVW 2024]
There is an online randomized $\approx \Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

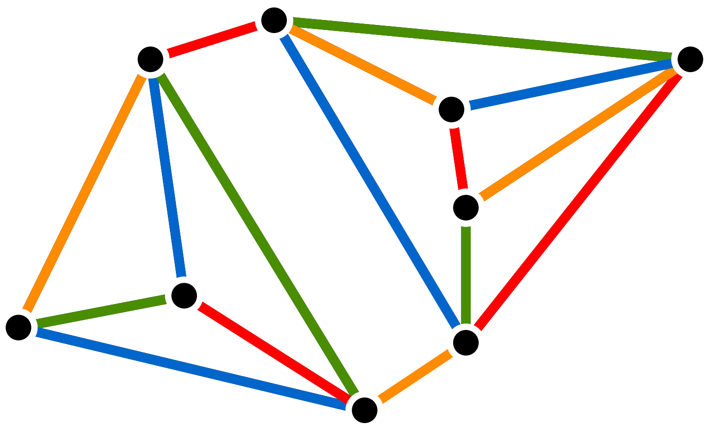


Summary

Greedy $2\Delta - 1$ coloring optimal when Δ small

Conjecture: [Bar-Noy/Motwani/Naor 1992] **Theorem:** [BSVW 2024]
There is an online randomized $\approx \Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Theorem: [This Paper]
Online deterministic $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.
Online randomized $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.



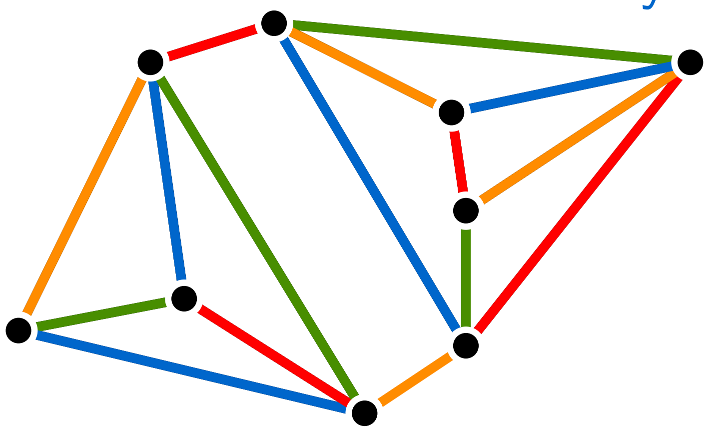
Summary

Greedy $2\Delta - 1$ coloring optimal when Δ small

Conjecture: [Bar-Noy/Motwani/Naor 1992] **Theorem:** [BSVW 2024]
There is an online randomized $\approx \Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Theorem: [This Paper]
Online deterministic $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.
Online randomized $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.

Exactly matches lower bounds



Summary

Greedy $2\Delta - 1$ coloring optimal when Δ small

Conjecture: [Bar-Noy/Motwani/Naor 1992] **Theorem:** [BSVW 2024]
There is an online randomized $\approx \Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Theorem: [This Paper]
Online deterministic $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.
Online randomized $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.

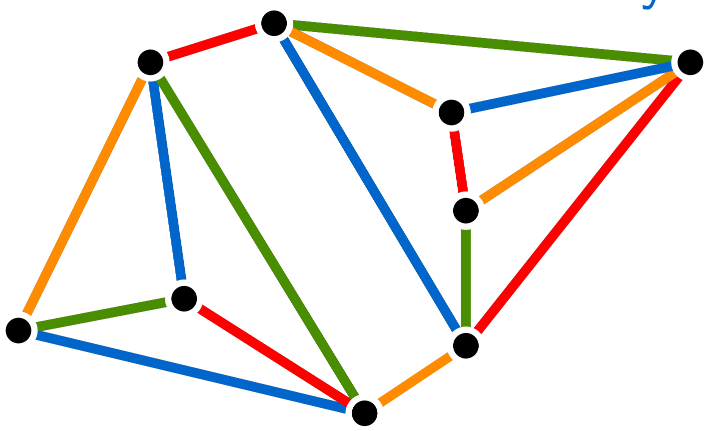
Exactly matches lower bounds

Techniques:

Bayesian algorithm

Deterministic $=$ Randomized vs Adaptive 😈

Martingale concentration $e^{-\Delta^2}$ 😇



Summary

Greedy $2\Delta - 1$ coloring optimal when Δ small

Conjecture: [Bar-Noy/Motwani/Naor 1992] **Theorem:** [BSVW 2024]
There is an online randomized $\approx \Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.

Theorem: [This Paper]
Online deterministic $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\log n)$.
Online randomized $(1 + o(1))\Delta$-edge-coloring algorithm when $\Delta = \omega(\sqrt{\log n})$.

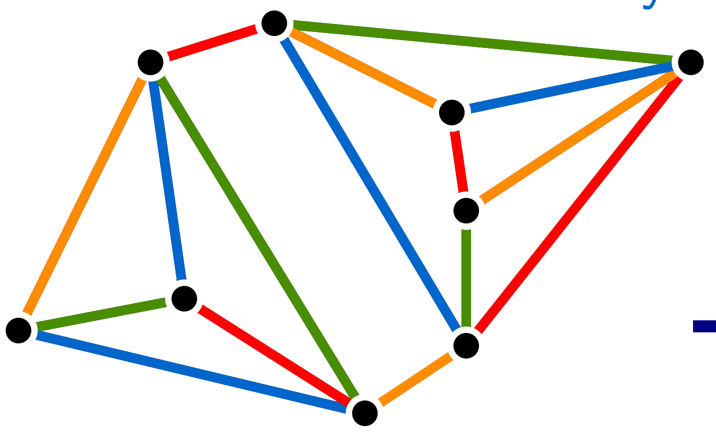
Exactly matches lower bounds

Techniques:

Bayesian algorithm

Deterministic = Randomized vs Adaptive 😈

Martingale concentration $e^{-\Delta^2}$ 😇



Thanks!

Extra Slides

Edge Coloring Open Problems:

- Hypergraphs & Multigraphs
- Pinpoint the $o(1)$ term:

  Our algo: $\Delta^{15/16}\log^{1/16}(n)$ extra colors

  Lowerbound: $\log n + \sqrt{\Delta}$

- Rounding fractional matchings
- Similar techniques for other problems: online weighted matching?
- List-Edge-Coloring Conjecture

Lower Bound: "Greedy is Optimal for Online Edge Coloring"

Theorem: No online algorithm can $(2\Delta - 2)$-edge-color every graph.

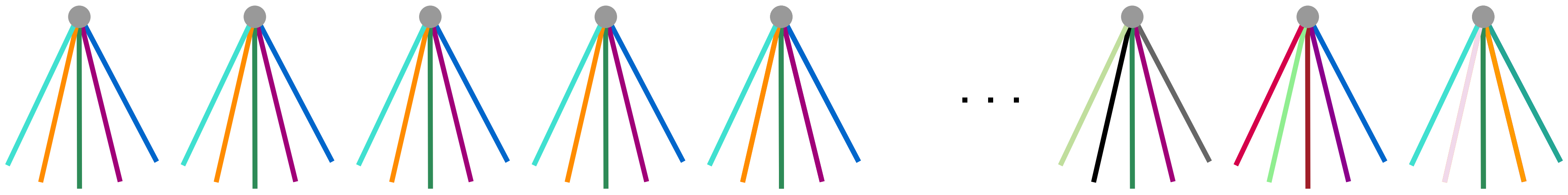
[Bar-Noy/Motwani/Naor 1992]

Lower Bound: "Greedy is Optimal for Online Edge Coloring"

Theorem: No online algorithm can $(2\Delta - 2)$-edge-color every graph.

[Bar-Noy/Motwani/Naor 1992]

Idea (Adversary): Create lots of $(\Delta - 1)$-stars



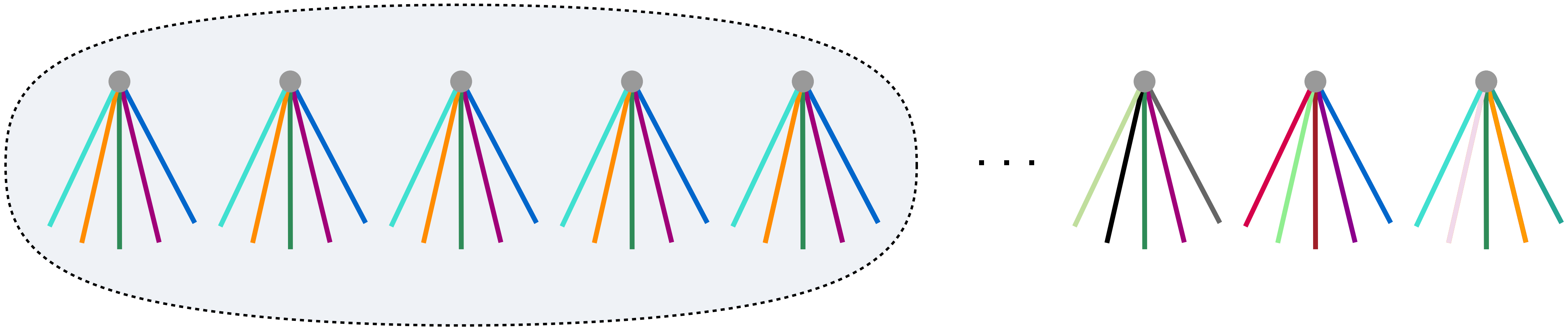
Lower Bound: "Greedy is Optimal for Online Edge Coloring"

Theorem: No online algorithm can $(2\Delta - 2)$-edge-color every graph.

[Bar-Noy/Motwani/Naor 1992]

Idea (Adversary): Create lots of $(\Delta - 1)$-stars

Eventually have Δ stars colored the same (pigeonhole principle)



Lower Bound: “Greedy is Optimal for Online Edge Coloring”

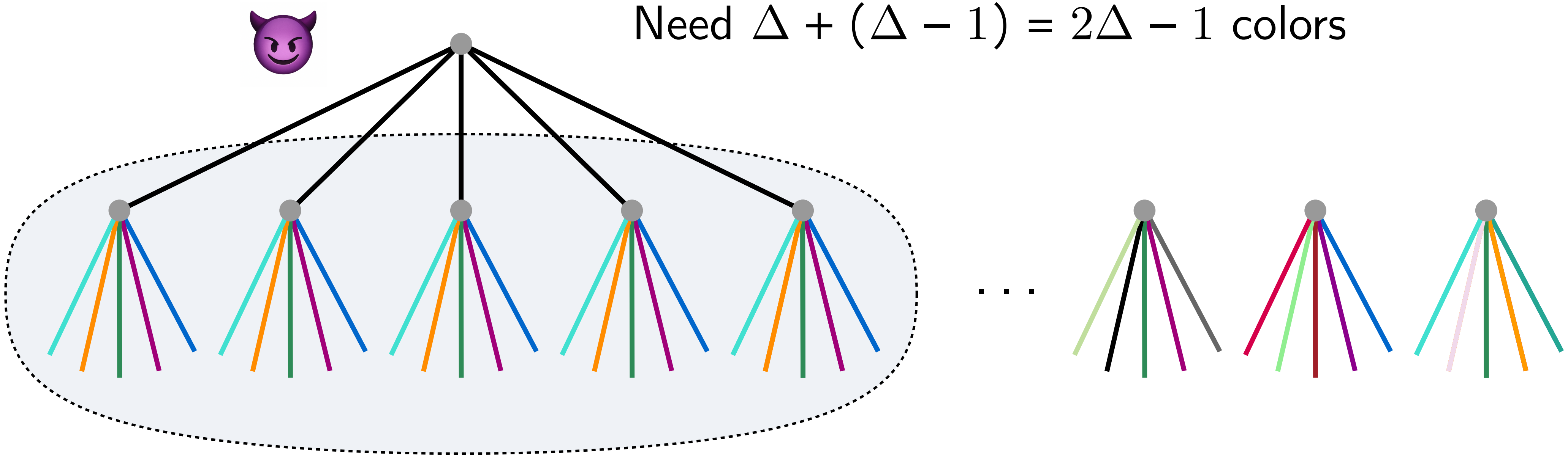
Theorem: No online algorithm can $(2\Delta - 2)$-edge-color every graph.

[Bar-Noy/Motwani/Naor 1992]

Idea (Adversary): Create lots of $(\Delta - 1)$-stars

Eventually have Δ stars colored the same (pigeonhole principle)

Need $\Delta + (\Delta - 1) = 2\Delta - 1$ colors



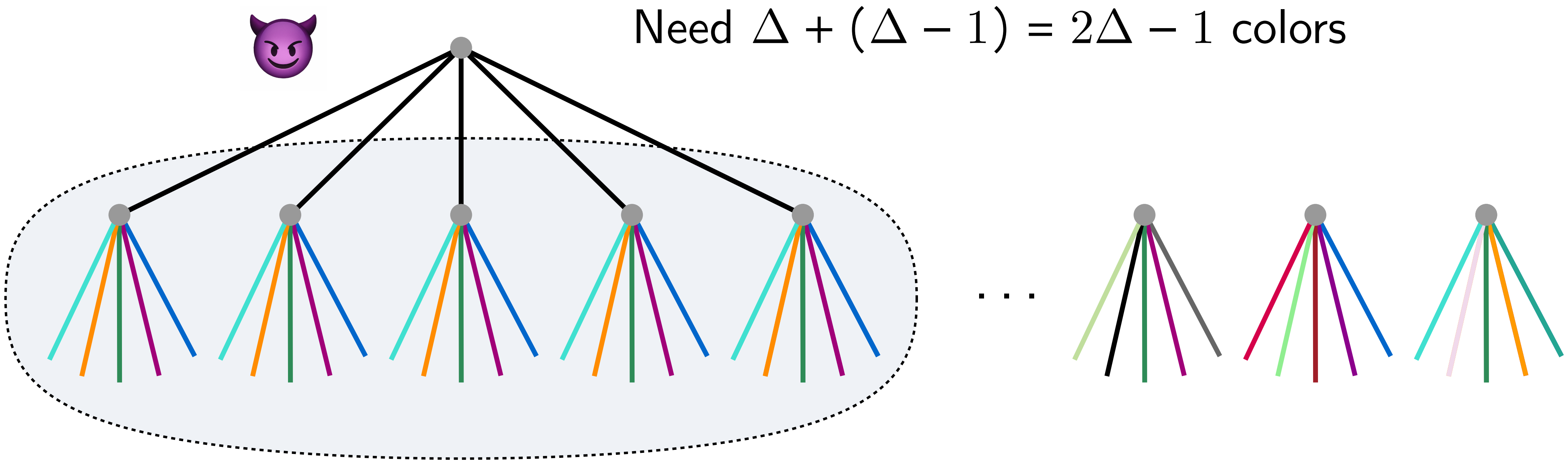
Lower Bound: "Greedy is Optimal for Online Edge Coloring"

Theorem: No online algorithm can $(2\Delta - 2)$-edge-color every graph.

[Bar-Noy/Motwani/Naor 1992]

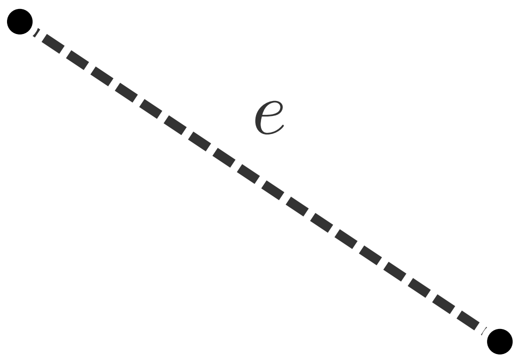
Idea (Adversary): Create lots of $(\Delta - 1)$-stars

Eventually have Δ stars colored the same (pigeonhole principle)



Our Algorithm – Main Palette

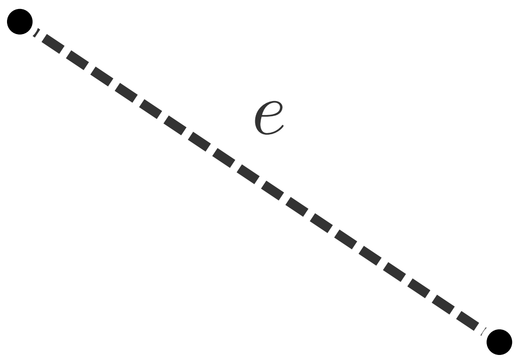
Focus on arriving edge e



Our Algorithm – Main Palette

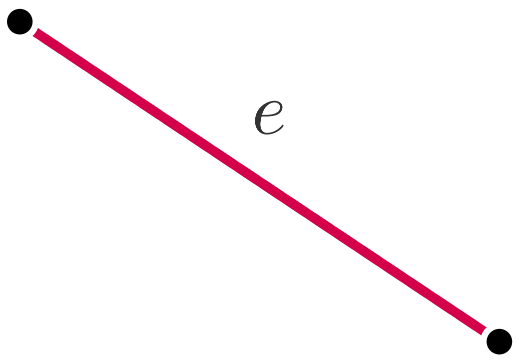
Focus on arriving edge e

Goal: $\Pr[\text{we assign } e \text{ color } c] = \frac{1-\varepsilon}{\Delta}$



Our Algorithm – Main Palette

Focus on arriving edge e



Goal: $\Pr[\text{we assign } e \text{ color } c] = \frac{1-\varepsilon}{\Delta}$

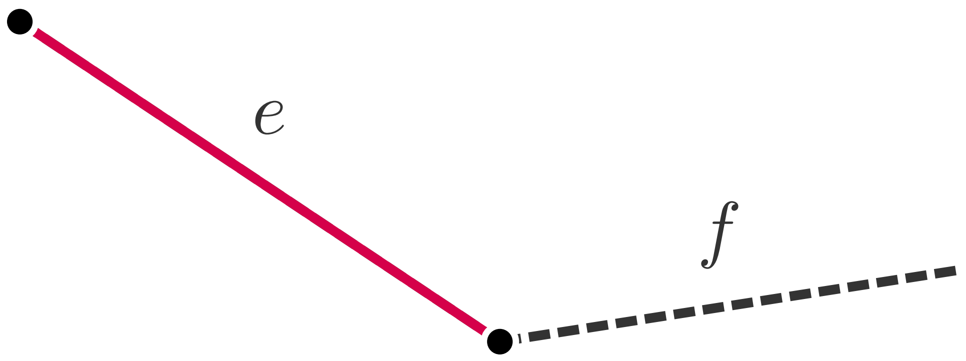
Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for each color $c \in \{1, 2 \ldots, \Delta\}$

When e arrives: $\Pr$ left uncolored $\rightarrow$ emergency palette

Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$

Our Algorithm – Main Palette

Focus on arriving edge e



Goal: $\Pr[\text{we assign } e \text{ color } c] = \frac{1-\varepsilon}{\Delta}$

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for each color $c \in \{1, 2 \ldots, \Delta\}$

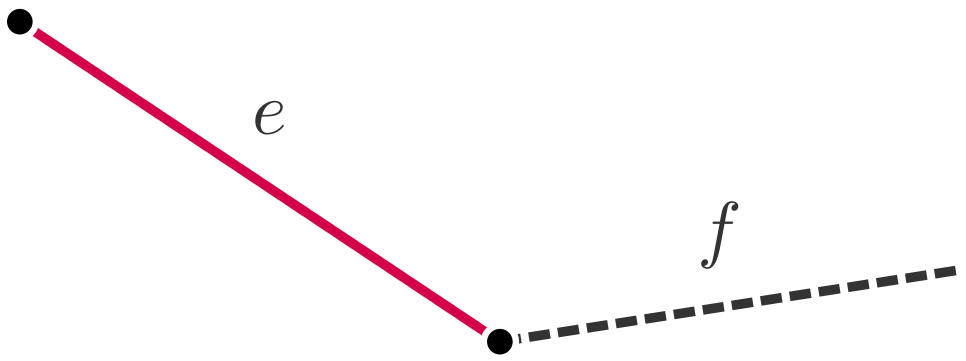
When e arrives:

$\Pr$ left uncolored → emergency palette

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Need to set $P^{\text{new}}_{f,c} \leftarrow 0$

Our Algorithm – Main Palette

Focus on arriving edge e



Goal: $\Pr[\text{we assign } e \text{ color } c] = \frac{1-\varepsilon}{\Delta}$

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for each color $c \in \{1, 2 \ldots, \Delta\}$

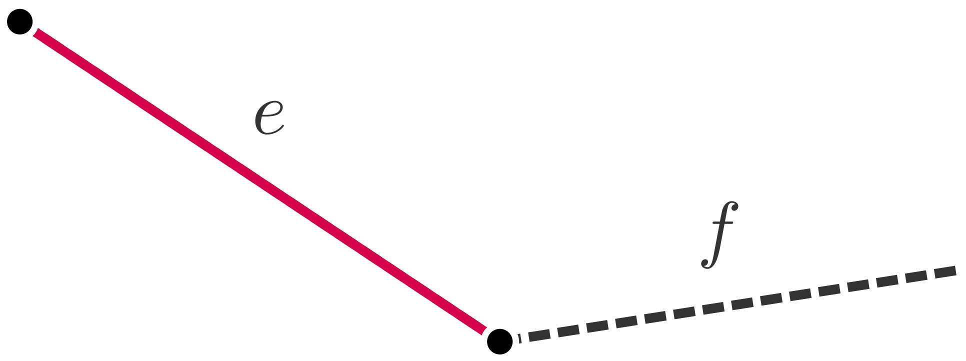
When e arrives:

$\Pr$ left uncolored $\rightarrow$ emergency palette

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Need to set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

Our Algorithm – Main Palette

Focus on arriving edge e



Goal: $\Pr[$we assign e color $c] = \frac{1-\varepsilon}{\Delta}$

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for each color $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

$\Pr$ left uncolored $\rightarrow$ emergency palette

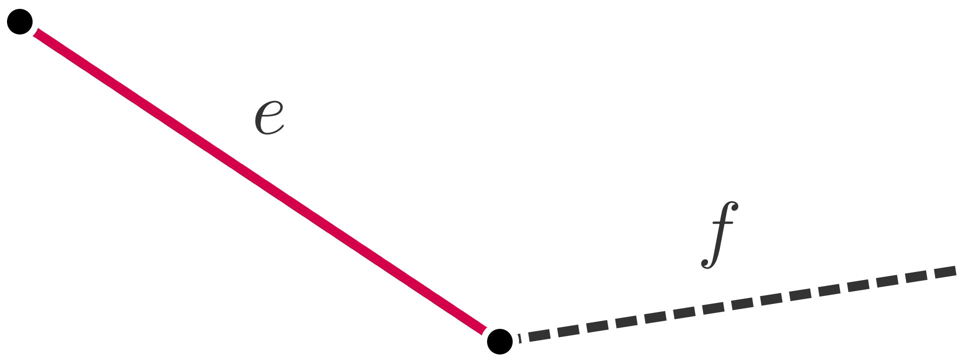
- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Need to set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

"Bayesian update"

$$\begin{aligned}
\Pr[f \text{ col } k] &= \\
&= \Pr[f \text{ col } k \mid e \text{ col } k]\Pr[e \text{ col } k] \\
&\quad + \Pr[f \text{ col } k \mid e \text{ not col } k]\Pr[e \text{ not col } k] \\
&= 0 \cdot P_{e,k} + P^{\text{new}}_{f,k} \cdot (1 - P_{e,k}) \\
&= P^{\text{old}}_{f,k}
\end{aligned}$$

Our Algorithm – Main Palette

Focus on arriving edge e



Goal: $\Pr[\text{we assign } e \text{ color } c] = \frac{1-\varepsilon}{\Delta}$

$\implies$ OK

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for each color $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

Pr left uncolored → emergency palette

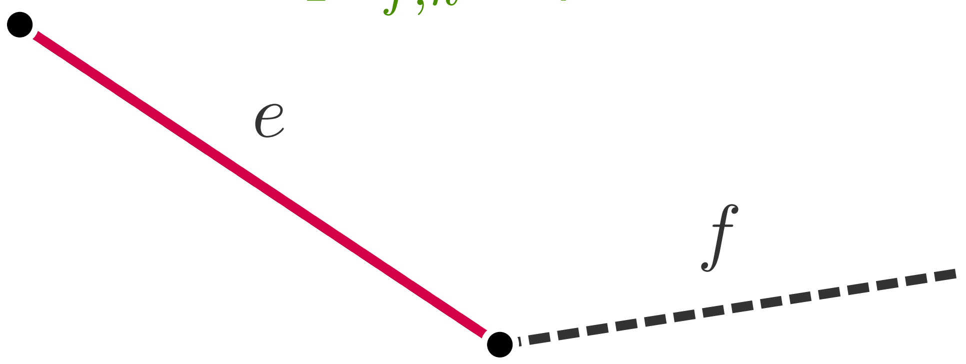
- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Need to set $P^{\text{new}}_{f,c} \leftarrow 0$
  - Update $P^{\text{new}}_{f,k} \leftarrow P^{\text{old}}_{f,k} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

"Bayesian update"

$$\begin{aligned}
\Pr[f \text{ col } k] &= \\
&= \Pr[f \text{ col } k \mid e \text{ col } k]\Pr[e \text{ col } k] \\
&\quad + \Pr[f \text{ col } k \mid e \text{ not col } k]\Pr[e \text{ not col } k] \\
&= 0 \cdot P_{e,k} + P^{\text{new}}_{f,k} \cdot (1 - P_{e,k}) \\
&= P^{\text{old}}_{f,k}
\end{aligned}$$

Our Algorithm – Main Palette

Focus on arriving edge e



$E[P_{f,k}^{(t+1)} \mid \text{past randomness}] = P_{f,k}^{(t)}$

Martingale!

Goal: $\Pr[\text{we assign } e \text{ color } c] = \frac{1-\varepsilon}{\Delta}$

$\implies$ OK

Initialize $P_{e,c} \leftarrow \frac{1-\varepsilon}{\Delta}$ for each color $c \in \{1, 2 \ldots, \Delta\}$

When e arrives:

$\Pr$ left uncolored $\rightarrow$ emergency palette

- Sample color c from $(P_{e,1}, P_{e,2}, \ldots, P_{e,\Delta}, 1 - \sum_c P_{e,c})$
- For potential future incident edges f:
  - Need to set $P_{f,c}^{\text{new}} \leftarrow 0$
  - Update $P_{f,k}^{\text{new}} \leftarrow P_{f,k}^{\text{old}} \cdot \frac{1}{1-P_{e,k}}$ for all colors $k \neq c$

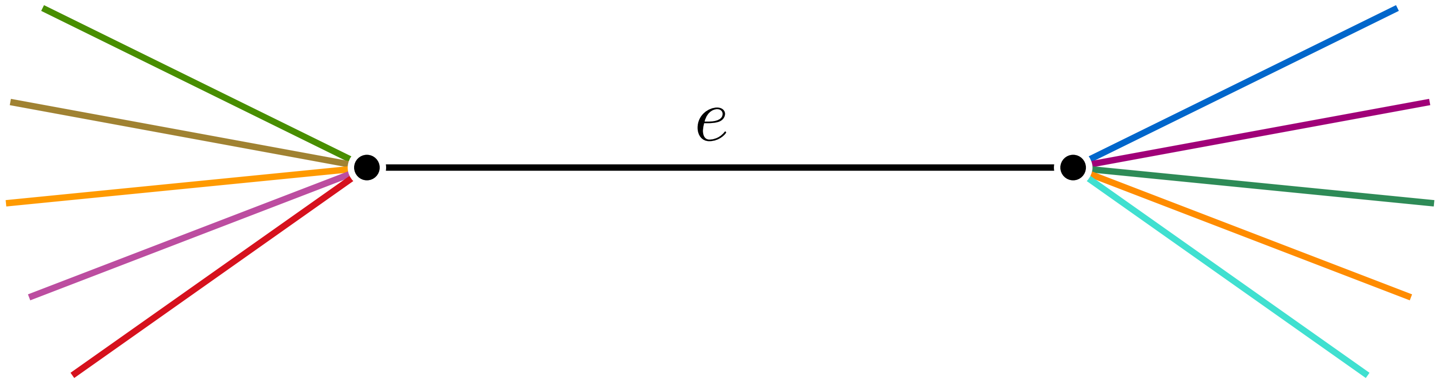
"Bayesian update"

$$\begin{aligned}
\Pr[f \text{ col } k] &= \\
&= \Pr[f \text{ col } k \mid e \text{ col } k]\Pr[e \text{ col } k] \\
&\quad + \Pr[f \text{ col } k \mid e \text{ not col } k]\Pr[e \text{ not col } k] \\
&= 0 \cdot P_{e,k} + P_{f,k}^{\text{new}} \cdot (1 - P_{e,k}) \\
&= P_{f,k}^{\text{old}}
\end{aligned}$$

Candidate Algorithm: "Randomized Greedy"

Use palette $\{1, 2, \ldots, (1+\varepsilon)\Delta\}$

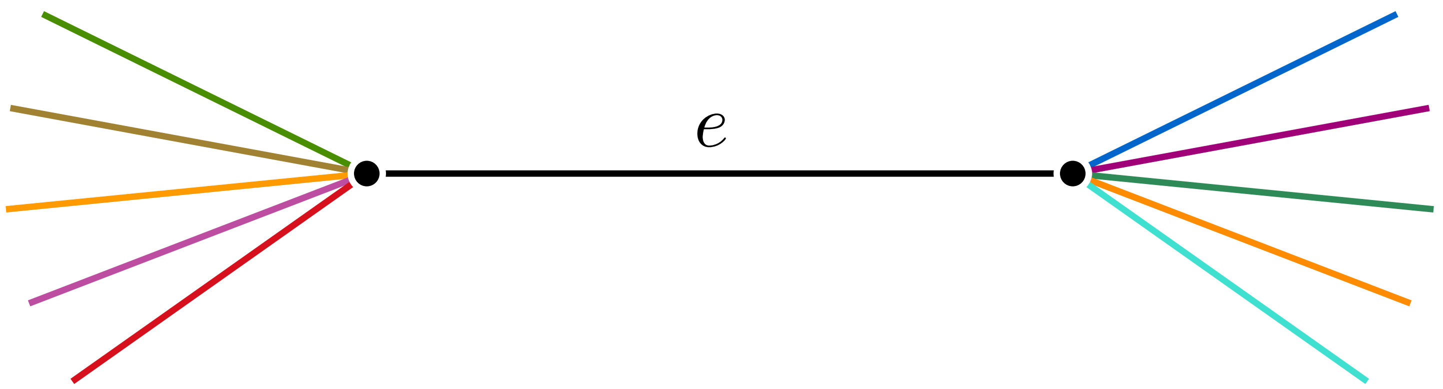
Color arriving edge e with an available color uniformly at random



Candidate Algorithm: “Randomized Greedy”

Use palette $\{1, 2, \ldots, (1+\varepsilon)\Delta\}$

Color arriving edge e with an available color uniformly at random

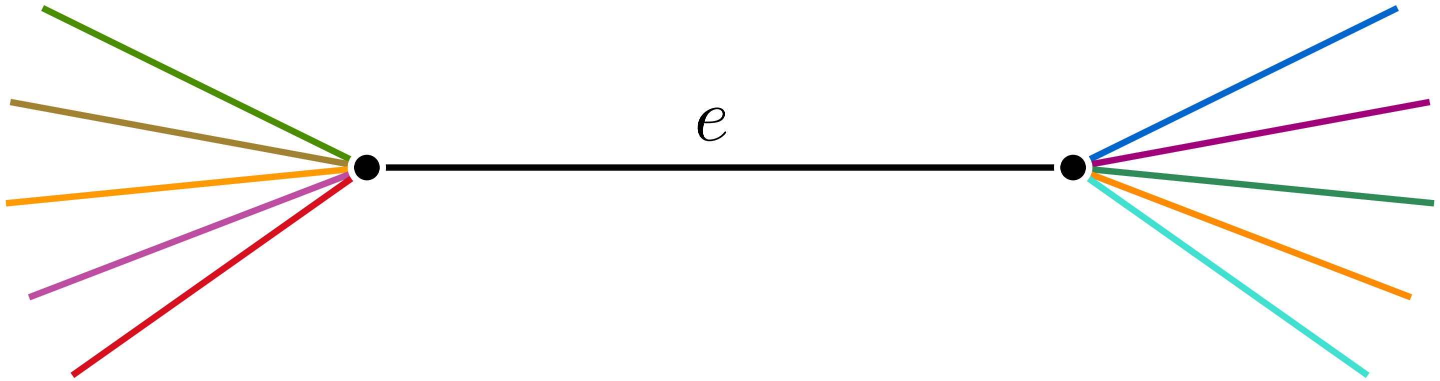


Does it work? (fail if no available color)

Candidate Algorithm: "Randomized Greedy"

Use palette $\{1, 2, \ldots, (1+\varepsilon)\Delta\}$

Color arriving edge e with an available color uniformly at random



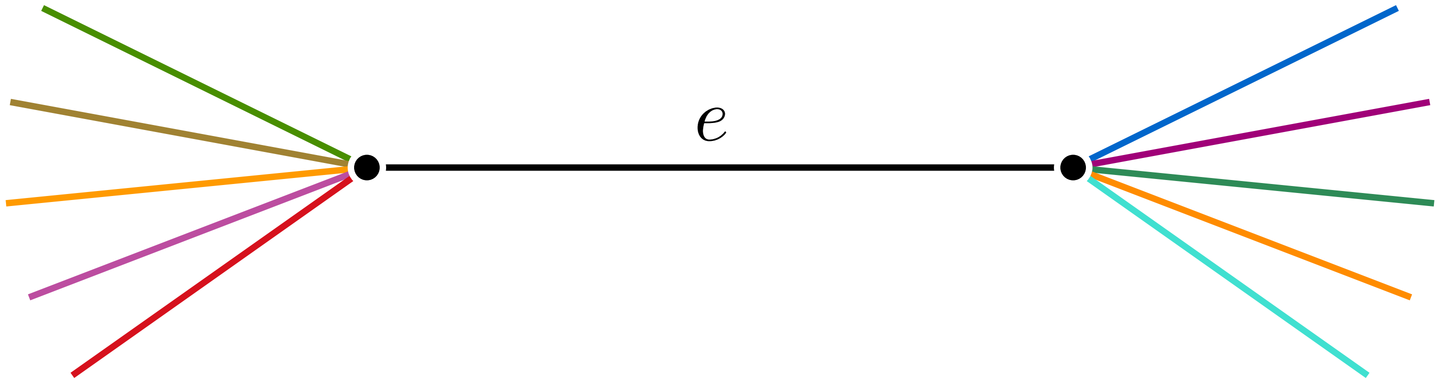
Does it work? (fail if no available color)

NO! two subtle problematic reasons:

Candidate Algorithm: "Randomized Greedy"

Use palette $\{1, 2, \ldots, (1+\varepsilon)\Delta\}$

Color arriving edge e with an available color uniformly at random



Does it work? (fail if no available color)

NO! two subtle problematic reasons:

(1) uses colors symmetrically (even fails in trees)

(2) "uniformly at random" allows adversary to amplify bias

Towards a Working Algorithm

Two subtle problematic reasons:

(1) uses colors symmetrically

Towards a Working Algorithm

Two subtle problematic reasons:

(1) uses colors symmetrically

Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

Towards a Working Algorithm

Two subtle problematic reasons:

(1) uses colors symmetrically

Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

Goal: $\Pr[\text{we assign } e \text{ color from main palette}] = 1 - \varepsilon$

Towards a Working Algorithm

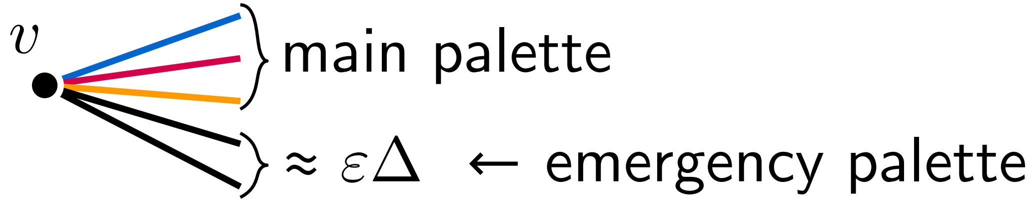
Two subtle problematic reasons:

(1) uses colors symmetrically

Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

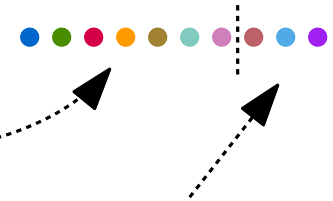
Goal: $\Pr[\text{we assign } e \text{ color from main palette}] = 1 - \varepsilon$



Towards a Working Algorithm

Two subtle problematic reasons:

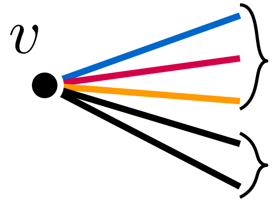
(1) uses colors symmetrically



Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

Goal: $\Pr[\text{we assign } e \text{ color from main palette}] = 1 - \varepsilon$



main palette

$\approx \varepsilon\Delta$ $\leftarrow$ emergency palette

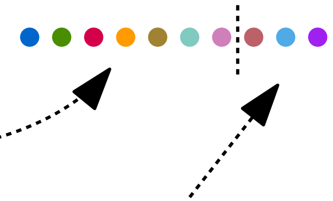
$\Delta' = \max\deg(\text{uncolored subgraph})$

Greedy: $2\Delta' \approx 2\varepsilon\Delta$

Towards a Working Algorithm

Two subtle problematic reasons:

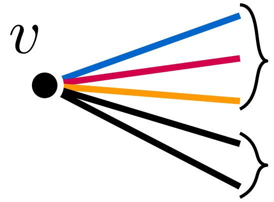
(1) uses colors symmetrically



Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

Goal: $\Pr[\text{we assign } e \text{ color from main palette}] = 1 - \varepsilon$



main palette

$\approx \varepsilon\Delta$ ← emergency palette

$\Delta' = \max\deg(\text{uncolored subgraph})$

Greedy: $2\Delta' \approx 2\varepsilon\Delta$

(2) "uniformly at random" allows adversary to amplify bias

Towards a Working Algorithm

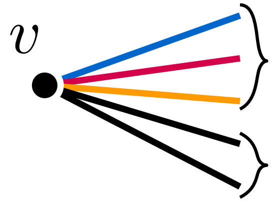
Two subtle problematic reasons:

(1) uses colors symmetrically

Main palette $\{1, 2, \ldots, \Delta\}$

Emergency palette $\{\Delta + 1, \ldots, \Delta + 2\varepsilon\Delta\}$ (only use when necessary)

Goal: $\Pr[\text{we assign } e \text{ color from main palette}] = 1 - \varepsilon$



main palette

$\approx \varepsilon\Delta$ $\leftarrow$ emergency palette

$\Delta' = \max\deg(\text{uncolored subgraph})$

Greedy: $2\Delta' \approx 2\varepsilon\Delta$

(2) "uniformly at random" allows adversary to amplify bias

Use a Bayesian "execution dependent" approach