First-Fit Coloring of Trees in Random Arrival Model

Bartłomiej Bosek, **Grzegorz Gutowski**, Michał Lasoń, Jakub Przybyło

Jagiellonian University, Polish Academy of Sciences, AGH University

金华市, 2024-05-08

arXiv: 2404.17011

Graph Coloring

Graph

Graph $G = (V, E)$ with $n = |V|$ vertices.

Graph Coloring

Graph

Graph $G = (V, E)$ with $n = |V|$ vertices.

Coloring

Function $c : V \mapsto \mathbb{Z}^+$ with $c(v) \neq c(w)$ for every $\{v, w\} \in E$.

Graph Coloring

Graph

Graph $G = (V, E)$ with $n = |V|$ vertices.

Coloring

Function $c : V \mapsto \mathbb{Z}^+$ with $c(v) \neq c(w)$ for every $\{v, w\} \in E$.

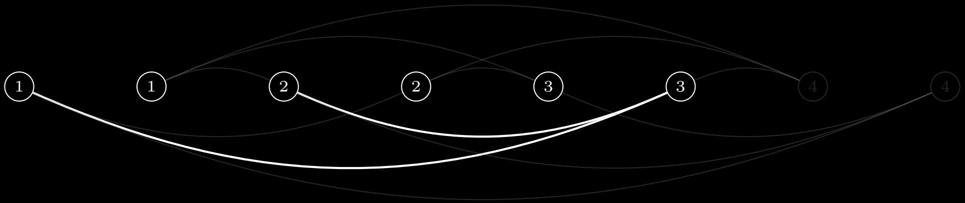
Chromatic number

$\chi(G)$ – minimum possible k with coloring $c : V \mapsto \{1, 2, \ldots, k\}$.

First-Fit Algorithm

Algorithm

1. Process vertices in order $v_1 \ll v_2 \ll \ldots \ll v_n$.
2. For vertex v_t use the first color that is not used as a color of any neighbor of v_j of v_t with $j < t$.



First-Fit Algorithm

Algorithm

1. Process vertices in order $v_1 \ll v_2 \ll \ldots \ll v_n$.
2. For vertex v_t use the first color that is not used as a color of any neighbor of v_j of v_t with $j < t$.

First-Fit chromatic number

$\chi_{\mathsf{FF}}(G, \ll)$ – maximum color used by First-Fit on G in order $\ll$.

Question

Is First-Fit a good algorithm? How much does $\chi_{FF}(G, \ll)$ differ from $\chi(G)$?

First-Fit Algorithm

Algorithm

1. Process vertices in order $v_1 \ll v_2 \ll \ldots \ll v_n$.
2. For vertex v_t use the first color that is not used as a color of any neighbor of v_j of v_t with $j < t$.

First-Fit chromatic number

$\chi_{\mathsf{FF}}(G, \ll)$ – maximum color used by First-Fit on G in order $\ll$.

Question

Is First-Fit a good algorithm? How much does $\chi_{FF}(G, \ll)$ differ from $\chi(G)$?

Answer

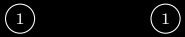
It all depends on the order $\ll$.

Optimistic Scenarios

Bipartite graph

Optimistic Scenarios

Bipartite graph

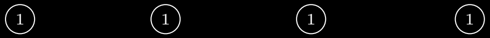


Optimistic Scenarios

Bipartite graph

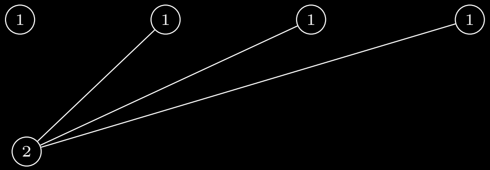
Optimistic Scenarios

Bipartite graph



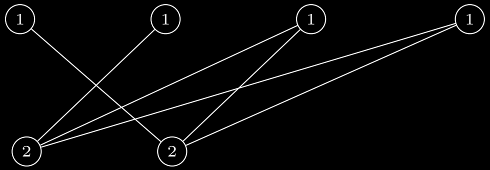
Optimistic Scenarios

Bipartite graph



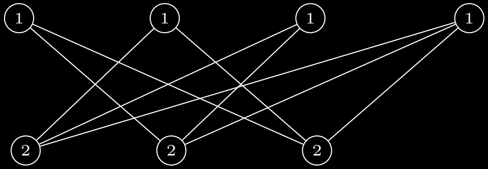
Optimistic Scenarios

Bipartite graph



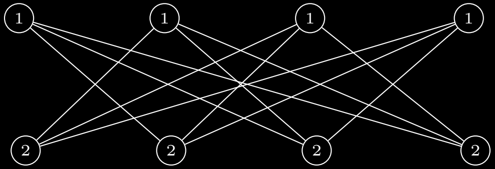
Optimistic Scenarios

Bipartite graph

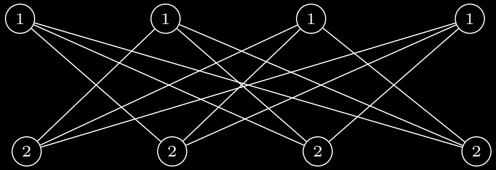


Optimistic Scenarios

Bipartite graph



Bipartite graph

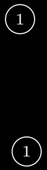


First-Fit can be optimal

1. Color G with k colors.
2. Construct $\ll$ by sorting vertices by their color.
3. First-Fit uses at most k colors in this order.

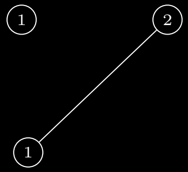
Bipartite graph

Bipartite graph



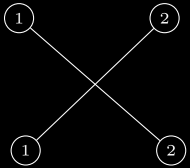
Pessimistic Scenarios

Bipartite graph



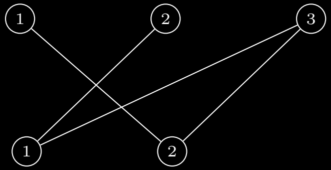
Pessimistic Scenarios

Bipartite graph



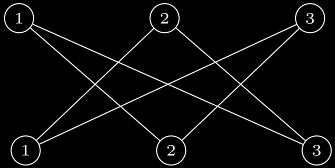
Pessimistic Scenarios

Bipartite graph



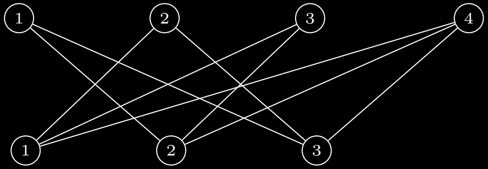
Pessimistic Scenarios

Bipartite graph



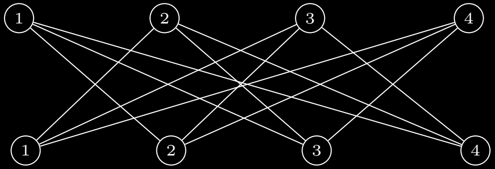
Pessimistic Scenarios

Bipartite graph



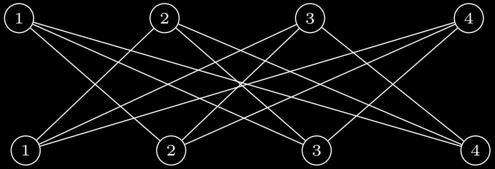
Pessimistic Scenarios

Bipartite graph



Pessimistic Scenarios

Bipartite graph



- $\chi(G) = 2$
- $\chi_{\mathsf{FF}}(G, \ll) = \frac{n}{2}$

Average Scenarios

Idea
Process vertices in random order.

Average Scenarios

Idea

Process vertices in random order.

Question

How many colors does First-Fit use on G, in expectation?

Average Scenarios

Idea

Process vertices in random order.

Question

How many colors does First-Fit use on G, in expectation?

Question

How many colors does First-Fit use on graphs in $\mathcal{G}$ with n vertices, in expectation?

Average Scenarios

Idea

Process vertices in random order.

Question

How many colors does First-Fit use on G, in expectation?

Question

How many colors does First-Fit use on graphs in $\mathcal{G}$ with n vertices, in expectation?

Question

How many colors does First-Fit use on trees with n vertices, in expectation?

Interesting Settings

Online Algorithms

First-Fit processes vertices without knowledge of the future.

Interesting Settings

Online Algorithms

First-Fit processes vertices without knowledge of the future.

Local Algorithms

First-Fit processes vertices without knowledge of the whole graph.

Interesting Settings

Online Algorithms

First-Fit processes vertices without knowledge of the future.

Local Algorithms

First-Fit processes vertices without knowledge of the whole graph.

Distributed Algorithms

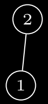
Adding a random delay in distributed coloring algorithm allows to process vertices in random order.

First-Fit on Forests in Adversarial Order

Tree

First-Fit on Forests in Adversarial Order

Tree

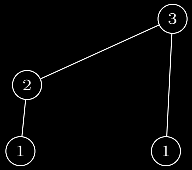


First-Fit on Forests in Adversarial Order

Tree

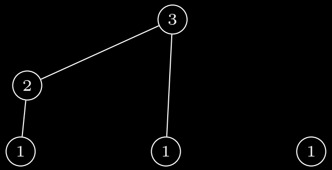
First-Fit on Forests in Adversarial Order

Tree



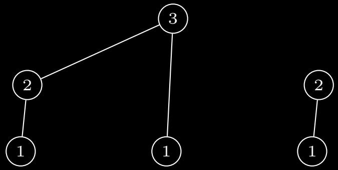
First-Fit on Forests in Adversarial Order

Tree



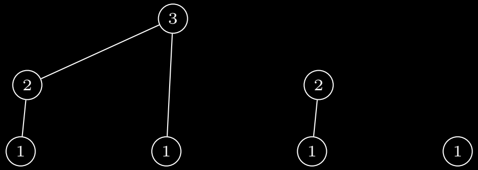
First-Fit on Forests in Adversarial Order

Tree



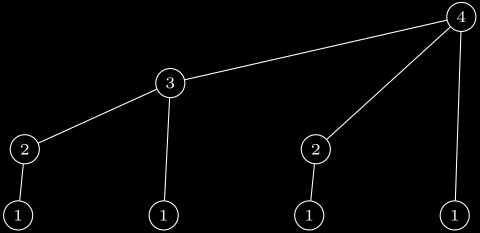
First-Fit on Forests in Adversarial Order

Tree



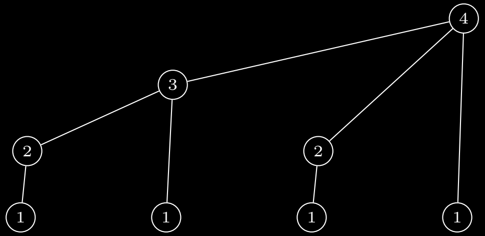
First-Fit on Forests in Adversarial Order

Tree



First-Fit on Forests in Adversarial Order

Tree



- $\chi(G) = 2$
- $\chi_{\mathsf{FF}}(G, \ll) = \log_2 n$

First-Fit on Forests in Random Order

Theorem

First-Fit uses $\left(\frac{1}{2} + o(1)\right) \cdot \frac{\ln n}{\ln \ln n}$ *colors on trees with* n *vertices.*

First-Fit on Forests in Random Order

Theorem

First-Fit uses $\left(\frac{1}{2} + o(1)\right) \cdot \frac{\ln n}{\ln \ln n}$ *colors on trees with* n *vertices.*

Meaning

$\max_{G \text{ – tree with } n \text{ vertices}} \mathbb{E}_{\ll \text{ – random order of } V}\left[\chi_{\mathrm{FF}}(G, \ll)\right] \approx k$ for $n \approx k!$.

Upper bound

Proof (of a slightly weaker statement)

- $G^{\ll}$ is a directed graph with each edge in G directed towards the vertex later in $\ll$.

Upper bound

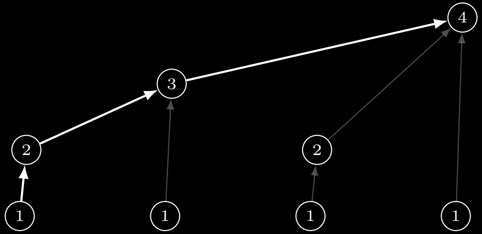
Proof (of a slightly weaker statement)

- $G^{\ll}$ is a directed graph with each edge in G directed towards the vertex later in $\ll$.
- First-Fit uses color $k \implies G^{\ll}$ contains a directed k-path.

Upper bound

Proof (of a slightly weaker statement)

- $G^{\ll}$ is a directed graph with each edge in G directed towards the vertex later in $\ll$.
- First-Fit uses color $k \implies G^{\ll}$ contains a directed k-path.



Upper bound

Proof (of a slightly weaker statement)

- $G^{\ll}$ is a directed graph with each edge in G directed towards the vertex later in $\ll$.
- First-Fit uses color $k \implies G^{\ll}$ contains a directed k-path.
- There are only n^2 different k-paths in the tree G.

Upper bound

Proof (of a slightly weaker statement)

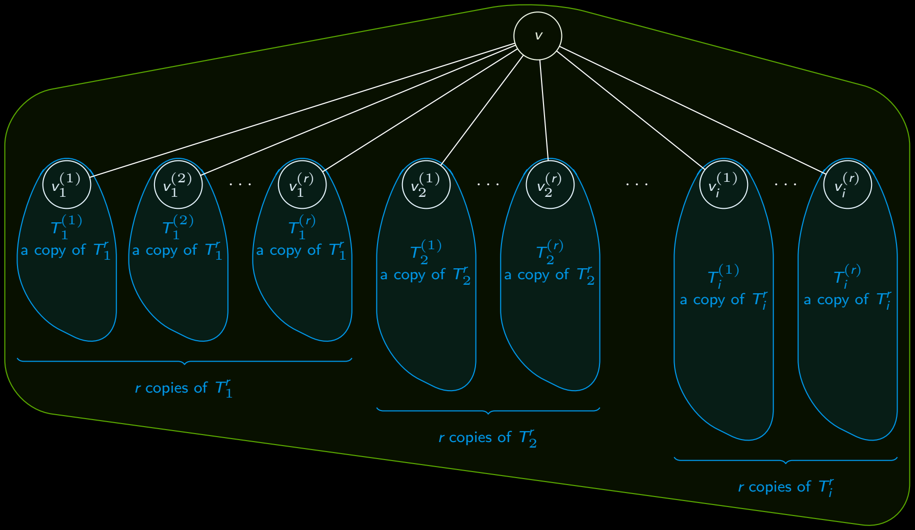
- $G^{\ll}$ is a directed graph with each edge in G directed towards the vertex later in $\ll$.
- First-Fit uses color $k \implies G^{\ll}$ contains a directed k-path.
- There are only n^2 different k-paths in the tree G.
- Probability that a fixed one is directed in $G^{\ll}$ is $\approx \frac{1}{k!}$.

Upper bound

Proof (of a slightly weaker statement)

- $G^{\ll}$ is a directed graph with each edge in G directed towards the vertex later in $\ll$.
- First-Fit uses color $k \implies G^{\ll}$ contains a directed k-path.
- There are only n^2 different k-paths in the tree G.
- Probability that a fixed one is directed in $G^{\ll}$ is $\approx \frac{1}{k!}$.
- For a constant probability of getting color k, we need $n^2 \approx k!$.

Lower bound



v
$v_1^{(1)}$
$T_1^{(1)}$
a copy of T_1^r
$v_1^{(2)}$
$T_1^{(2)}$
a copy of T_1^r
...
$v_1^{(r)}$
$T_1^{(r)}$
a copy of T_1^r
$v_2^{(1)}$
$T_2^{(1)}$
a copy of T_2^r
...
$v_2^{(r)}$
$T_2^{(r)}$
a copy of T_2^r
...
$v_i^{(1)}$
$T_i^{(1)}$
a copy of T_i^r
...
$v_i^{(r)}$
$T_i^{(r)}$
a copy of T_i^r
r copies of T_1^r
r copies of T_2^r
r copies of T_i^r

Lower bound

Proof (of a slightly weaker statement)

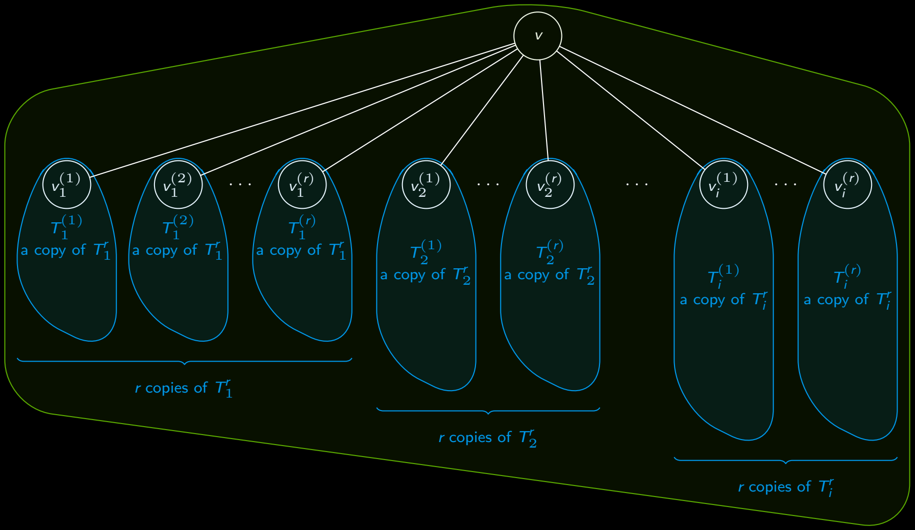
- Fix k – target number of colors.
- Fix $\gamma > 0$ – allowed error.
- Let $r \approx k \ln k / \gamma^2$.

Lower bound

Proof (of a slightly weaker statement)

- Fix k – target number of colors.
- Fix $\gamma > 0$ – allowed error.
- Let $r \approx k \ln k / \gamma^2$.
- T_1^r is a single vertex, T_{i+1}^r has r copies of each $T_1^r, \ldots, T_i^r$.

Lower bound



v
$v_1^{(1)}$
$T_1^{(1)}$
a copy of T_1^r
$v_1^{(2)}$
$T_1^{(2)}$
a copy of T_1^r
$v_1^{(r)}$
$T_1^{(r)}$
a copy of T_1^r
r copies of T_1^r
$v_2^{(1)}$
$T_2^{(1)}$
a copy of T_2^r
$v_2^{(r)}$
$T_2^{(r)}$
a copy of T_2^r
r copies of T_2^r
$v_i^{(1)}$
$T_i^{(1)}$
a copy of T_i^r
$v_i^{(r)}$
$T_i^{(r)}$
a copy of T_i^r
r copies of T_i^r

Lower bound

Proof (of a slightly weaker statement)

- $\mathbb{P}_{\ll}(\chi_{\mathsf{FF}}(T_k^r, \ll) \geqslant k) > 1 - \gamma.$

Lower bound

Proof (of a slightly weaker statement)

- $\mathbb{P}_{\ll}(\chi_{\mathsf{FF}}(T^r_k, \ll) \geqslant k) > 1 - \gamma$.
- Instead of using a random order, use a random position in $[0, 1]$ for each vertex.

Lower bound

Proof (of a slightly weaker statement)

- $\mathbb{P}_{\ll}(\chi_{\mathsf{FF}}(T_k^r, \ll) \geqslant k) > 1 - \gamma$.
- Instead of using a random order, use a random position in $[0, 1]$ for each vertex.
- Inductively prove that root of T_i^r has $1 - \varepsilon_i$ probability of getting color i. $\varepsilon_i = \frac{i\gamma}{k}$.

Lower bound

Proof (of a slightly weaker statement)

- $\mathbb{P}_{\ll}(\chi_{\mathsf{FF}}(T_k^r, \ll) \geqslant k) > 1 - \gamma$.
- Instead of using a random order, use a random position in $[0, 1]$ for each vertex.
- Inductively prove that root of T_i^r has $1 - \varepsilon_i$ probability of getting color i. $\varepsilon_i = \frac{i\gamma}{k}$.
- For a fixed position x of the root, the probability that a child root of T_j^r does not provide color j is bounded by $1 - x + \varepsilon_j$.

Lower bound

Proof (of a slightly weaker statement)

- $\mathbb{P}_{\ll}(\chi_{\mathsf{FF}}(T^r_k, \ll) \geqslant k) > 1 - \gamma$.
- Instead of using a random order, use a random position in $[0,1]$ for each vertex.
- Inductively prove that root of T^r_i has $1 - \varepsilon_i$ probability of getting color i. $\varepsilon_i = \frac{i\gamma}{k}$.
- For a fixed position x of the root, the probability that a child root of T^r_j does not provide color j is bounded by $1 - x + \varepsilon_j$.

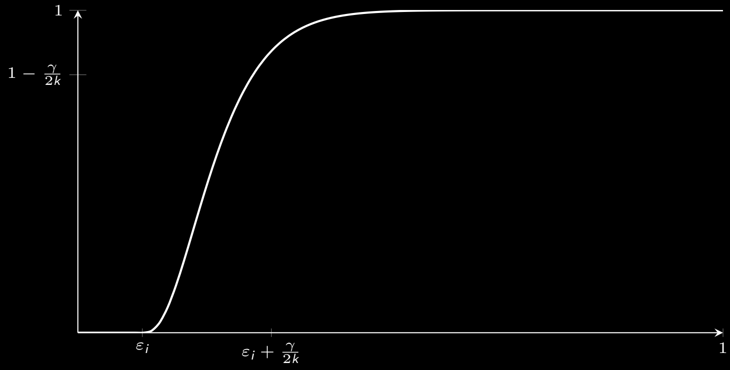
$$\int_{\varepsilon_i}^{1} \left(1 - (1 - x + \varepsilon_i)^r\right)^i \partial x > 1 - \varepsilon_{i+1}$$

Integral

$$\int_{\varepsilon_i}^{1} \left(1 - (1 - x + \varepsilon_i)^r\right)^i \partial x > 1 - \varepsilon_{i+1}$$

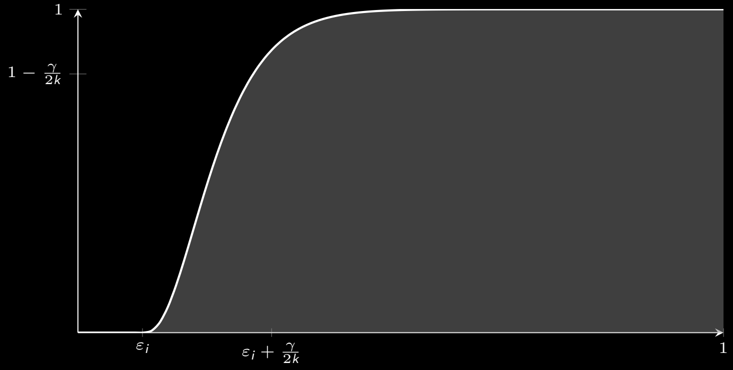
Integral

$$\int_{\varepsilon_i}^{1} \left(1-(1-x+\varepsilon_i)^r\right)^i \partial x > 1-\varepsilon_{i+1}$$



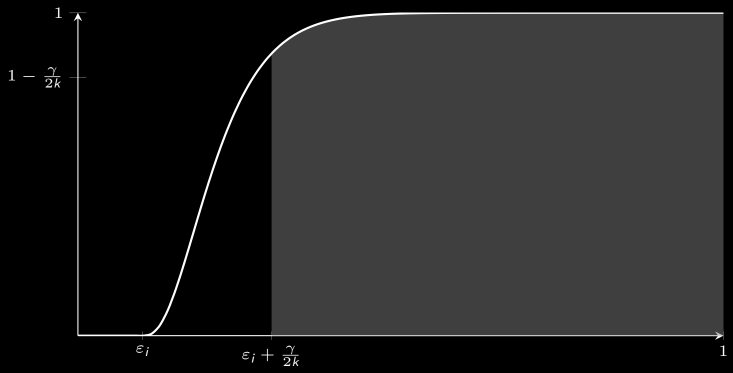
Integral

$$\int_{\varepsilon_i}^{1} \left(1 - (1 - x + \varepsilon_i)^r\right)^i \partial x > 1 - \varepsilon_{i+1}$$



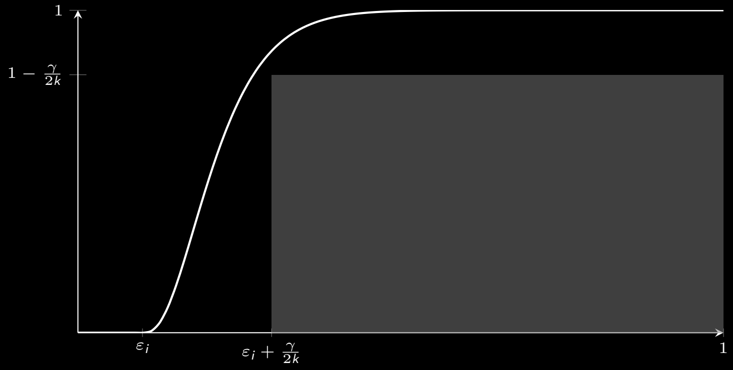
Integral

$$\int_{\varepsilon_i}^{1} \left(1-(1-x+\varepsilon_i)^r\right)^i \partial x > 1-\varepsilon_{i+1}$$



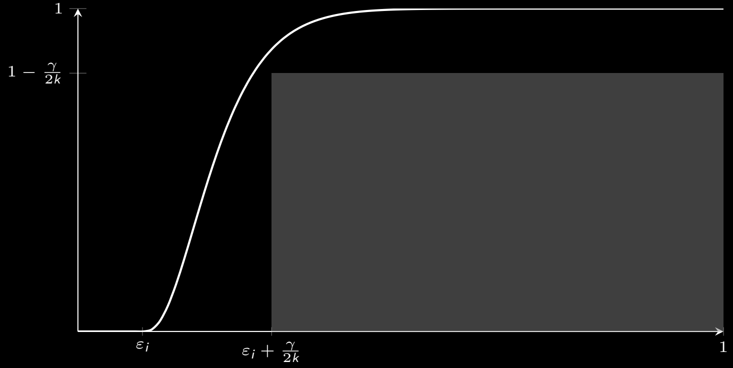
Integral

$$\int_{\varepsilon_i}^{1} \left(1-(1-x+\varepsilon_i)^r\right)^i \partial x > 1-\varepsilon_{i+1}$$



Integral

$$\left(1-\left(1-\frac{\gamma}{2k}\right)^r\right)^i > 1-\frac{\gamma}{2k}$$



Lower bound

Proof (of a slightly weaker statement)

- $\mathbb{P}_{\ll}(\chi_{\mathsf{FF}}(T^r_k, \ll) \geqslant k) > 1 - \gamma$.
- Instead of using a random order, use a random position in $[0, 1]$ for each vertex.
- Inductively prove that root of T^r_i has $1 - \varepsilon_i$ probability of getting color i. $\varepsilon_i = \frac{i\gamma}{k}$.
- For a fixed position x of the root, the probability that a child root of T^r_j does not provide color j is bounded by $1 - x + \varepsilon_j$.
- $\int_{\varepsilon_i}^{1} (1 - (1 - x + \varepsilon_i)^r)^i \partial x > 1 - \varepsilon_{i+1}$

Lower bound

Proof (of a slightly weaker statement)

- $\mathbb{P}_{\ll}(\chi_{\mathsf{FF}}(T^r_k, \ll) \geqslant k) > 1 - \gamma$.
- Instead of using a random order, use a random position in $[0, 1]$ for each vertex.
- Inductively prove that root of T^r_i has $1 - \varepsilon_i$ probability of getting color i. $\varepsilon_i = \frac{i\gamma}{k}$.
- For a fixed position x of the root, the probability that a child root of T^r_j does not provide color j is bounded by $1 - x + \varepsilon_j$.
- $\int_{\varepsilon_i}^{1}(1 - (1 - x + \varepsilon_i)^r)^i \partial x > 1 - \varepsilon_{i+1}$
- T^r_k has $\approx k^k$ vertices.

Summary

Theorem

First-Fit uses $\left(\frac{1}{2}+o(1)\right)\cdot\frac{\ln n}{\ln\ln n}$ *colors on trees with* n *vertices.*

Summary

Theorem

First-Fit uses $\left(\frac{1}{2} + o(1)\right) \cdot \frac{\ln n}{\ln \ln n}$ *colors on trees with* n *vertices.*

Question

First-Fit on bipartite graphs

Thank You!