The Complexity of Tree Multicolorings

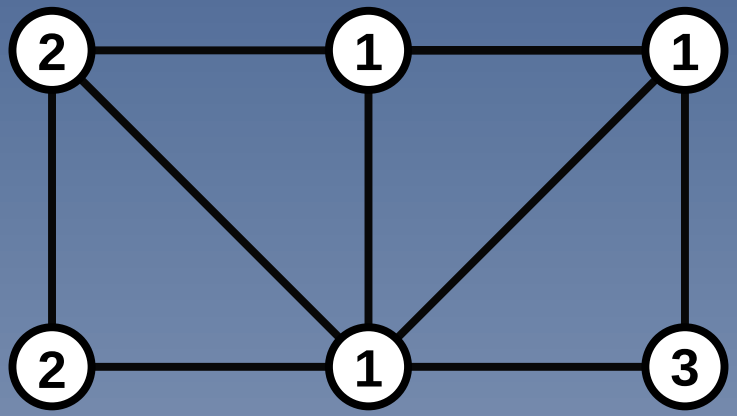
Dániel Marx
Budapest University of Technology and Economics
dmarx@cs.bme.hu

Minimum sum multicoloring

- **Given:** a graph $G(V, E)$, and demand function $x : V \to \mathbb{N}$
- **Find:** an assignment Ψ of $x(v)$ colors (integers) to every vertex v, such that neighbors receive disjoint sets
- **Goal:** The *finish time* $f(v)$ of vertex v is the largest color (integer) assigned to it in the coloring. Minimize $\sum_{v \in V} f(v)$, the *sum of the coloring*.

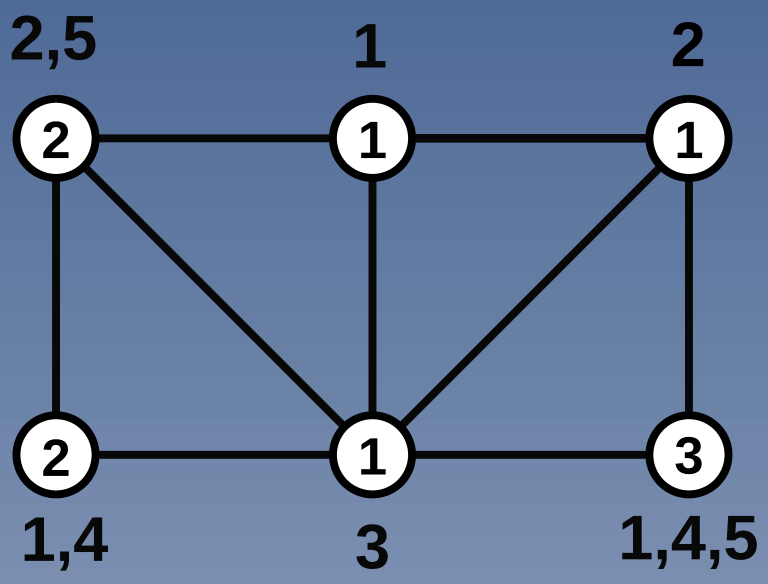
Minimum sum multicoloring

- **Given:** a graph $G(V, E)$, and demand function $x : V \to \mathbb{N}$
- **Find:** an assignment Ψ of $x(v)$ colors (integers) to every vertex v, such that neighbors receive disjoint sets
- **Goal:** The *finish time* $f(v)$ of vertex v is the largest color (integer) assigned to it in the coloring. Minimize $\sum_{v \in V} f(v)$, the *sum of the coloring*.



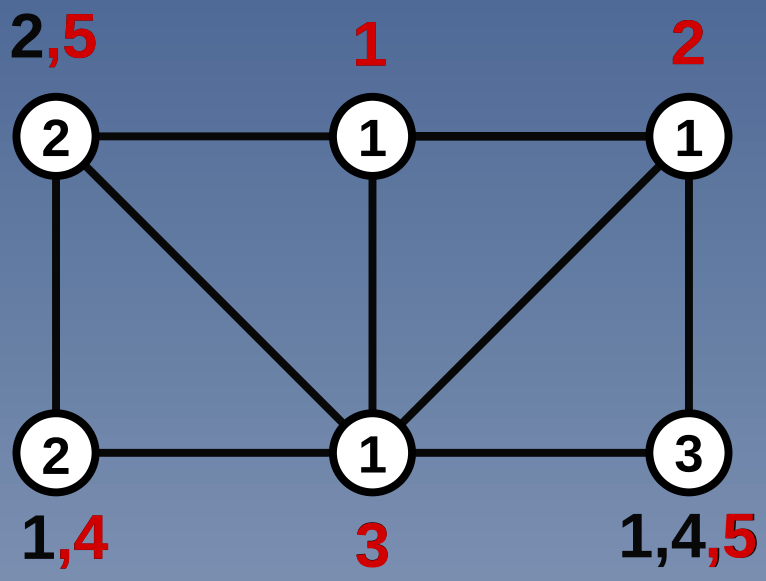
Minimum sum multicoloring

- **Given:** a graph $G(V, E)$, and demand function $x : V \to \mathbb{N}$
- **Find:** an assignment Ψ of $x(v)$ colors (integers) to every vertex v, such that neighbors receive disjoint sets
- **Goal:** The *finish time* $f(v)$ of vertex v is the largest color (integer) assigned to it in the coloring. Minimize $\sum_{v \in V} f(v)$, the *sum of the coloring*.



Minimum sum multicoloring

- **Given:** a graph $G(V, E)$, and demand function $x : V \to \mathbb{N}$
- **Find:** an assignment Ψ of $x(v)$ colors (integers) to every vertex v, such that neighbors receive disjoint sets
- **Goal:** The *finish time* $f(v)$ of vertex v is the largest color (integer) assigned to it in the coloring. Minimize $\sum_{v \in V} f(v)$, the *sum of the coloring*.



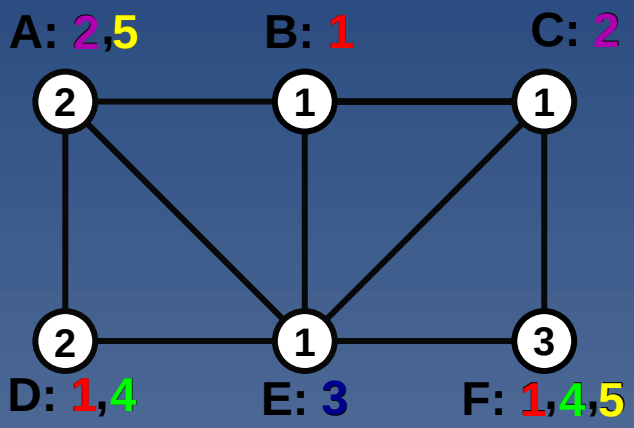
Sum of the coloring:
5 + 1 + 2 + 4 + 3 + 5 = 20

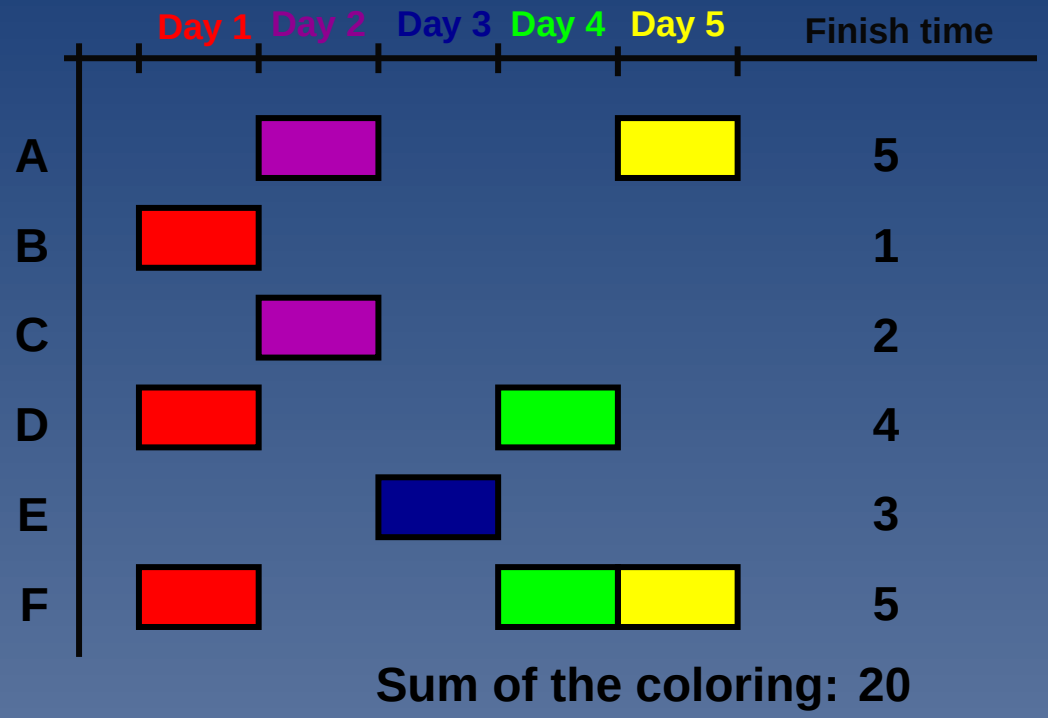
Application in scheduling

Scheduling of interfering jobs, minimizing the sum of completion times (same as minimizing the average completion times)

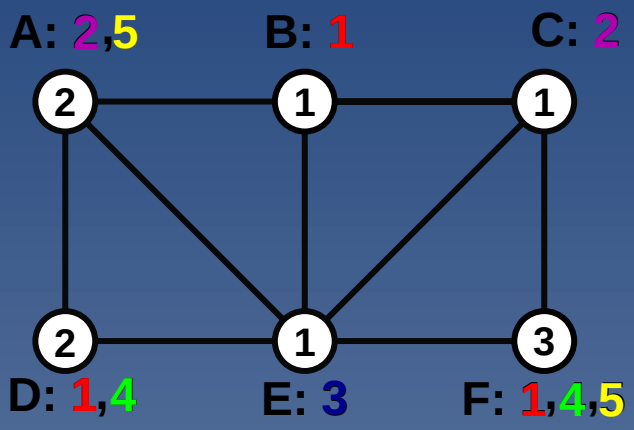
vertices	$\Longleftrightarrow$	jobs
demands	$\Longleftrightarrow$	days required
edges	$\Longleftrightarrow$	interfering pairs of jobs
colors	$\Longleftrightarrow$	days
assignment of colors	$\Longleftrightarrow$	assignment of days
finish time of a vertex	$\Longleftrightarrow$	day when the job is finished
sum of the coloring	$\Longleftrightarrow$	sum of the completion times

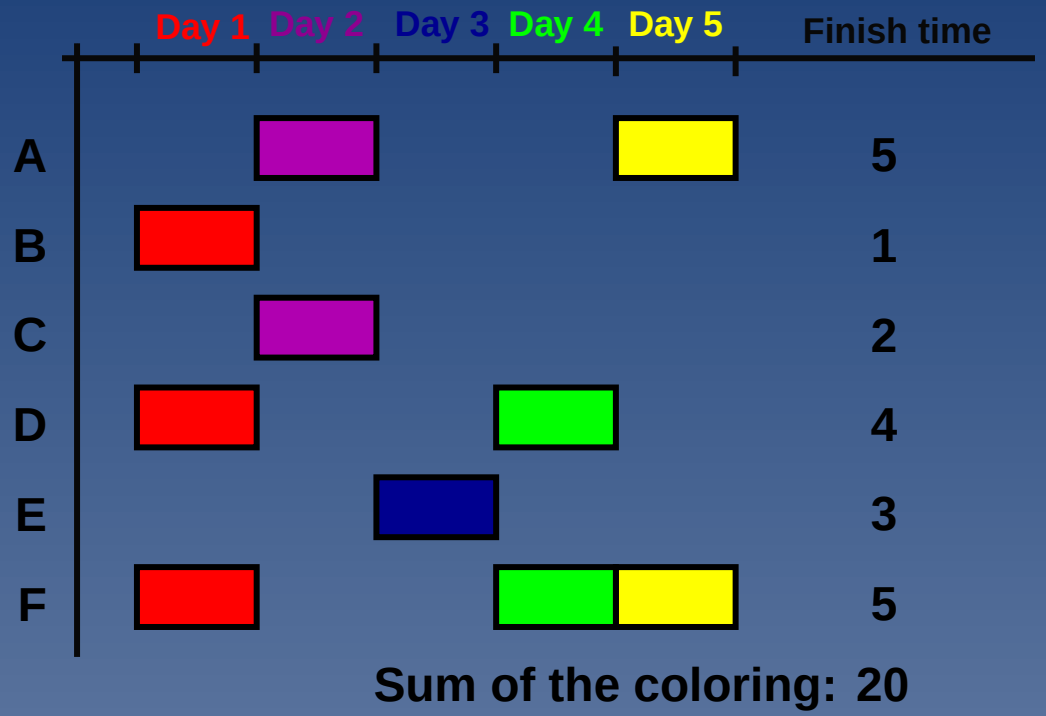
Example


A: 2,5
B: 1
C: 2
2
1
1
2
1
3
D: 1,4
E: 3
F: 1,4,5


Day 1
Day 2
Day 3
Day 4
Day 5
Finish time
A
B
C
D
E
F
5
1
2
4
3
5
Sum of the coloring: 20

Example





Preemptive scheduling: the jobs can be interrupted

Known results

Special case, the chromatic sum problem: $x(v) = 1,\ \forall v \in V$

- **General graphs:**
  - ⋆ cannot be approximated within $n^{1-\epsilon}$ even if every demand is 1 (unless **NP** $=$ **ZPP**) [Bar-Noy et al., 1998],
  - ⋆ $O(n/log^2 n)$ approximation for sum multicoloring [Bar-Noy et al., 2000]
- **Bipartite graphs:**
  - ⋆ 1.5-approximation for sum multicoloring [Bar-Noy and Kortsarz, 1998]
  - ⋆ **APX**-hard, even if every demand is 1

Known results

- **Planar graphs:**
  - $(1+\epsilon)$-approximation for sum multicoloring [Halldórsson and Kortsarz, 1999]
  - **NP**-complete even if every demand is 1
- **Trees:**
  - $(1+\epsilon)$-approximation for sum multicoloring [Halldórsson et al., 1999]
  - polynomial time solvable if every demand is 1 [Kubicka, 1989],

Known results

- **Planar graphs:**
  - $(1+\epsilon)$-approximation for sum multicoloring [Halldórsson and Kortsarz, 1999]
  - **NP**-complete even if every demand is 1
- **Trees:**
  - $(1+\epsilon)$-approximation for sum multicoloring [Halldórsson et al., 1999]
  - polynomial time solvable if every demand is 1 [Kubicka, 1989],

New result: Minimum sum multicoloring is **NP**-hard on binary trees, even if every demand is polynomially bounded (in the size of the tree)

List multicoloring

As a first step of the proof, we introduce another problem where trees are difficult to color:

List multicoloring

- **Given:** a graph $G(V, E)$, a demand function $x : V \to \mathbb{N}$, and a set of available colors $L(v)$ for every vertex
- **Find:** an assignment Ψ of $x(v)$ colors to every vertex v, such that
  - ⋆ neighbors receive disjoint sets and
  - ⋆ $\Psi(v) \subseteq L(v)$

List multicoloring

As a first step of the proof, we introduce another problem where trees are difficult to color:

List multicoloring

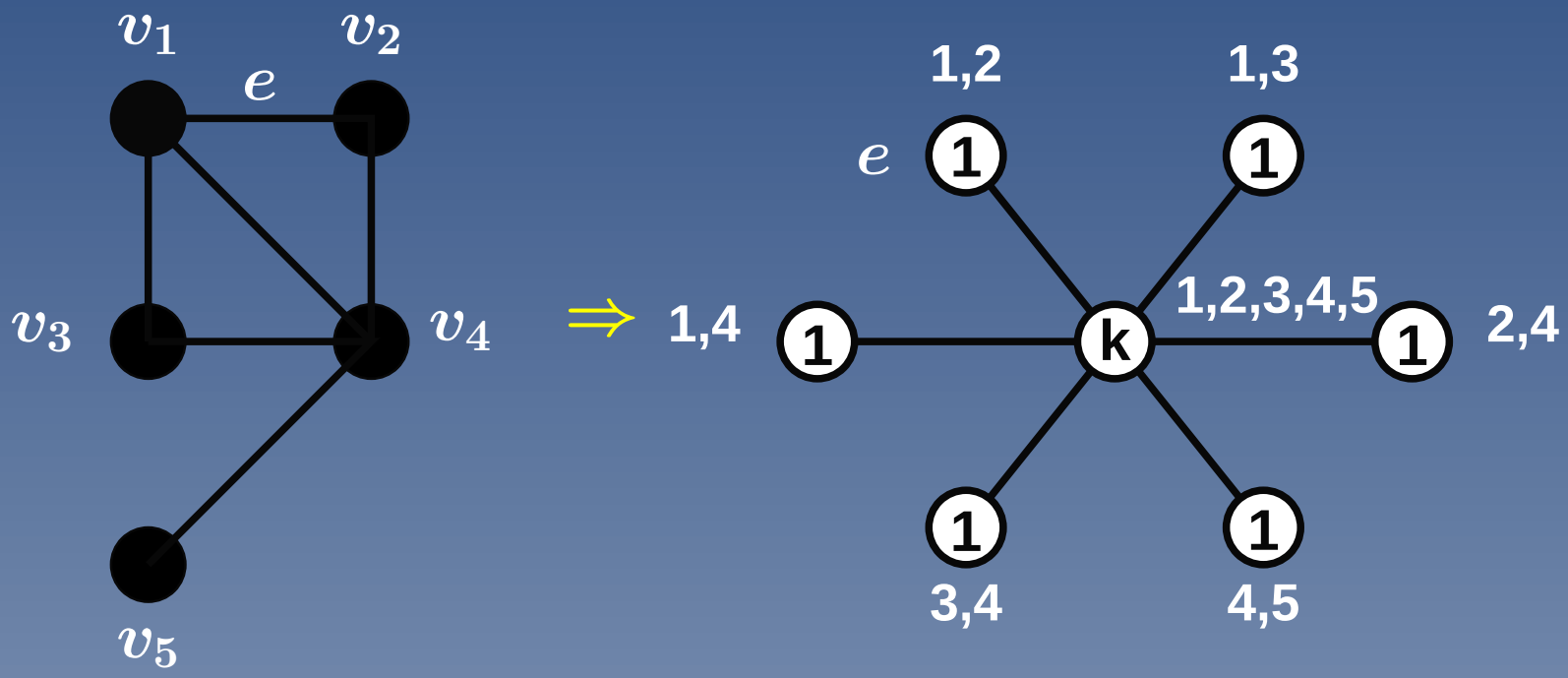
- **Given:** a graph $G(V, E)$, a demand function $x : V \to \mathbb{N}$, and a set of available colors $L(v)$ for every vertex
- **Find:** an assignment Ψ of $x(v)$ colors to every vertex v, such that
  - ⋆ neighbors receive disjoint sets and
  - ⋆ $\Psi(v) \subseteq L(v)$

New result: List multicoloring is **NP**-complete in binary trees.

Theorem: List multicoloring is **NP**-complete in trees.

(Sketch of proof) Reduction from the maximum independent set problem ("Is there an independent set of size k?")

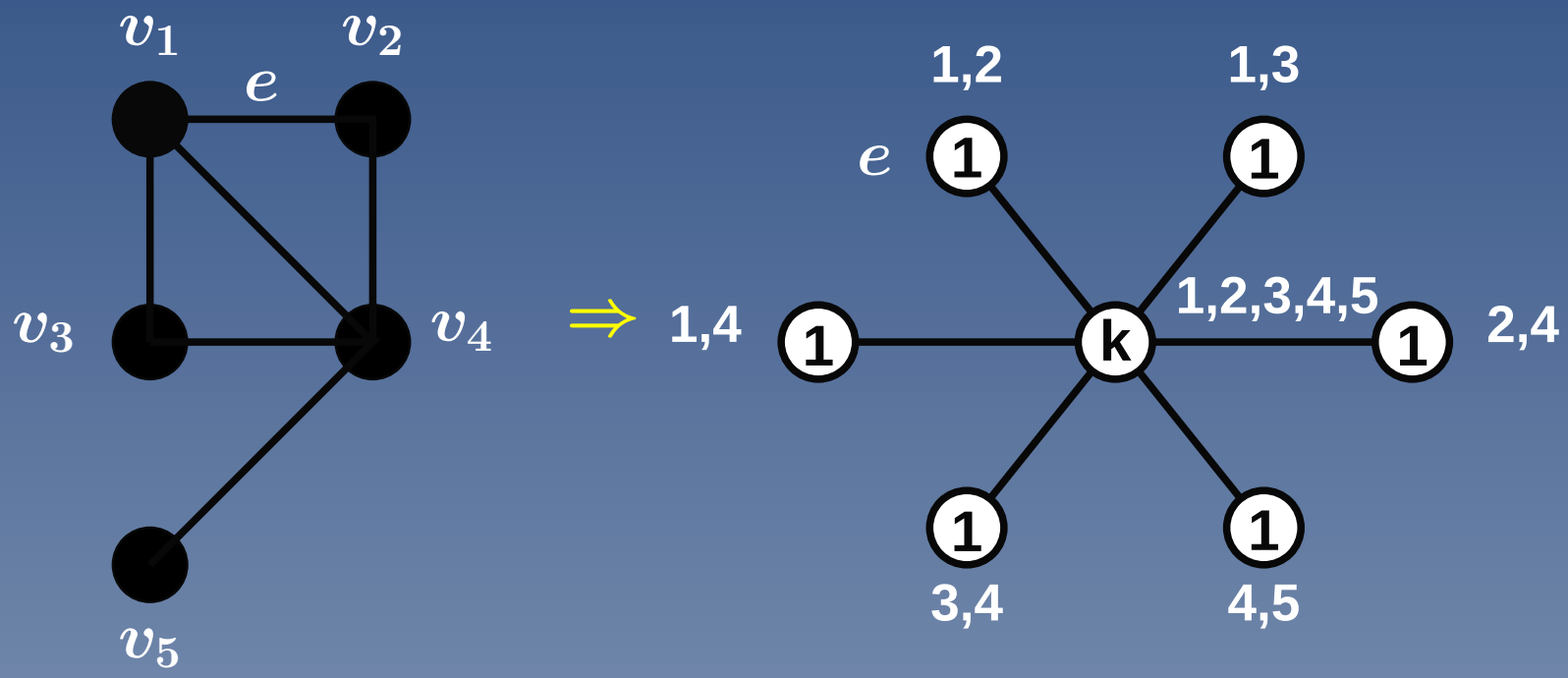
The tree is a star with one leaf for each edge.
For every edge $v_x v_y$, let $\{x, y\}$ be the list of the corresponding leaf.
The list of the central node v contains every color.



Theorem: List multicoloring is **NP**-complete in trees.

(Sketch of proof) Reduction from the maximum independent set problem ("Is there an independent set of size k?")

The tree is a star with one leaf for each edge.
For every edge $v_x v_y$, let $\{x, y\}$ be the list of the corresponding leaf.
The list of the central node v contains every color.

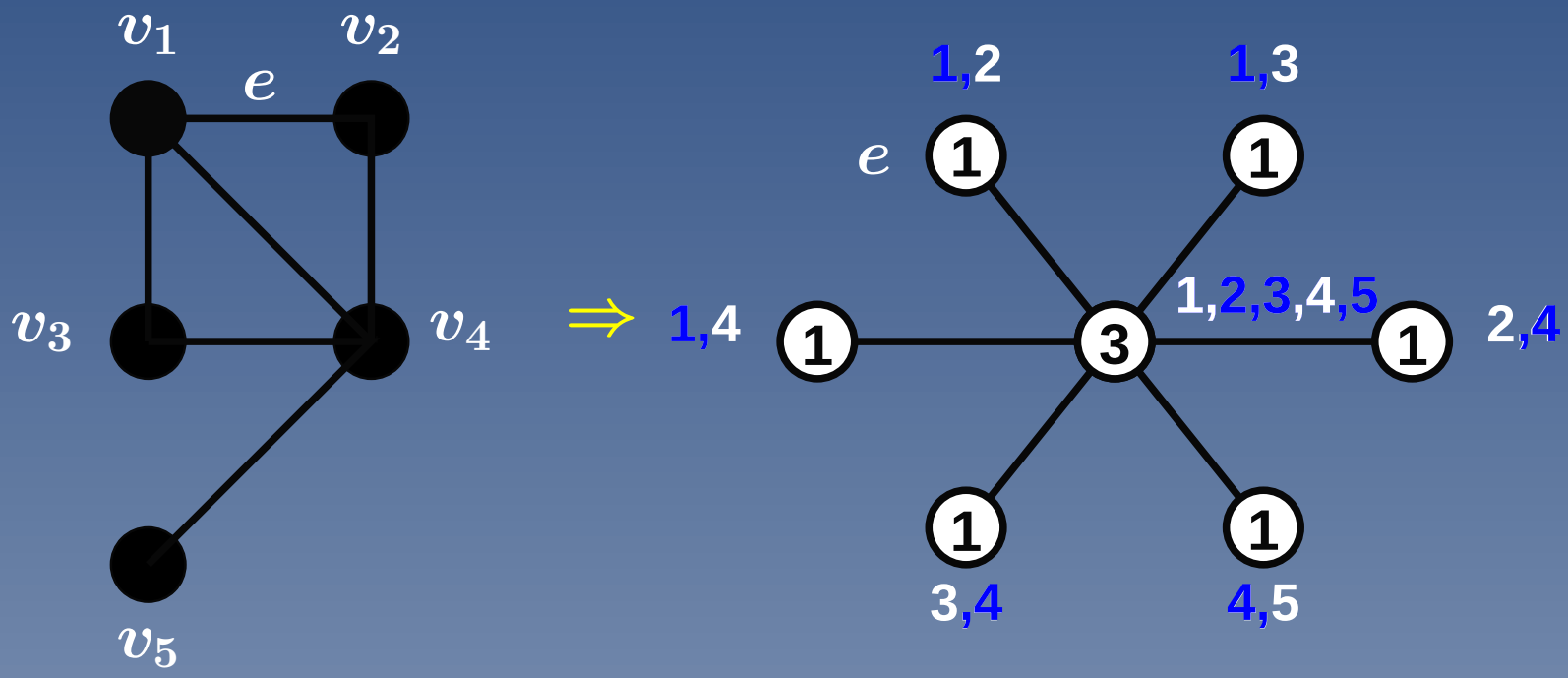


Claim: In every list coloring of the star, the colors assigned to the central node form an independent set.

Theorem: List multicoloring is **NP**-complete in trees.

(Sketch of proof) Reduction from the maximum independent set problem ("Is there an independent set of size k?")

The tree is a star with one leaf for each edge.
For every edge $v_x v_y$, let $\{x, y\}$ be the list of the corresponding leaf.
The list of the central node v contains every color.

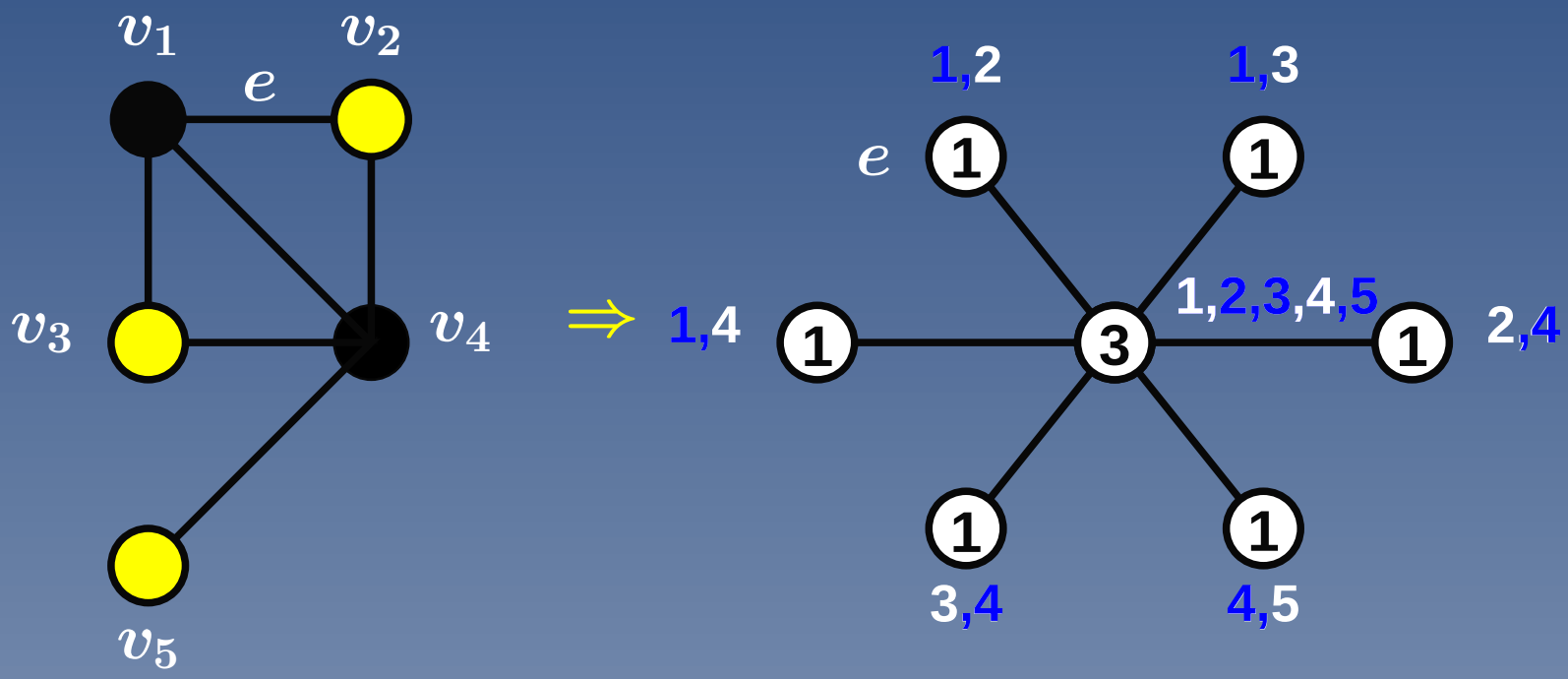


Claim: In every list coloring of the star, the colors assigned to the central node form an independent set.

Theorem: List multicoloring is **NP**-complete in trees.

(Sketch of proof) Reduction from the maximum independent set problem ("Is there an independent set of size k?")

The tree is a star with one leaf for each edge.
For every edge $v_x v_y$, let $\{x, y\}$ be the list of the corresponding leaf.
The list of the central node v contains every color.



Claim: In every list coloring of the star, the colors assigned to the central node form an independent set.

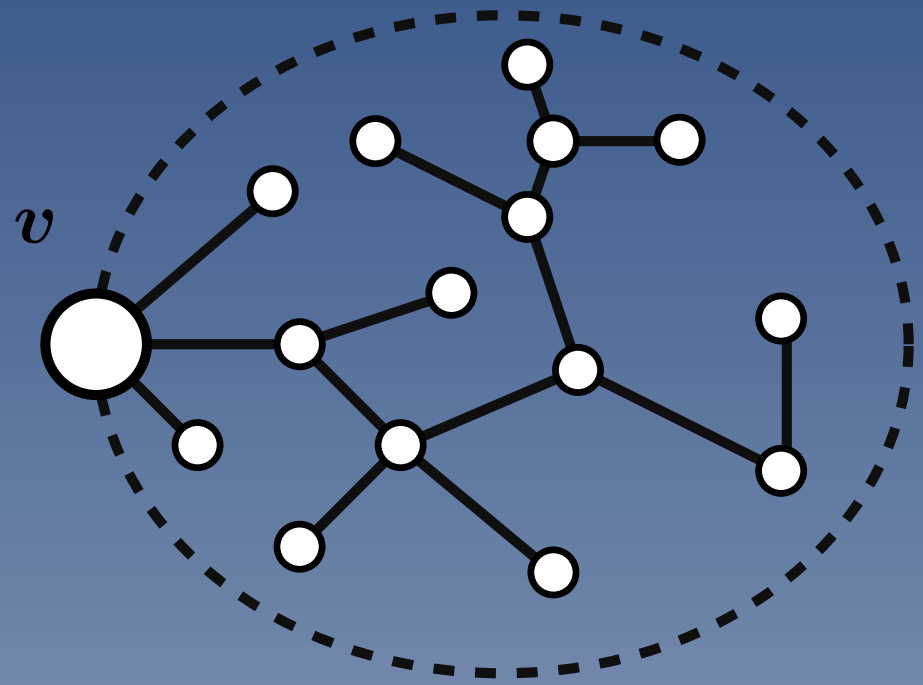
Returning to minimum sum multicoloring. (There are no lists, the goal is to minimize $\sum_{v \in V} f(v)$, where $f(v)$ is the largest color assigned to v.)

The **NP**-hardness of minimum sum coloring in trees is proved by a similar reduction. The lists are simulated by **“penalty gadgets”**.

Returning to minimum sum multicoloring. (There are no lists, the goal is to minimize $\sum_{v \in V} f(v)$, where $f(v)$ is the largest color assigned to v.)

The **NP**-hardness of minimum sum coloring in trees is proved by a similar reduction. The lists are simulated by **"penalty gadgets"**.

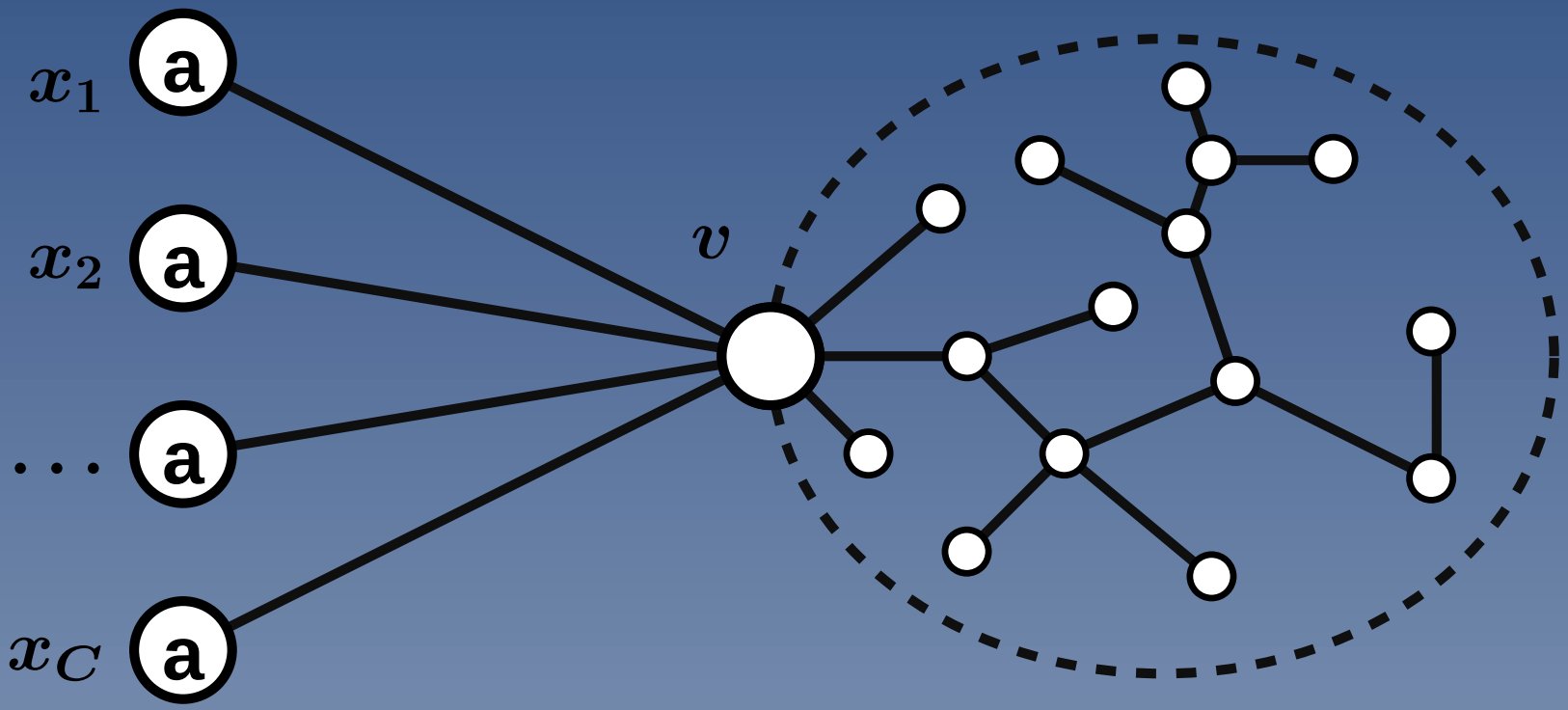
Illustration: forcing vertex v to use only colors greater than a

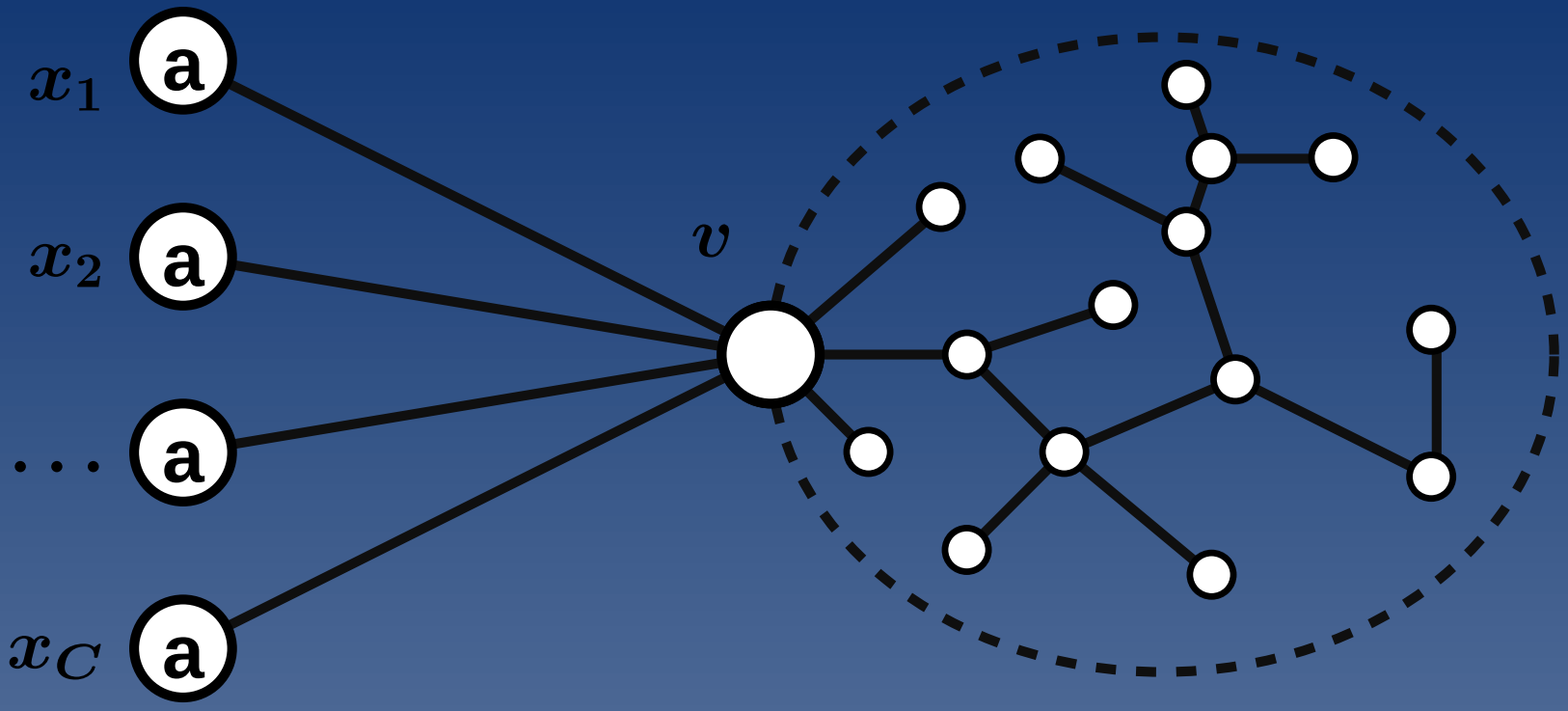


Returning to minimum sum multicoloring. (There are no lists, the goal is to minimize $\sum_{v \in V} f(v)$, where $f(v)$ is the largest color assigned to v.)

The **NP**-hardness of minimum sum coloring in trees is proved by a similar reduction. The lists are simulated by **"penalty gadgets"**.

Illustration: forcing vertex v to use only colors greater than a





Every vertex x_i has demand $a \Rightarrow$ the sum of vertices x_i is at least aC. If C is "very large", then this forces v to have only colors greater than a:

- If v has only colors greater than $a \Rightarrow$ every vertex x_i can receive colors $\{1, \dots, a\} \Rightarrow$ their total sum is aC.
- If v has a color $\leq a \Rightarrow$ every x_i has a color greater than $a \Rightarrow$ their total sum is at least $aC + C$.

Remaining steps

- A similar gadget can force v to have only colors less than b
- Using these two gadgets, we can force v to have colors from a given set
  $\Rightarrow$ we can prove that minimum sum multicoloring is **NP**-complete in trees
- With a more complicated construction, we can make penalty gadgets with maximum degree 3
  $\Rightarrow$ we can prove that minimum sum multicoloring is **NP**-complete in **binary** trees

Summary

- **Coloring problem:** Minimum sum multicoloring (minimize the sum of the finish times)

- **Previous positive result:** Minimum sum multicoloring is polynomial in trees if every demand is 1 (or bounded by a constant)

  More general result: if every demand is at most p, then the problem can be solved in $O(n \cdot (p \log n)^p)$ time $\Rightarrow$ polynomial time, if every demand is $O(\log n / \log \log n)$

- **Previous positive result:** $(1+\epsilon)$-approximation algorithm for minimum sum multicoloring in trees.

- **New negative result:** Minimum sum multicoloring is **NP**-complete in binary trees.

- List multicoloring is **NP**-complete in binary trees.