The (Parameterized) complexity of Equitable Coloring

Guilherme C. M. Gomes
Matheus R. Guedes
Vinicius F. dos Santos
Carlos Vinicius G. C. Lima

Universidade Federal de Minas Gerais

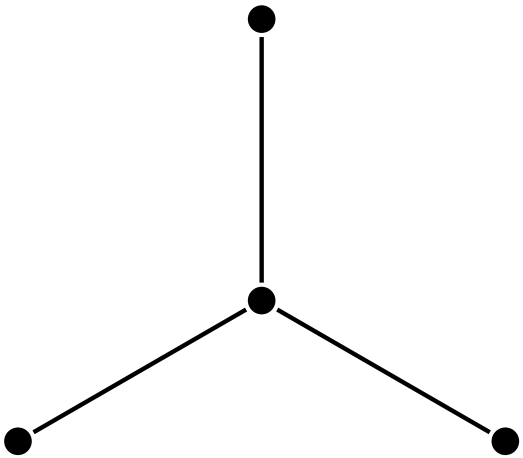
Equitable Coloring

Equitable (Vertex) Coloring

Can we k-color G such that the size of two color classes differ by ≤ 1?

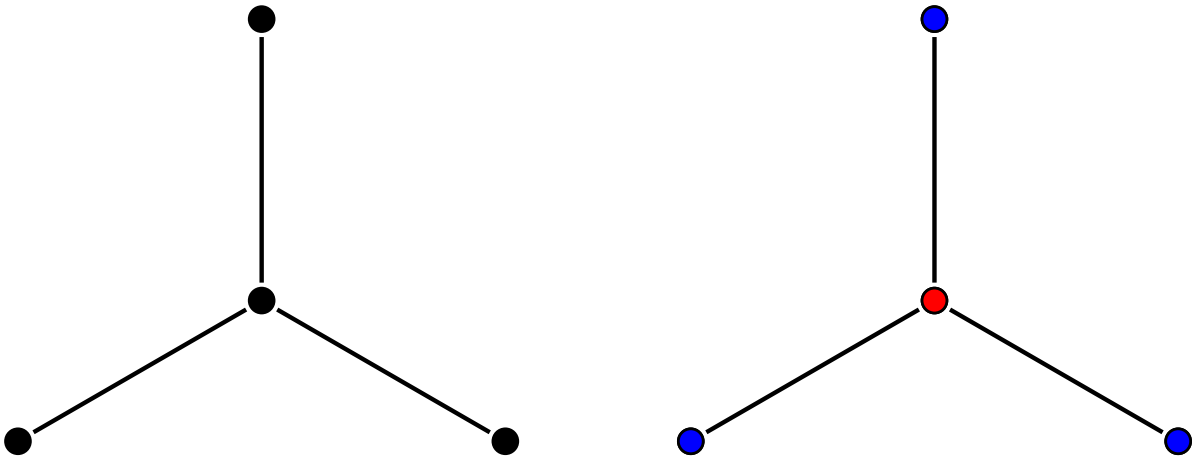
Equitable (Vertex) Coloring

Can we k-color G such that the size of two color classes differ by ≤ 1?



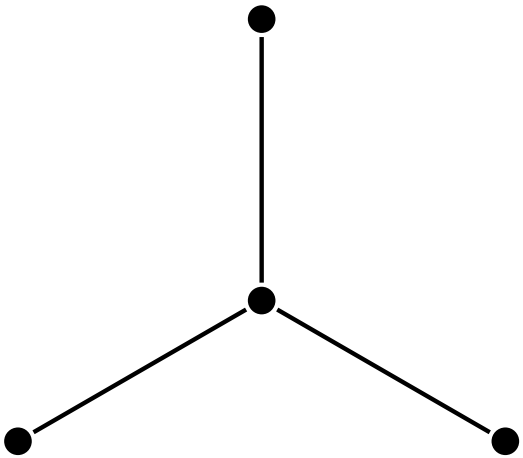
Equitable (Vertex) Coloring

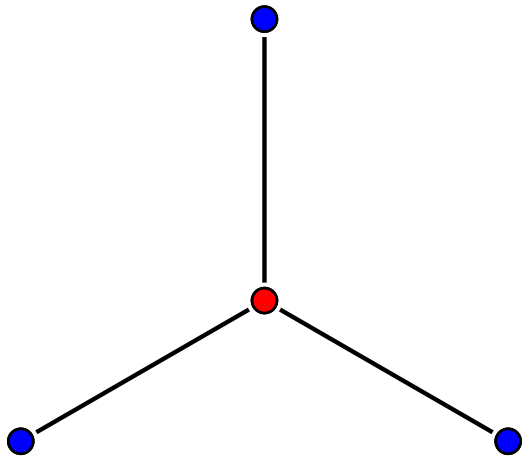
Can we k-color G such that the size of two color classes differ by ≤ 1?

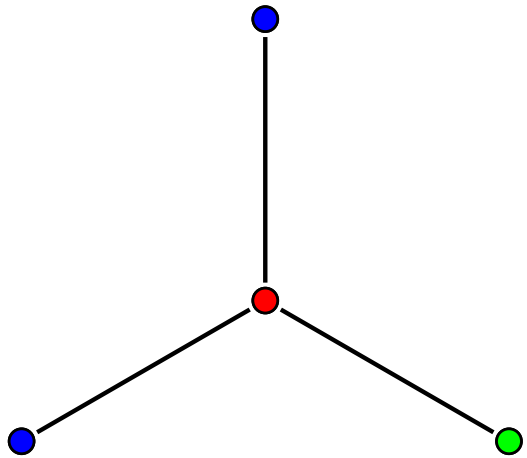


Equitable (Vertex) Coloring

Can we k-color G such that the size of two color classes differ by ≤ 1?







Applications

- Municipal garbage collection;
- Processor task scheduling;
- Communication control;
- Server load balancing.

Some important stuff I

Equitable chromatic number

The smallest integer k such that G is equitably k-colorable is the *equitable chromatic number* $\chi_=(G)$.

Some important stuff I

Equitable chromatic number

The smallest integer k such that G is equitably k-colorable is the *equitable chromatic number* $\chi_=(G)$.

Equitable Coloring Conjecture

For every connected graph G which is neither a complete graph nor an odd-hole, $\chi_=(G) \leq \Delta(G)$.

Some important stuff II

Equitable chromatic threshold

The smallest integer k such that for every $k' \geq k$, G is equitably k'-colorable is its *equitable chromatic threshold* $\chi^*_=(G)$.

Some important stuff II

Equitable chromatic threshold

The smallest integer k such that for every $k' \geq k$, G is equitably k'-colorable is its *equitable chromatic threshold* $\chi^*_=(G)$.

Hajnal-Szmerédi Theorem

Any graph G is equitably k-colorable if $k \geq \Delta(G) + 1$. Equivalently, $\chi^*_=(G) \leq \Delta(G) + 1$.

Some important stuff II

Equitable chromatic threshold

The smallest integer k such that for every $k' \geq k$, G is equitably k'-colorable is its *equitable chromatic threshold* $\chi^*_=(G)$.

Hajnal-Szmerédi Theorem

Any graph G is equitably k-colorable if $k \geq \Delta(G) + 1$. Equivalently, $\chi^*_=(G) \leq \Delta(G) + 1$.

Equitable Δ Coloring Conjecture

For every connected graph G which is not a complete graph, an odd-hole nor $K_{2n+1,2n+1}$, for any $n \geq 1$, $\chi^*_=(G) \leq \Delta(G)$ holds.

Some important stuff II

Equitable chromatic threshold

The smallest integer k such that for every $k' \geq k$, G is equitably k'-colorable is its *equitable chromatic threshold* $\chi^*_=(G)$.

Hajnal-Szmerédi Theorem

Any graph G is equitably k-colorable if $k \geq \Delta(G) + 1$. Equivalently, $\chi^*_=(G) \leq \Delta(G) + 1$.

Equitable Δ Coloring Conjecture

For every connected graph G which is not a complete graph, an odd-hole nor $K_{2n+1,2n+1}$, for any $n \geq 1$, $\chi^*_=(G) \leq \Delta(G)$ holds.

Ko-Wei Lih. "Equitable coloring of graphs". In: *Handbook of combinatorial optimization*. Springer, 2013, pp. 1199–1248

The story so far...

Class	Complexity
Trees	P
Forests	P
Bipartite	NP-complete, even if $k = 3$
Co-bipartite	P
Cographs	NP-complete, P for each fixed k
Bounded Treewidth	P
Chordal	NP-complete
Block	?
Split	P
Unipolar	?
Interval	NP-complete
Co-interval	P

After this talk

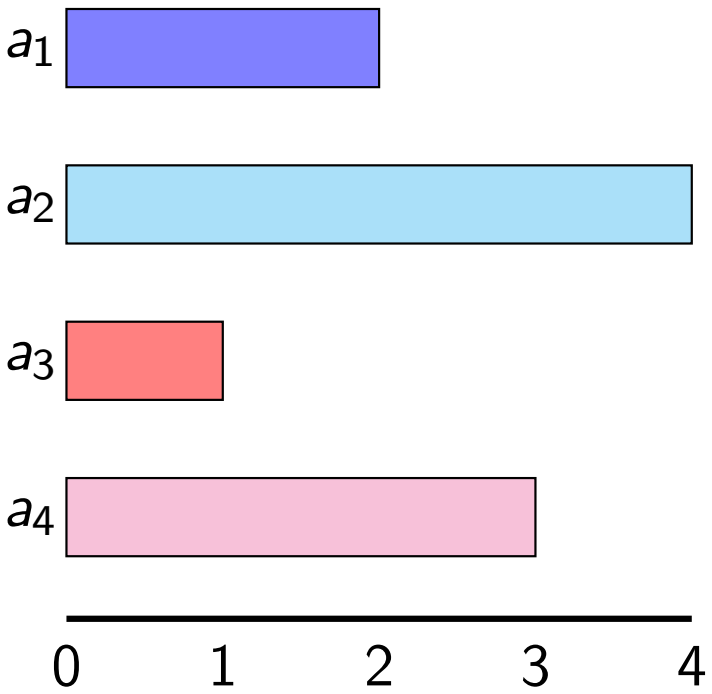
Class	Complexity
Trees	P
Forests	P
Bipartite	NP-complete, even if $k = 3$
Co-bipartite	P
Cographs	NP-complete, P for each fixed k
Bounded Treewidth	P
Chordal	NP-complete
Block	NP-complete
Split	P
Unipolar	P
Interval	NP-complete
Co-interval	P

Bin Packing

Can we partition $A = \{a_1, \ldots, a_n\}$ in k bins such that $\sum_{a_j \in \text{bin}_i} a_j = B$?

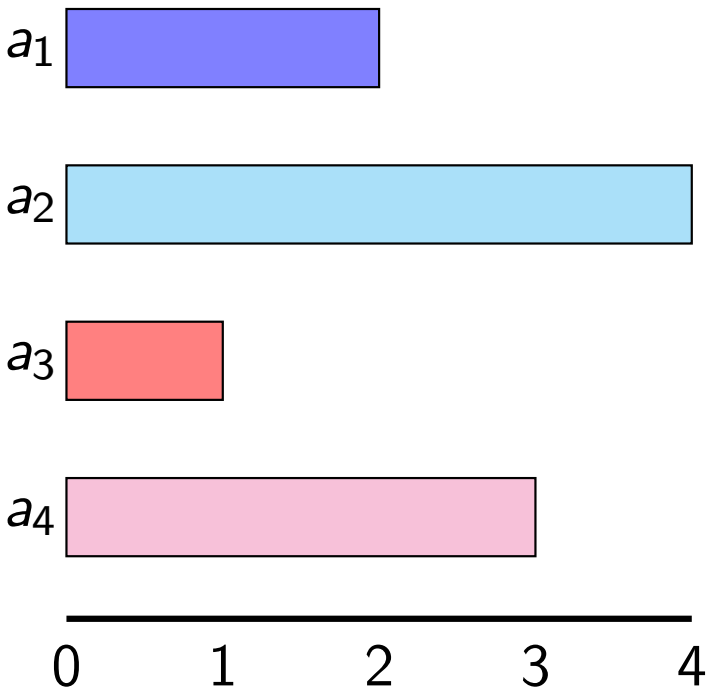
Bin Packing

Can we partition $A = \{a_1, \ldots, a_n\}$ in k bins such that $\sum_{a_j \in \text{bin}_i} a_j = B$?



Bin Packing

Can we partition $A = \{a_1, \ldots, a_n\}$ in k bins such that $\sum_{a_j \in \text{bin}_i} a_j = B$?



For each item of A, build a gadget with some **key** vertices.

All key vertices must have the same color.

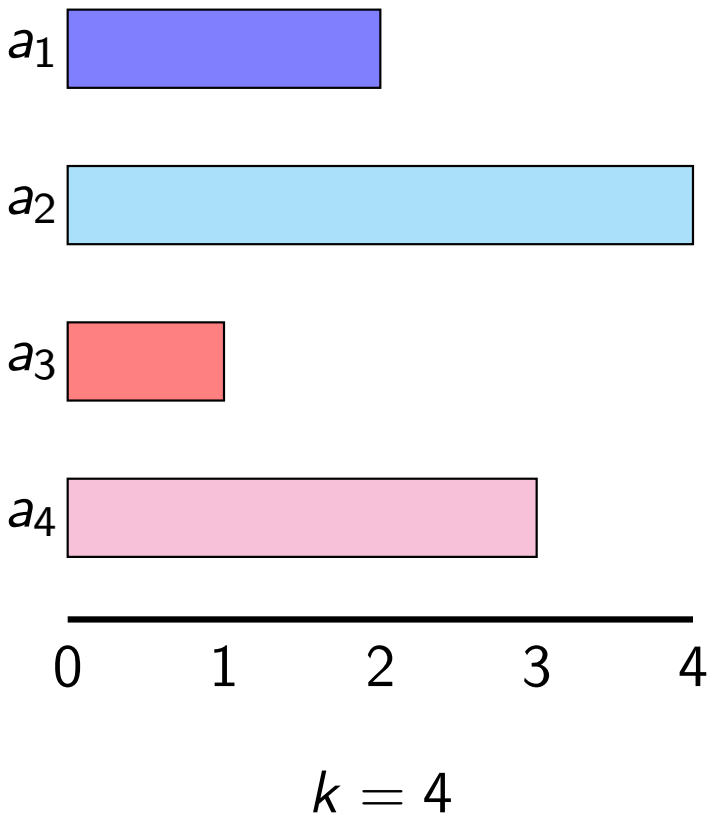
Key vertices with color $i \rightarrow$ item in i-th bin.

(a, k)-flowers

Create $a + 1$ cliques with $k - 1$ vertices and add one universal vertex.

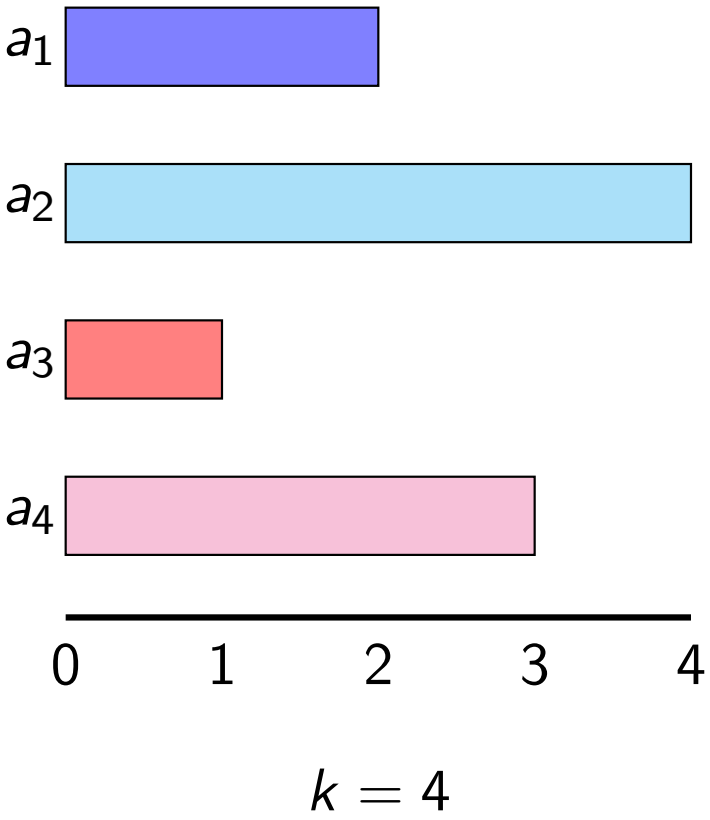
(a, k)-flowers

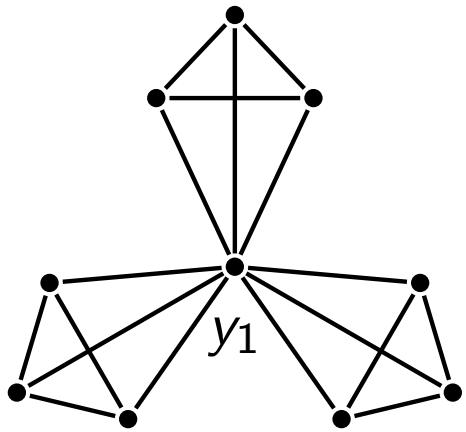
Create $a + 1$ cliques with $k - 1$ vertices and add one universal vertex.

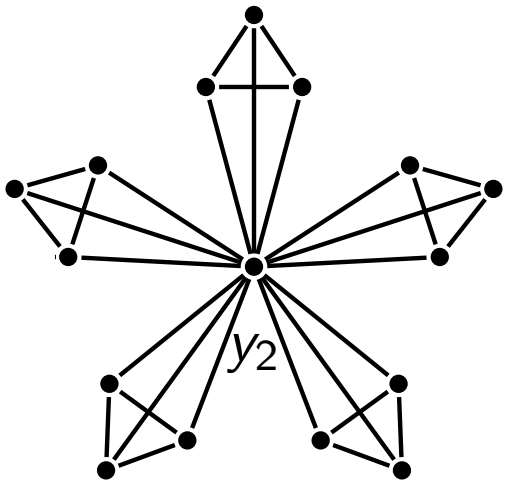


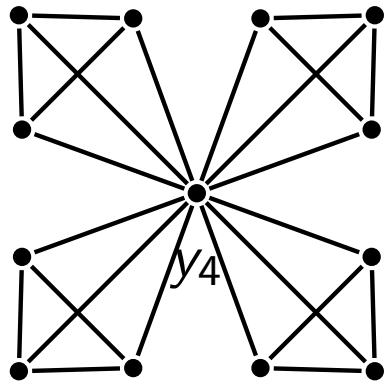
(a, k)-flowers

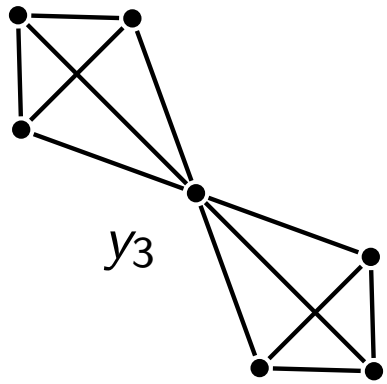
Create $a + 1$ cliques with $k - 1$ vertices and add one universal vertex.



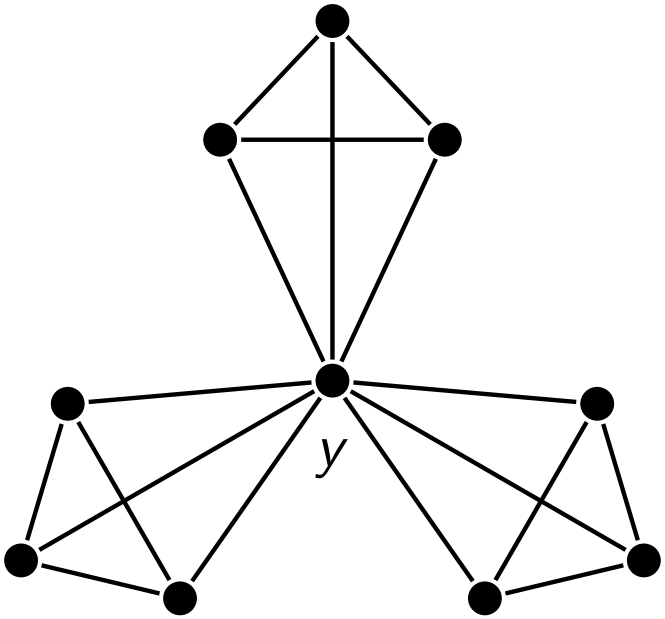




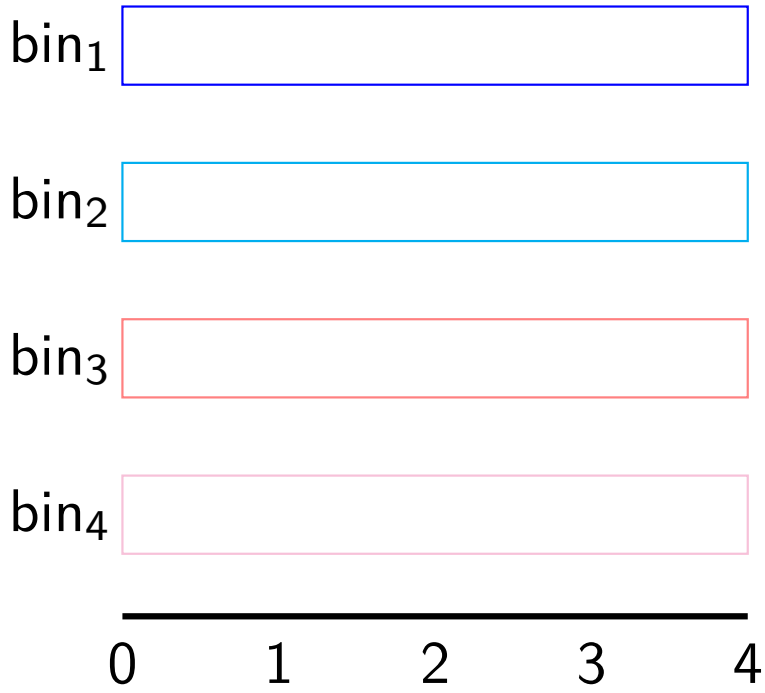




Flowers, colors, and bins

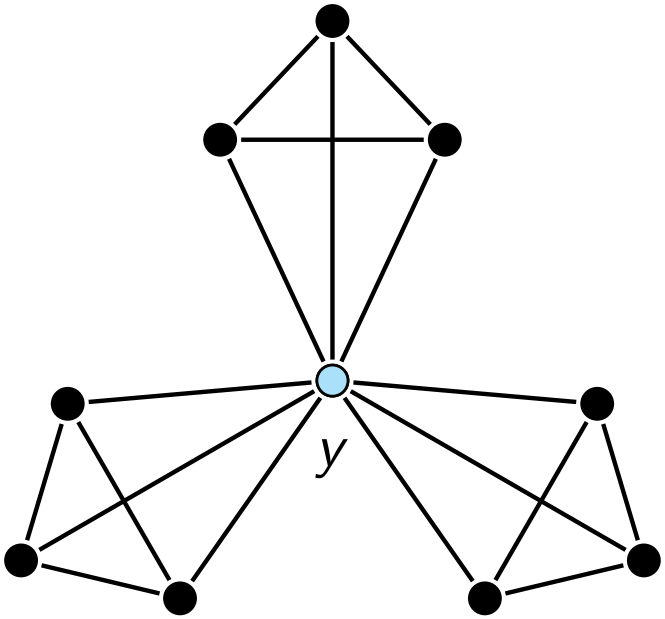


$a_j = 2, k = 4$

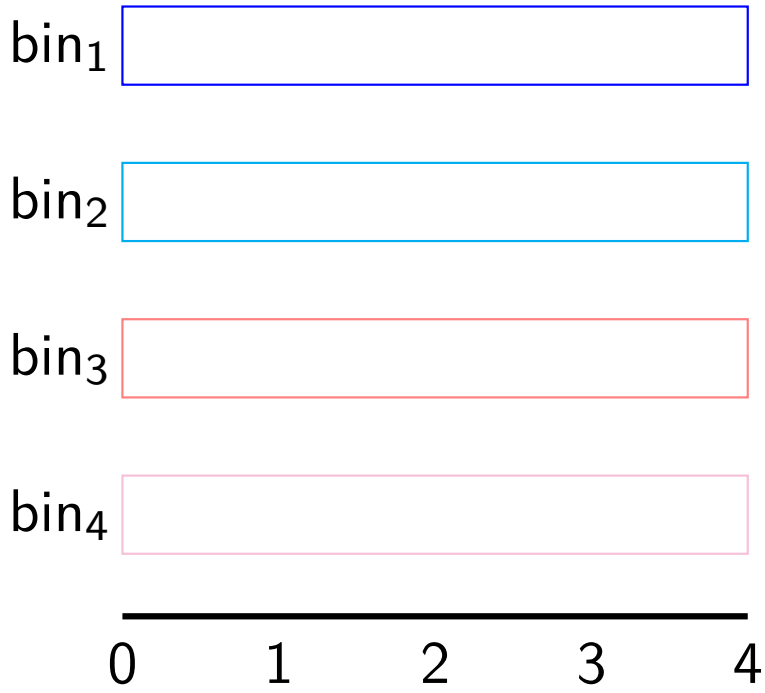


$k = 4$

Flowers, colors, and bins

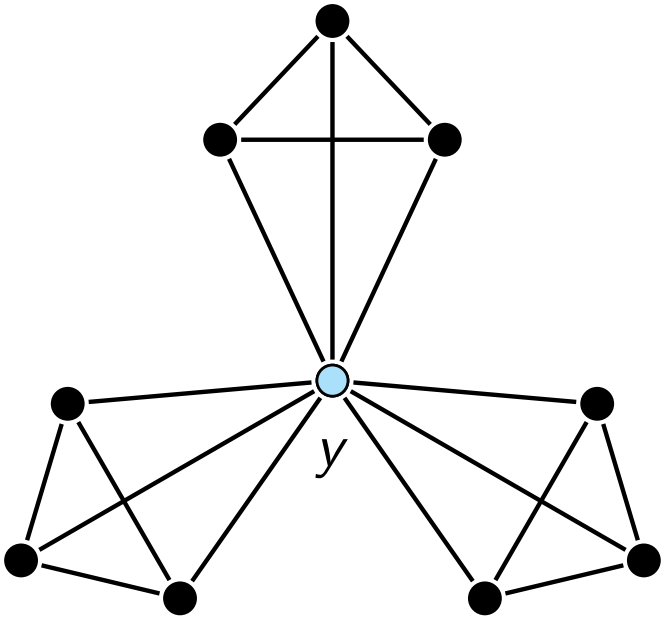


$a_j = 2, k = 4$

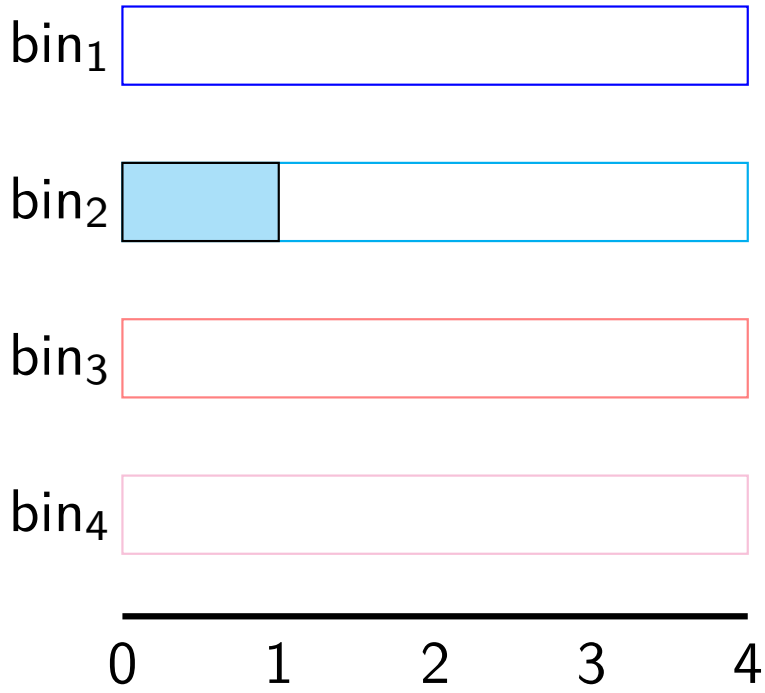


$k = 4$

Flowers, colors, and bins

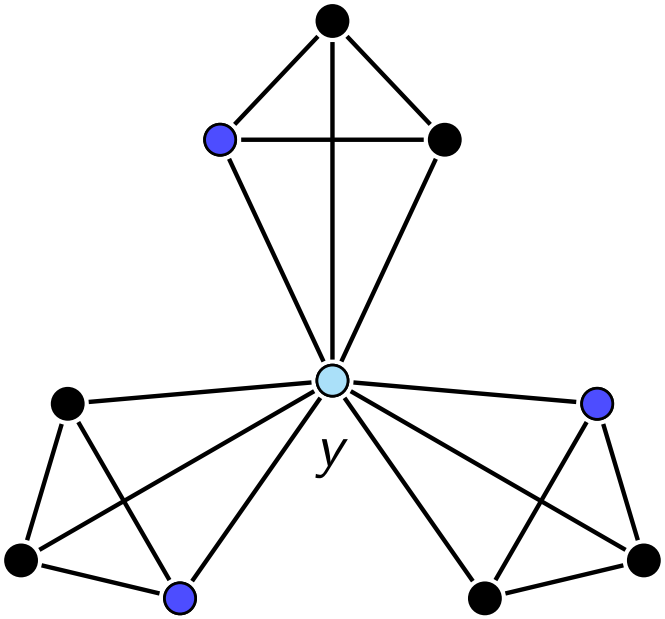


$a_j = 2, k = 4$

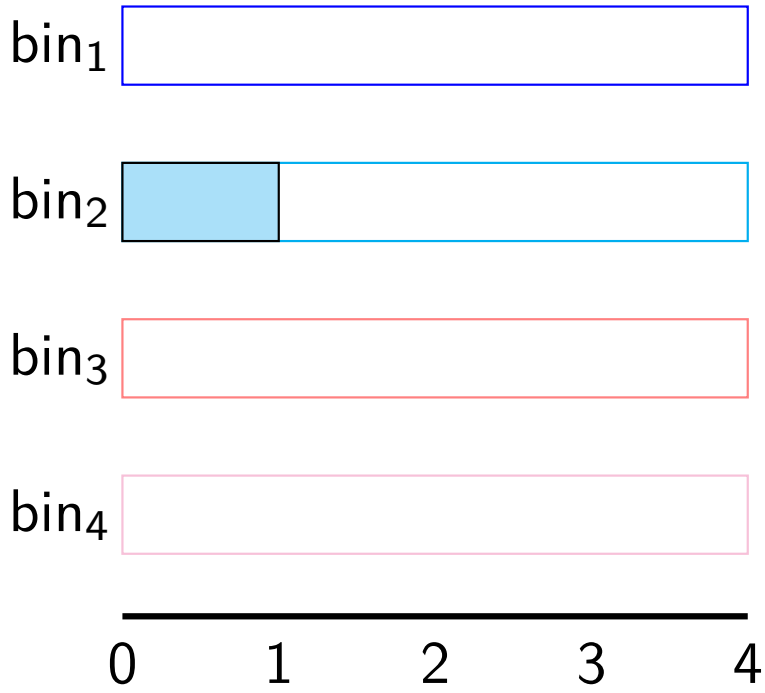


$k = 4$

Flowers, colors, and bins

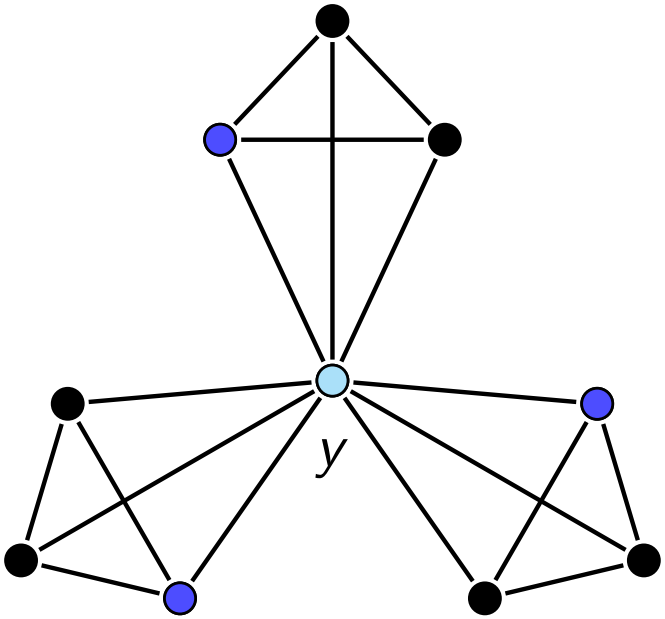


$a_j = 2, k = 4$

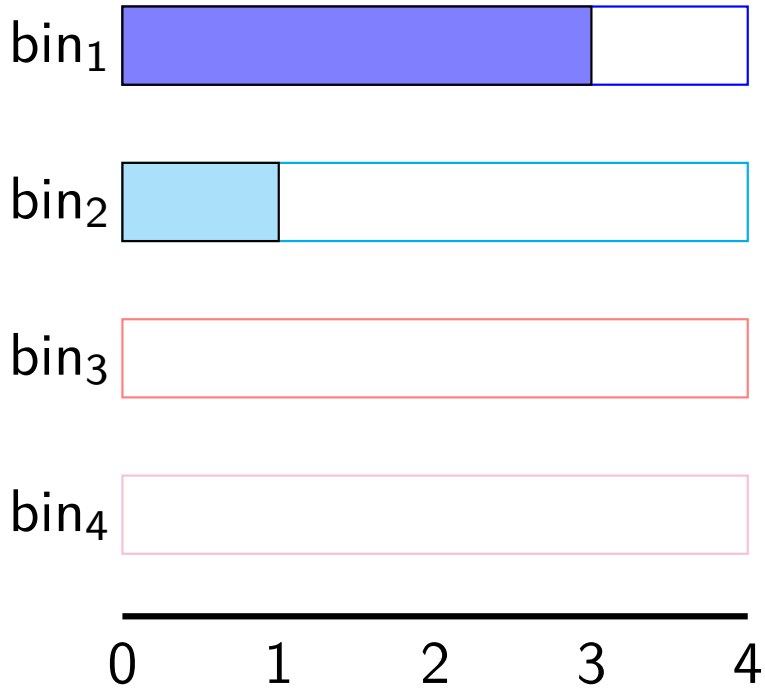


$k = 4$

Flowers, colors, and bins

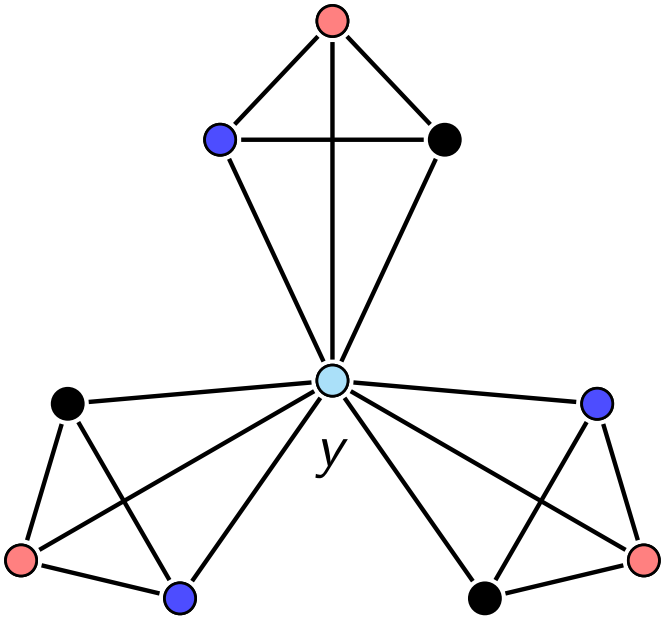


$a_j = 2, k = 4$

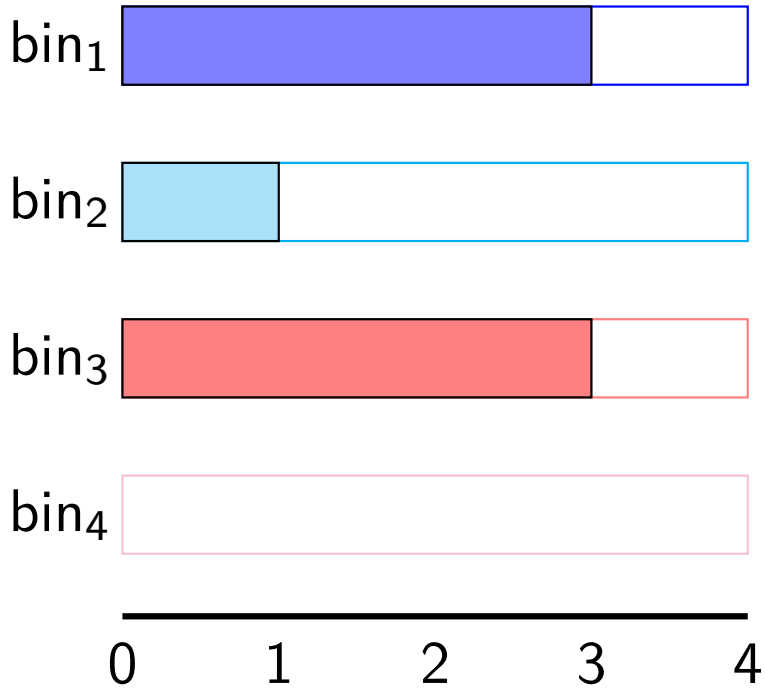


$k = 4$

Flowers, colors, and bins

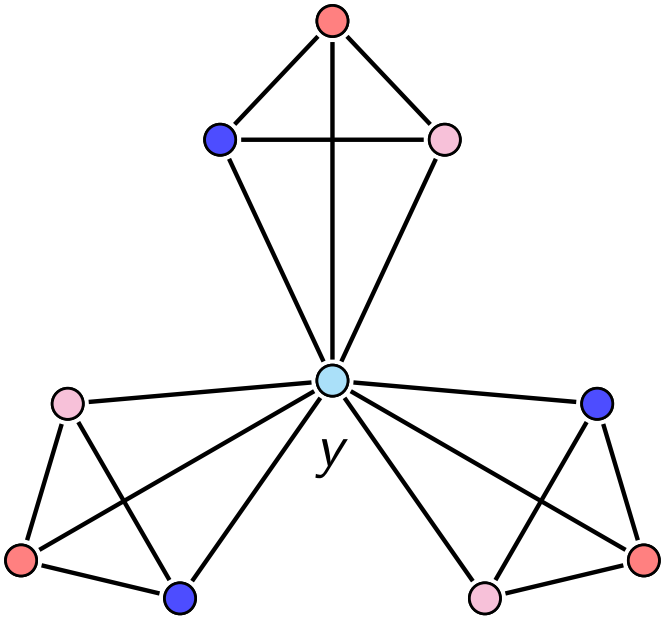


$a_j = 2, k = 4$

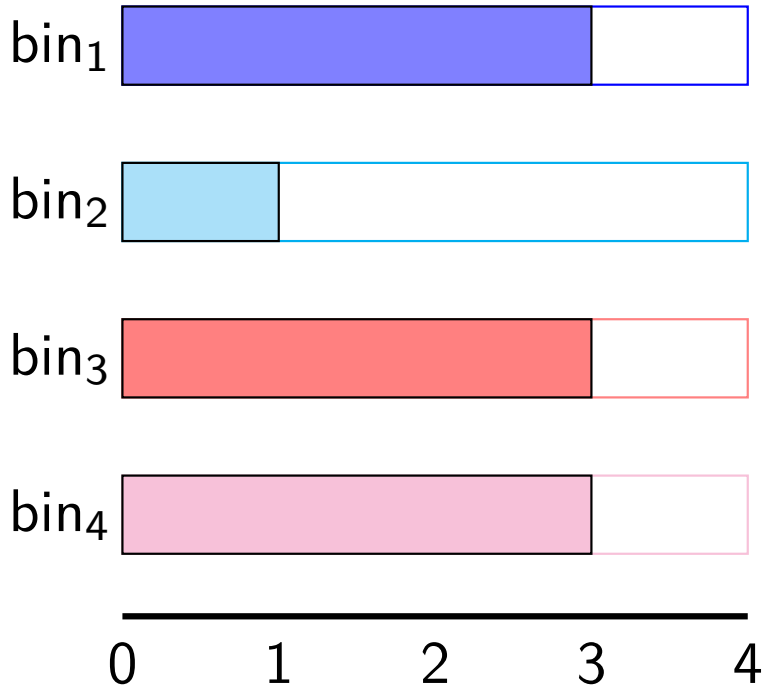


$k = 4$

Flowers, colors, and bins



$a_j = 2, k = 4$



$k = 4$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Construct a graph G as the disjoint union of flowers $F_j = F(a_j, k)$ and try to equitably k-color it.

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Construct a graph G as the disjoint union of flowers $F_j = F(a_j, k)$ and try to equitably k-color it.

$$|V(G)| = \sum_{j \in [n]} ((a_j + 1)(k-1) + 1)$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Construct a graph G as the disjoint union of flowers $F_j = F(a_j, k)$ and try to equitably k-color it.

$$|V(G)| = \sum_{j \in [n]} ((a_j + 1)(k-1) + 1)$$

$$= (k-1)\left(n + \sum_{j \in [n]} a_j\right) + n$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Construct a graph G as the disjoint union of flowers $F_j = F(a_j, k)$ and try to equitably k-color it.

$$\begin{aligned} |V(G)| &= \sum_{j \in [n]} ((a_j + 1)(k-1) + 1) \\ &= (k-1)\left(n + \sum_{j \in [n]} a_j\right) + n \\ &= (k-1)(n + kB) + n \end{aligned}$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Construct a graph G as the disjoint union of flowers $F_j = F(a_j, k)$ and try to equitably k-color it.

$$\begin{aligned} |V(G)| &= \sum_{j\in[n]} ((a_j+1)(k-1)+1) \\ &= (k-1)\left(n + \sum_{j\in[n]} a_j\right) + n \\ &= (k-1)(n+kB) + n \\ &= kn + k^2B - n - kB + n \end{aligned}$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Construct a graph G as the disjoint union of flowers $F_j = F(a_j, k)$ and try to equitably k-color it.

$$
\begin{aligned}
|V(G)| &= \sum_{j \in [n]} ((a_j + 1)(k-1) + 1) \\
&= (k-1)\left(n + \sum_{j \in [n]} a_j\right) + n \\
&= (k-1)(n + kB) + n \\
&= kn + k^2 B - n - kB + n \\
&= k(kB - B + n)
\end{aligned}
$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution φ to BIN-PACKING, $\psi(y_j) = i$ if $a_j \in \varphi_i$.

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution φ to BIN-PACKING, $\psi(y_j) = i$ if $a_j \in \varphi_i$.

$$|\psi_i| = |\varphi_i| + \sum_{j | y_j \notin \psi_i} (a_j + 1)$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution φ to BIN-PACKING, $\psi(y_j) = i$ if $a_j \in \varphi_i$.

$$|\psi_i| = |\varphi_i| + \sum_{j|y_j \notin \psi_i} (a_j + 1)$$

$$= |\varphi_i| + \sum_{j \in [n]} (a_j + 1) - \sum_{j|y_j \in \psi_i} (a_j + 1)$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution φ to BIN-PACKING, $\psi(y_j) = i$ if $a_j \in \varphi_i$.

$$
\begin{aligned}
|\psi_i| &= |\varphi_i| + \sum_{j|y_j \notin \psi_i} (a_j + 1) \\
&= |\varphi_i| + \sum_{j \in [n]} (a_j + 1) - \sum_{j|y_j \in \psi_i} (a_j + 1) \\
&= |\varphi_i| + n + kB - B - |\varphi_i|
\end{aligned}
$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution φ to BIN-PACKING, $\psi(y_j) = i$ if $a_j \in \varphi_i$.

$$
\begin{aligned}
|\psi_i| &= |\varphi_i| + \sum_{j|y_j \notin \psi_i} (a_j + 1) \\
&= |\varphi_i| + \sum_{j \in [n]} (a_j + 1) - \sum_{j|y_j \in \psi_i} (a_j + 1) \\
&= |\varphi_i| + n + kB - B - |\varphi_i| \\
&= kB - B + n = \frac{|V(G)|}{k}
\end{aligned}
$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution ψ to EQUITABLE COLORING, put a_j in φ_i if $\psi(y_j) = i$.

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution ψ to EQUITABLE COLORING, put a_j in φ_i if $\psi(y_j) = i$.

$$kB - B + n = |\psi_i|$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution ψ to EQUITABLE COLORING, put a_j in φ_i if $\psi(y_j) = i$.

$$\begin{aligned} kB - B + n &= |\psi_i| \\ &= \sum_{j|y_j \in \psi_i} 1 + \sum_{j|y_j \notin \psi_i} (a_j + 1) \end{aligned}$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution ψ to EQUITABLE COLORING, put a_j in φ_i if $\psi(y_j) = i$.

$$\begin{aligned} kB - B + n &= |\psi_i| \\ &= \sum_{j|y_j \in \psi_i} 1 + \sum_{j|y_j \notin \psi_i} (a_j + 1) \\ &= \sum_{j|y_j \in \psi_i} 1 + \sum_{j \in [n]} (a_j + 1) - \sum_{j|y_j \in \psi_i} (a_j + 1) \end{aligned}$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution ψ to EQUITABLE COLORING, put a_j in φ_i if $\psi(y_j) = i$.

$$\begin{aligned} kB - B + n &= |\psi_i| \\ &= \sum_{j|y_j \in \psi_i} 1 + \sum_{j|y_j \notin \psi_i} (a_j + 1) \\ &= \sum_{j|y_j \in \psi_i} 1 + \sum_{j \in [n]} (a_j + 1) - \sum_{j|y_j \in \psi_i} (a_j + 1) \\ &= kB + n - \sum_{j|y_j \in \psi_i} a_j \end{aligned}$$

Block Graphs

Theorem

EQUITABLE COLORING *of block graphs is* NP-complete.

Given a solution ψ to EQUITABLE COLORING, put a_j in φ_i if $\psi(y_j) = i$.

$$
\begin{aligned}
kB - B + n &= |\psi_i| \\
&= \sum_{j|y_j \in \psi_i} 1 + \sum_{j|y_j \notin \psi_i} (a_j + 1) \\
&= \sum_{j|y_j \in \psi_i} 1 + \sum_{j \in [n]} (a_j + 1) - \sum_{j|y_j \in \psi_i} (a_j + 1) \\
&= kB + n - \sum_{j|y_j \in \psi_i} a_j
\end{aligned}
$$

$$
B = \sum_{j|y_j \in \psi_i} a_j
$$

Disjoint union of split graphs (complete k-partite)

Theorem

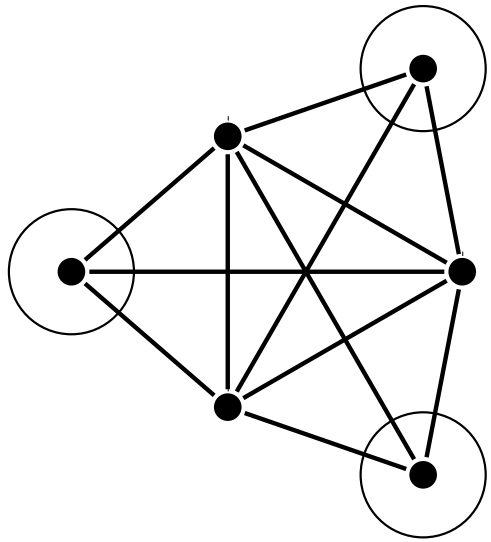
Equitable Coloring *of disjoint union of split graphs is* NP-complete.

Disjoint union of split graphs (complete k-partite)

Theorem

Equitable Coloring *of disjoint union of split graphs is* NP-complete.

Each a_j becomes a split graph with $k-1$ vertices in the clique and a_j+1 vertices in the independent set (key vertices).



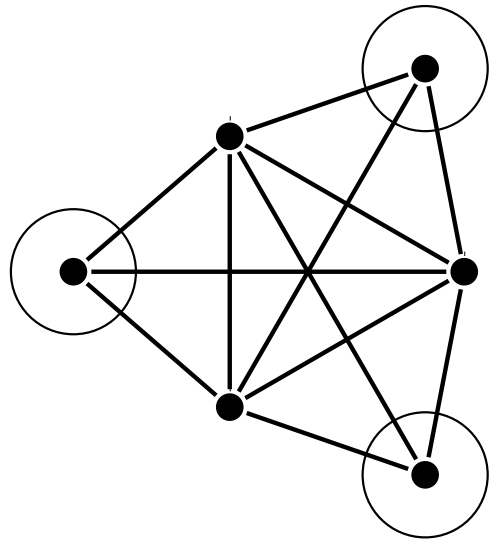
Disjoint union of split graphs (complete k-partite)

Theorem

EQUITABLE COLORING *of disjoint union of split graphs is* NP-complete.

Each a_j becomes a split graph with $k-1$ vertices in the clique and a_j+1 vertices in the independent set (key vertices).

$|V(G)| = \sum_{a_j \in A}(k-1)+(a_j+1) = k(n+B)$.



Disjoint union of split graphs (complete k-partite)

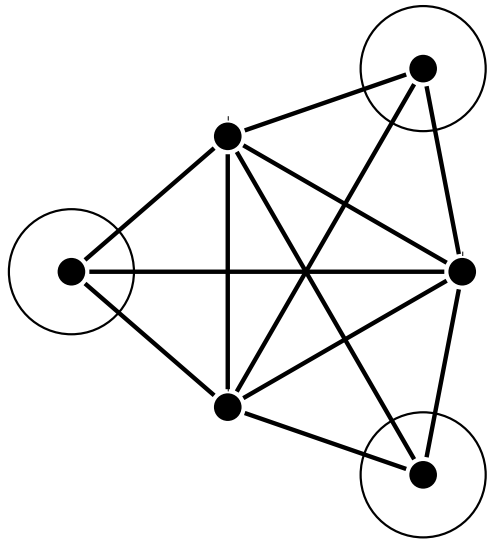
Theorem

Equitable Coloring *of disjoint union of split graphs is* NP-complete.

Each a_j becomes a split graph with $k-1$ vertices in the clique and a_j+1 vertices in the independent set (key vertices).

$|V(G)| = \sum_{a_j \in A}(k-1)+(a_j+1) = k(n+B)$.

Try equitably k-color it.



$K_{1,r}$-free interval graphs

Theorem

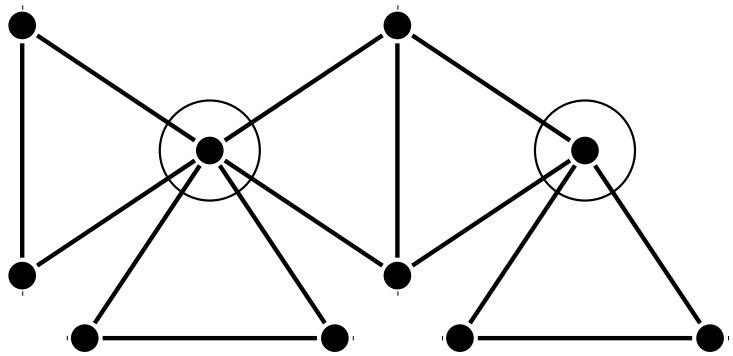
EQUITABLE COLORING *of* $K_{1,r}$*-free interval graphs is* NP-complete *if* $r \geq 4$*, otherwise it is solvable in polynomial time (consequence of de Werra'85).*

$K_{1,r}$-free interval graphs

Theorem

EQUITABLE COLORING *of* $K_{1,r}$*-free interval graphs is* NP*-complete if* $r \geq 4$*, otherwise it is solvable in polynomial time (consequence of de Werra'85).*

Each a_j becomes a sequence of a_j cliques of size $k-1$. Add one universal vertex to each pair of consecutive cliques. Said vertex also has an extra clique of size $k-1$ attached to it.



$k = 3 \qquad a_j = 2$

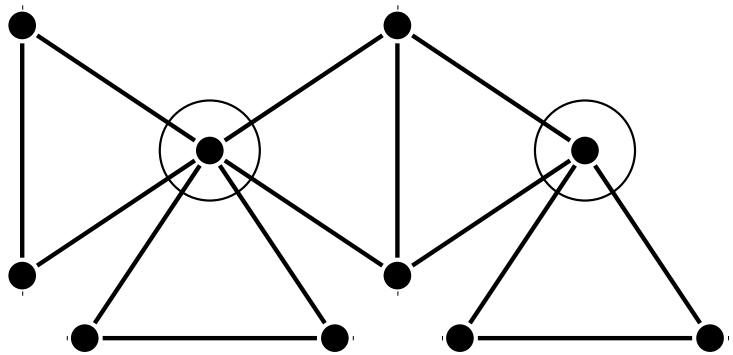
$K_{1,r}$-free interval graphs

Theorem

EQUITABLE COLORING *of* $K_{1,r}$*-free interval graphs is* NP*-complete if* $r \geq 4$*, otherwise it is solvable in polynomial time (consequence of de Werra'85).*

Each a_j becomes a sequence of a_j cliques of size $k-1$. Add one universal vertex to each pair of consecutive cliques. Said vertex also has an extra clique of size $k-1$ attached to it.

$|V(G)| = \sum_{a_j \in A} a_j(k-1) + a_j k = k(2kB - B)$.



$k = 3 \qquad a_j = 2$

$K_{1,r}$-free interval graphs

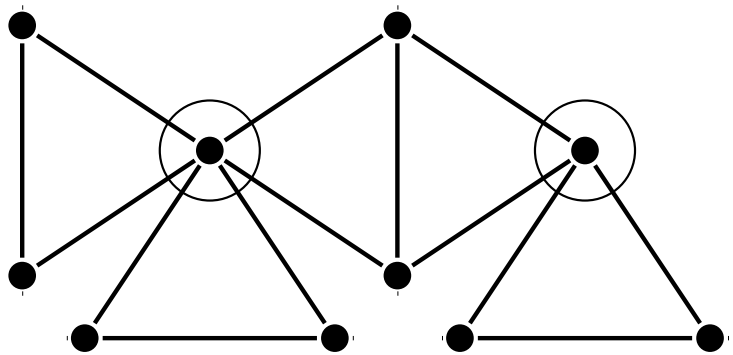
Theorem

EQUITABLE COLORING *of* $K_{1,r}$*-free interval graphs is* NP*-complete if* $r \geq 4$*, otherwise it is solvable in polynomial time (consequence of de Werra'85).*

Each a_j becomes a sequence of a_j cliques of size $k-1$. Add one universal vertex to each pair of consecutive cliques. Said vertex also has an extra clique of size $k-1$ attached to it.

$|V(G)| = \sum_{a_j \in A} a_j(k-1) + a_j k = k(2kB - B)$.

Again, try to equitably k-color G.



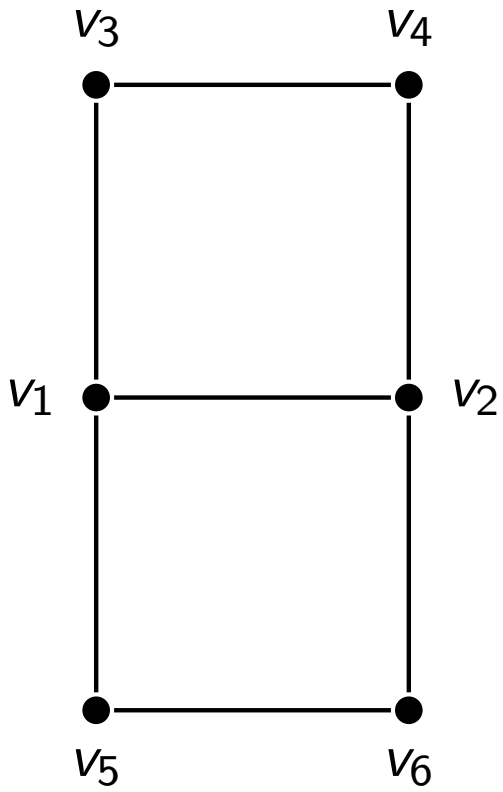
$k = 3 \qquad a_j = 2$

Unipolar graphs

A graph G is unipolar if it has a clique Q such that $G - Q$ is a disjoint union of cliques.

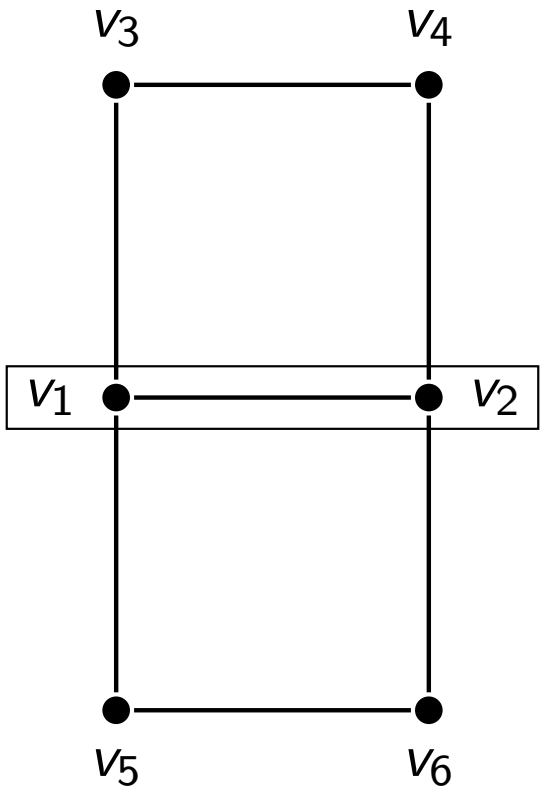
Unipolar graphs

A graph G is unipolar if it has a clique Q such that $G - Q$ is a disjoint union of cliques.

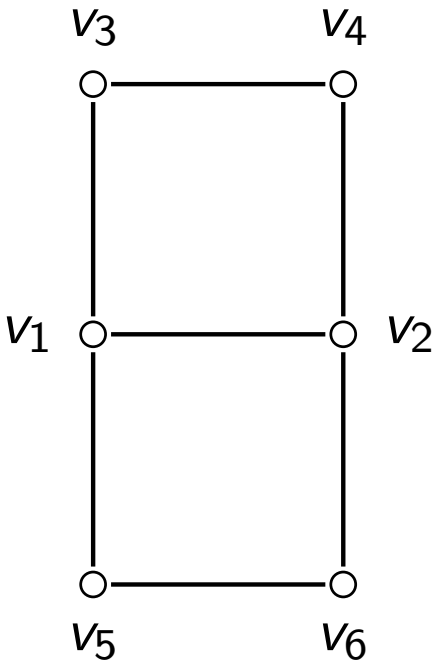


Unipolar graphs

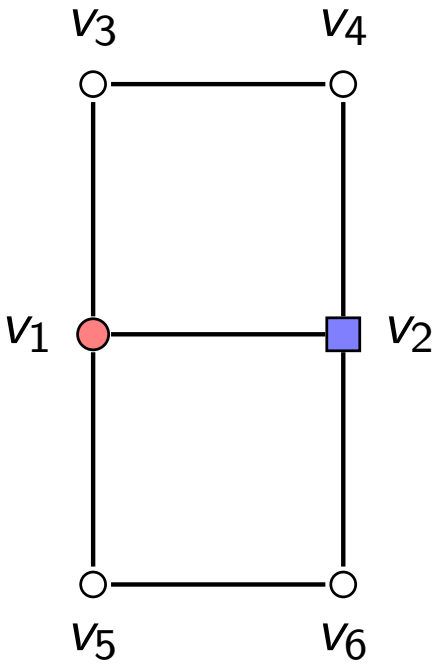
A graph G is unipolar if it has a clique Q such that $G - Q$ is a disjoint union of cliques.



A max-flow based algorithm



A max-flow based algorithm

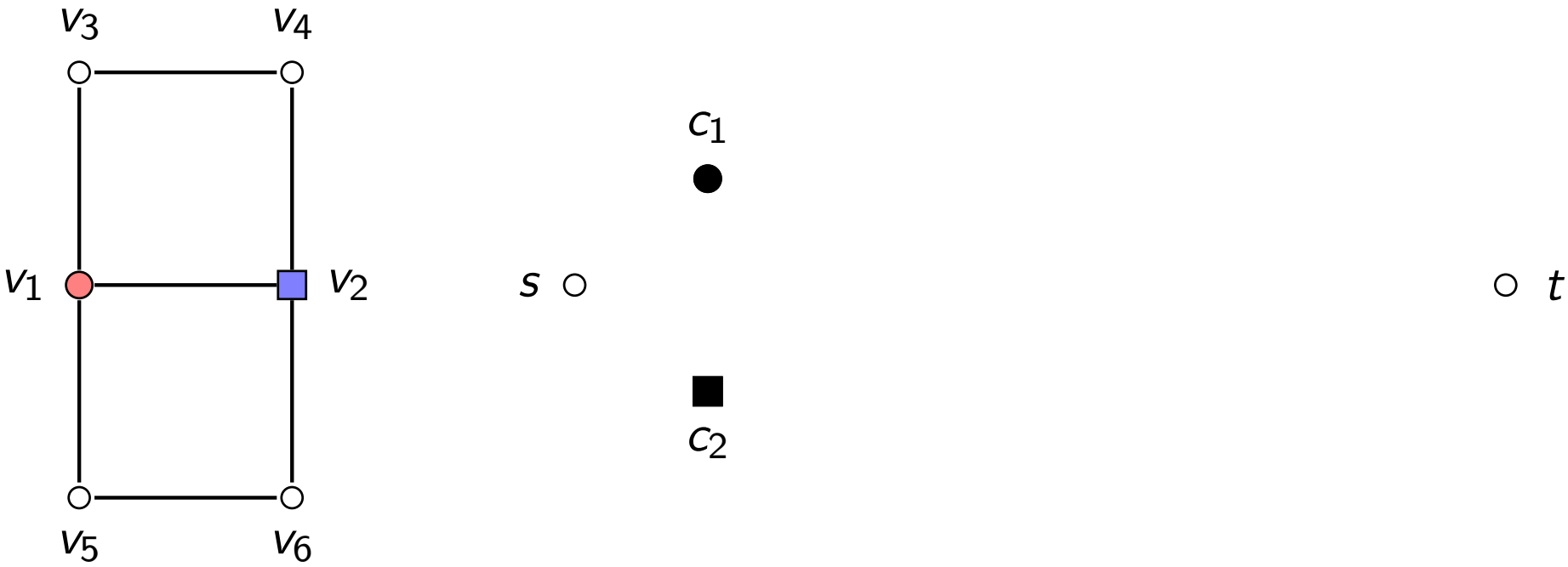


A max-flow based algorithm

Vertices

Source s, sink t,

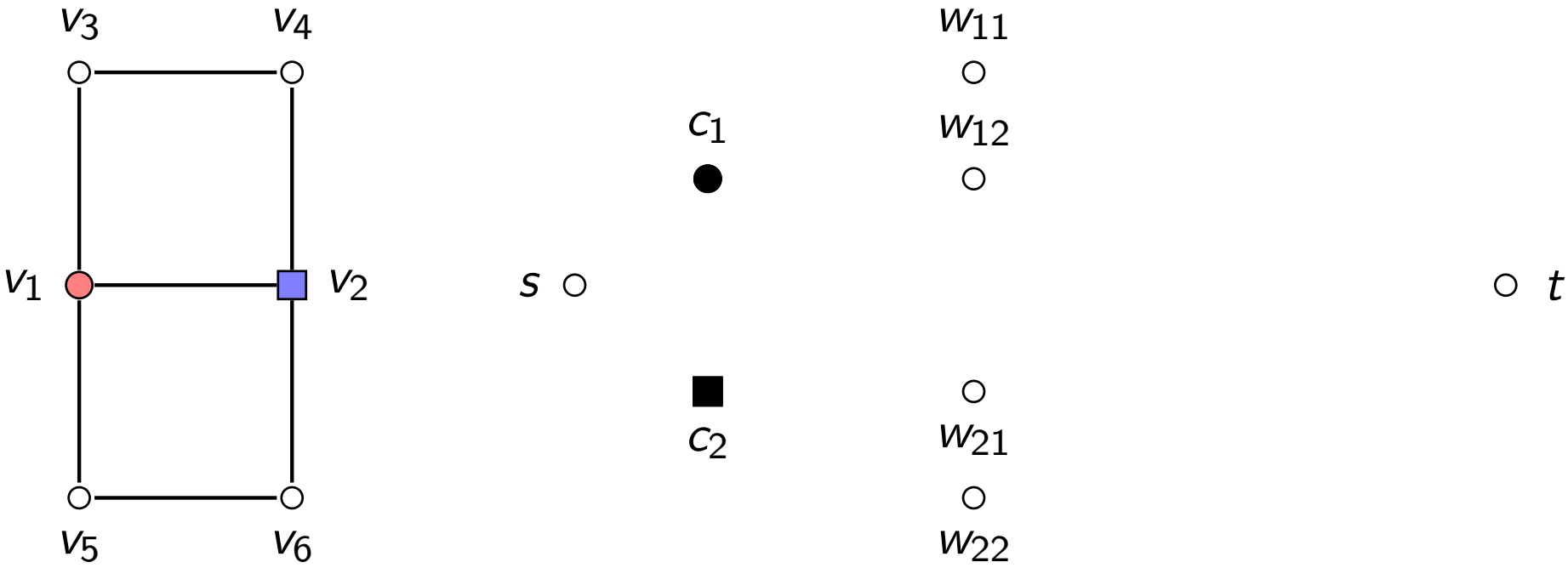
A max-flow based algorithm



Vertices

Source s, sink t, for each color i, c_i,

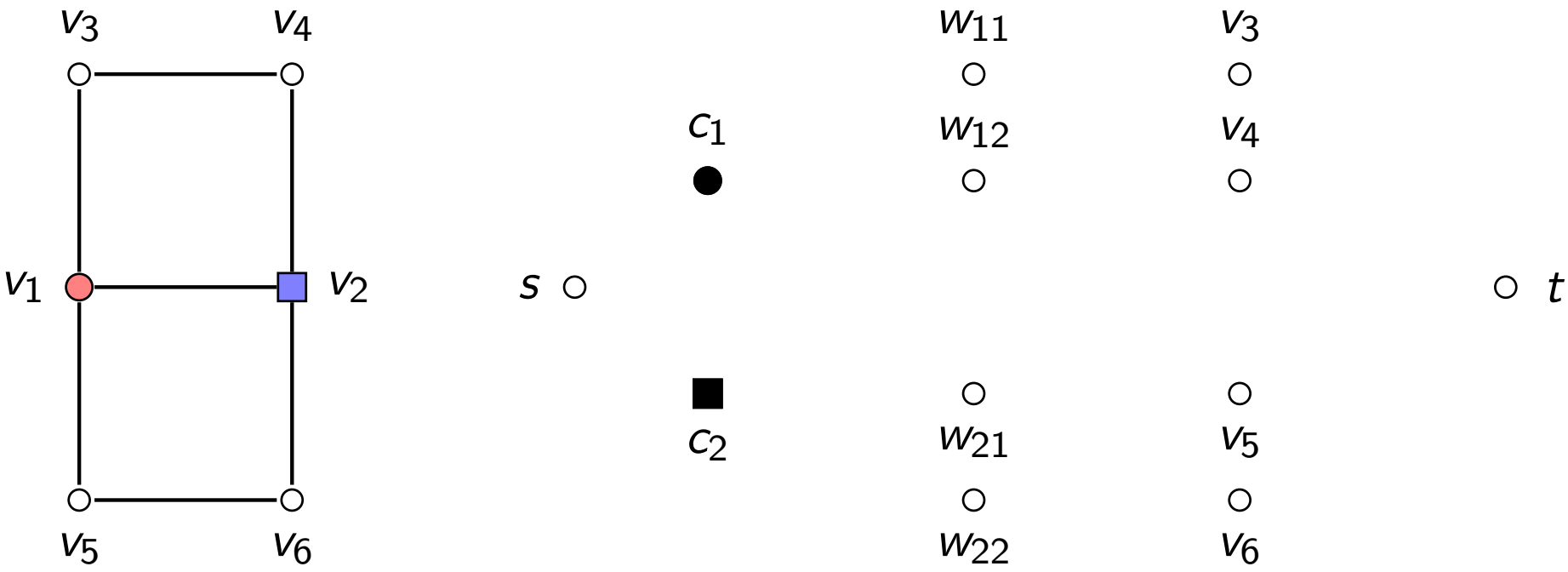
A max-flow based algorithm



Vertices

Source s, sink t, for each color i, c_i, for each color i and clique j, w_{ij},

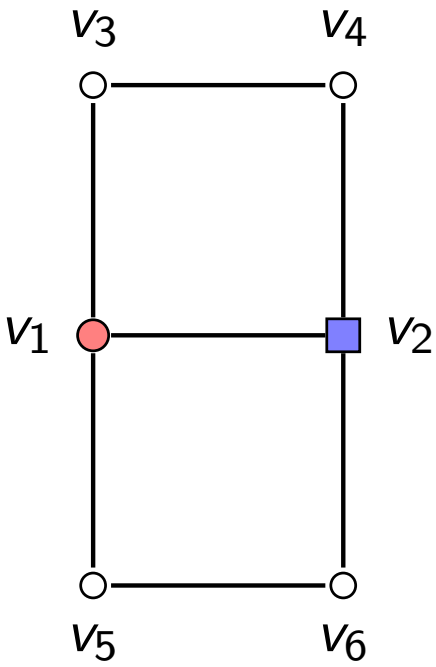
A max-flow based algorithm

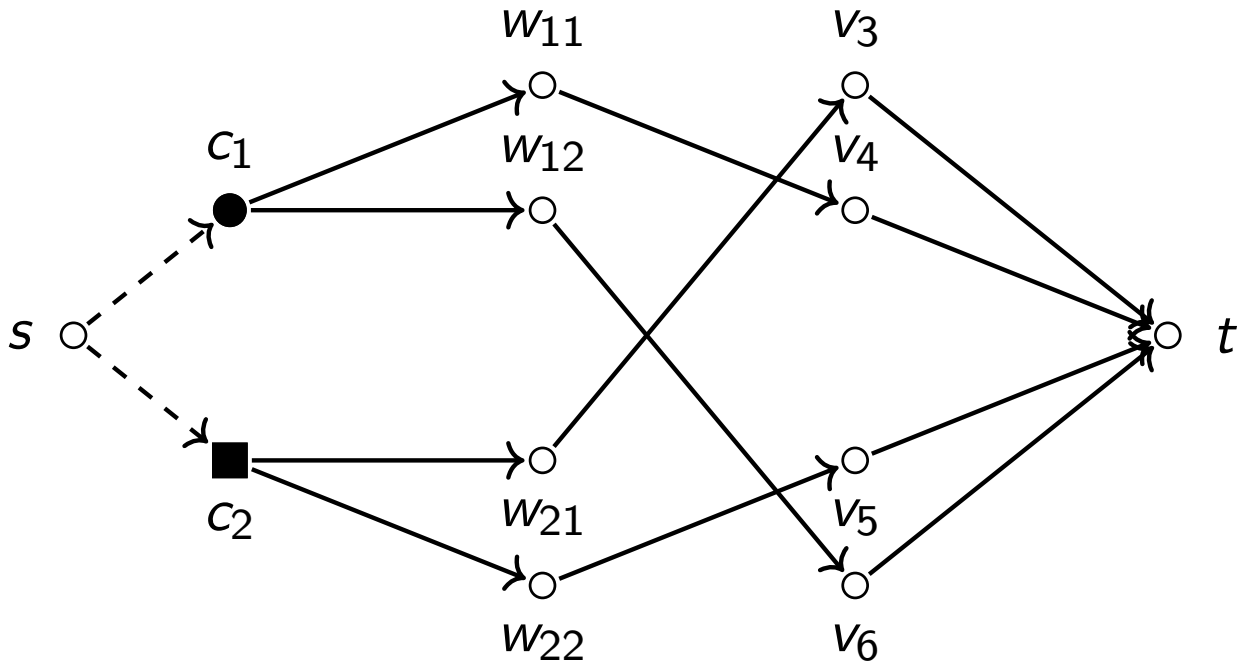


Vertices

Source s, sink t, for each color i, c_i, for each color i and clique j, w_{ij}, for vertex $v_\ell \notin Q$, v_ℓ.

A max-flow based algorithm





Vertices

Source s, sink t, for each color i, c_i, for each color i and clique j, w_{ij}, for vertex $v_\ell \notin Q$, v_ℓ.

Each flow unit gives the color of one vertex. Solid arcs have unit capacity.

Parameterized Complexity

A primer on parameterized complexity

If a problem is NP-hard (we believe that) there is no *exact algorithm* that solves *all instances* in time *polynomial* on the size input.

A primer on parameterized complexity

If a problem is NP-hard (we believe that) there is no *exact algorithm* that solves *all instances* in time *polynomial* on the size input.

A primer on parameterized complexity

If a problem is NP-hard (we believe that) there is no *exact algorithm* that solves *all instances* in time *polynomial* on the size input.

Instances are now a pair (x, k) where x is the input to the problem and the *parameter* k describes some property of x.

A primer on parameterized complexity

If a problem is NP-hard (we believe that) there is no *exact algorithm* that solves *all instances* in time *polynomial* on the size input.

Instances are now a pair (x, k) where x is the input to the problem and the *parameter* k describes some property of x. Very useful when $k << |x|$.

A primer on parameterized complexity

If a problem is NP-hard (we believe that) there is no *exact algorithm* that solves *all instances* in time *polynomial* on the size input.

Instances are now a pair (x, k) where x is the input to the problem and the *parameter* k describes some property of x. Very useful when $k << |x|$.

FPT Problem can be solved in $f(k)n^{\mathcal{O}(1)}$ time.
Example: Vertex Cover parameterized by size of the solution.

A primer on parameterized complexity

If a problem is NP-hard (we believe that) there is no *exact algorithm* that solves *all instances* in time *polynomial* on the size input.

Instances are now a pair (x, k) where x is the input to the problem and the *parameter* k describes some property of x. Very useful when $k << |x|$.

FPT Problem can be solved in $f(k)n^{\mathcal{O}(1)}$ time.
Example: Vertex Cover parameterized by size of the solution.

Kernelization x can be compressed into a *kernel* x' so that $|x'| \leq g(k)$.
There exists an FPT algorithm if and only if there is a kernel.

A primer on parameterized complexity

If a problem is NP-hard (we believe that) there is no *exact algorithm* that solves *all instances* in time *polynomial* on the size input.

Instances are now a pair (x, k) where x is the input to the problem and the *parameter* k describes some property of x. Very useful when $k << |x|$.

FPT Problem can be solved in $f(k)n^{\mathcal{O}(1)}$ time.
Example: Vertex Cover parameterized by size of the solution.

Kernelization x can be compressed into a *kernel* x' so that $|x'| \leq g(k)$.
There exists an FPT algorithm if and only if there is a kernel.

W[1]-hard No $f(k)n^{\mathcal{O}(1)}$ time algorithm unless FPT $=$ W[1].
Example: Clique parameterized by size of the solution.

The parameterized story so far...

Class	Parameterized Complexity
Bipartite	W[1]-hard parameterized by #colors
Cographs	W[1]-hard parameterized by #colors
Chordal	W[1]-hard parameterized by #colors
Block	?
Disjoint union of Split	?
interval	?
Independent set $+kv$	FPT param. by k
Split $+kv$	W[1]-hard parameterized by k
Cluster $+kv$	?
Co-cluster $+kv$	?
Forest $+kv$	W[1]-hard parameterized by $k+$ #colors
Path $+kv$	?
Max leaf ℓ	FPT parameterized by ℓ

After this talk

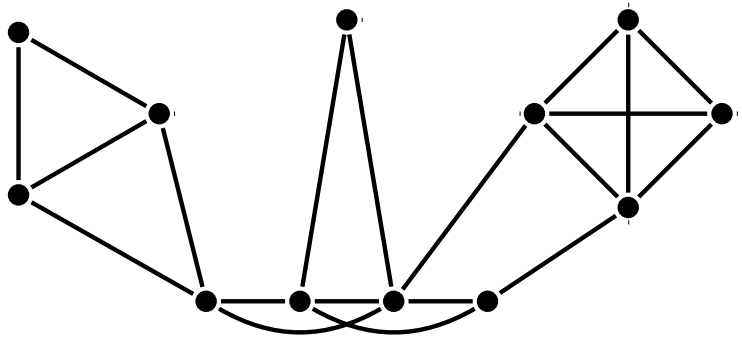
Class	Parameterized Complexity
Bipartite	W[1]-hard parameterized by #colors
Cographs	W[1]-hard param. by #colors
Chordal	W[1]-hard param. by #colors
Block	W[1]-hard param. by #colors + treedepth
Disjoint union of Split	W[1]-hard param. by #colors + tw
Interval	W[1]-hard param. by #colors + bandwidth
Independent set $+kv$	FPT param. by k
Split $+kv$	W[1]-hard param. by k
Cluster $+kv$	FPT param. by k
Co-cluster $+kv$	FPT param. by k
Forest $+kv$	W[1]-hard param. by k + #colors
Path $+kv$	W[1]-hard param. by k + #colors
Max leaf ℓ	Cubic kernel parameterized by ℓ

Cluster $+kv$

G is a cluster graph if each of its connected components is a clique (cluster).

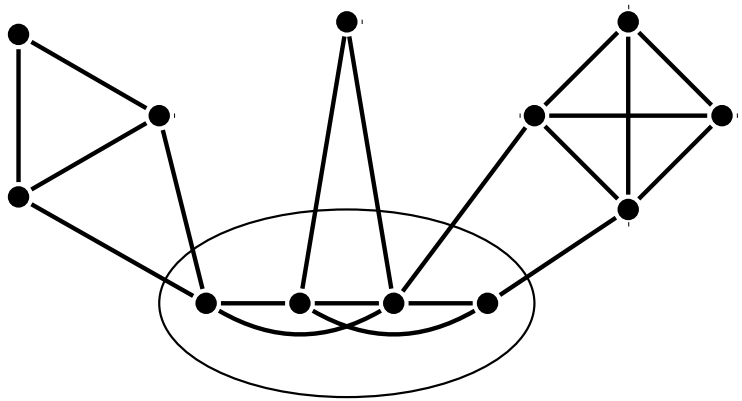
Cluster $+kv$

G is a cluster graph if each of its connected components is a clique (cluster).



Cluster $+kv$

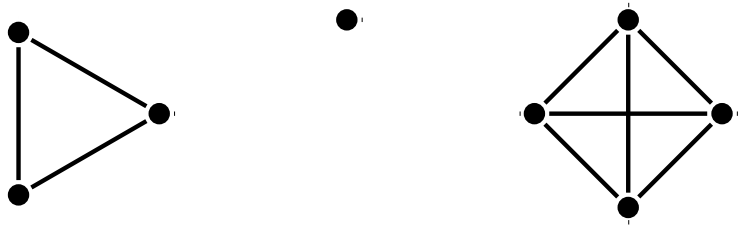
G is a cluster graph if each of its connected components is a clique (cluster).



G is a cluster $+kv$ graph if there is a set $U \subset V(G)$ of size k such that $G - U$ is a cluster graph, with clusters $\{C_1, \ldots, C_\ell\}$.

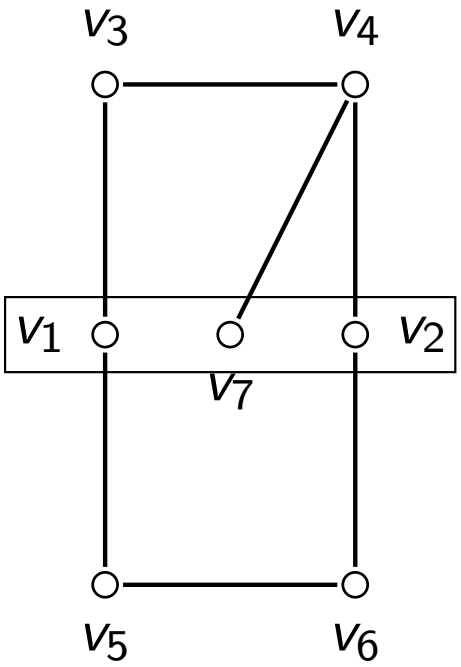
Cluster $+kv$

G is a cluster graph if each of its connected components is a clique (cluster).

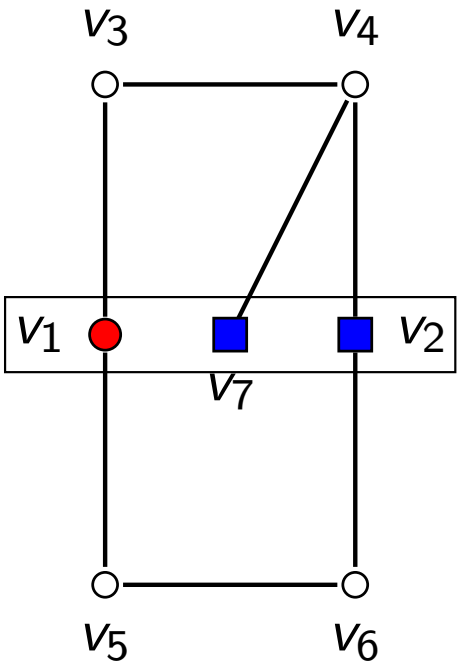


G is a cluster $+kv$ graph if there is a set $U \subset V(G)$ of size k such that $G - U$ is a cluster graph, with clusters $\{C_1, \ldots, C_\ell\}$.

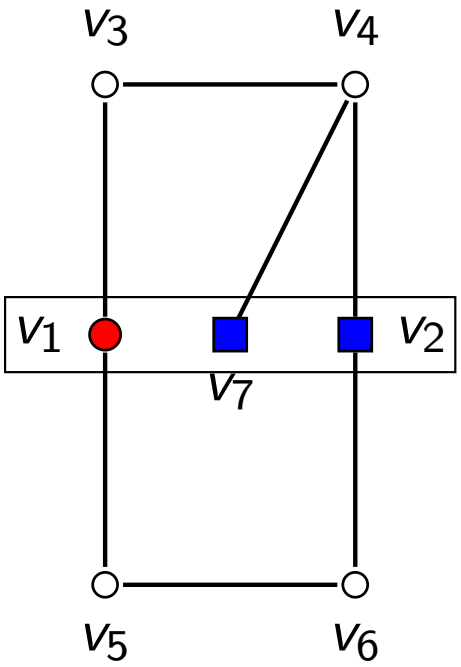
Cluster $+kv$: max-flow again

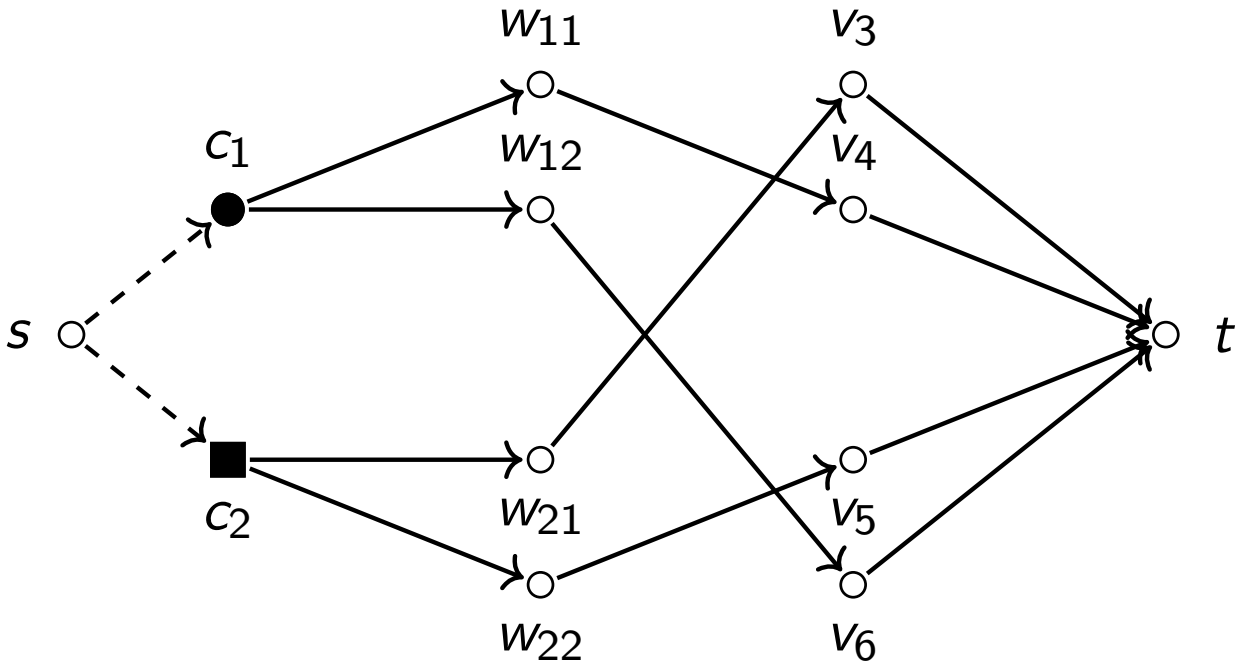


Cluster $+kv$: max-flow again



Cluster $+kv$: max-flow again

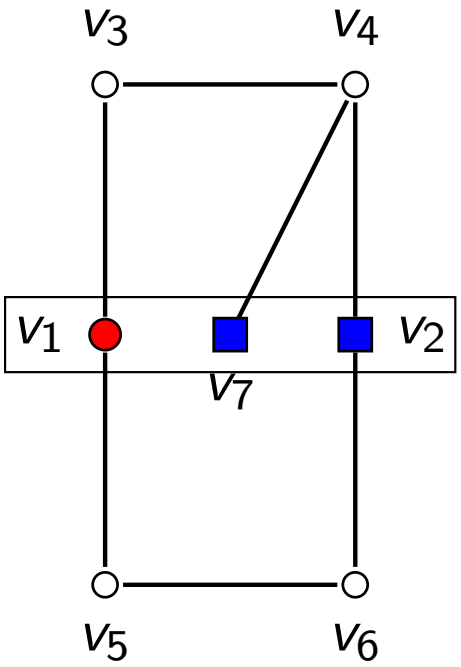


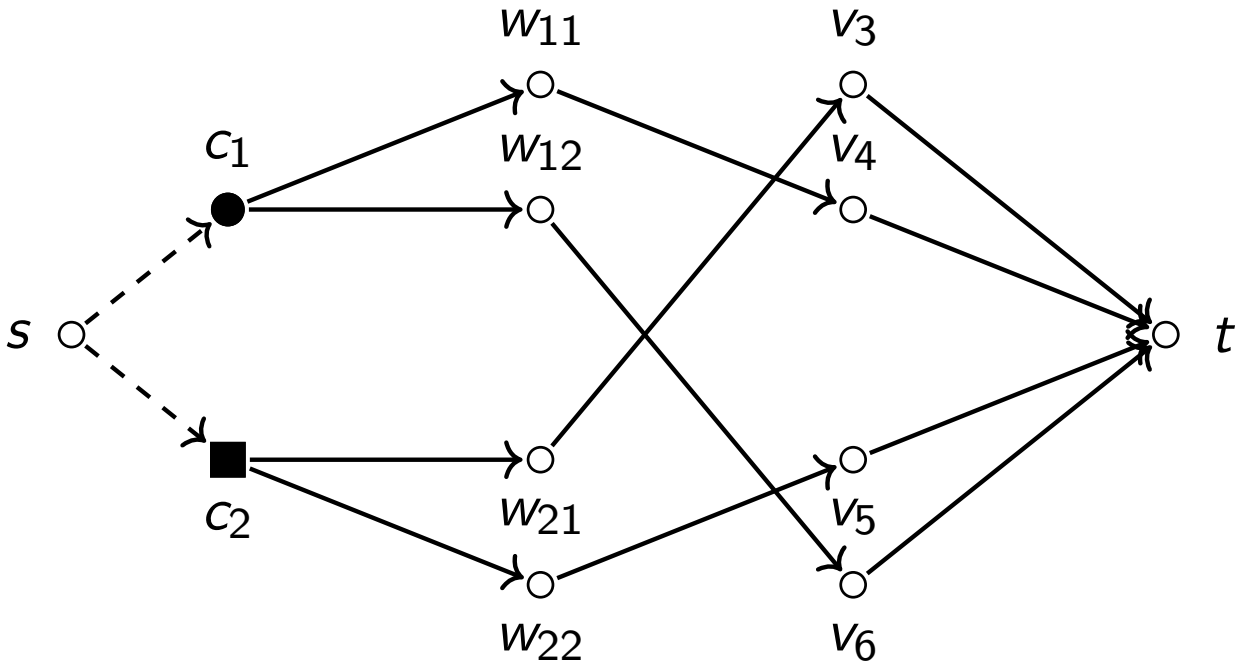


Algorithm

For each of the k^k colorings of U, construct the auxiliary graph.

Cluster $+kv$: max-flow again





Algorithm

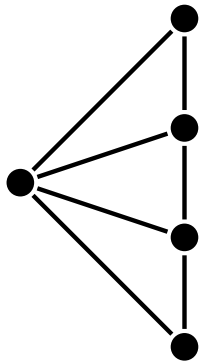
For each of the k^k colorings of U, construct the auxiliary graph. Take into account the #times color i was used in U on the capacity of the (s, c_i) arcs.

Max leaf number

The max leaf number ℓ of a graph G is the maximum number of leaves on a spanning tree of G.

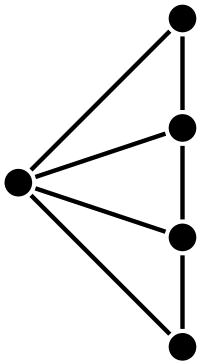
Max leaf number

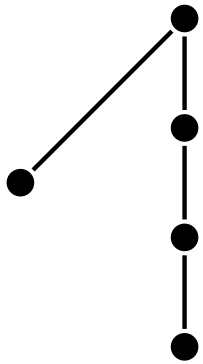
The max leaf number ℓ of a graph G is the maximum number of leaves on a spanning tree of G.



Max leaf number

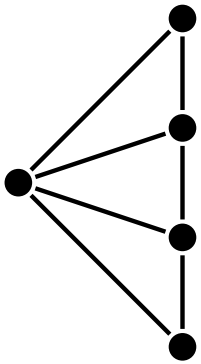
The max leaf number ℓ of a graph G is the maximum number of leaves on a spanning tree of G.

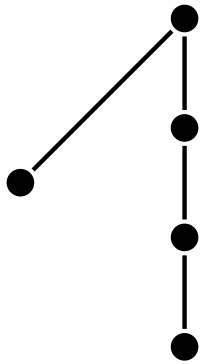


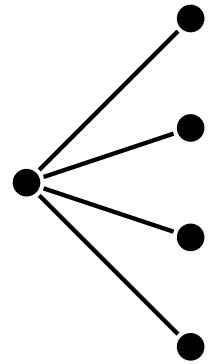


Max leaf number

The max leaf number ℓ of a graph G is the maximum number of leaves on a spanning tree of G.

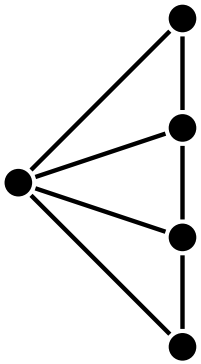
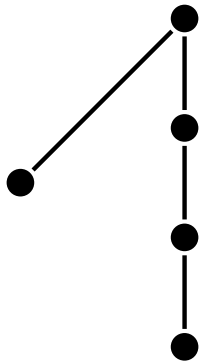
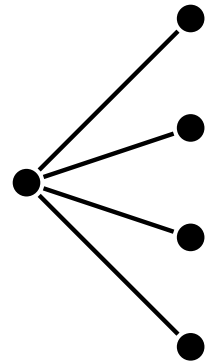






Max leaf number

The max leaf number ℓ of a graph G is the maximum number of leaves on a spanning tree of G.

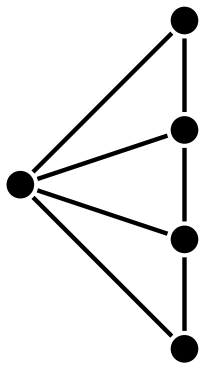
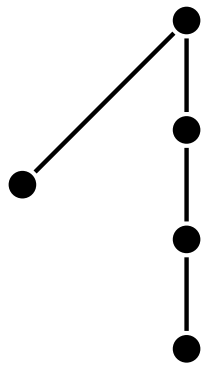
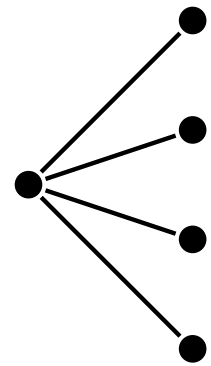




Theorem (Estivill-Castro et al. (2005))

G has max leaf number number ℓ if and only if it is the subdivision of a graph H on 4ℓ vertices.

Max leaf number

The max leaf number ℓ of a graph G is the maximum number of leaves on a spanning tree of G.

Theorem (Estivill-Castro et al. (2005))

G has max leaf number number ℓ if and only if it is the subdivision of a graph H on 4ℓ vertices.

Theorem (Cappelle, G., dos Santos (LAGOS2021?))

$G \setminus H$ is a collection of at most 5.5ℓ paths and contains most vertices of G.

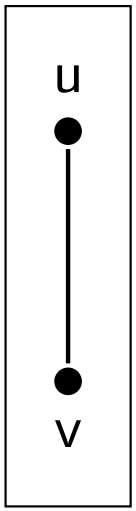
Improving the balance of the force/coloring

Let φ be a k-coloring of H and $\alpha = |\varphi_k| - |\varphi_1|$.

Improving the balance of the force/coloring

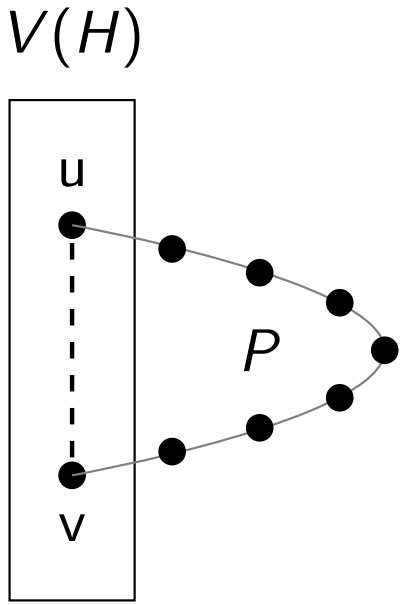
Let φ be a k-coloring of H and $\alpha = |\varphi_k| - |\varphi_1|$.

$V(H)$



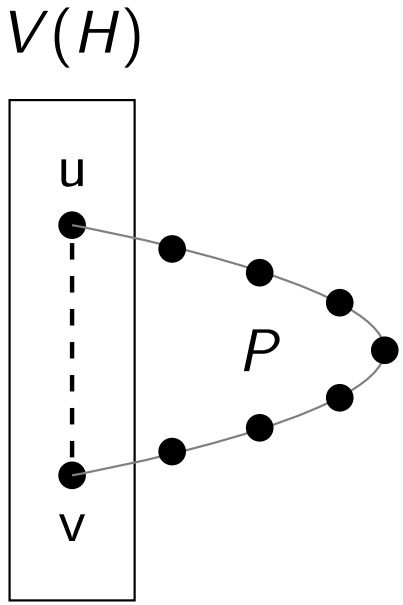
Improving the balance of the force/coloring

Let φ be a k-coloring of H and $\alpha = |\varphi_k| - |\varphi_1|$.



Improving the balance of the force/coloring

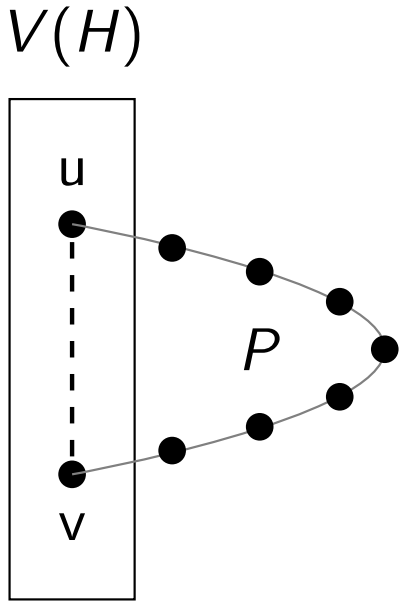
Let φ be a k-coloring of H and $\alpha = |\varphi_k| - |\varphi_1|$.



If $|P| \geq k + 2$, we can extend φ to $H \cup P$ while maintaining it α-balanced or making it 1-balanced if $\alpha = 0$.

Improving the balance of the force/coloring

Let φ be a k-coloring of H and $\alpha = |\varphi_k| - |\varphi_1|$.



If $|P| \geq k + 2$, we can extend φ to $H \cup P$ while maintaining it α-balanced or making it 1-balanced if $\alpha = 0$.

If $|P| \geq k + 2 + x(k + 1)$, we can extend φ to a coloring that is $\alpha - x$ or 1-balanced.

A kernel parameterized by k and ℓ

Let Q be the subgraph of G obtained by removing all paths with $\geq 2k+3$ vertices.

A kernel parameterized by k and ℓ

Let Q be the subgraph of G obtained by removing all paths with $\geq 2k+3$ vertices.

Observation 1

Q has $\mathcal{O}(k\ell)$ vertices: Q is a supergraph of H and at most 5.5ℓ paths with at most $2k+2$ vertices each remain in Q.

A kernel parameterized by k and ℓ

Let Q be the subgraph of G obtained by removing all paths with $\geq 2k+3$ vertices.

Observation 1

Q has $\mathcal{O}(k\ell)$ vertices: Q is a supergraph of H and at most 5.5ℓ paths with at most $2k+2$ vertices each remain in Q.

Observation 2

If $|V(G)| \leq |Q|(k+2)+11\ell(k+1)$, G is already a kernel, so we are done.

A kernel parameterized by k and ℓ

Let Q be the subgraph of G obtained by removing all paths with $\geq 2k+3$ vertices.

Observation 1

Q has $\mathcal{O}(k\ell)$ vertices: Q is a supergraph of H and at most 5.5ℓ paths with at most $2k+2$ vertices each remain in Q.

Observation 2

If $|V(G)| \leq |Q|(k+2) + 11\ell(k+1)$, G is already a kernel, so we are done.

Observation 3

Otherwise, the paths in $G \setminus Q$ satisfy $\sum_{P_i \in G \setminus Q} x_i(k+1) \geq |Q|(k+1)$, so we can bring any coloring of Q to G and make it equitable.

But you promised a kernel on ℓ...

Since H has 4ℓ vertices, the maximum degree of G is $4\ell - 1$.

But you promised a kernel on ℓ...

Since H has 4ℓ vertices, the maximum degree of G is $4\ell - 1$.

Using the Brooks-like Hajnal-Szemerédi theorem, any interesting instance asks for a coloring with at most Δ colors.

But you promised a kernel on ℓ...

Since H has 4ℓ vertices, the maximum degree of G is $4\ell - 1$.

Using the Brooks-like Hajnal-Szemerédi theorem, any interesting instance asks for a coloring with at most Δ colors.

Replacing on Observation 2: G has at most $176\ell^3 + \mathcal{O}\left(\ell^2\right)$ vertices.

Back to the future: Vertex Cover

Known to yield an FPT algorithm for Equitable Coloring and a poly kernel for Vertex Coloring.

Back to the future: Vertex Cover

Known to yield an FPT algorithm for Equitable Coloring and a poly kernel for Vertex Coloring.

Theorem

Equitable Coloring *does not admit a polynomial kernel when parameterized by vertex cover and number of colors, unless* NP $\subseteq$ coNP/poly.

Back to the future: Vertex Cover

Known to yield an FPT algorithm for Equitable Coloring and a poly kernel for Vertex Coloring.

Theorem

Equitable Coloring *does not admit a polynomial kernel when parameterized by vertex cover and number of colors, unless* NP $\subseteq$ coNP/poly.

The proof is by a (way too technical) OR-cross-composition from Multicolored Clique: given H and a partition $\mathcal{V}$ of $V(H)$, find a clique with one vertex in each part of $\mathcal{V}$.

Back to the future: Vertex Cover

Known to yield an FPT algorithm for Equitable Coloring and a poly kernel for Vertex Coloring.

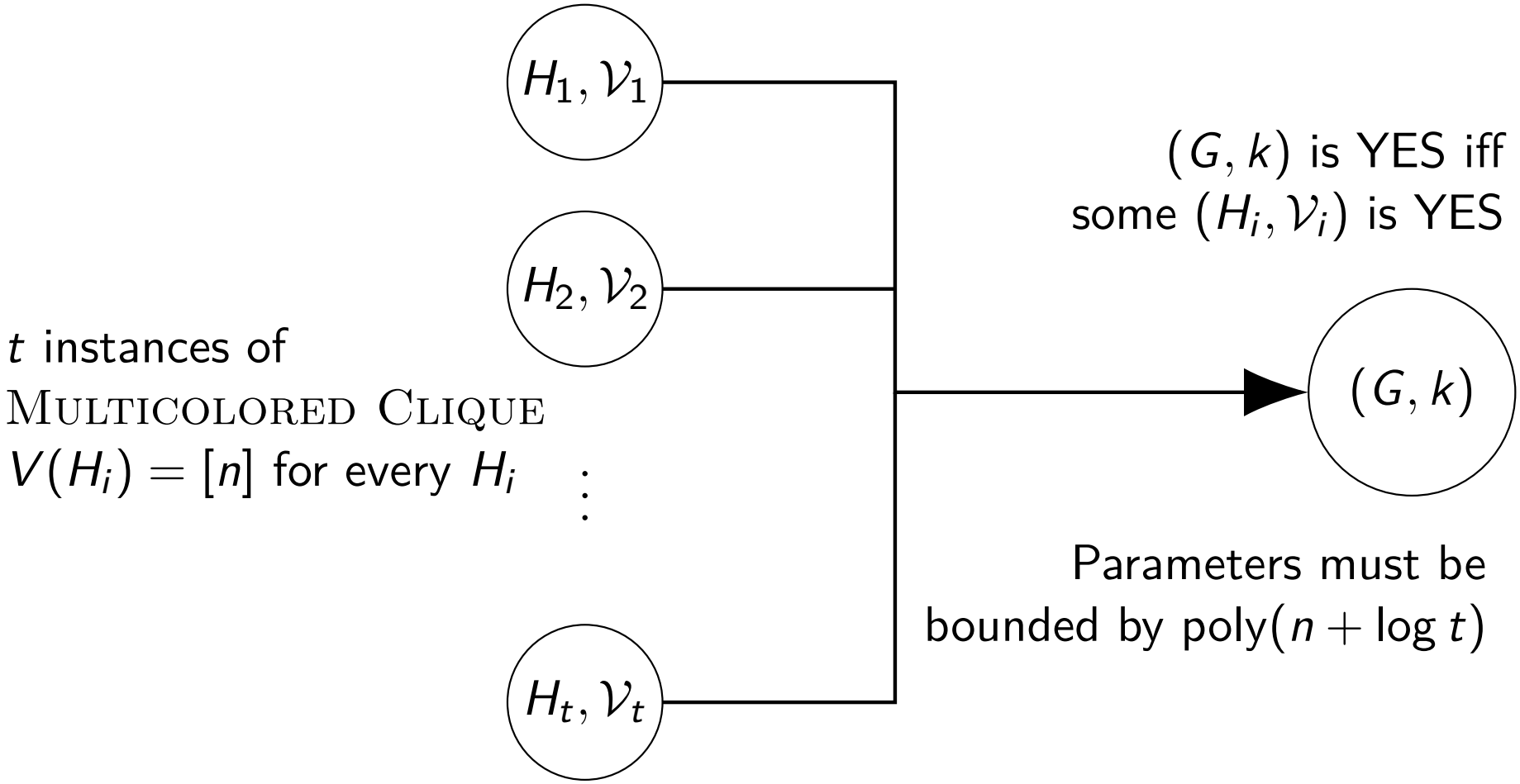
Theorem

Equitable Coloring *does not admit a polynomial kernel when parameterized by vertex cover and number of colors, unless* NP $\subseteq$ coNP/poly.

The proof is by a (way too technical) OR-cross-composition from Multicolored Clique: given H and a partition $\mathcal{V}$ of $V(H)$, find a clique with one vertex in each part of $\mathcal{V}$.

But lets try to get some very high level intuition.

OR-cross-composition



OR-cross-composition

Key idea 1: Overlay the gadgets for each $(H_i, \mathcal{V}_i)$: 1 gadget belongs to many instances.

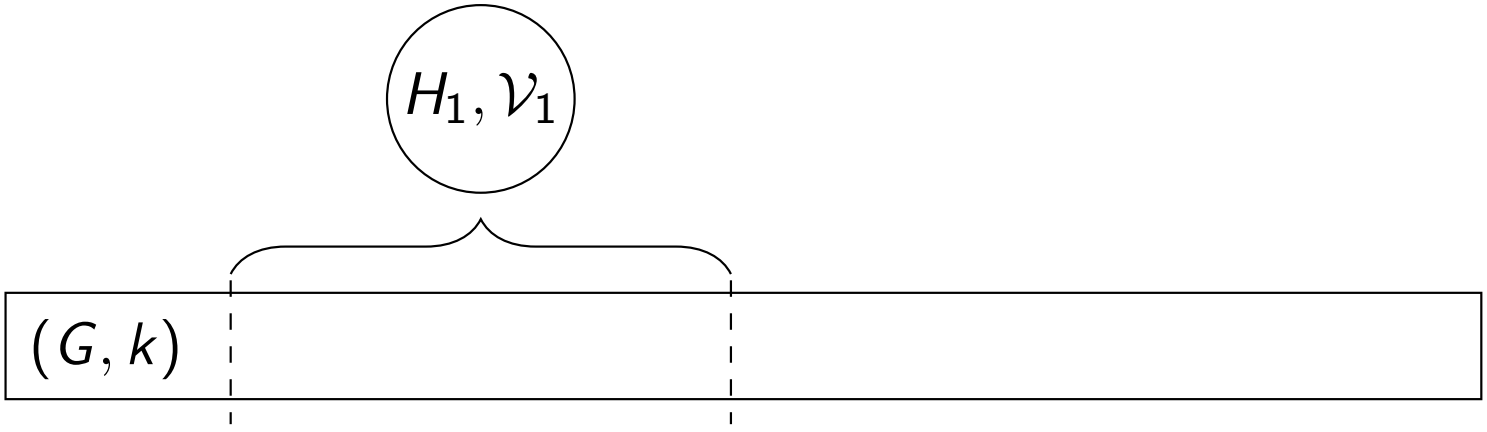
OR-cross-composition

Key idea 1: Overlay the gadgets for each $(H_i, \mathcal{V}_i)$: 1 gadget belongs to many instances.

(G, k)

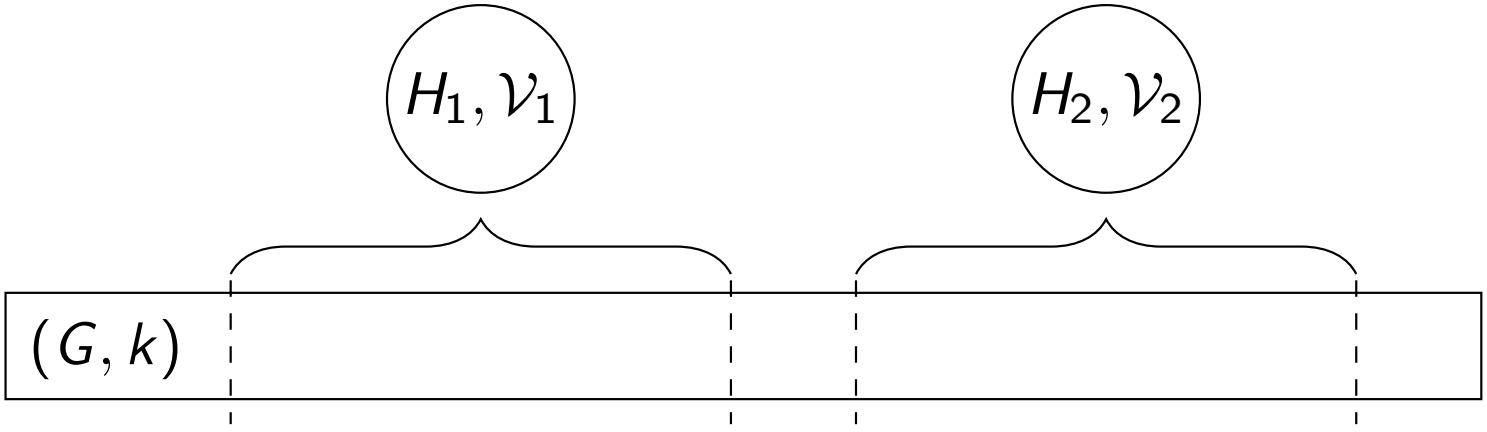
OR-cross-composition

Key idea 1: Overlay the gadgets for each $(H_i, \mathcal{V}_i)$: 1 gadget belongs to many instances.



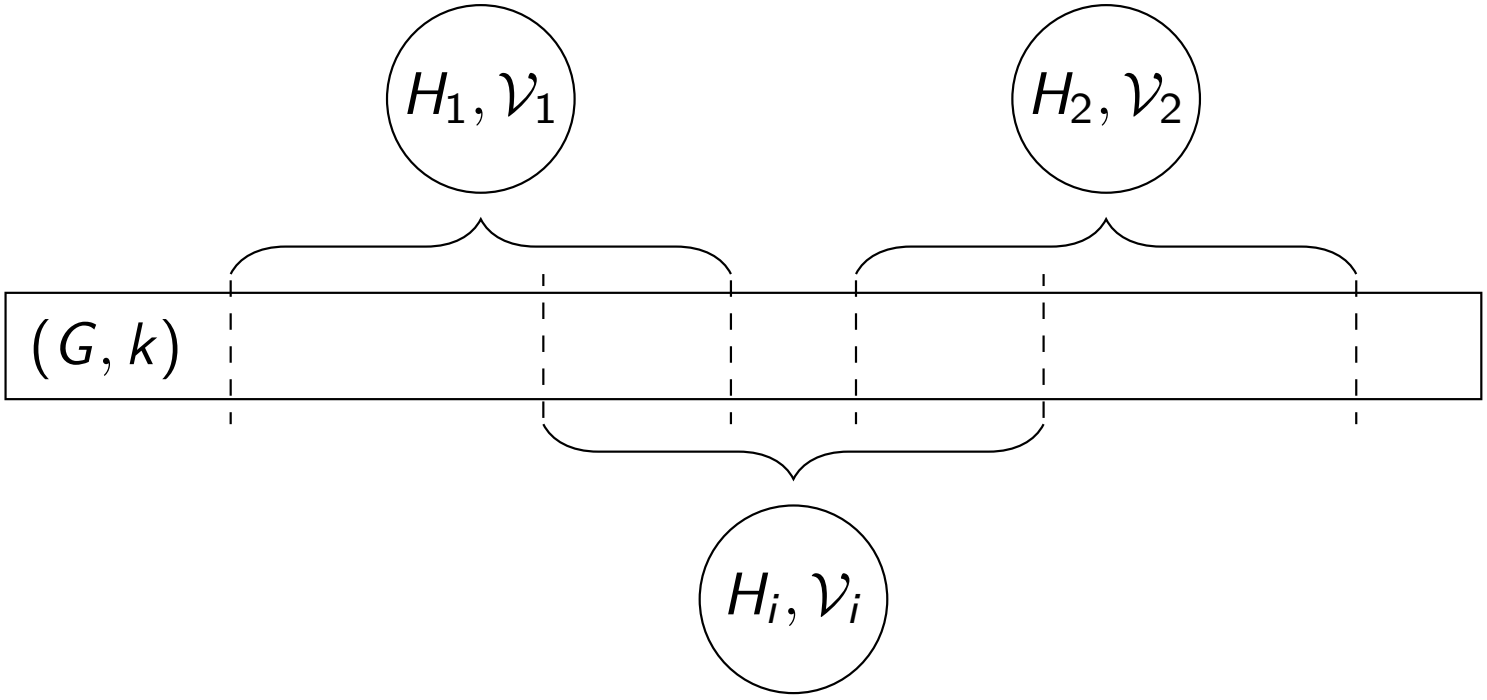
OR-cross-composition

Key idea 1: Overlay the gadgets for each $(H_i, \mathcal{V}_i)$: 1 gadget belongs to many instances.



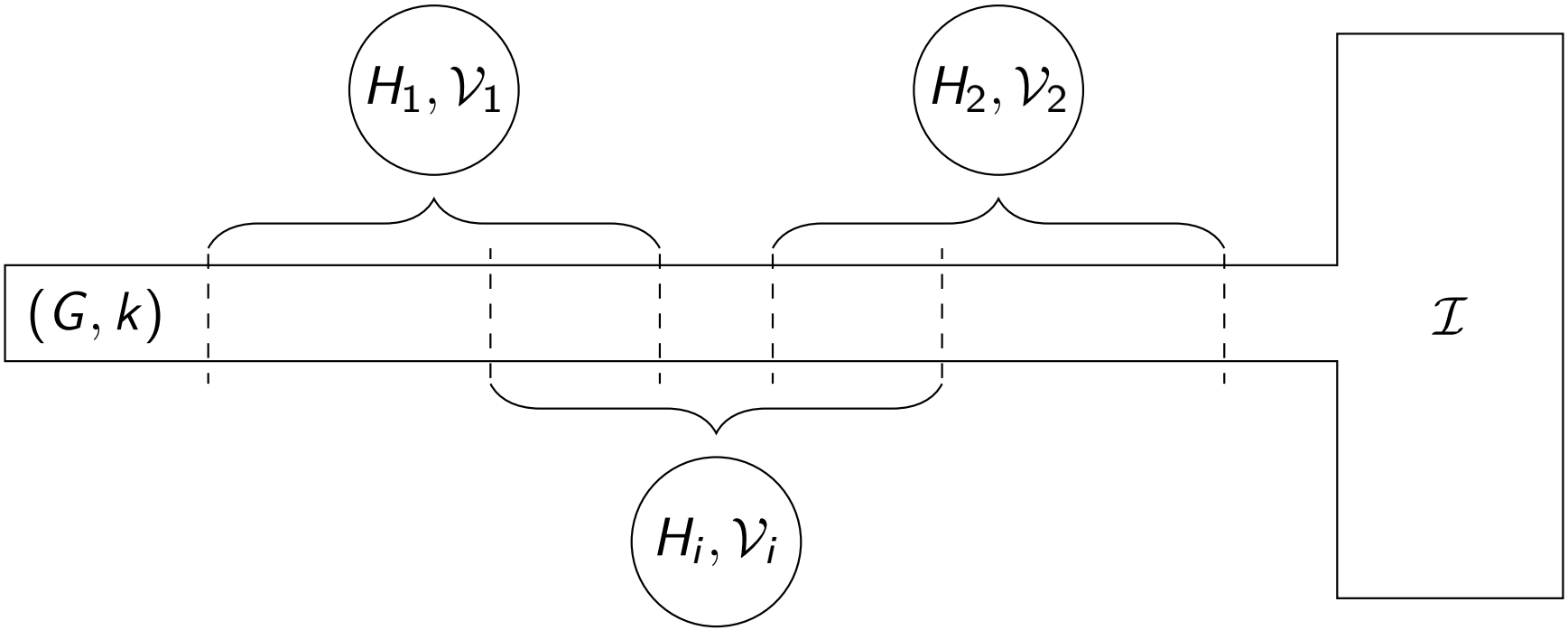
OR-cross-composition

Key idea 1: Overlay the gadgets for each $(H_i, \mathcal{V}_i)$: 1 gadget belongs to many instances.



OR-cross-composition

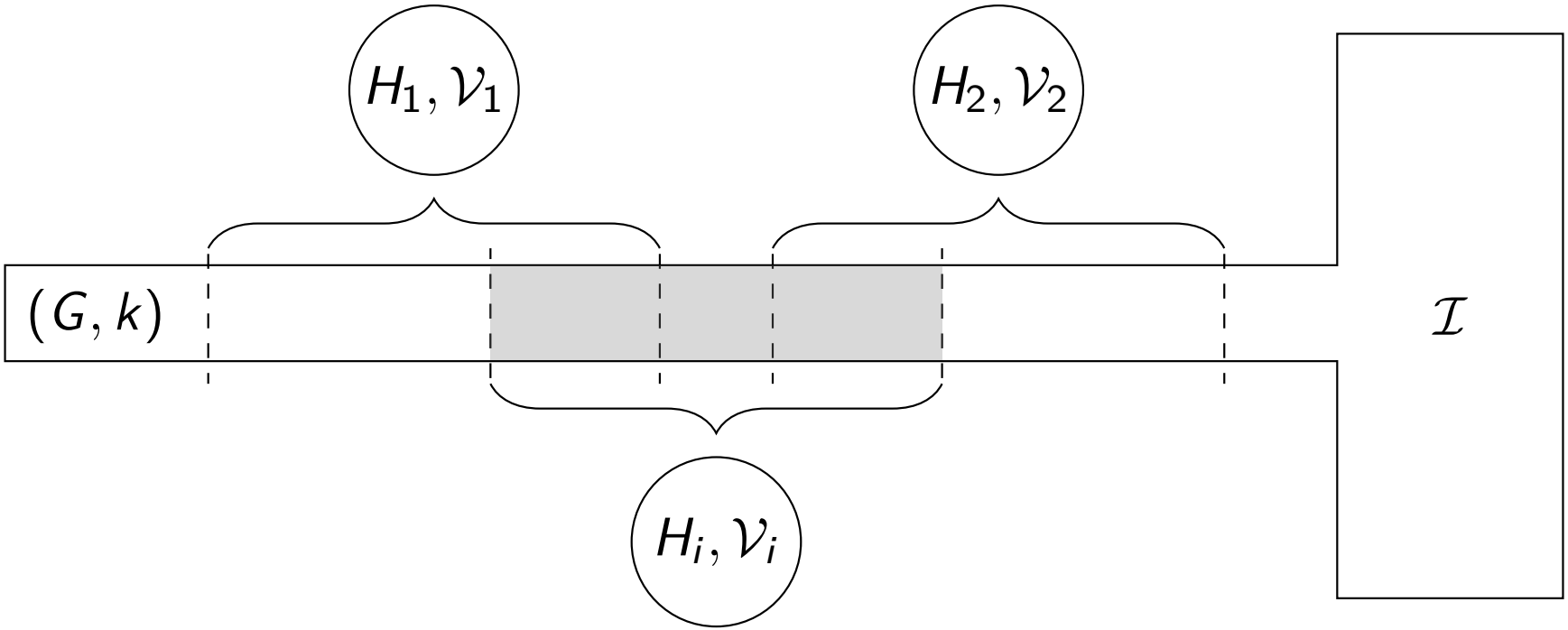
Key idea 1: Overlay the gadgets for each $(H_i, \mathcal{V}_i)$: 1 gadget belongs to many instances.



Key idea 2: Use an instance selector gadget $\mathcal{I}$ to isolate the gadgets of $G \setminus \mathcal{I}$ that encode the instance with the solution.

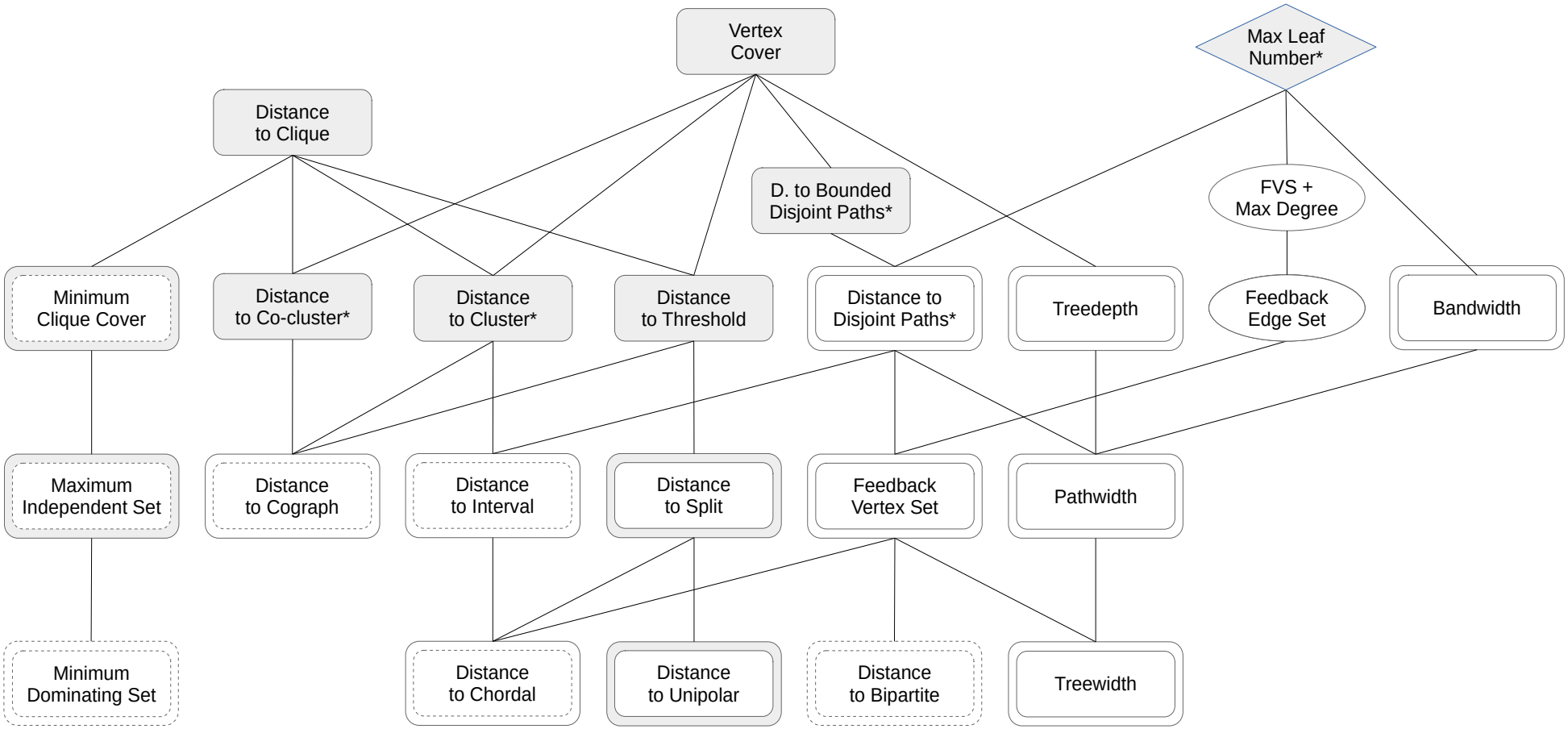
OR-cross-composition

Key idea 1: Overlay the gadgets for each $(H_i, \mathcal{V}_i)$: 1 gadget belongs to many instances.



Key idea 2: Use an instance selector gadget $\mathcal{I}$ to isolate the gadgets of $G \setminus \mathcal{I}$ that encode the instance with the solution.

Parameterized landscape



Open Problems

- Can we find a subcubic kernel when parameterizing by the max leaf number?
- Using ETH, can we prove that currently known algorithms have optimal running time?
- Is Equitable Coloring FPT when parameterized by feedback edge set?

Thank you!