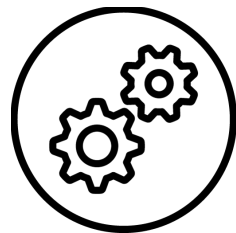

Parameterized Graph Modification: A Modern Perspective

Neeldhara Misra

Department of Computer Science and Automation,
Indian Institute of Science

Parameterized Graph Modification: A Modern Perspective

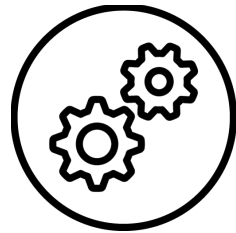
Parameterized Graph Modification: A Modern Perspective



The General Setup

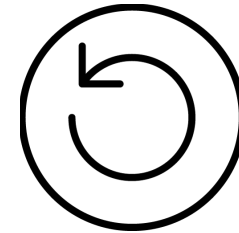
A brief introduction to parameterized algorithmics

Parameterized Graph Modification: A Modern Perspective



The General Setup

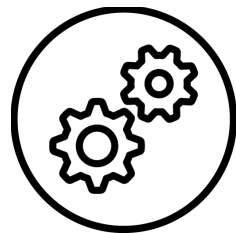
A brief introduction to parameterized algorithmics



One Size Fits All?

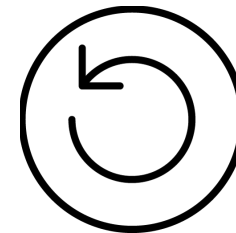
A meta-algorithm for a class of graph-theoretic problems

Parameterized Graph Modification: A Modern Perspective



The General Setup

A brief introduction to parameterized algorithmics

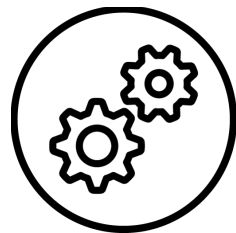


One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

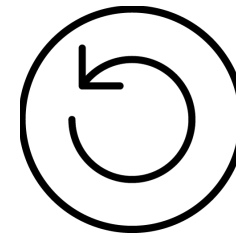
The general notion of a “graph modification problem”.

Parameterized Graph Modification: A Modern Perspective



The General Setup

A brief introduction to parameterized algorithmics



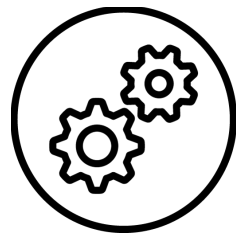
One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

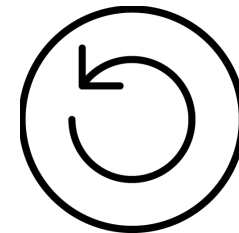
Implications of the Graph Minor Theorem.

Parameterized Graph Modification: A Modern Perspective



The General Setup

A brief introduction to parameterized algorithmics



One Size Fits All?

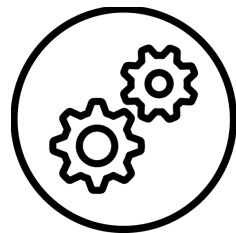
A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

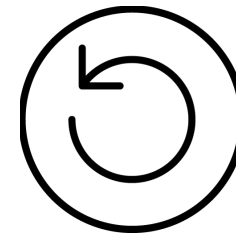
Vertex Cover, Revisited.

Parameterized Graph Modification: A Modern Perspective



The General Setup

A brief introduction to parameterized algorithmics



One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

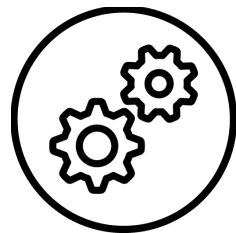
The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

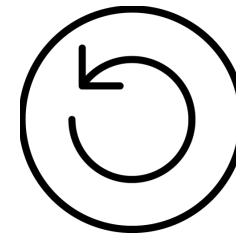
Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

Parameterized Graph Modification: A Modern Perspective



The General Setup

A brief introduction to parameterized algorithmics



One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

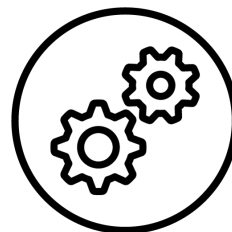
Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

Planar F-Deletion.

Parameterized Graph Modification: A Modern Perspective

The General Setup

A brief introduction to parameterized algorithmics



The General Setup

A brief introduction to parameterized algorithms

The General Setup

A brief introduction to parameterized algorithms

The Classical Perspective

The General Setup

A brief introduction to parameterized algorithms

The Classical Perspective

Evaluate running times of algorithms as a function of the size of input alone.

The General Setup

A brief introduction to parameterized algorithmics

The Classical Perspective

Evaluate running times of algorithms as a function of the size of input alone.

The Parameterized Perspective

The General Setup

A brief introduction to parameterized algorithmics

The Classical Perspective

Evaluate running times of algorithms as a function of the size of input alone.

The Parameterized Perspective

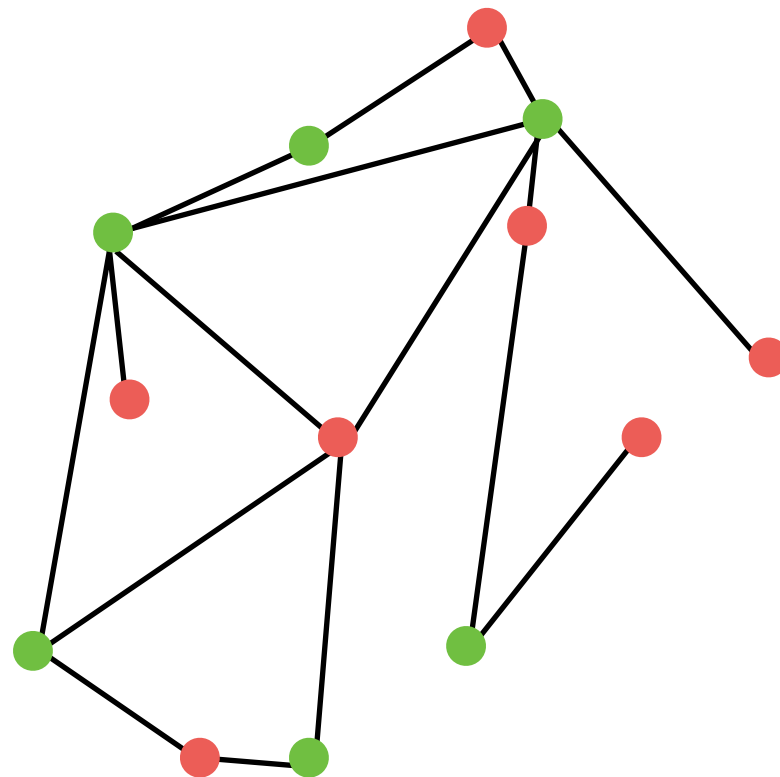
Evaluate running times of algorithms as a function of the size of input **and** additional structural parameters.

Vertex Cover

Does there exist a subset S of at most k vertices whose removal makes the graph independent?

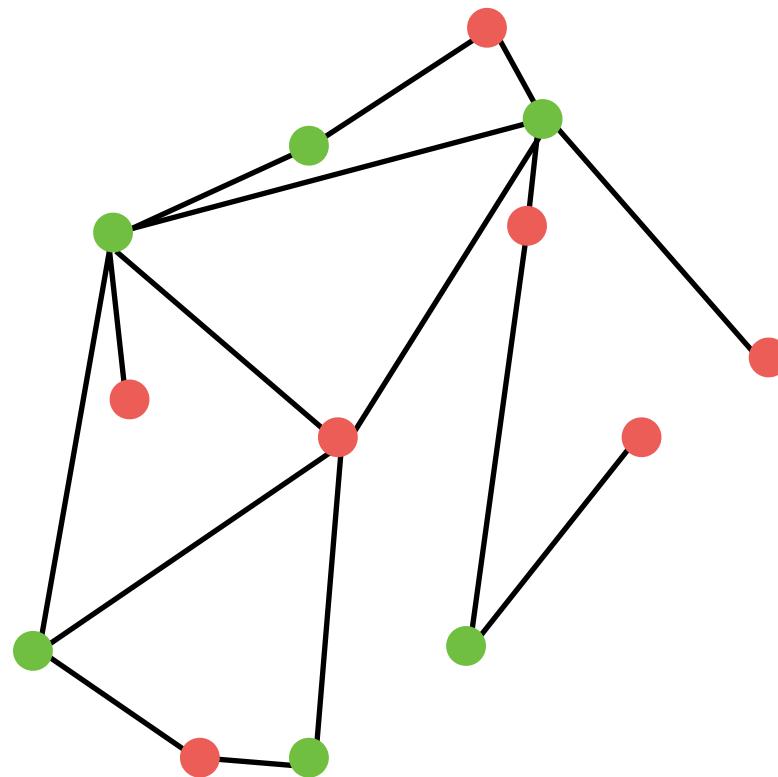
Vertex Cover

Does there exist a subset S of at most k vertices whose removal makes the graph independent?



Vertex Cover

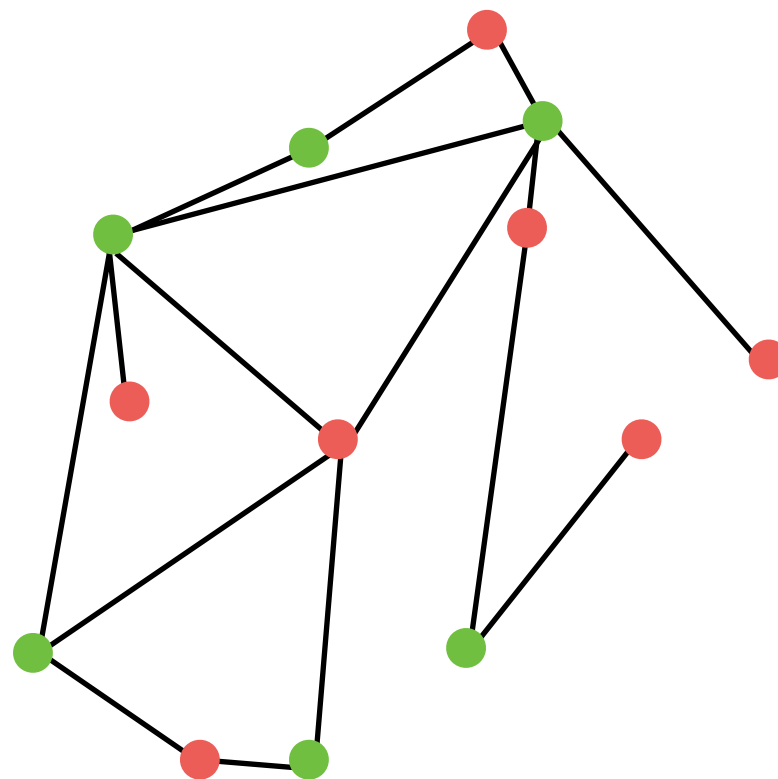
Does there exist a subset S of at most k vertices whose removal makes the graph independent?



Brute Force

Vertex Cover

Does there exist a subset S of at most k vertices whose removal makes the graph independent?



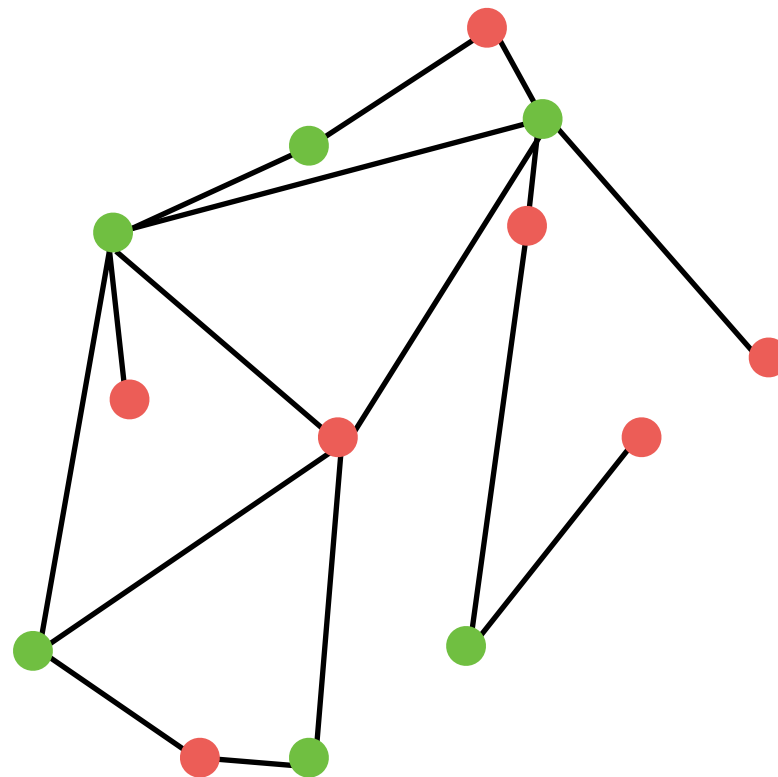
Brute Force

$$n^k$$

Vertex Cover

Does there exist a subset S of at most k vertices whose removal makes the graph independent?

Pruned Search Tree



Brute Force

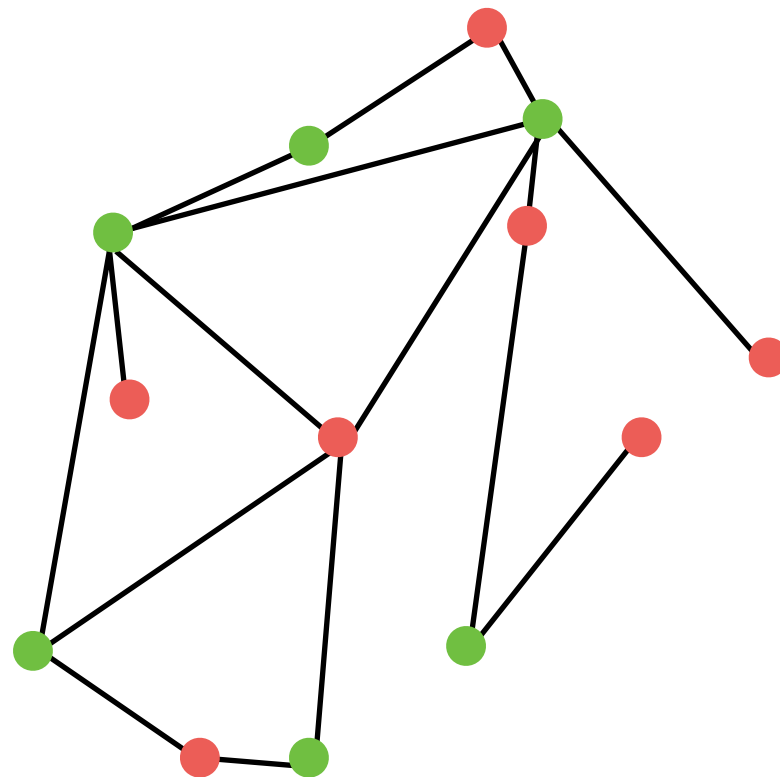
$$n^k$$

Vertex Cover

Does there exist a subset S of at most k vertices whose removal makes the graph independent?

Pruned Search Tree

$$2^k n^{O(1)}$$



Brute Force

$$n^k$$

Vertex Cover

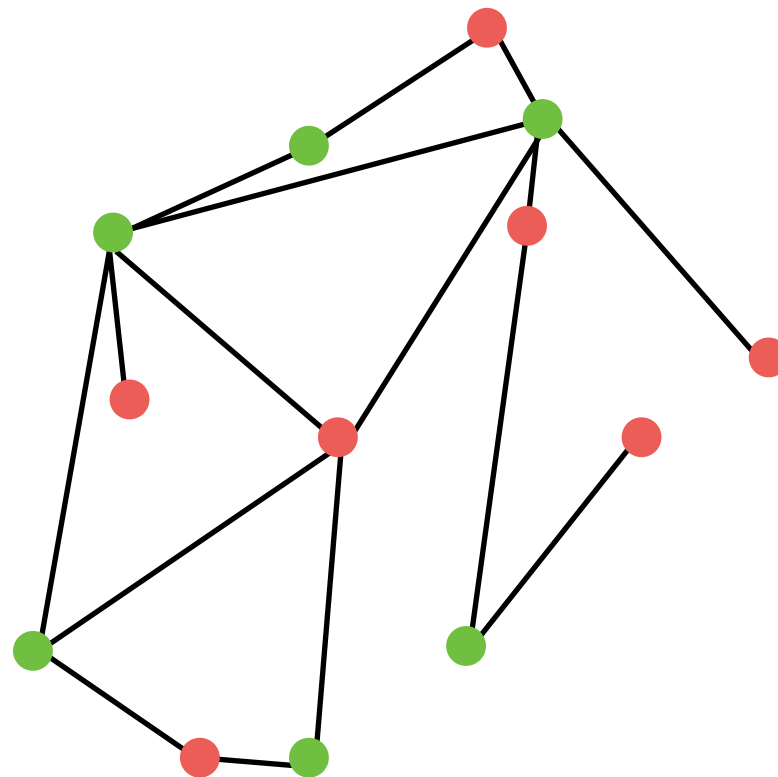
Does there exist a subset S of at most k vertices whose removal makes the graph independent?

Pruned Search Tree

$$2^k n^{O(1)}$$

Brute Force

$$n^k$$



(Implemented by CBRG, ETH Zurich)

Exploiting Structure

For real-world computations, we always know more about the problem.

Exploiting Structure

For real-world computations, we always know more about the problem.

Small Diameter

Exploiting Structure

For real-world computations, we always know more about the problem.

Small Diameter

Planar graphs

Exploiting Structure

For real-world computations, we always know more about the problem.

Small Diameter

Planar graphs

Small Nesting depth

Exploiting Structure

For real-world computations, we always know more about the problem.

Small Diameter

Planar graphs

Small Nesting depth

Graphs without cycles

Exploiting Structure

Acknowledge the presence of additional structure.

Exploiting Structure

Acknowledge the presence of additional structure.

This will be a **secondary measurement**, a parameter.

Exploiting Structure

Acknowledge the presence of additional structure.

This will be a **secondary measurement**, a parameter.

Design algorithms that restrict the **combinatorial explosion** to a **function of the parameter**.

Fixed-parameter Tractable Algorithms

$$f(k)p(n)$$

Fixed-parameter Tractable Algorithms

a structural parameter

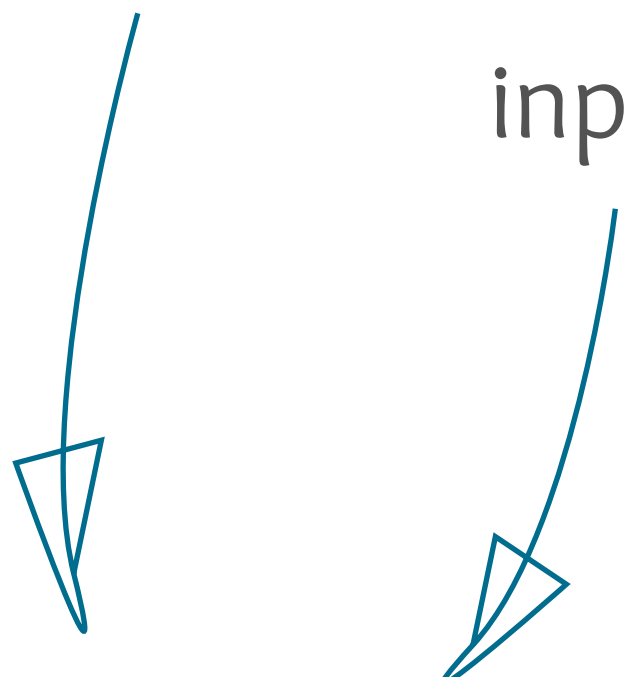


$$f(k)p(n)$$

Fixed-parameter Tractable Algorithms

a structural parameter

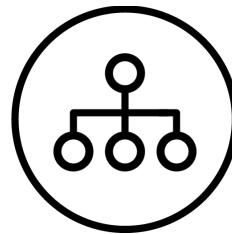
input size


$$f(k)p(n)$$

Parameterized Graph Modification: A Modern Perspective

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems



One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

Planar F-Deletion.

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

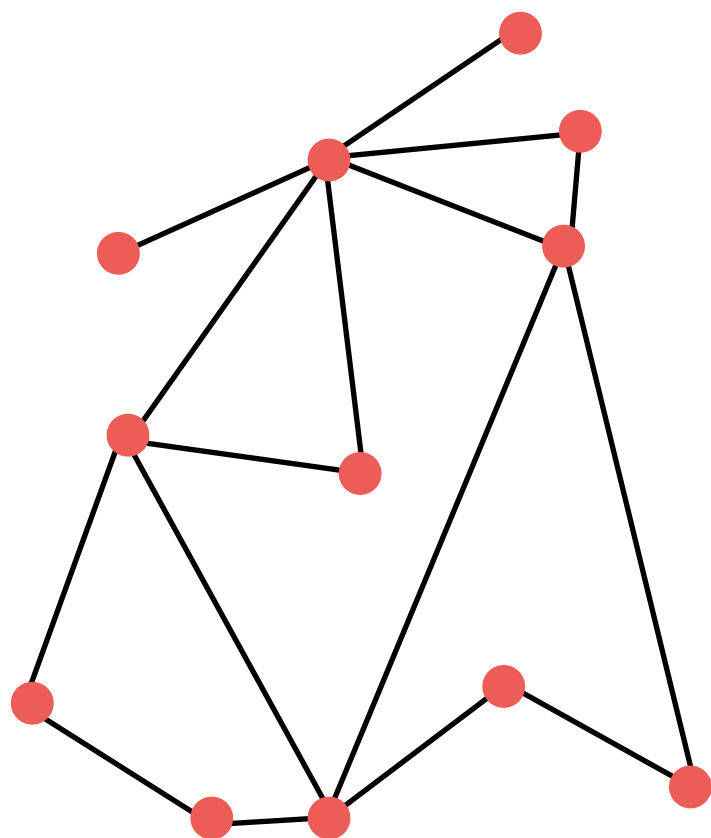
The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

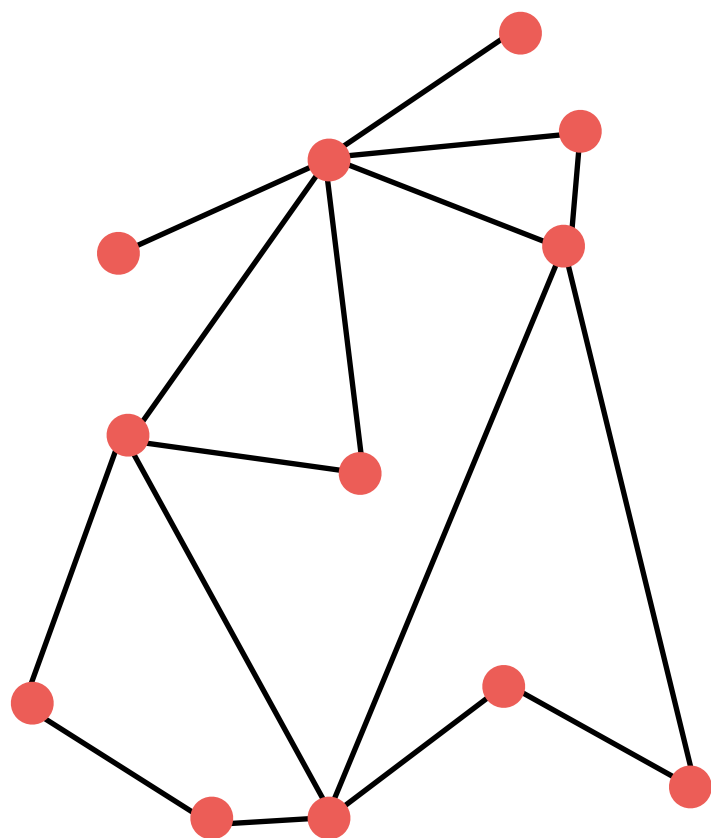
Vertex Cover, Revisited.

Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

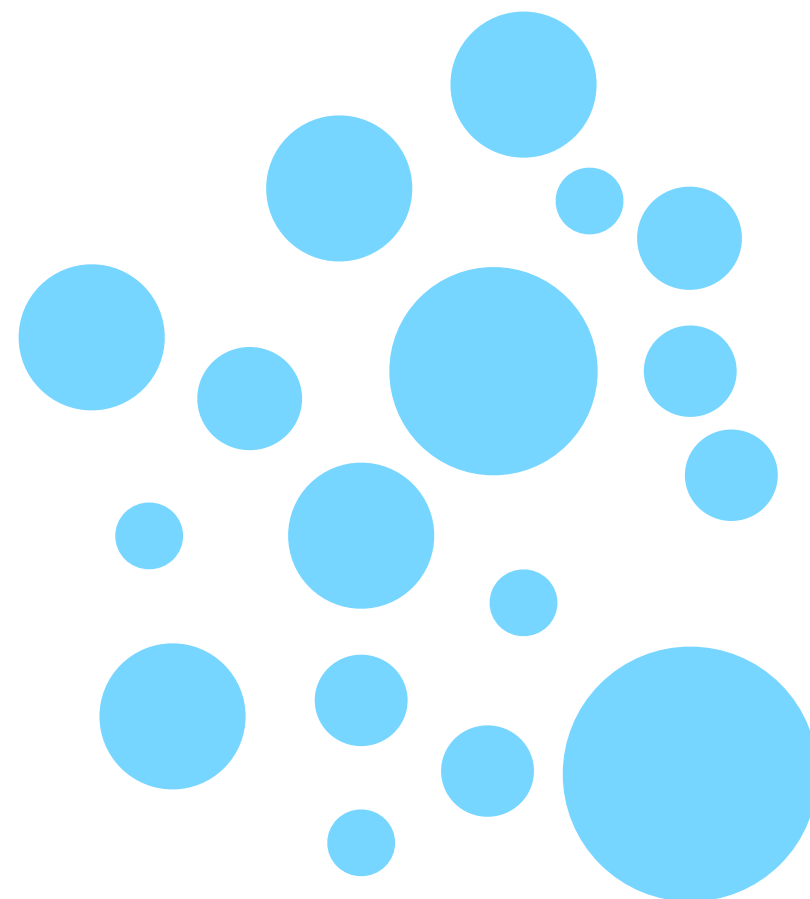
Planar F-Deletion.



Input Graph, G

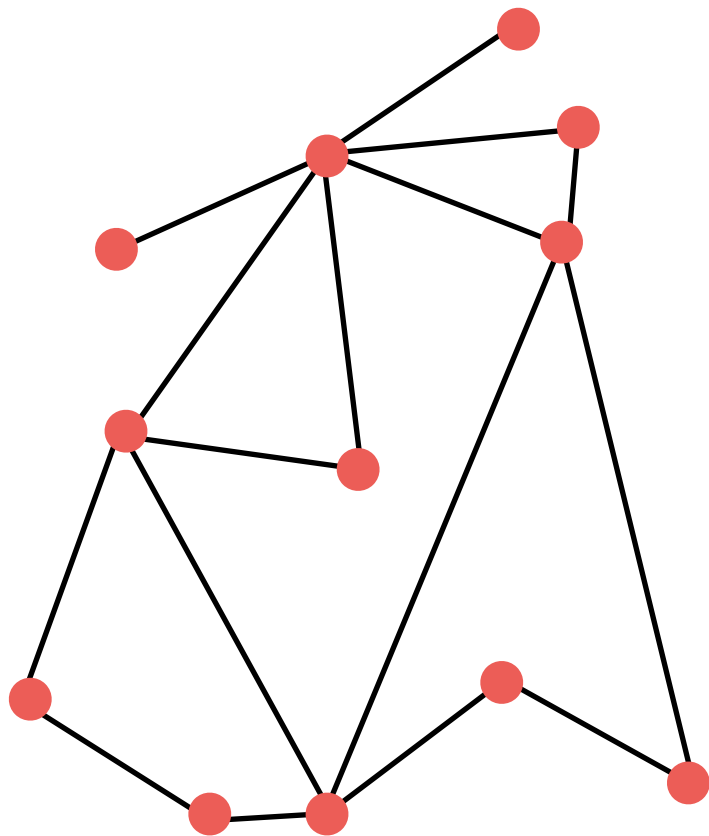


Input Graph, G

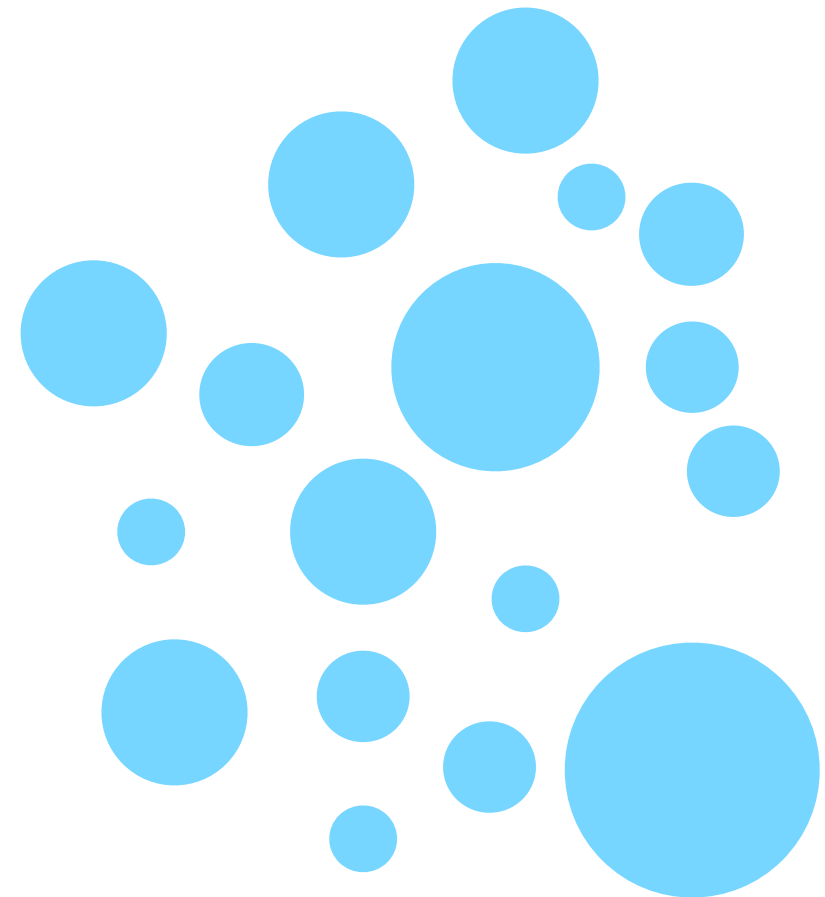


Graph Class or "property", π

Goal: Transform G so that it belongs to Π with "minimum damages".

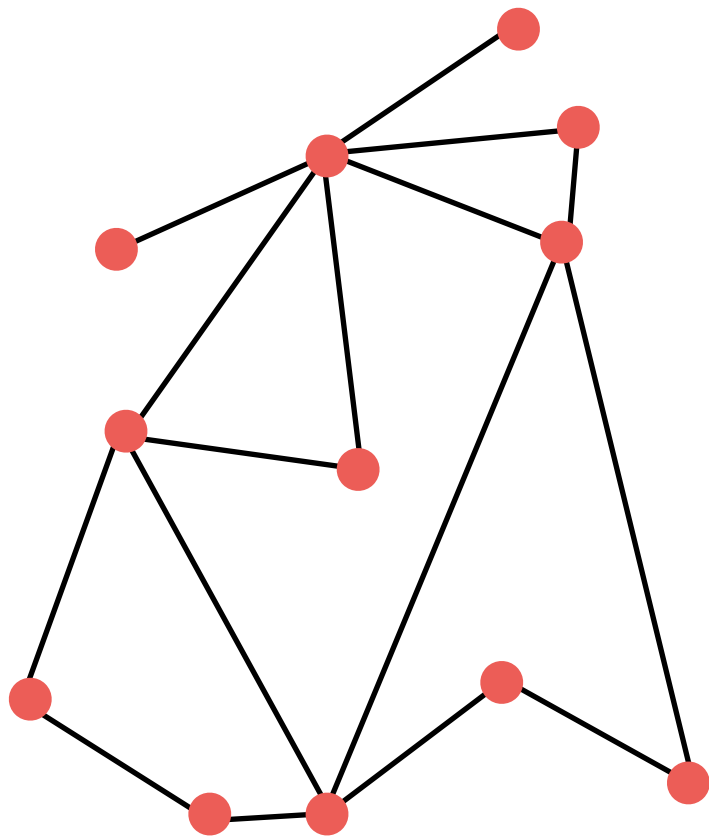


Input Graph, G

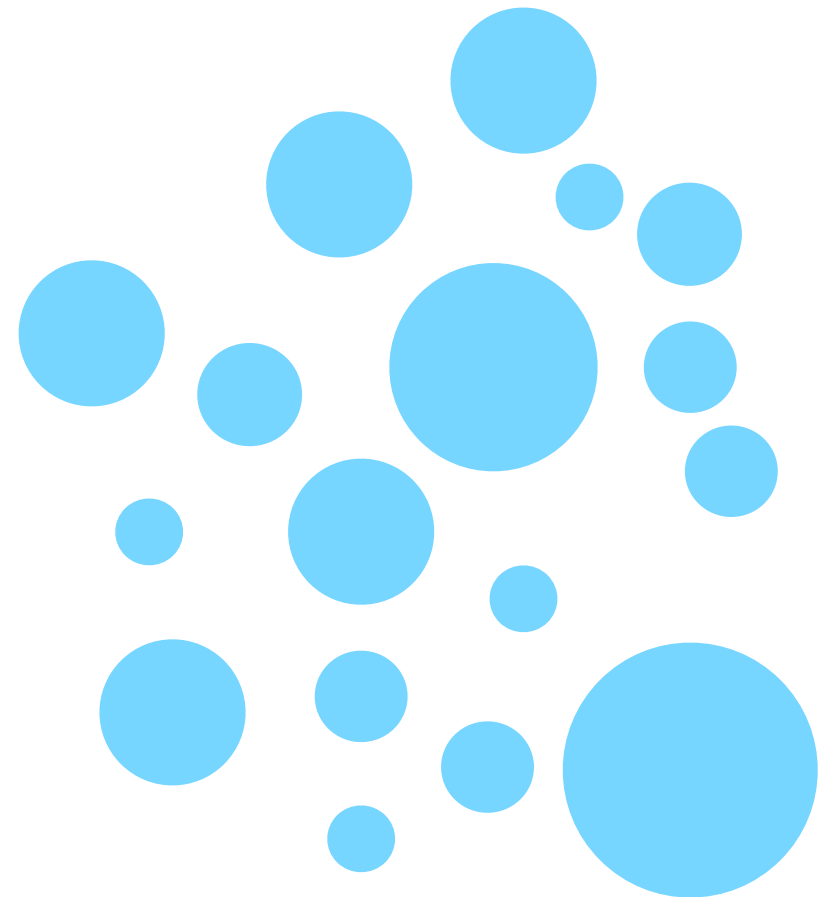


Graph Class or "property", Π

Goal: Transform G so that it belongs to Π with "minimum damages".



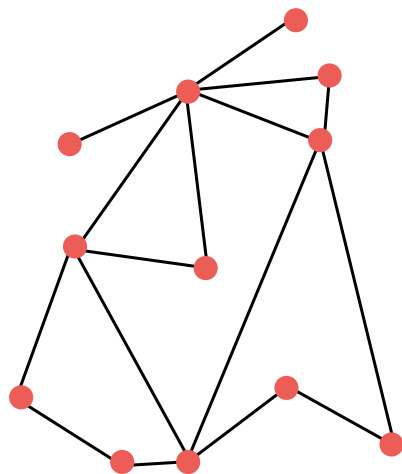
Input Graph, G



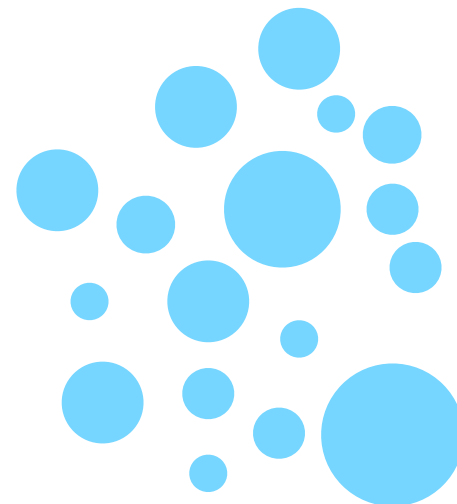
Graph Class or "property", Π

Goal: Transform G so that it belongs to Π with "minimum damages".

Input Graph, G



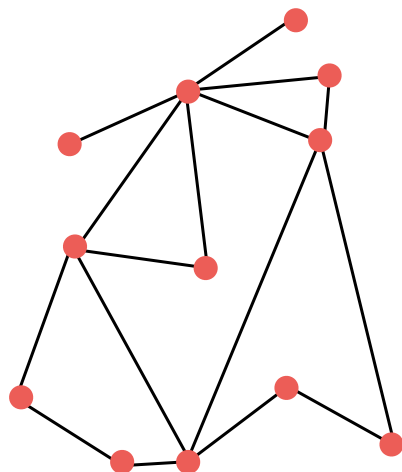
Graph Class or "property", Π



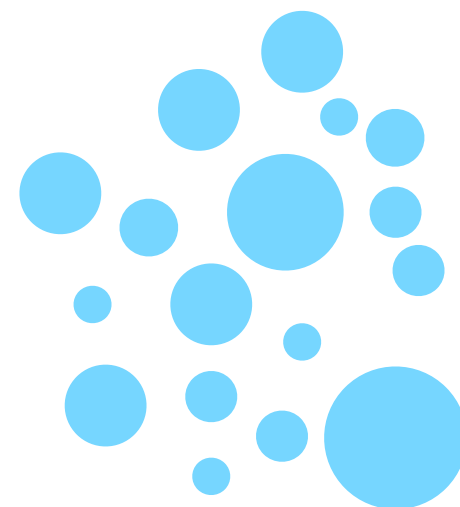
Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions

Input Graph, G



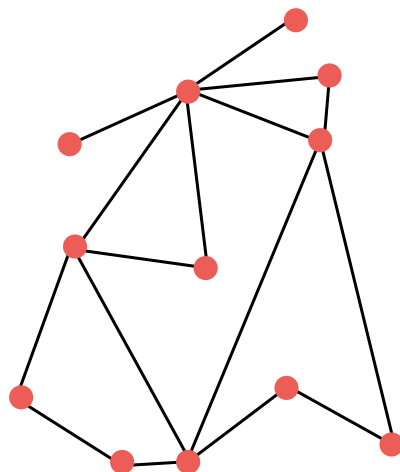
Graph Class or "property", Π



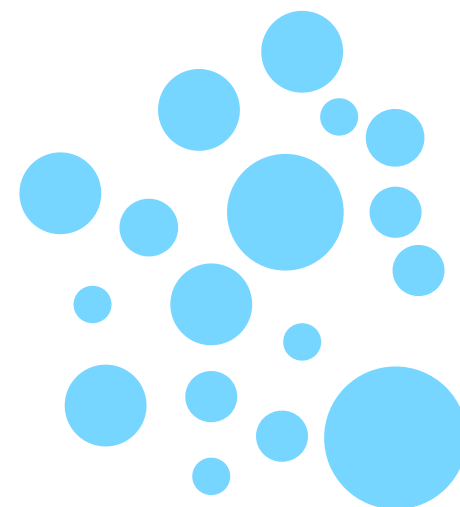
Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions edge deletions

Input Graph, G



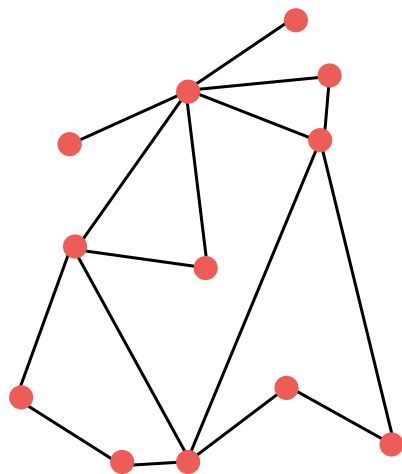
Graph Class or "property", Π



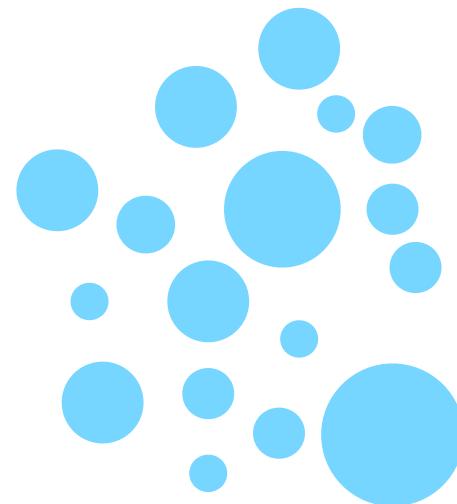
Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions edge deletions edge additions

Input Graph, G



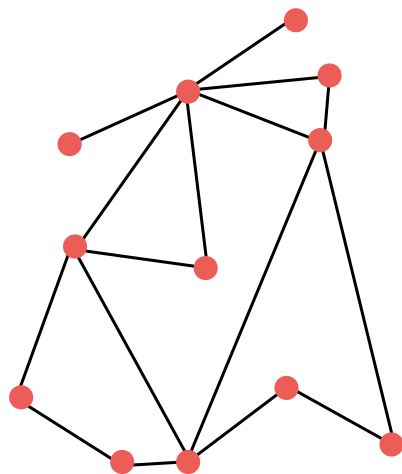
Graph Class or "property", Π



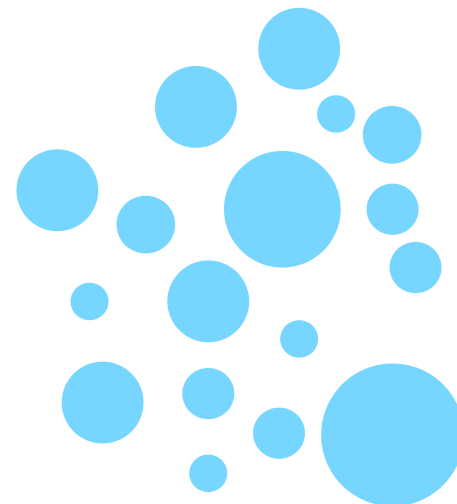
Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions edge deletions edge additions edge contractions

Input Graph, G



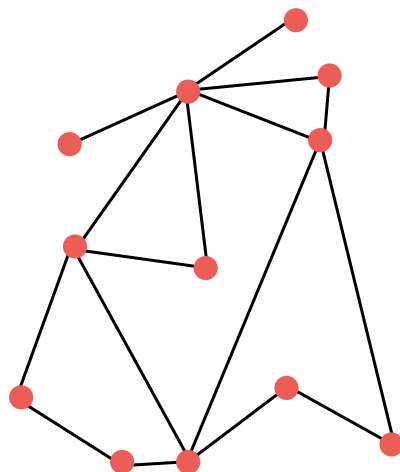
Graph Class or "property", Π



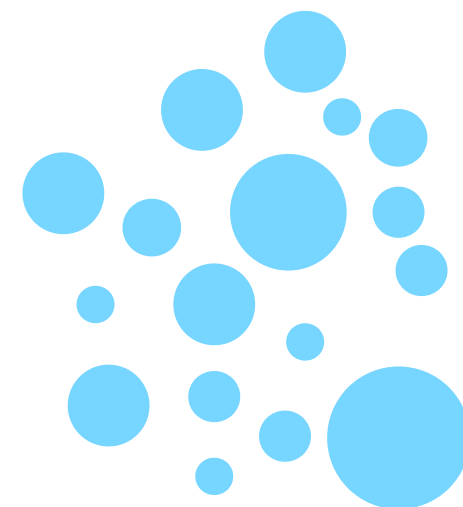
Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions edge deletions edge additions edge contractions edge editing

Input Graph, G



Graph Class or "property", Π



Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions

edge deletions

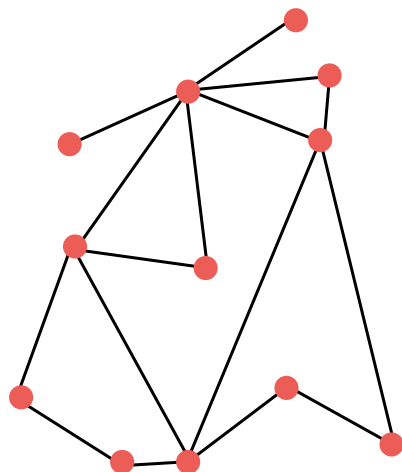
edge additions

edge contractions

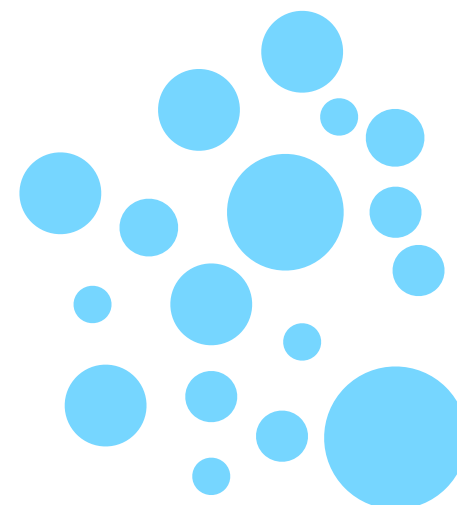
edge editing

VERTEX COVER

Input Graph, G



Edgeless Graphs



Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions

edge deletions

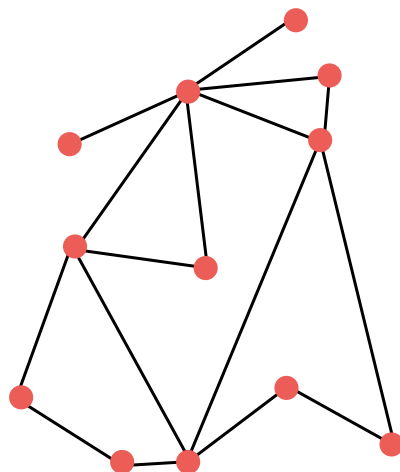
edge additions

edge contractions

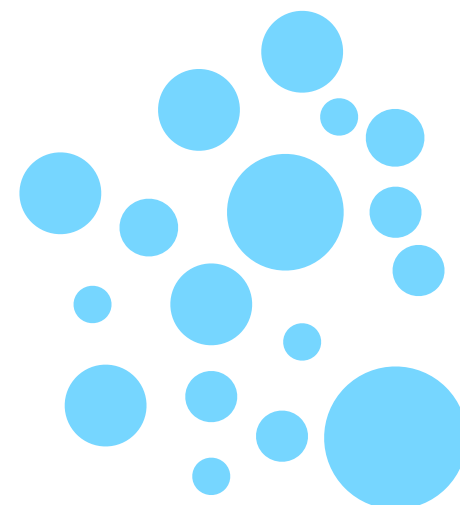
edge editing

FEEDBACK VERTEX SET

Input Graph, G



Acyclic Graphs



Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions

edge deletions

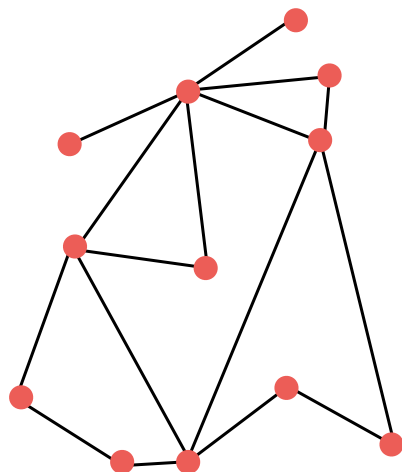
edge additions

edge contractions

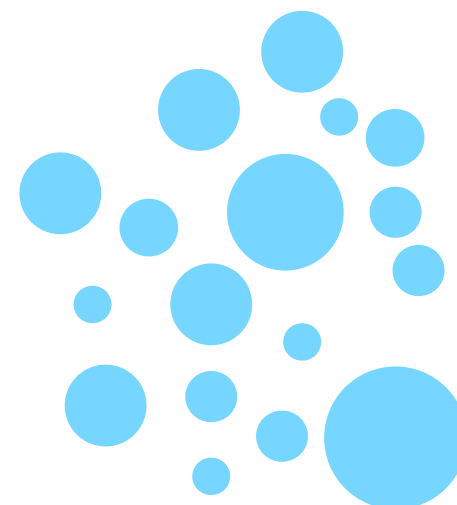
edge editing

MINIMUM FILL-IN

Input Graph, G



Chordal Graphs



Goal: Transform G so that it belongs to Π with "minimum damages".

vertex deletions

edge deletions

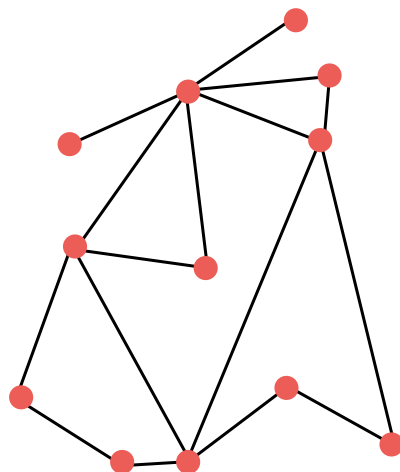
edge additions

edge contractions

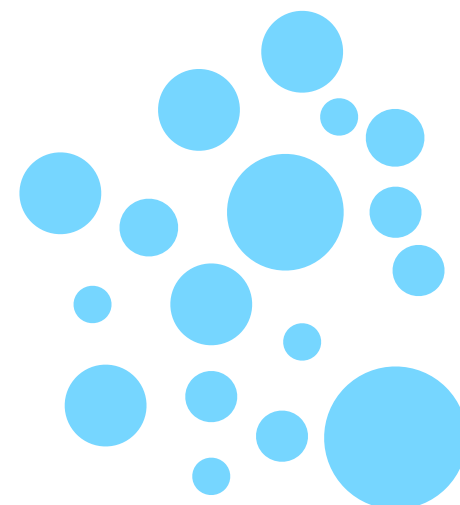
edge editing

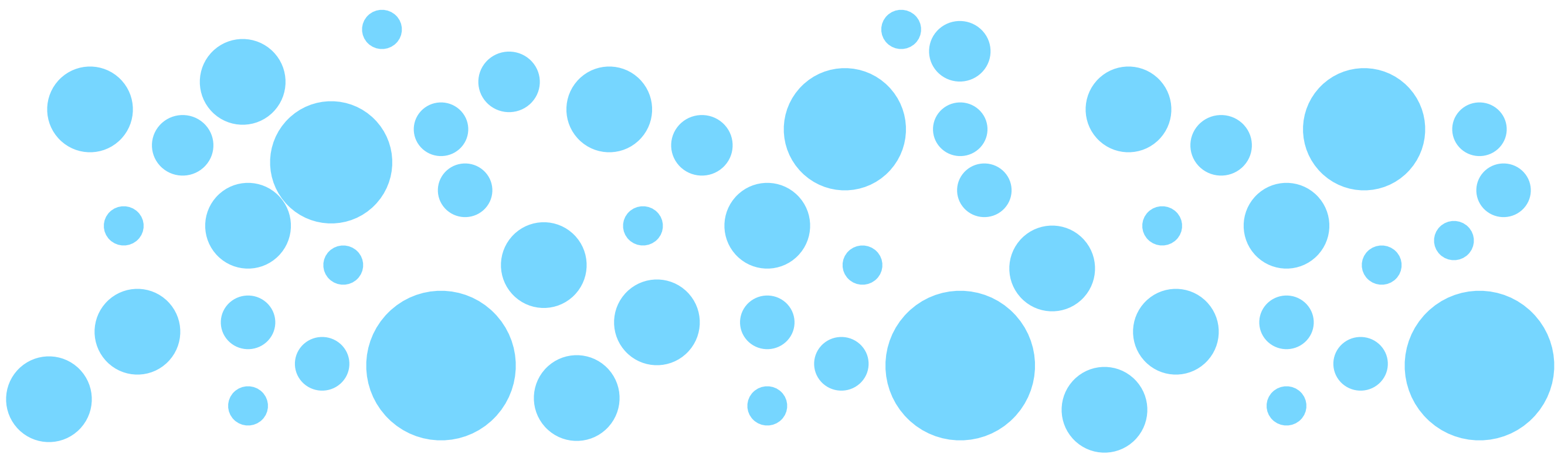
CLUSTER EDITING

Input Graph, G



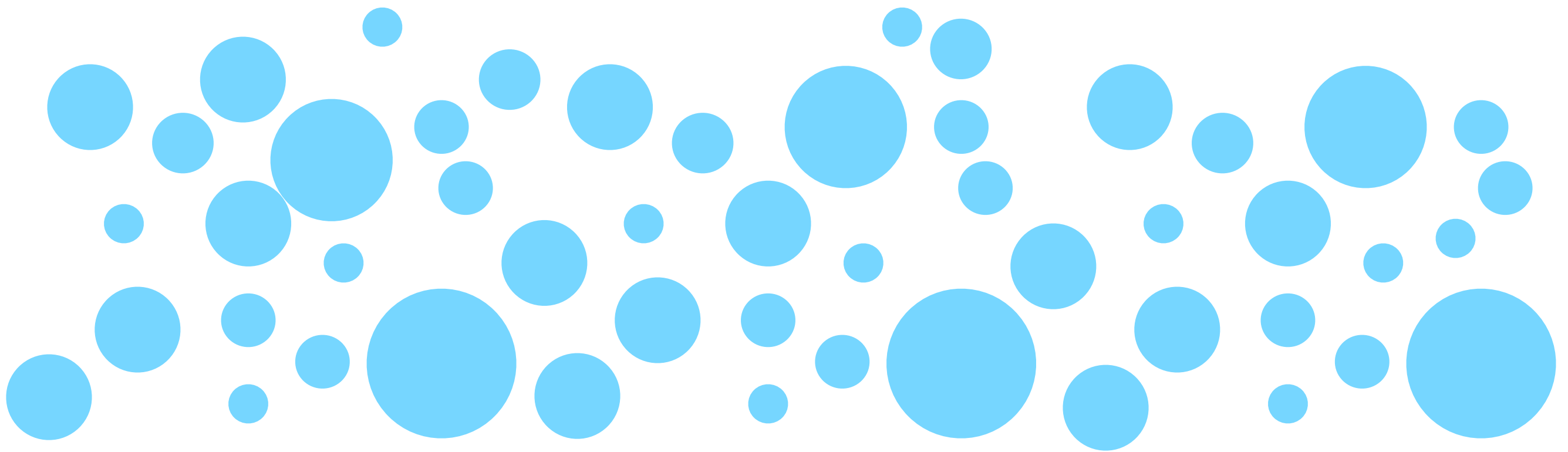
Cluster Graphs





Graph Class or “property”, Π

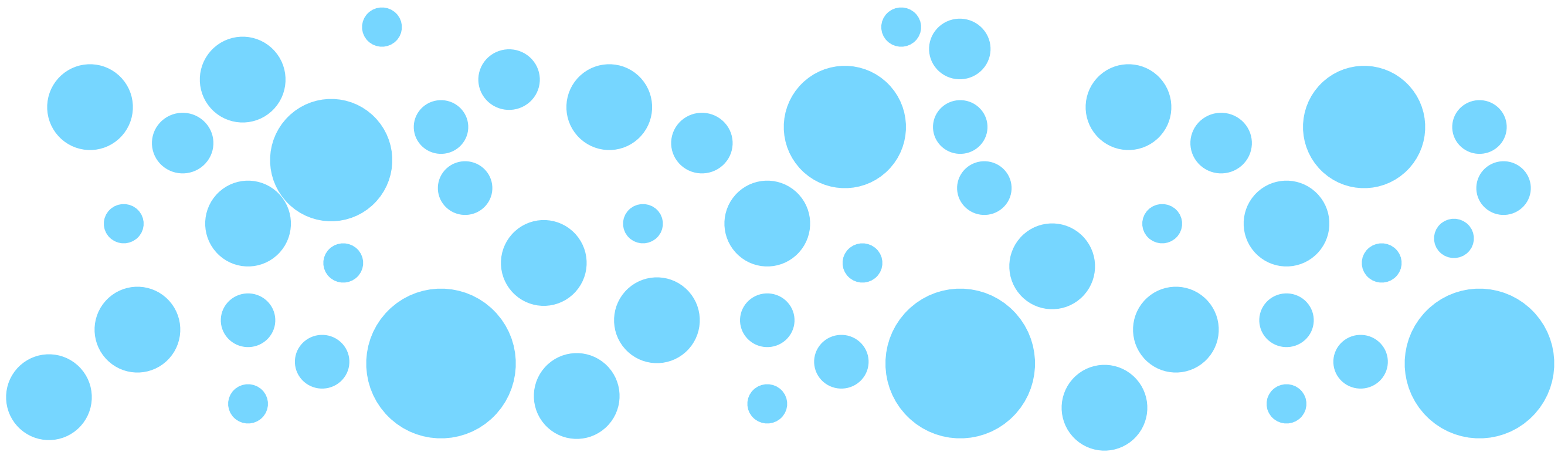
Non-Trivial Π admits infinitely many graphs, and also excludes infinitely many graphs.



Graph Class or “property”, Π

Non-Trivial Π admits infinitely many graphs, and also excludes infinitely many graphs.

Monotone If G belongs to Π , then all **subgraphs** of G also belong to Π .

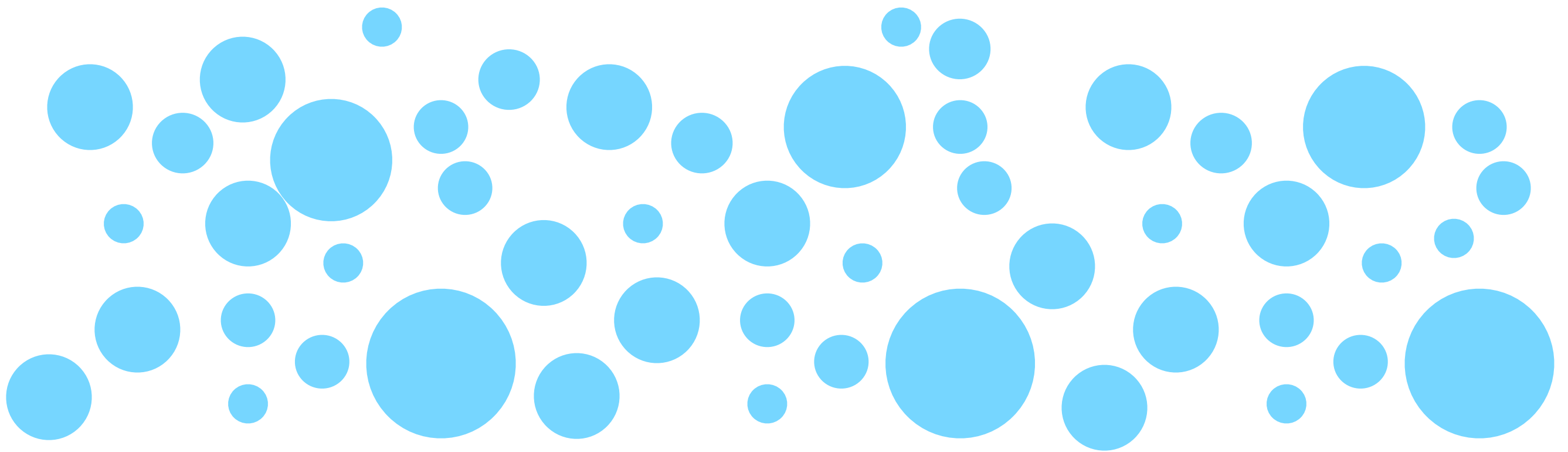


Graph Class or “property”, Π

Non-Trivial Π admits infinitely many graphs, and also excludes infinitely many graphs.

Monotone If G belongs to Π , then all **subgraphs** of G also belong to Π .

Hereditary If G belongs to Π , then all **induced subgraphs** of G also belong to Π .



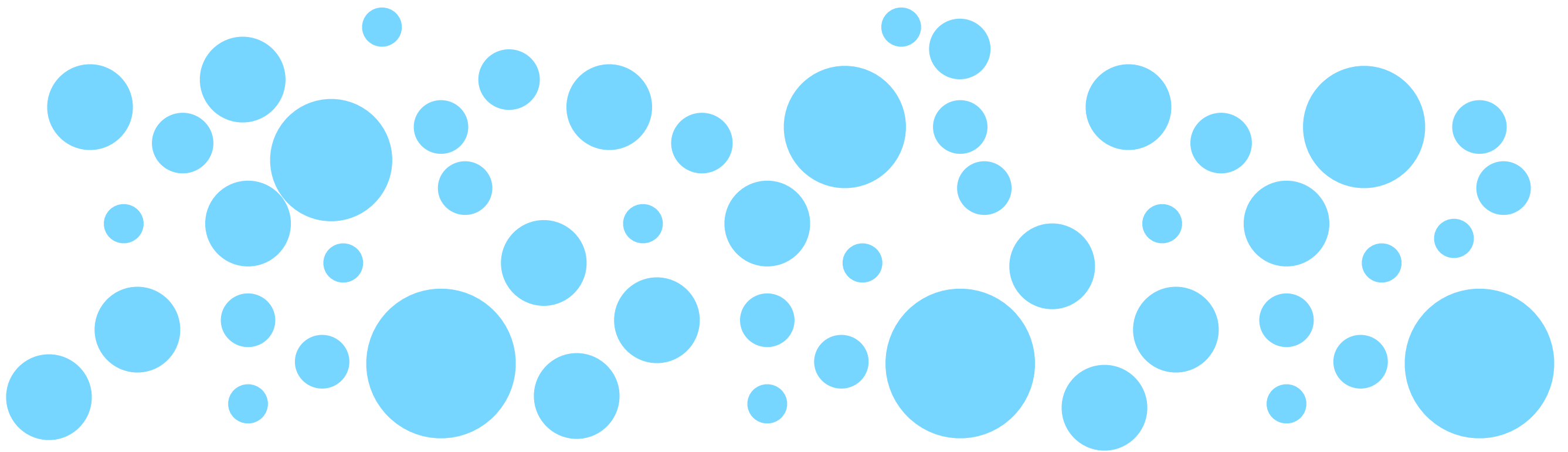
Graph Class or “property”, Π

Non-Trivial Π admits infinitely many graphs, and also excludes infinitely many graphs.

Monotone If G belongs to Π , then all **subgraphs** of G also belong to Π .

(Bipartite, Planar)

Hereditary If G belongs to Π , then all **induced subgraphs** of G also belong to Π .



Graph Class or “property”, Π

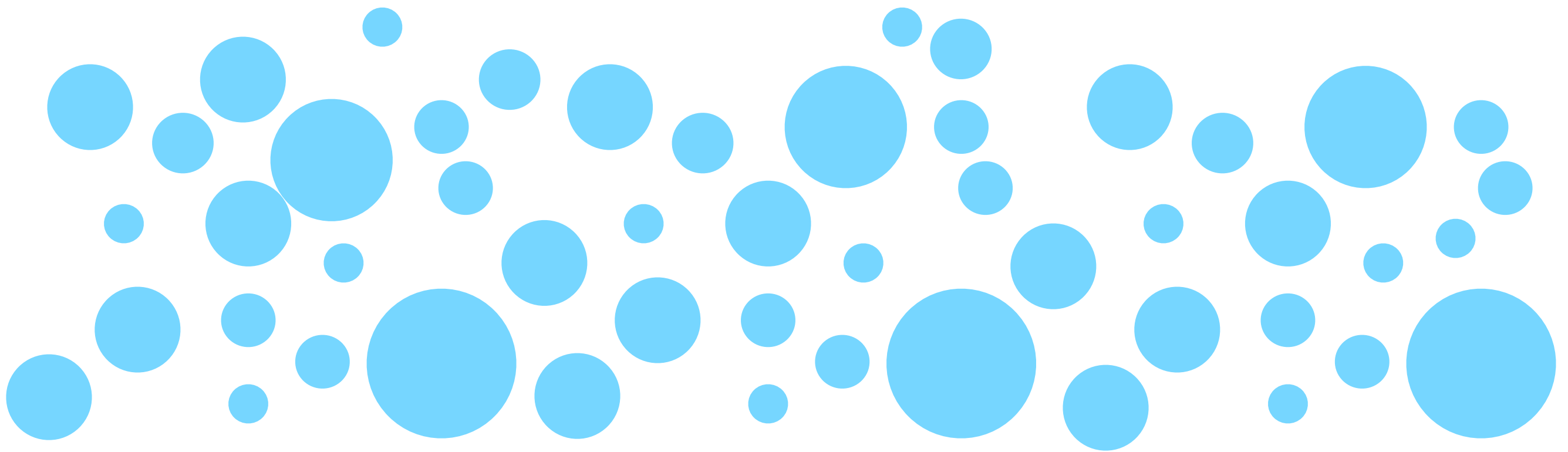
Non-Trivial Π admits infinitely many graphs, and also excludes infinitely many graphs.

Monotone If G belongs to Π , then all **subgraphs** of G also belong to Π .

(Bipartite, Planar)

Hereditary If G belongs to Π , then all **induced subgraphs** of G also belong to Π .

(Complete graphs, Forests)



Graph Class or “property”, Π

Non-Trivial Π admits infinitely many graphs, and also excludes infinitely many graphs.

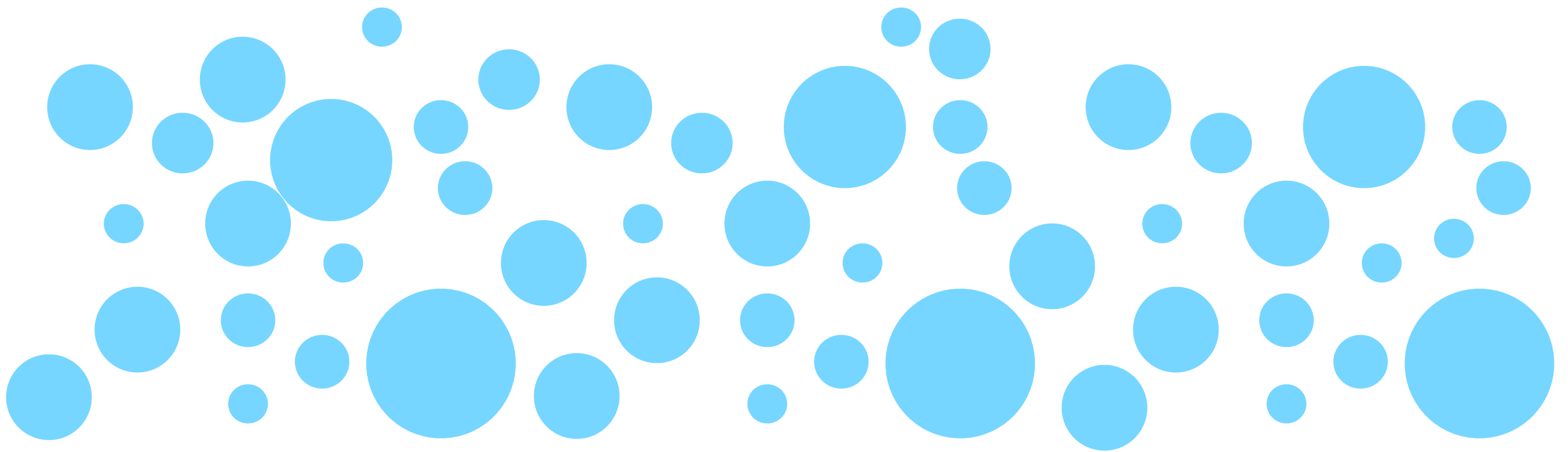
Monotone If G belongs to Π , then all **subgraphs** of G also belong to Π .

(Bipartite, Planar)

Hereditary If G belongs to Π , then all **induced subgraphs** of G also belong to Π .

(Complete graphs, Forests)

Connectivity, Biconnectivity, Trees, Stars, Eulerian, etc.



Graph Class or “property”, Π

Lewis &
Yannakakis
[1980]

The vertex-deletion problem is **NP-complete** for non-trivial,
hereditary properties.

Lewis &
Yannakakis
[1980]

The vertex-deletion problem is **NP-complete** for non-trivial,
hereditary properties.

(Edgeless, Acyclic, etc.)

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Remove edges to make the input graph edgeless.

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Remove edges to make the input graph edgeless.

Remove edges to make the input graph acyclic.

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Remove edges to make the input graph edgeless.

Remove edges to make the input graph acyclic.

Remove edges to make the input graph bipartite.

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Remove edges to make the input graph edgeless.

Remove edges to make the input graph acyclic.

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Remove edges to make the input graph edgeless.

Remove edges to make the input graph acyclic.

Alon, Shapira
& Sudakov
[2009]

Given a property Π such that Π holds for every bipartite graph, the Minimum Π -Deletion problem is NP-hard.

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Remove edges to make the input graph edgeless.

Remove edges to make the input graph acyclic.

Alon, Shapira
& Sudakov
[2009]

Given a property Π such that Π holds for every bipartite graph, the Minimum Π -Deletion problem is NP-hard.

(Triangle-free)

Edge-Deletion or Completion problems are sometimes easier than their vertex-deletion counterparts.

Add edges to make the input graph a cluster graph.

Remove edges to make the input graph edgeless.

Remove edges to make the input graph acyclic.

Alon, Shapira
& Sudakov
[2009]

Given a property Π such that Π holds for every bipartite graph, the Minimum Π -Deletion problem is NP-hard.

(Triangle-free)

Colbourn &
El-Mallah
[1988]

Making a graph P_k -free by deleting the minimum number of edges is NP-hard for every $k > 2$.

Finding large induced subgraphs with property Π .

Khot and Raman, (2002)

Finding large induced subgraphs with property Π .

Khot and Raman, (2002)

If Π is a hereditary property that contains all independent sets and all cliques,
or if Π excludes some independent sets and some cliques,
then the problem of finding an induced subgraph of size at least k , satisfying Π ,
is FPT.

Finding large induced subgraphs with property Π .

Khot and Raman, (2002)

If Π is a hereditary property that contains all independent sets and all cliques,
or if Π excludes some independent sets and some cliques,
then the problem of finding an induced subgraph of size at least k , satisfying Π ,
is FPT.

If Π is a hereditary property contains all independent sets but not all cliques
(or vice versa),
then the problem of finding an induced subgraph of size at least k , satisfying Π ,
is $W[1]$ -hard.

Finding large induced subgraphs with property Π .

Khot and Raman, (2002)

Ramsey Numbers!

If Π is a hereditary property that contains all independent sets and all cliques,
or if Π excludes some independent sets and some cliques,
then the problem of finding an induced subgraph of size at least k , satisfying Π ,
is FPT.

Reduction from Independent Set

If Π is a hereditary property contains all independent sets but not all cliques
(or vice versa),
then the problem of finding an induced subgraph of size at least k , satisfying Π ,
is $W[1]$ -hard.

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

Implications of the Graph Minor Theorem.

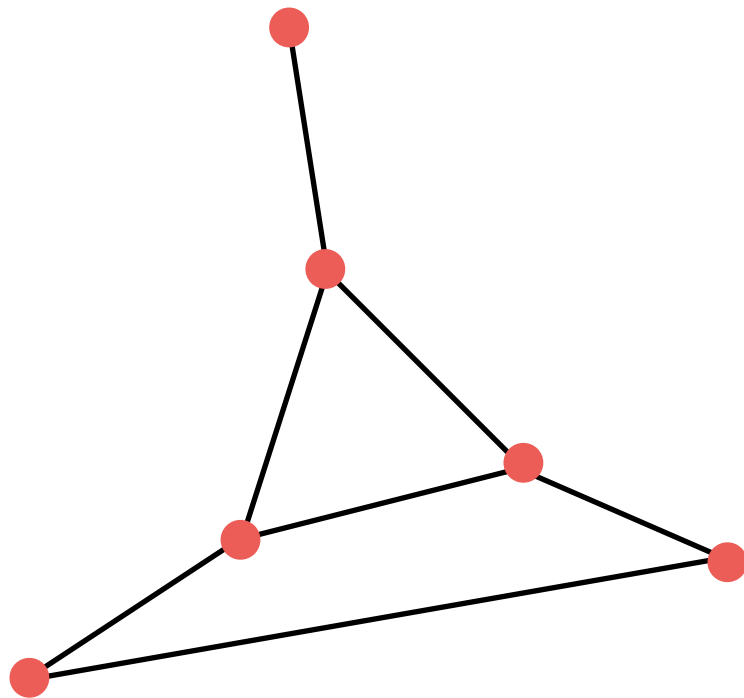
Vertex Cover, Revisited.

Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

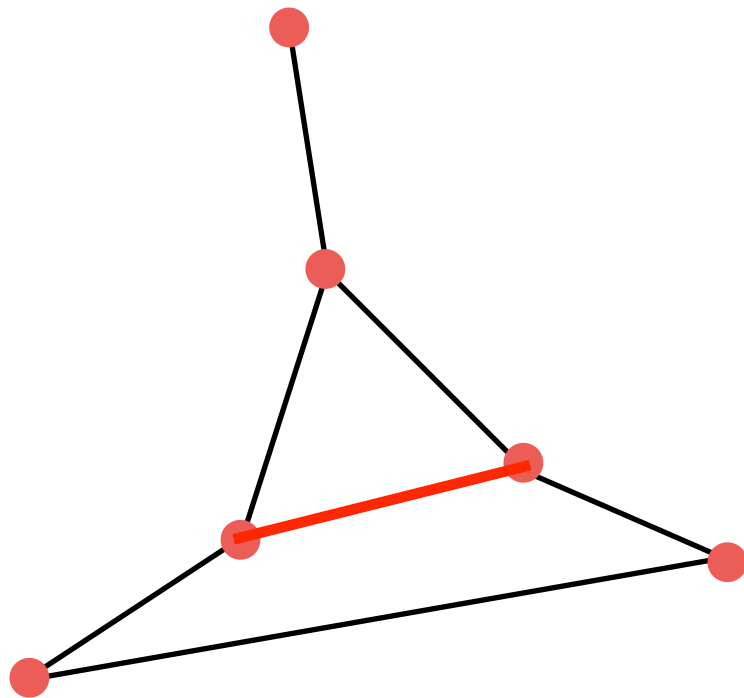
Planar F-Deletion.

A graph G contains a graph H as a minor if H can be obtained from G by a sequence of vertex or edge deletions; and edge contractions.

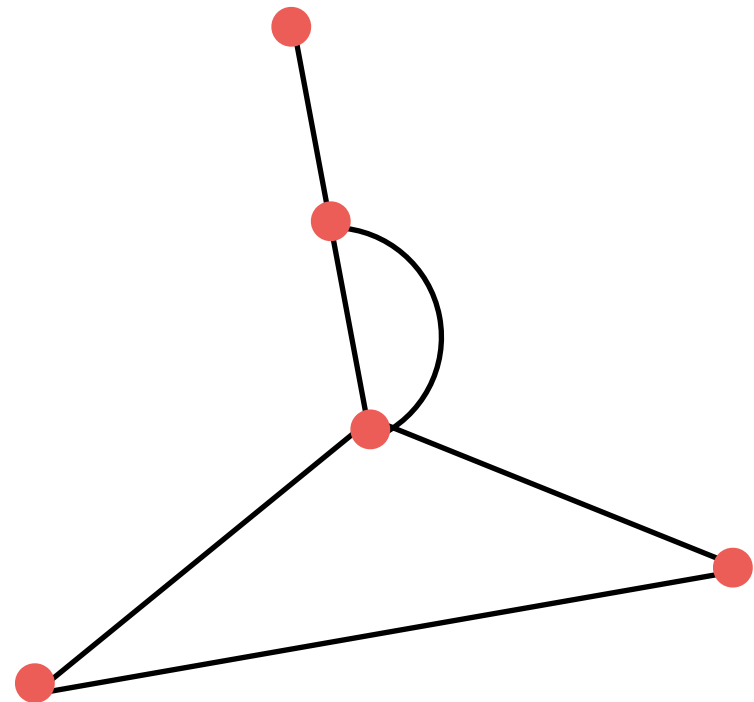
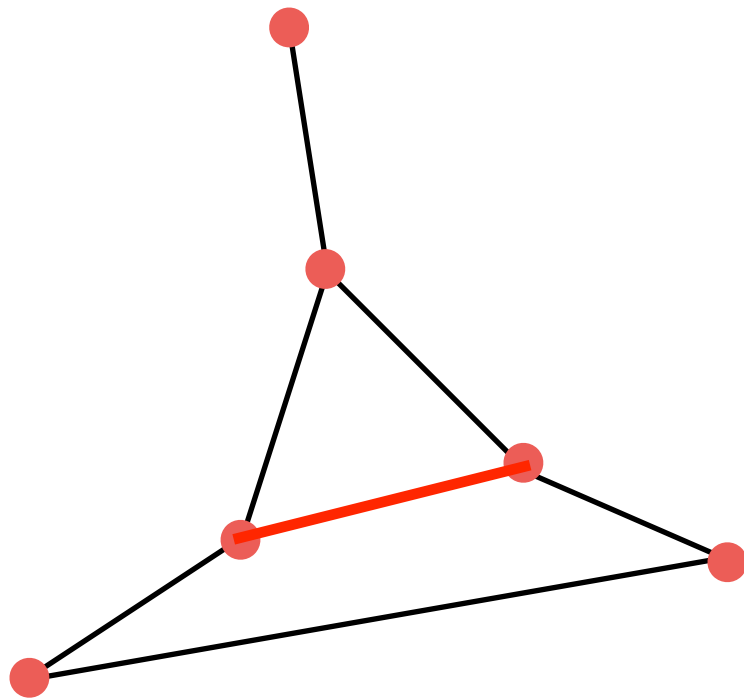
A graph G contains a graph H as a minor if H can be obtained from G by a sequence of vertex or edge deletions; and edge contractions.

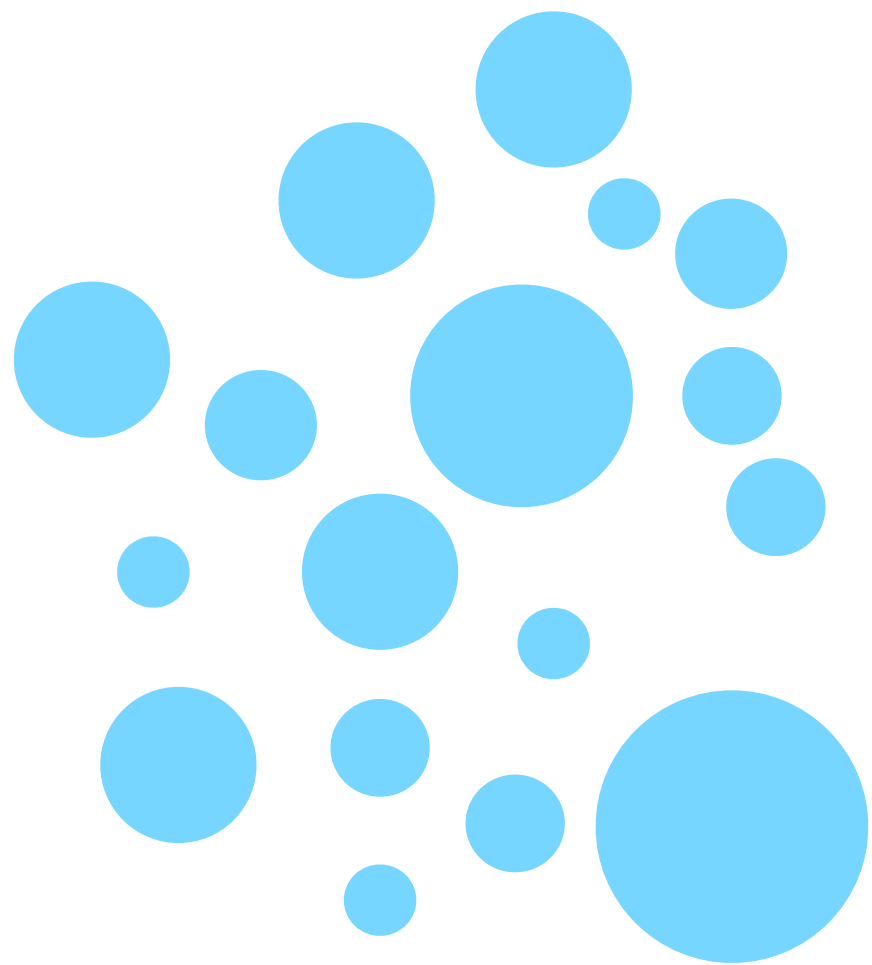


A graph G contains a graph H as a minor if H can be obtained from G by a sequence of vertex or edge deletions; and edge contractions.



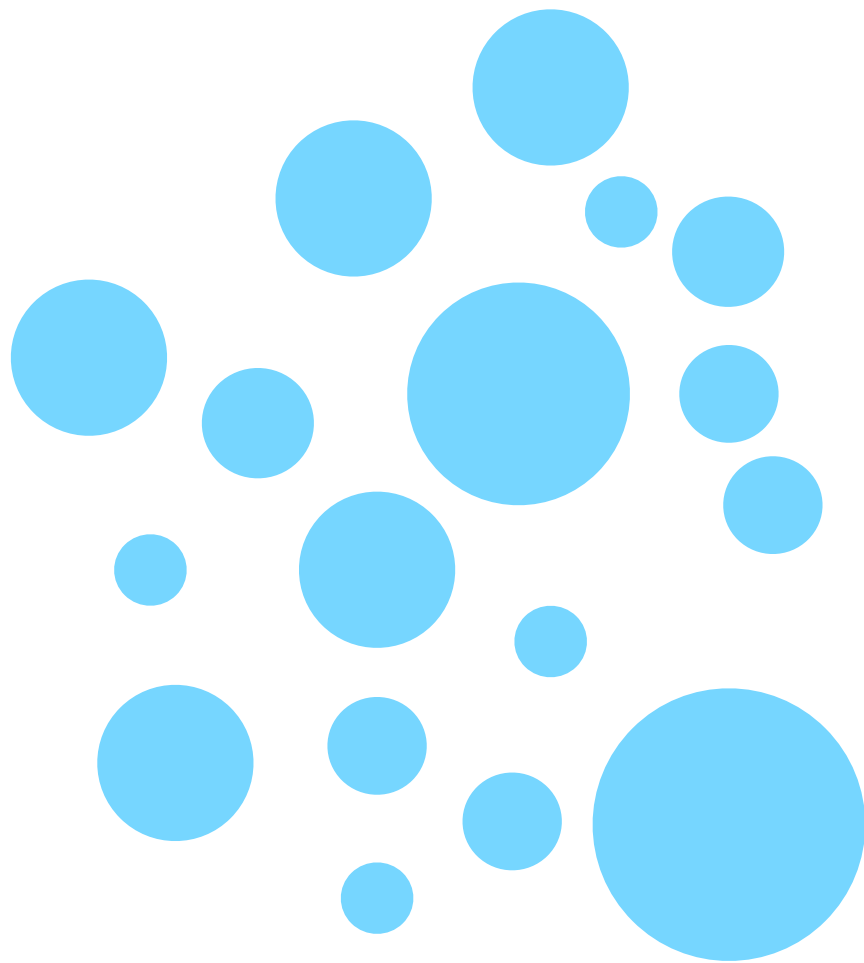
A graph G contains a graph H as a minor if H can be obtained from G by a sequence of vertex or edge deletions; and edge contractions.





Graph Class or “property”, Π

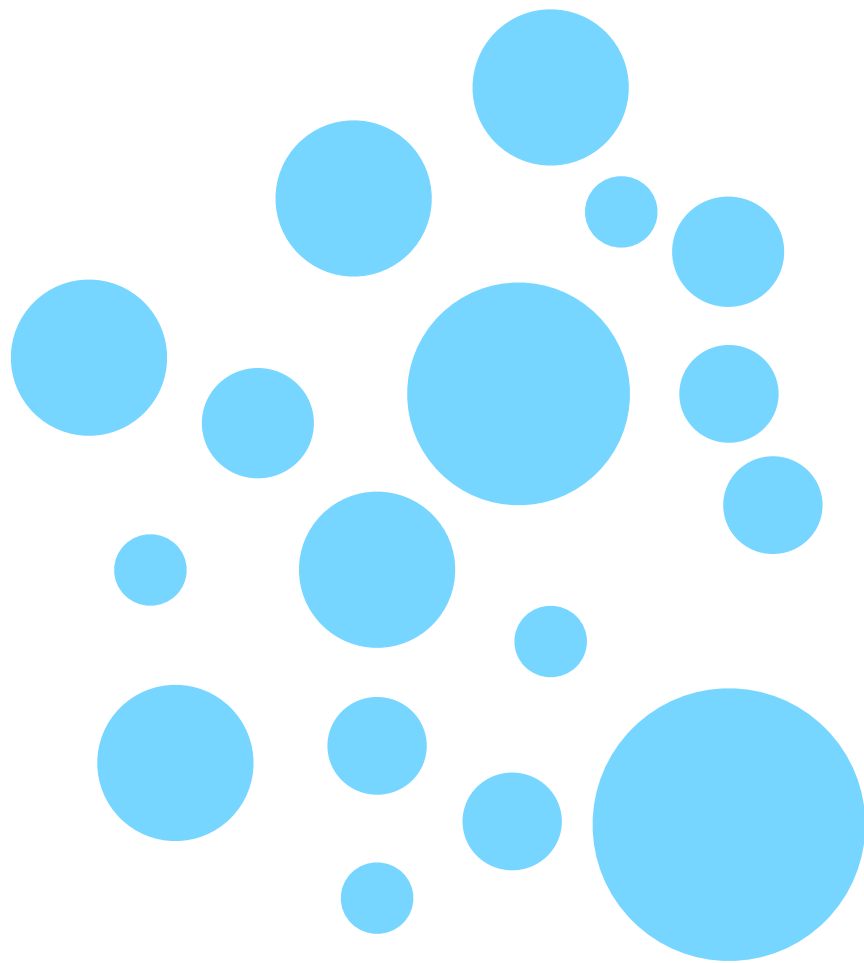
Forbidden Minor Characterization



Graph Class or “property”, Π

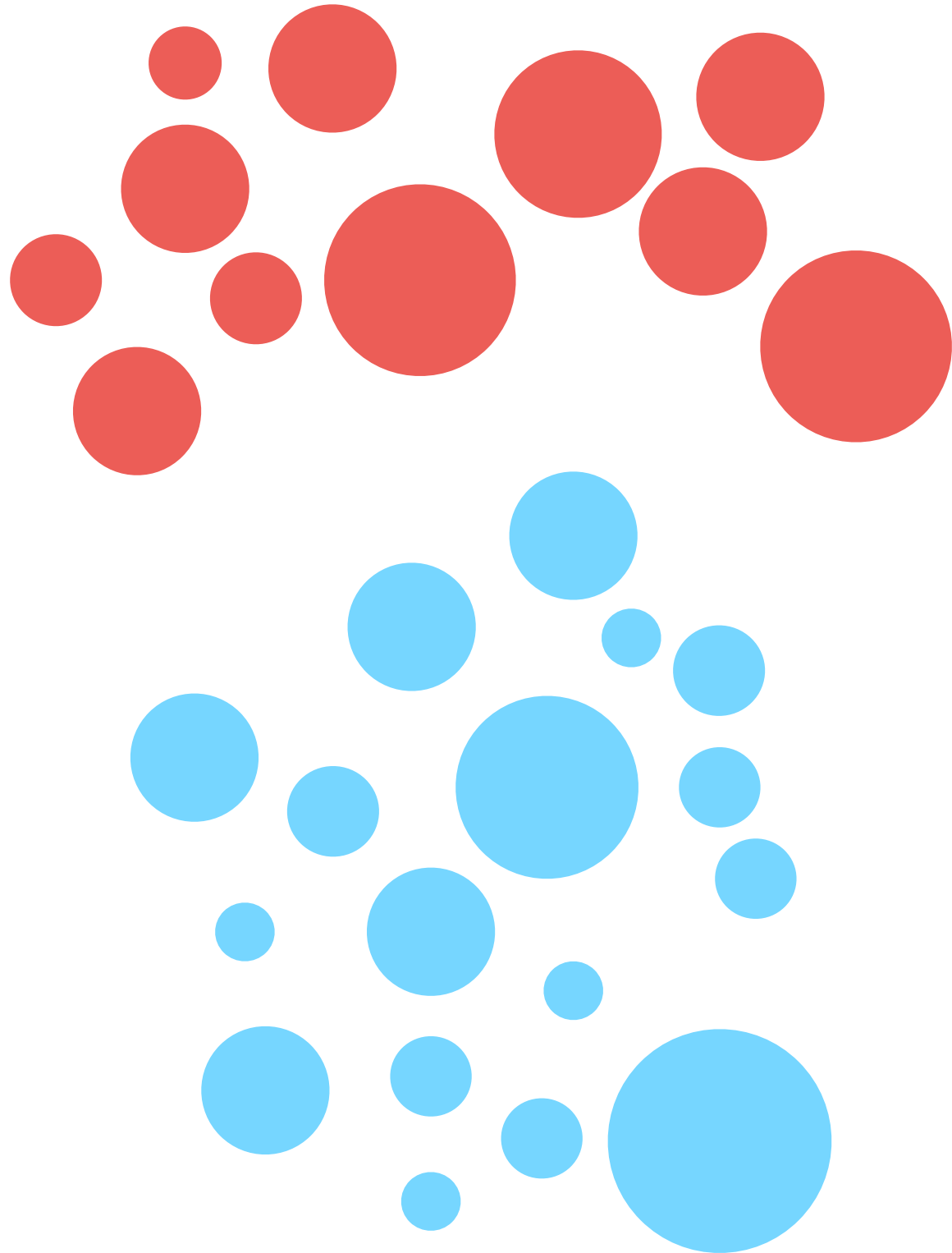
Forbidden Minor Characterization

A graph G belongs to Π if, and only if,
 G does not contain any graph from $\text{Forb}(\Pi)$
as a **minor**.



Forbidden Minor Characterization

A graph G belongs to Π if, and only if,
 G does not contain any graph from $\text{Forb}(\Pi)$
as a **minor**.

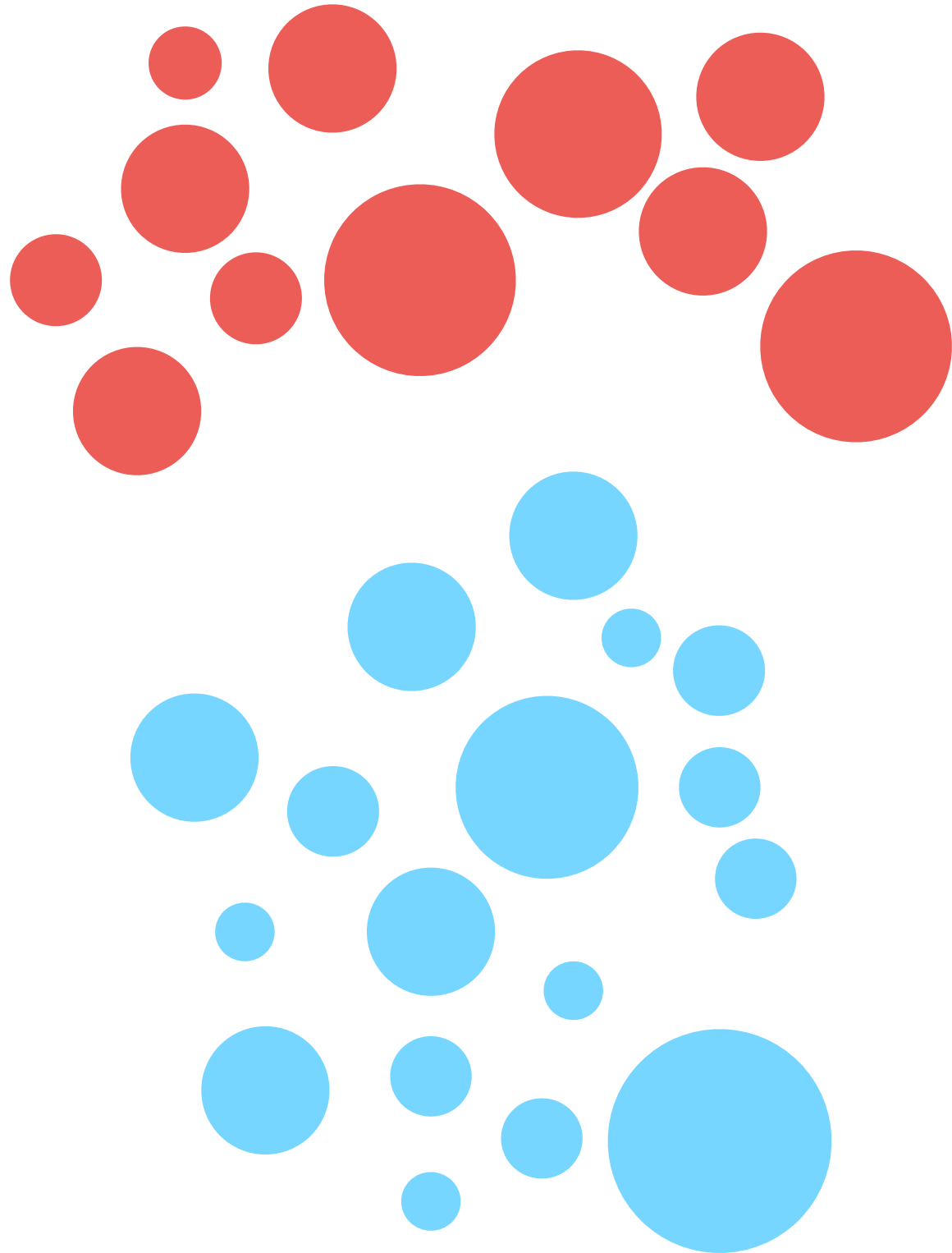


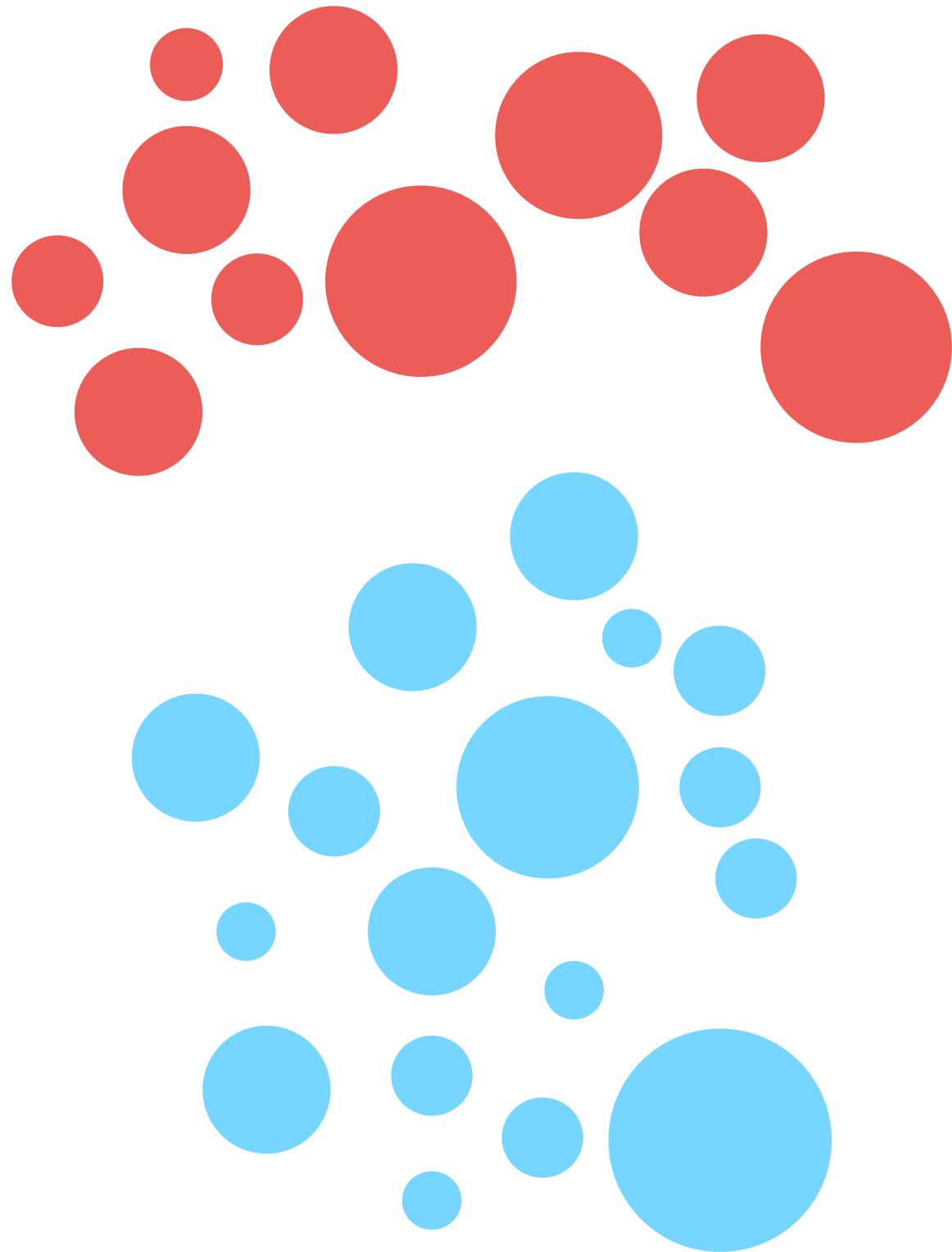
Graph Class or “property”, Π

Forbidden Minor Characterization

A graph G belongs to Π if, and only if,
 G does not contain any graph from $\text{Forb}(\Pi)$
as a **minor**.

A graph admits a **forbidden minor characterization**
if, and only if,
it is **closed under taking minors**.



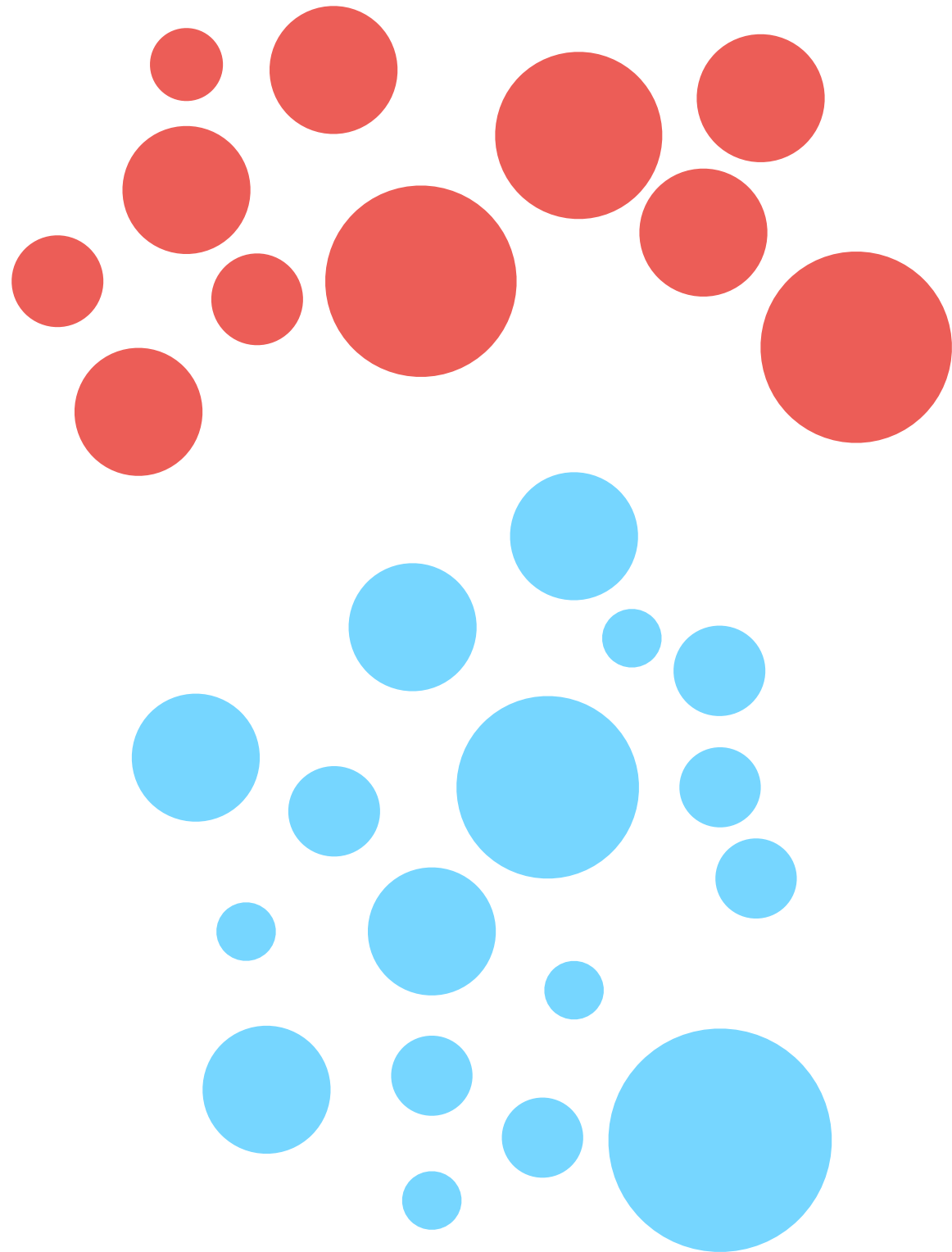


Forbidden Minor Characterization

A graph G belongs to Π if, and only if,
 G does not contain any graph from $\text{Forb}(\Pi)$
as a **minor**.

A graph admits a **forbidden minor characterization**
if, and only if,
it is **closed under taking minors**.

In fact, there is always a finite forbidden set!
(Robertson-Seymour; Graph Minors Theorem)



Forbidden Minor Characterization

A graph G belongs to Π if, and only if,
 G does not contain any graph from $\text{Forb}(\Pi)$
as a **minor**.

A graph admits a **forbidden minor characterization**
if, and only if,
it is **closed under taking minors**.

In fact, there is always a finite forbidden set!
(Robertson-Seymour; Graph Minors Theorem)

Suppose the class $\Pi + (\text{modifications})$ is
closed under taking minors...



Graph Class or “property”, Π + (modifications)

Forbidden Minor Characterization

A graph G belongs to Π if, and only if,
 G does not contain any graph from $\text{Forb}(\Pi)$
as a **minor**.

A graph admits a **forbidden minor characterization**
if, and only if,
it is **closed under taking minors**.

In fact, there is always a finite forbidden set!
(Robertson-Seymour; Graph Minors Theorem)

Suppose the class Π + (modifications) is
closed under taking minors...



Graph Class or “property”, Π + (modifications)

Forbidden Minor Characterization

A graph G belongs to Π if, and only if,
 G does not contain any graph from $\text{Forb}(\Pi)$
as a **minor**.

A graph admits a **forbidden minor characterization**
if, and only if,
it is **closed under taking minors**.

In fact, there is always a finite forbidden set!
(Robertson-Seymour; Graph Minors Theorem)

Suppose the class Π + (modifications) is
closed under taking minors...

Now: the graph modification problem has boiled down
to a membership testing problem, which can be
done in cubic time, given the finite forbidden set.

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

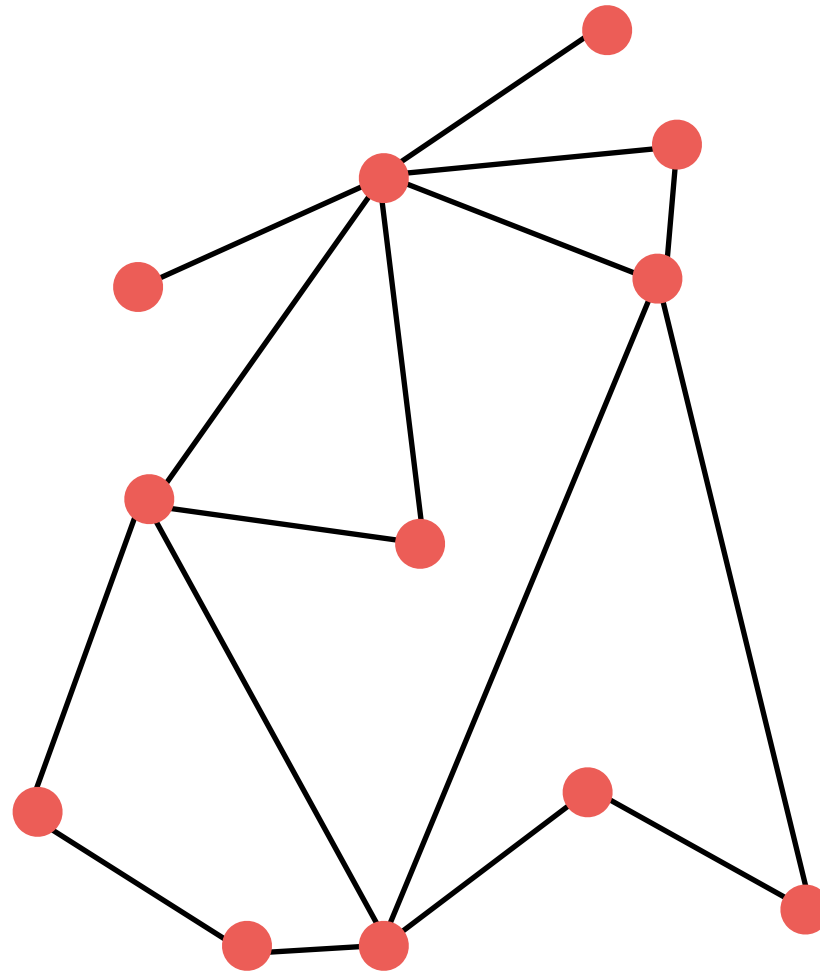
Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

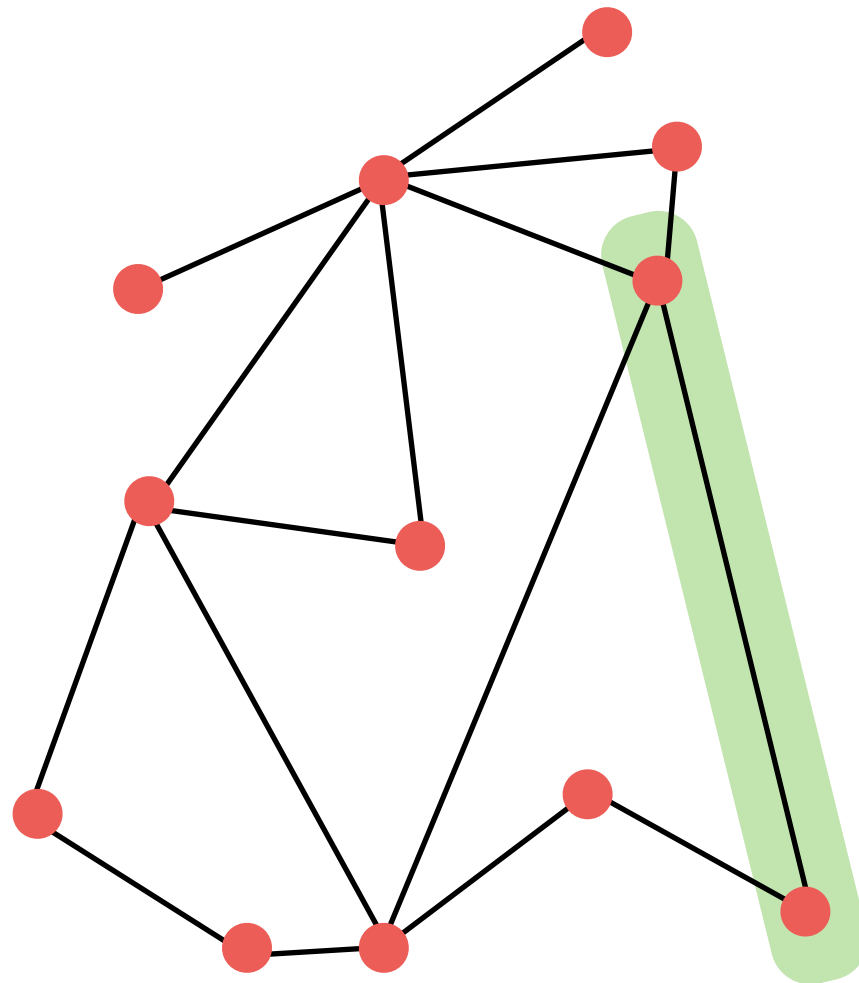
Planar F-Deletion.

Can G be made edgeless by the removal of at most k vertices?



A simple randomized algorithm with an error probability of 2^{-k} .

Can G be made edgeless by the removal of at most k vertices?



A simple randomized algorithm with an error probability of 2^{-k} .

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

Define a class of graph modification problems.

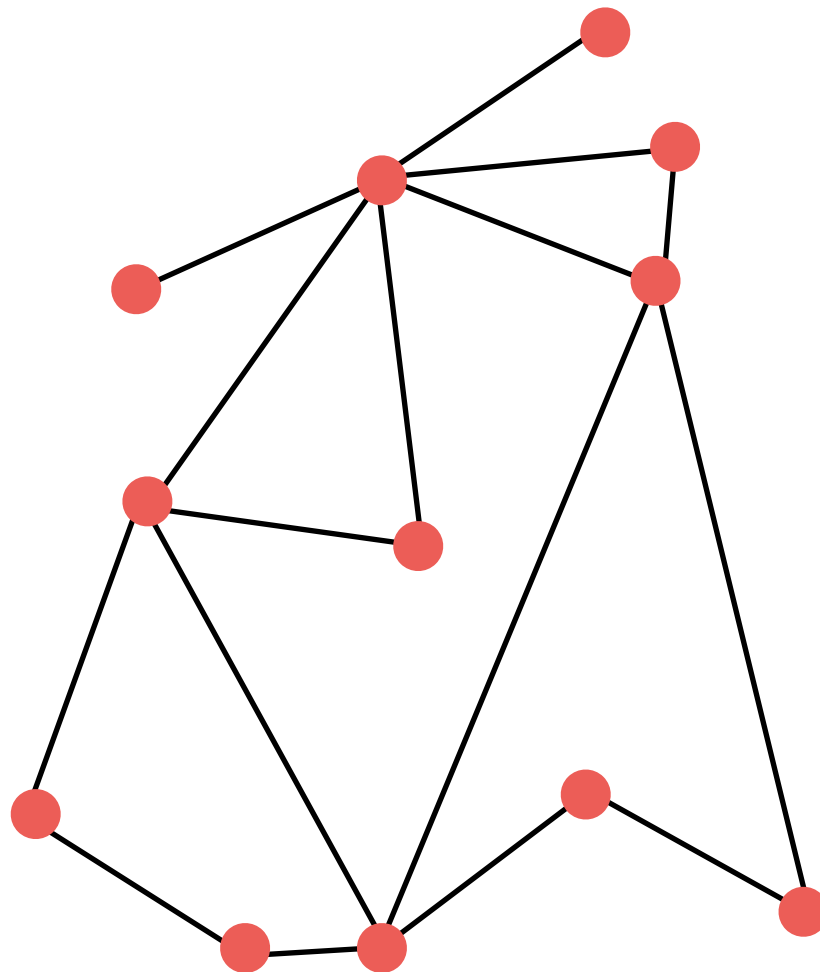
Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

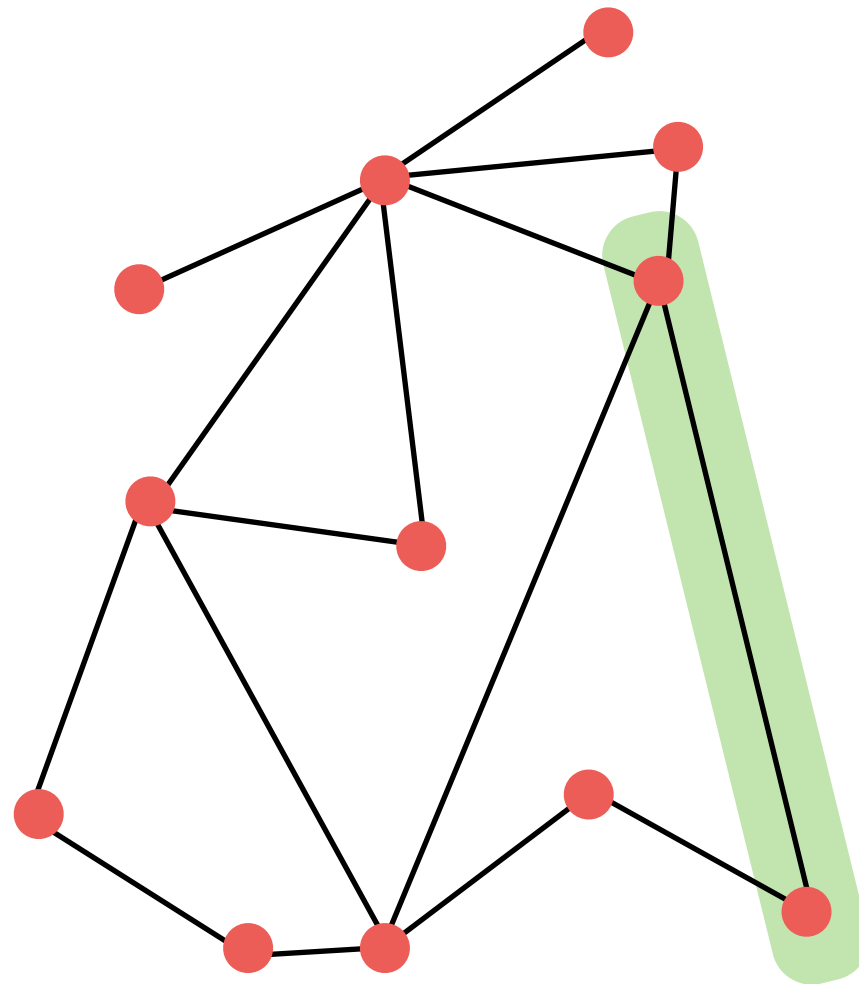
Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

Planar F-Deletion.

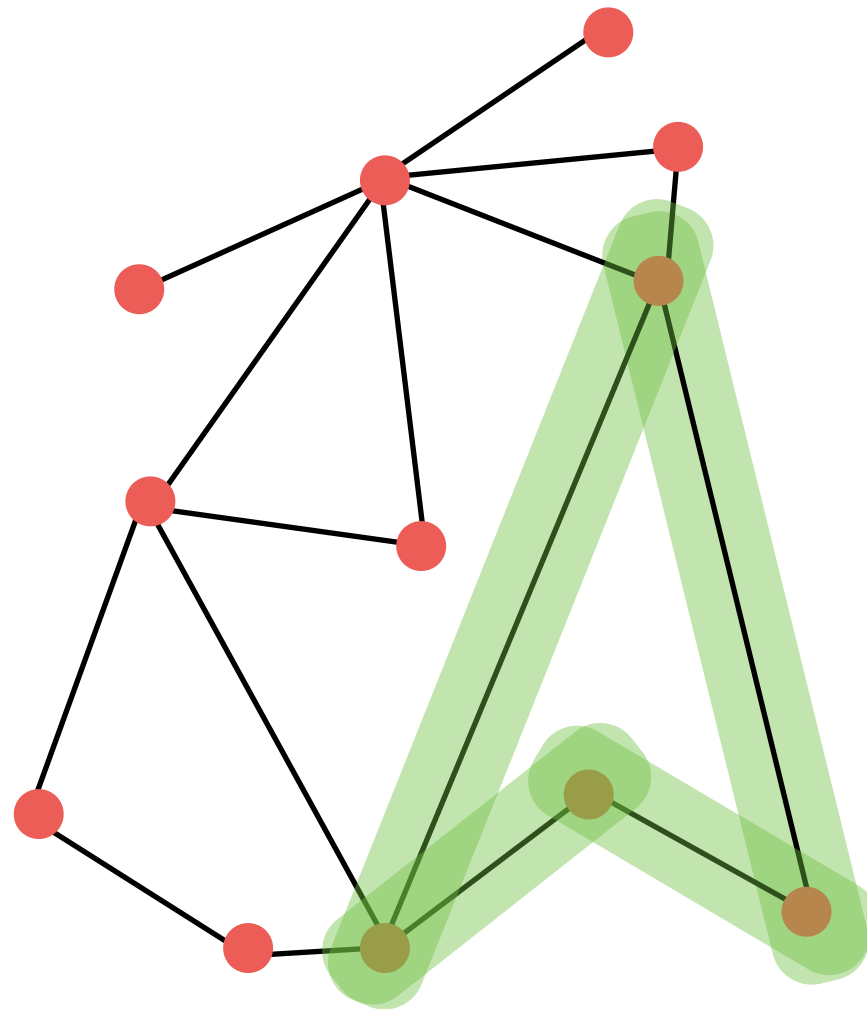
Can G be made acyclic by the removal of at most k vertices?



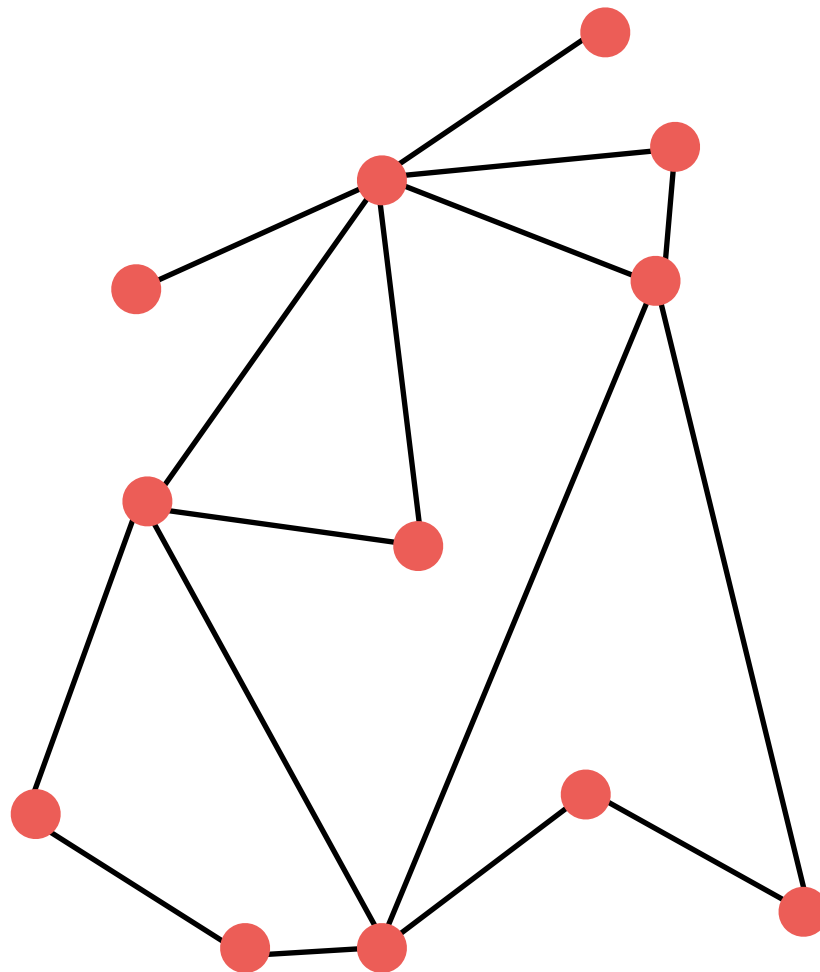
Can G be made acyclic by the removal of at most k vertices?



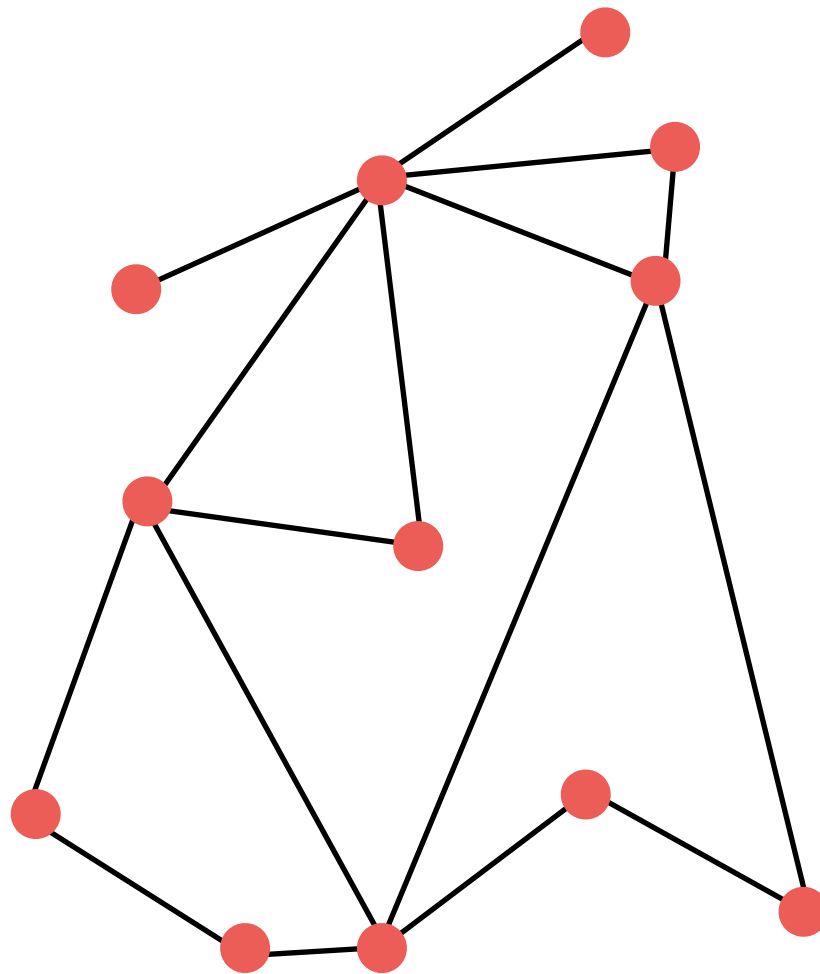
Can G be made acyclic by the removal of at most k vertices?



Can G be made acyclic by the removal of at most k vertices?



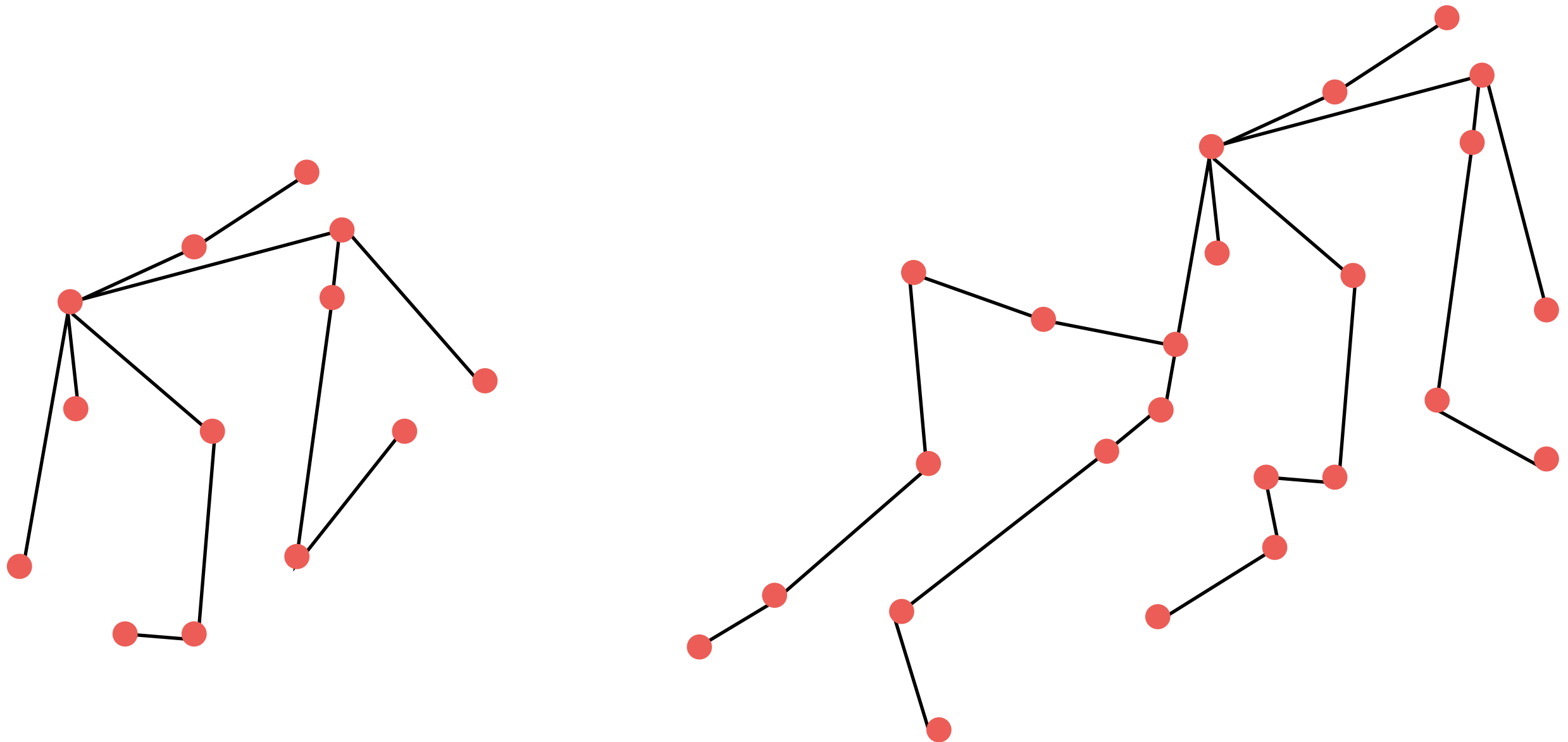
Can G be made acyclic by the removal of at most k vertices?



Let us consider the *structure* of a YES-instance.

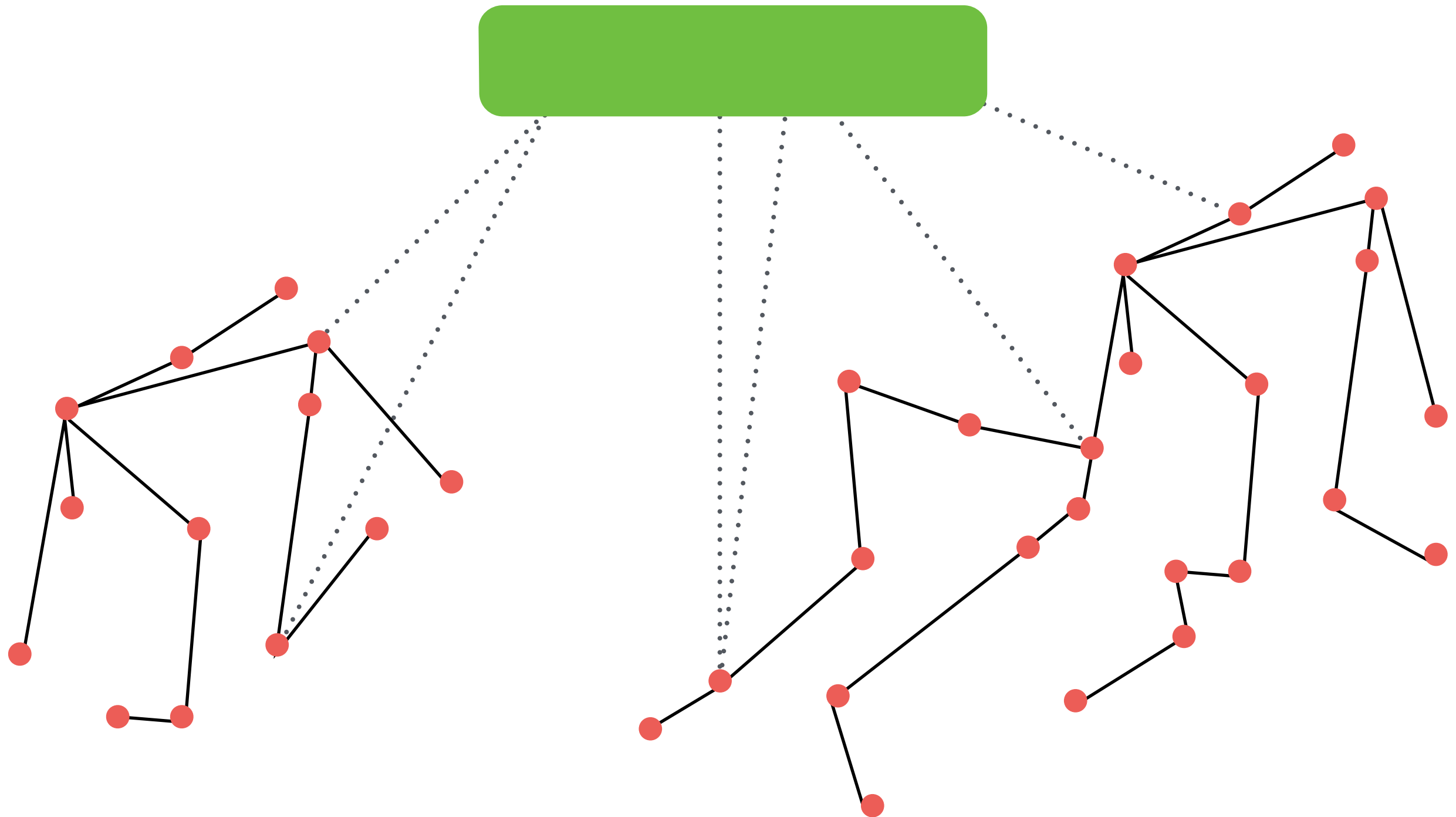
Let us also restrict ourselves to graphs of constant maximum degree, say five.

The k vertices corresponding to the FVS

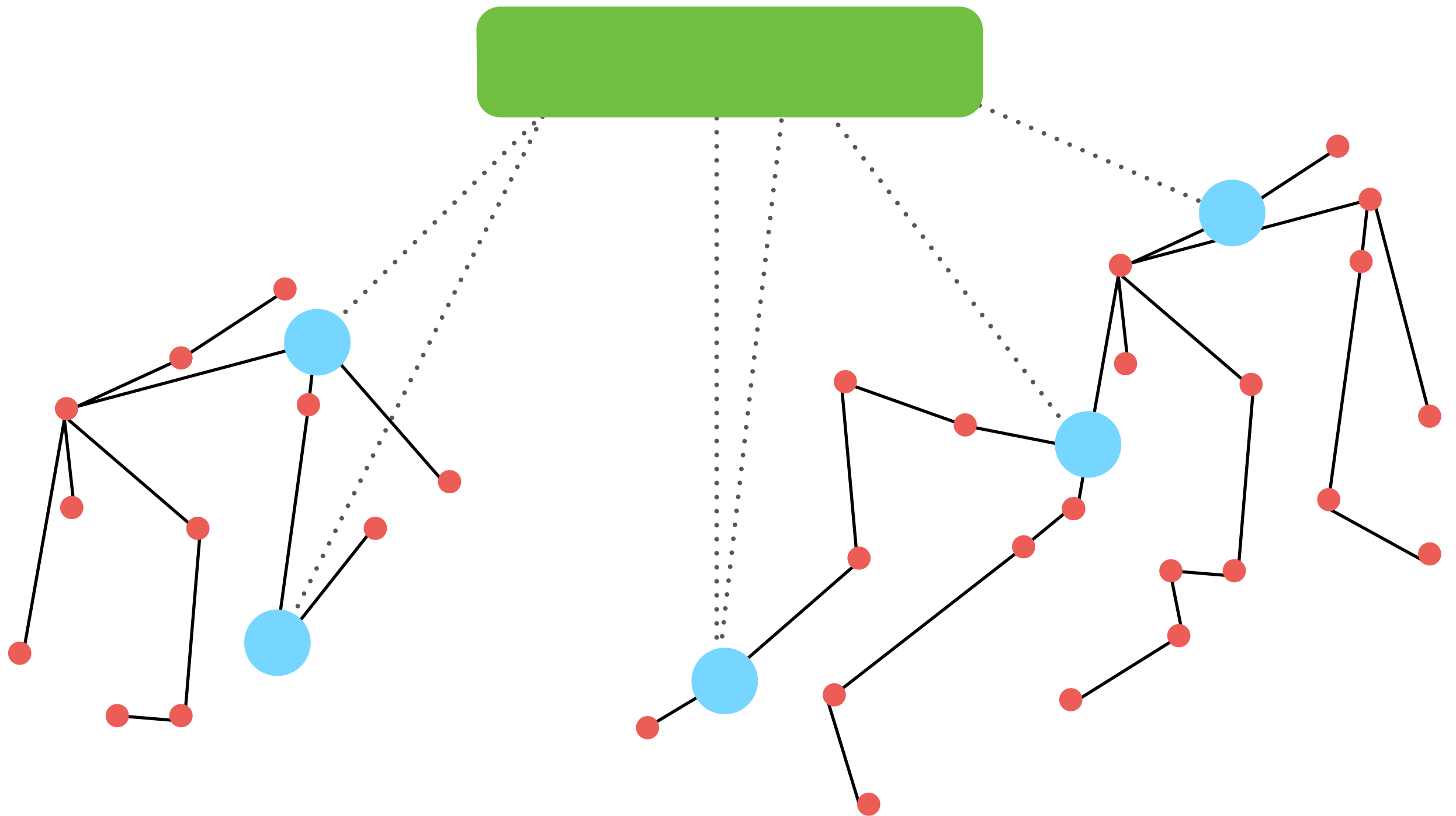


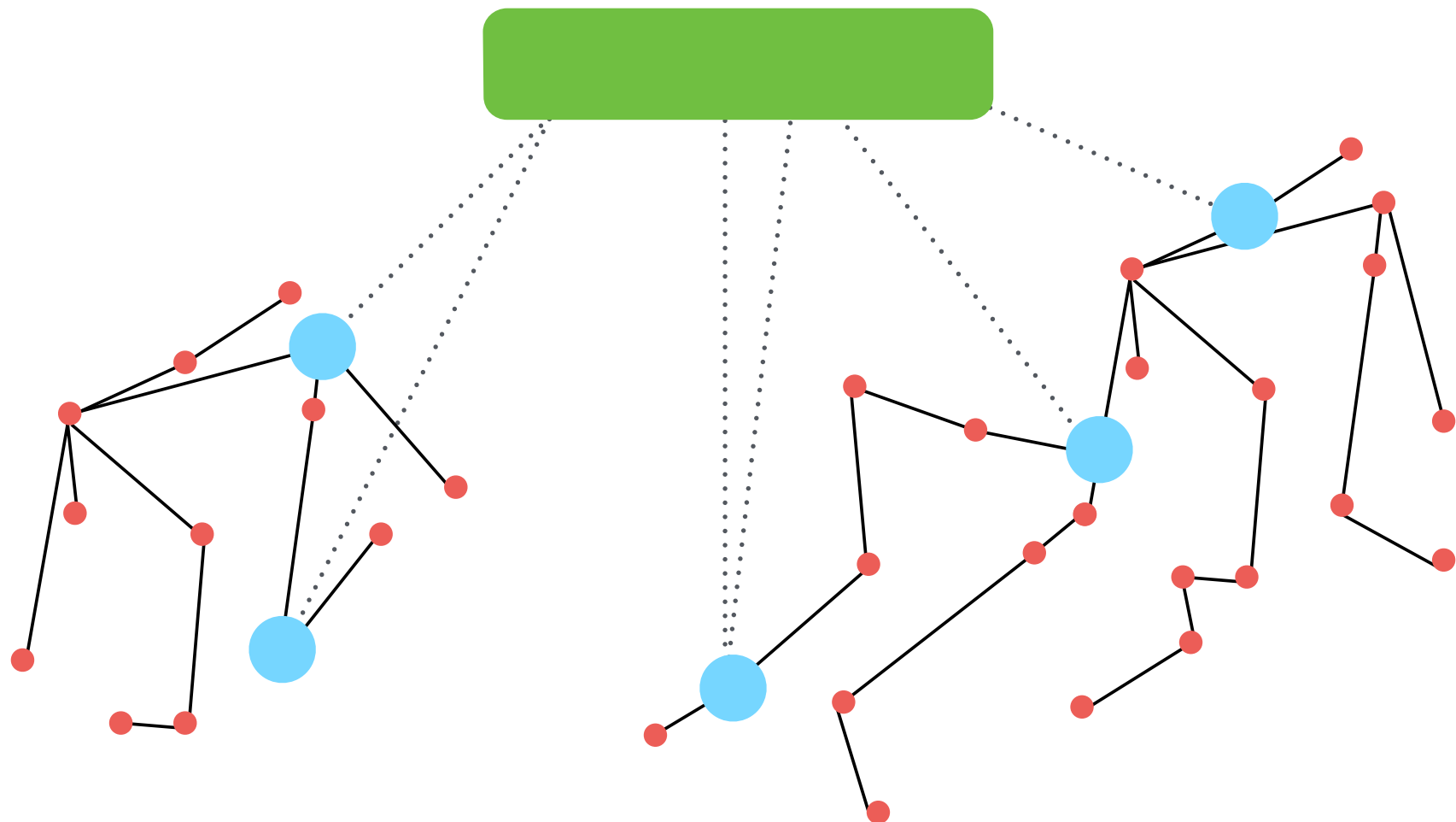
$(n-k)$ vertices and at most $(n-k-1)$ edges “outside”.

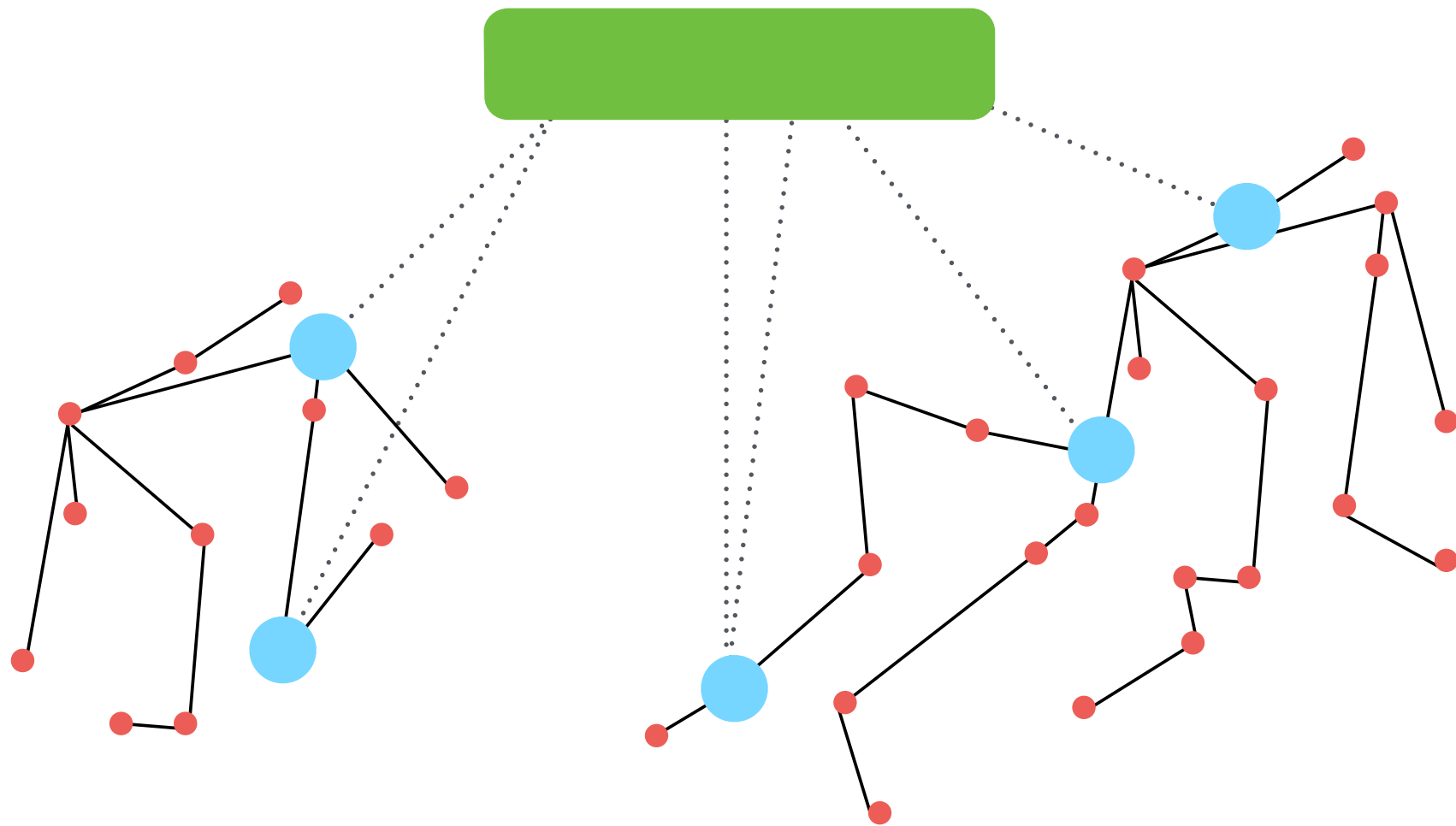
The k vertices corresponding to the FVS



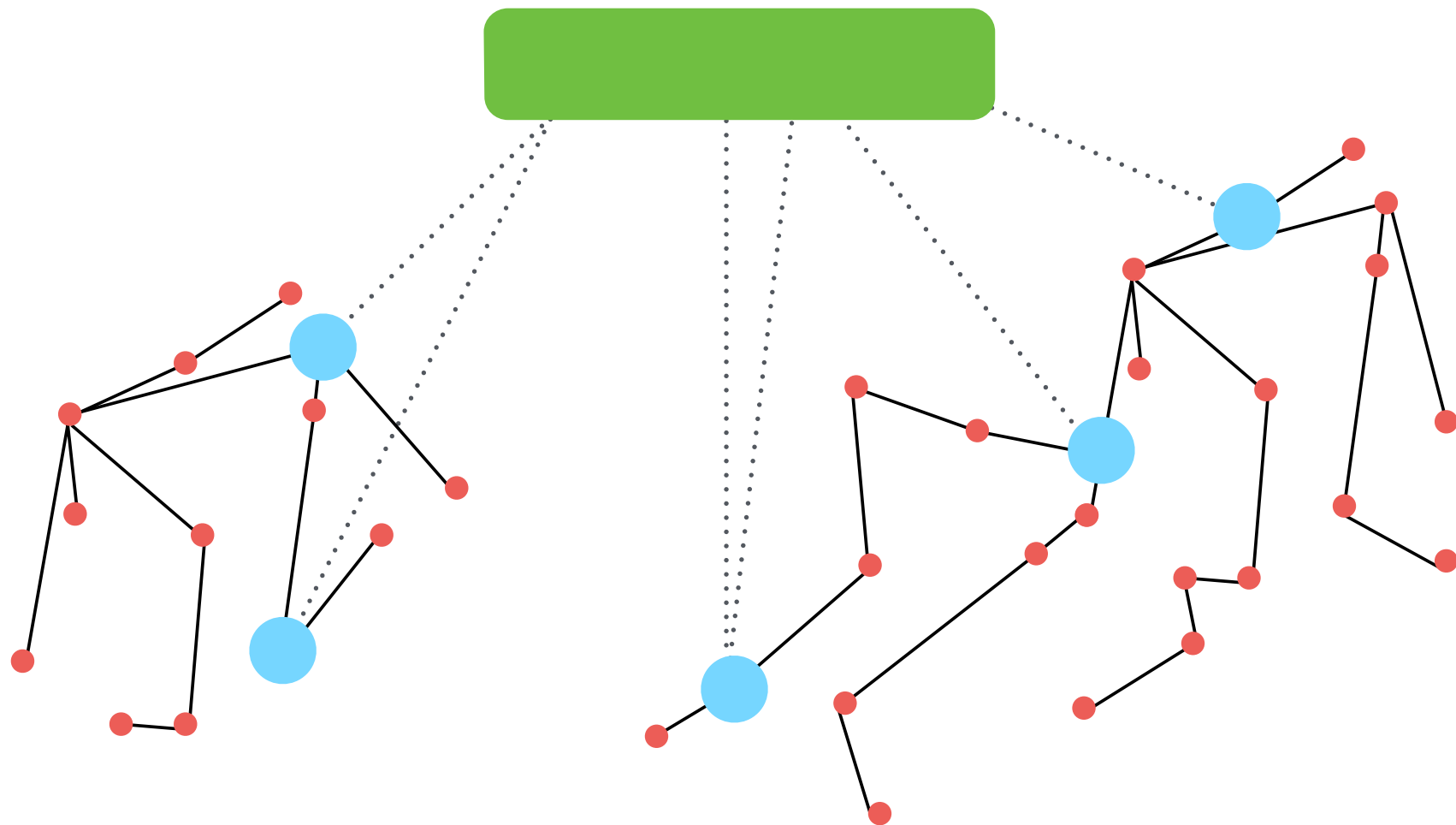
$(n-k)$ vertices and at most $(n-k-1)$ edges “outside”.





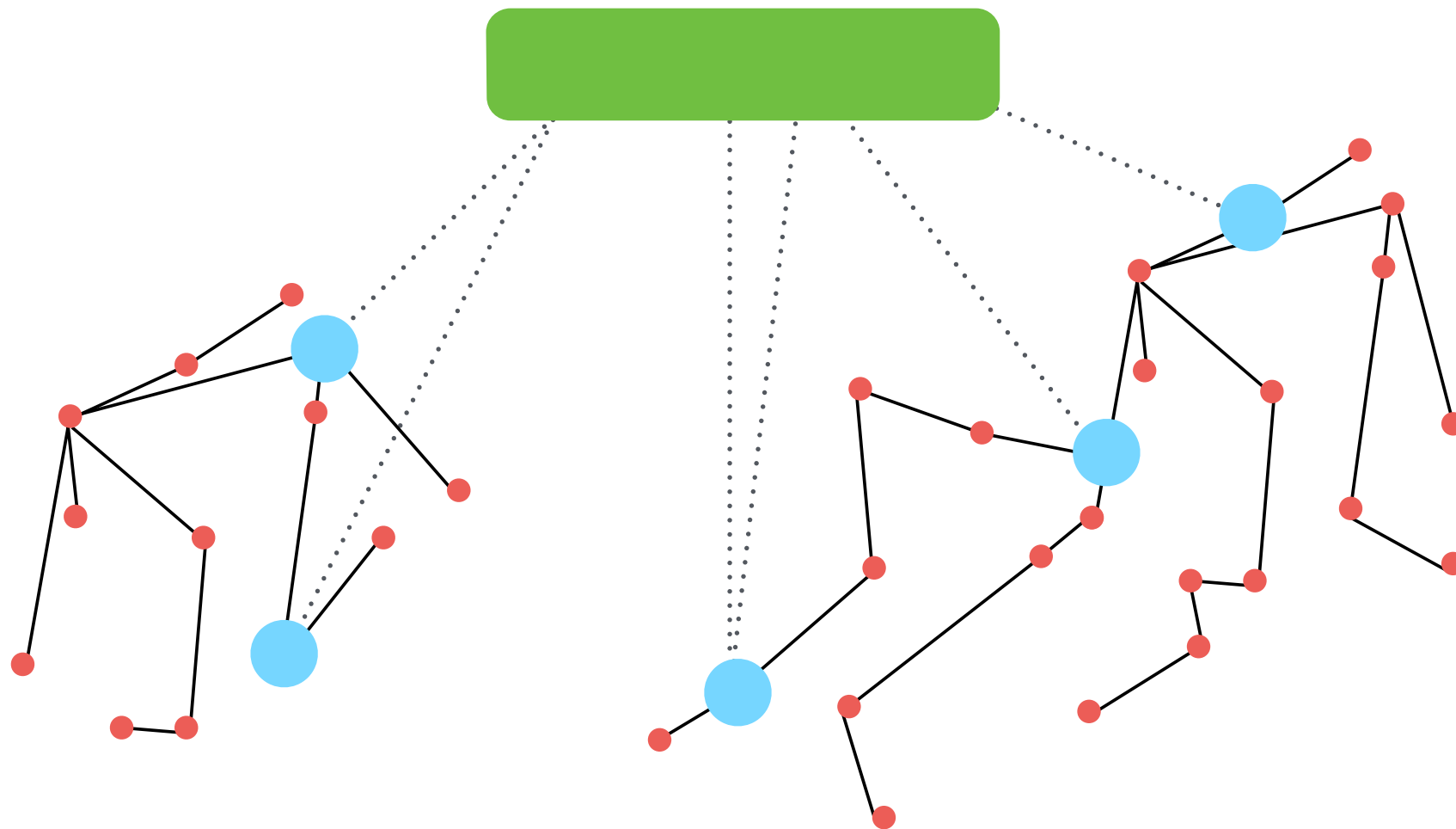


t “points of contact” between S and $G \setminus S$...



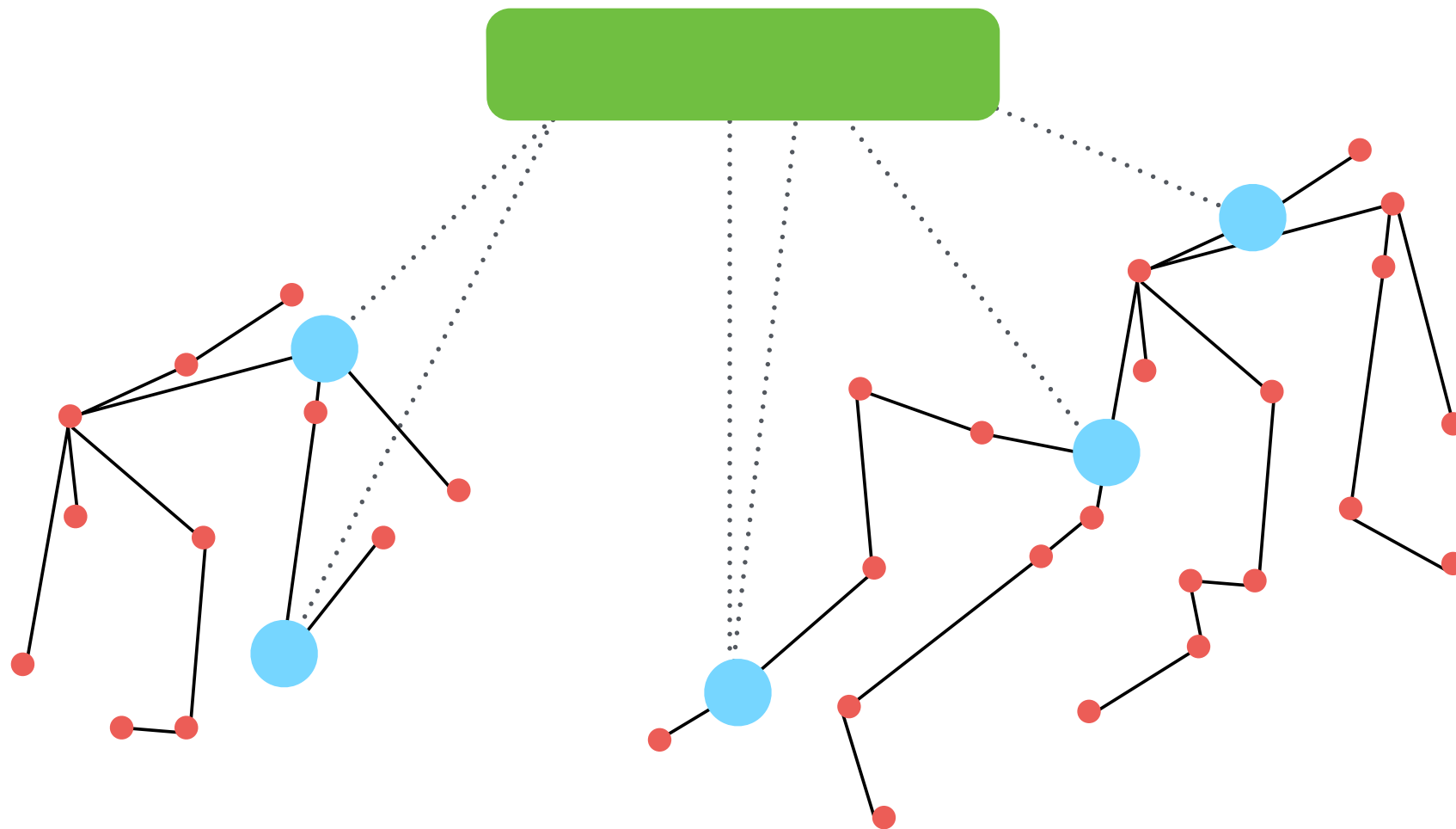
t “points of contact” between S and $G \setminus S$...

at least t edges going across the two sets.



t “points of contact” between S and $G \setminus S$... at least t edges going across the two sets.

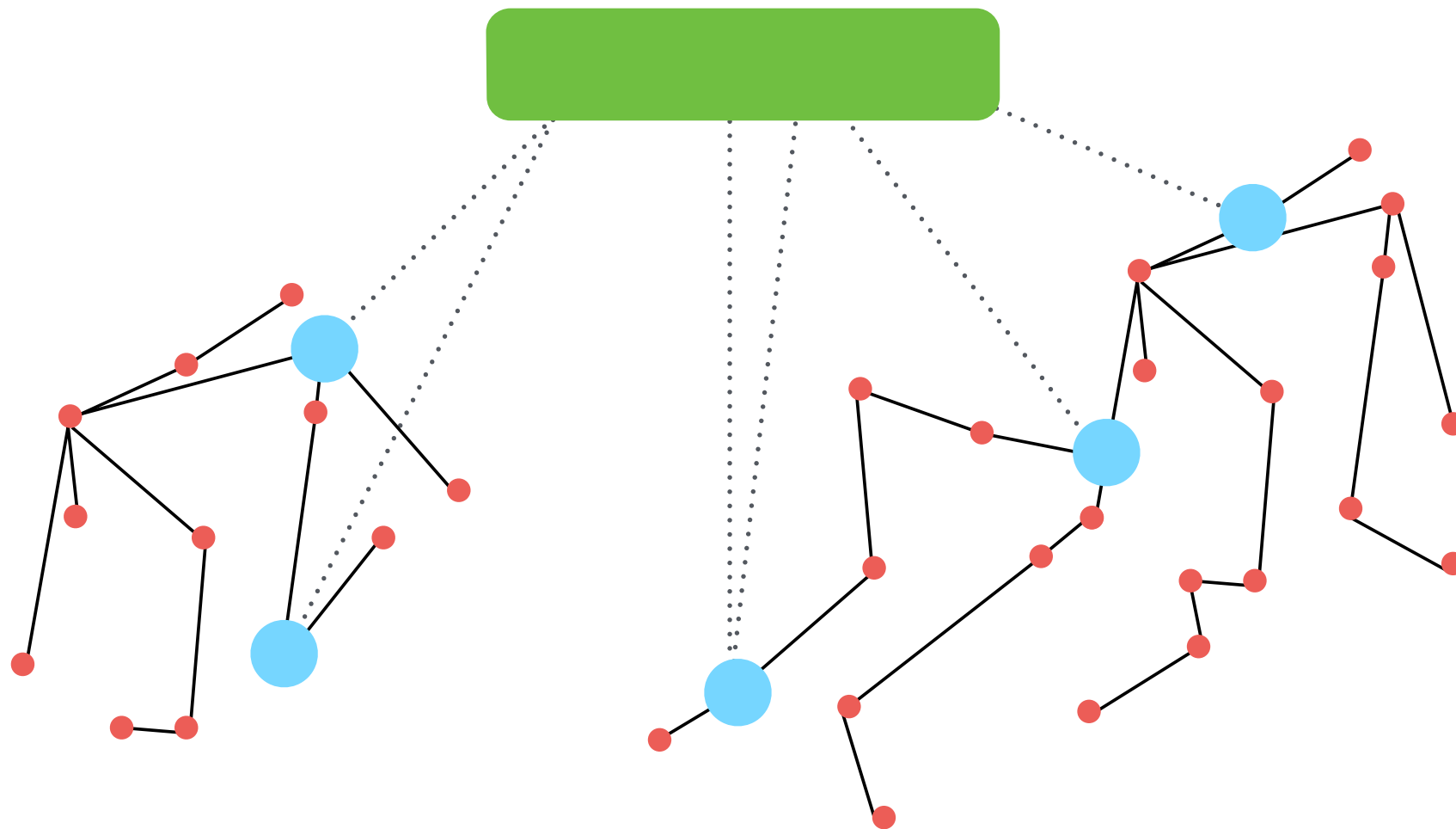
Delete the points of contact in $G \setminus S$ to get at most $5t$ “pieces”.



t “points of contact” between S and $G \setminus S$... at least t edges going across the two sets.

Delete the points of contact in $G \setminus S$ to get at most $5t$ “pieces”.

Suppose each piece has constant size, say c .

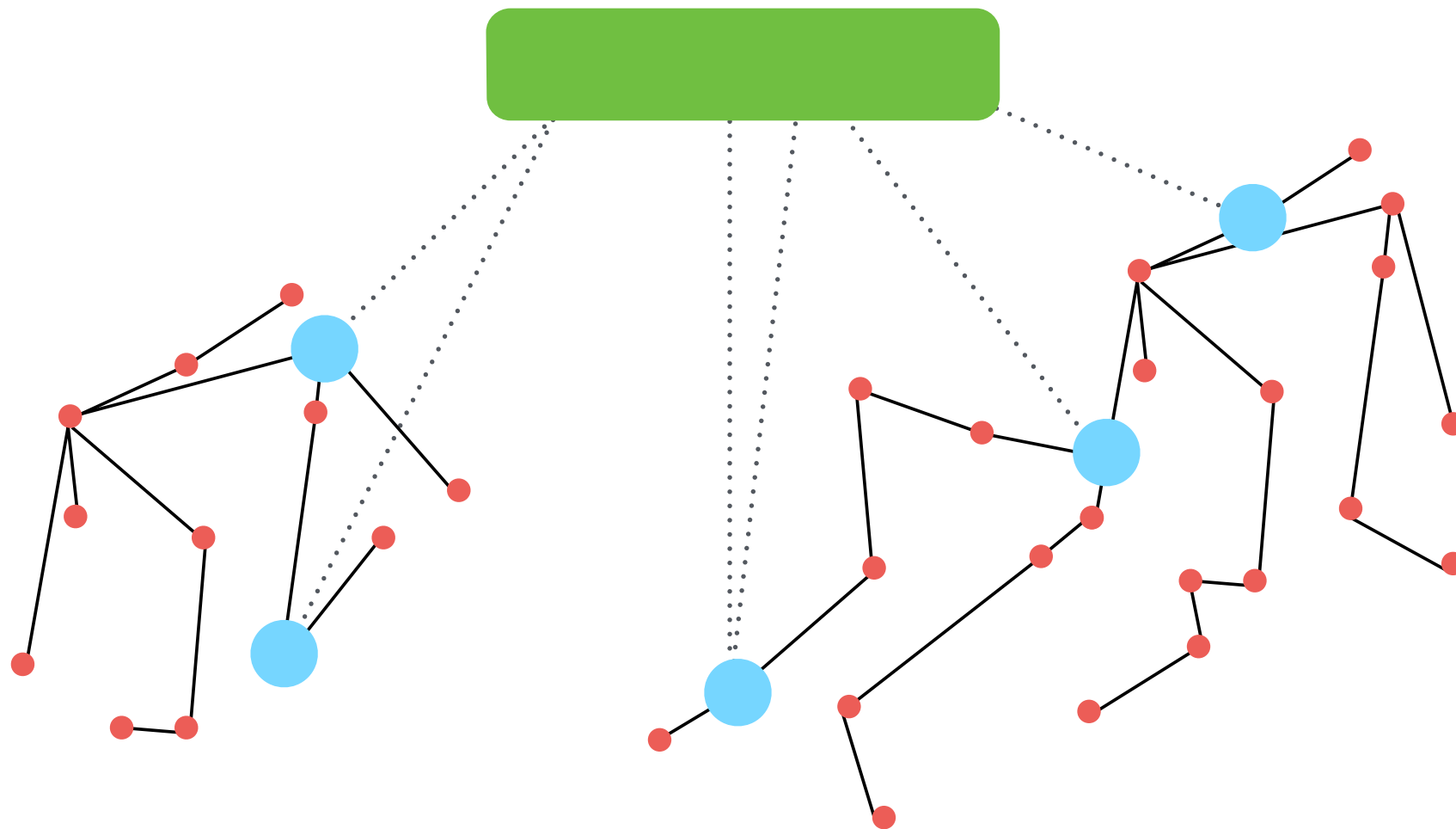


t “points of contact” between S and $G \setminus S$... at least t edges going across the two sets.

Delete the points of contact in $G \setminus S$ to get at most $5t$ “pieces”.

Suppose each piece has constant size, say c .

The total number of edges in $G \setminus S$ is at most $5ct$.



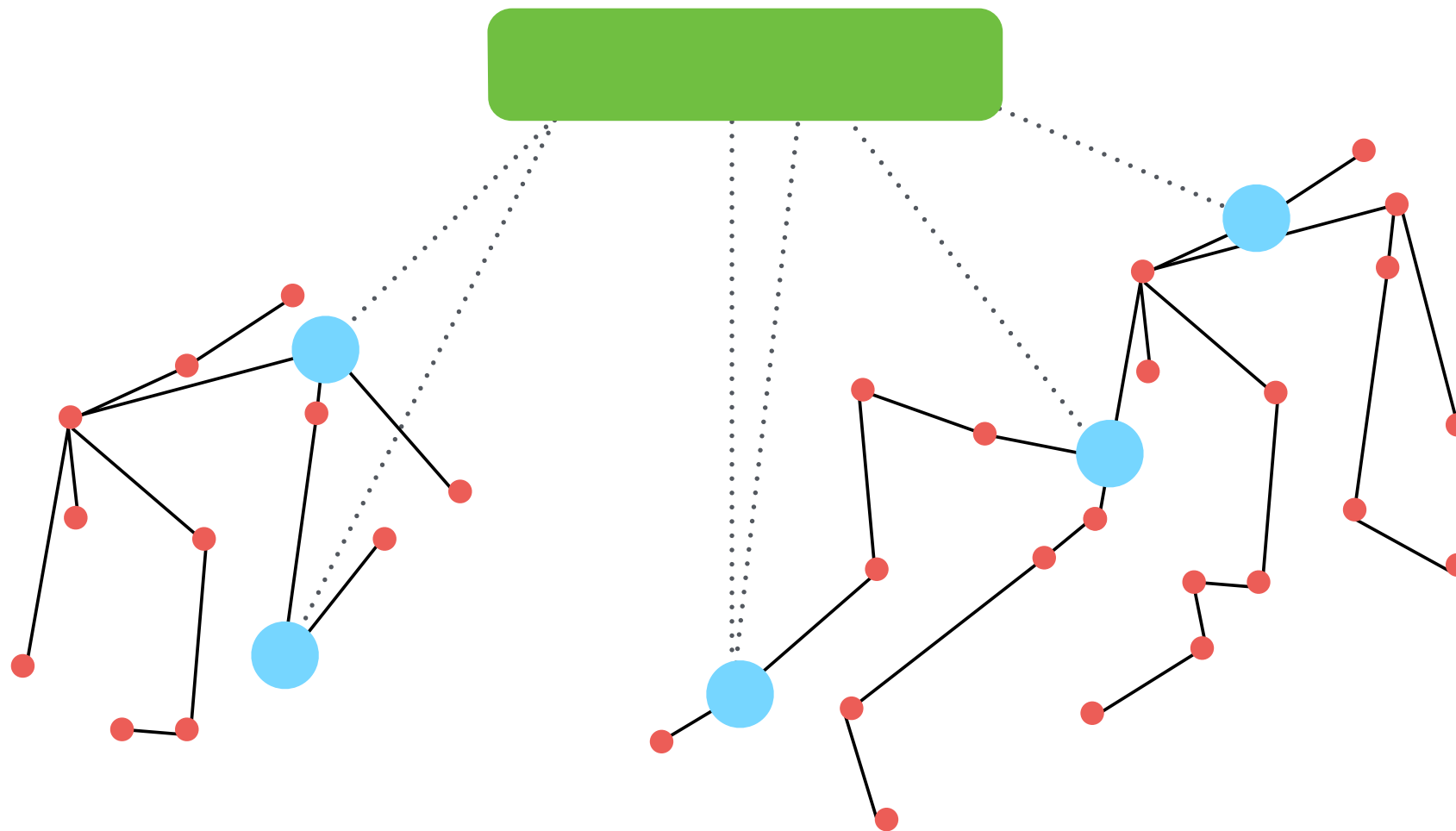
t “points of contact” between S and $G \setminus S$... at least t edges going across the two sets.

Delete the points of contact in $G \setminus S$ to get at most $5t$ “pieces”.

Suppose each piece has constant size, say c .

The total number of edges in $G \setminus S$ is at most $5ct$.

The number of **good edges** is at least t , and the number of **bad edges** is at most $5ct$.



t “points of contact” between S and $G \setminus S$... at least t edges going across the two sets.

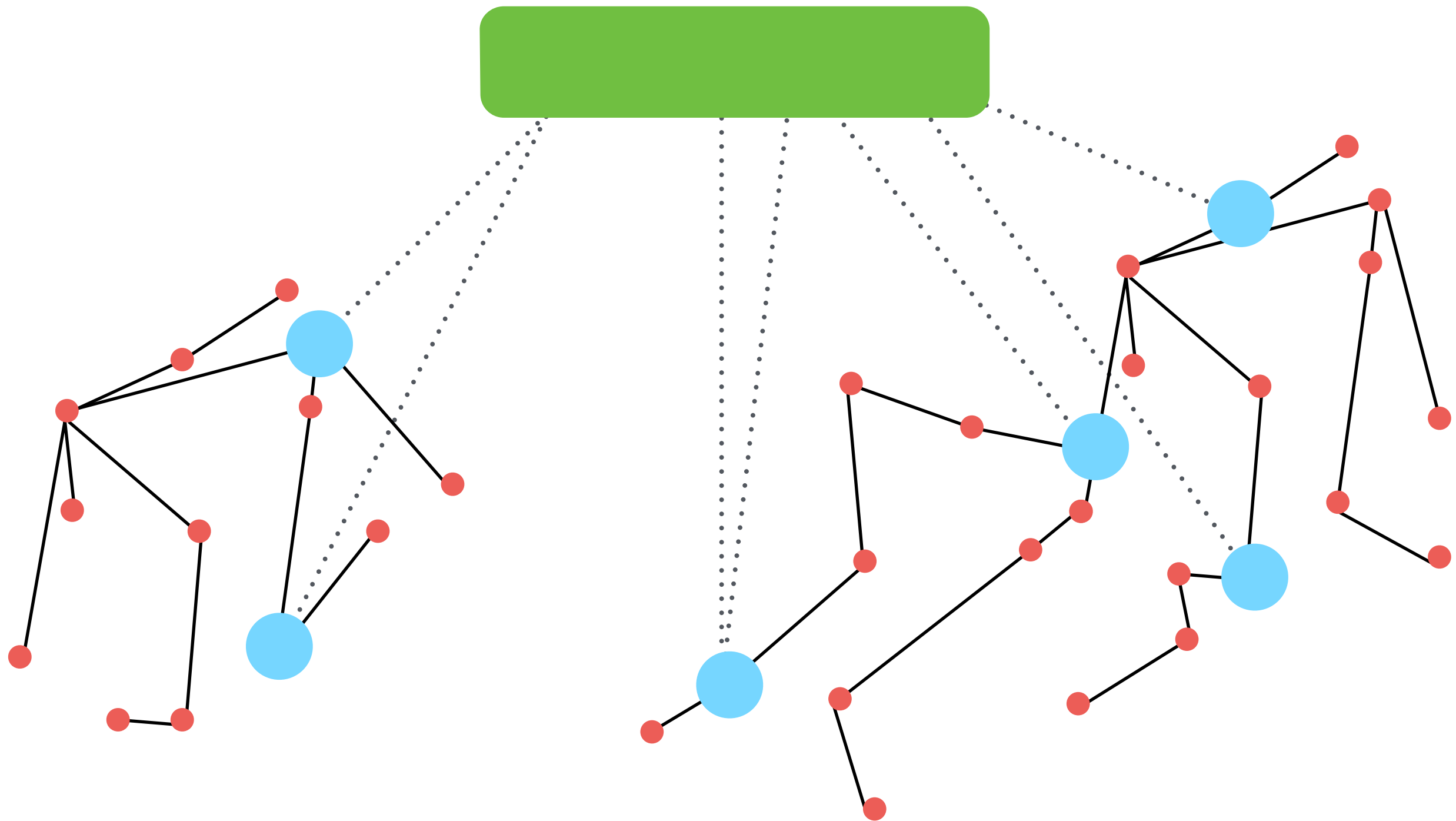
Delete the points of contact in $G \setminus S$ to get at most $5t$ “pieces”.

Suppose each piece has constant size, say c .

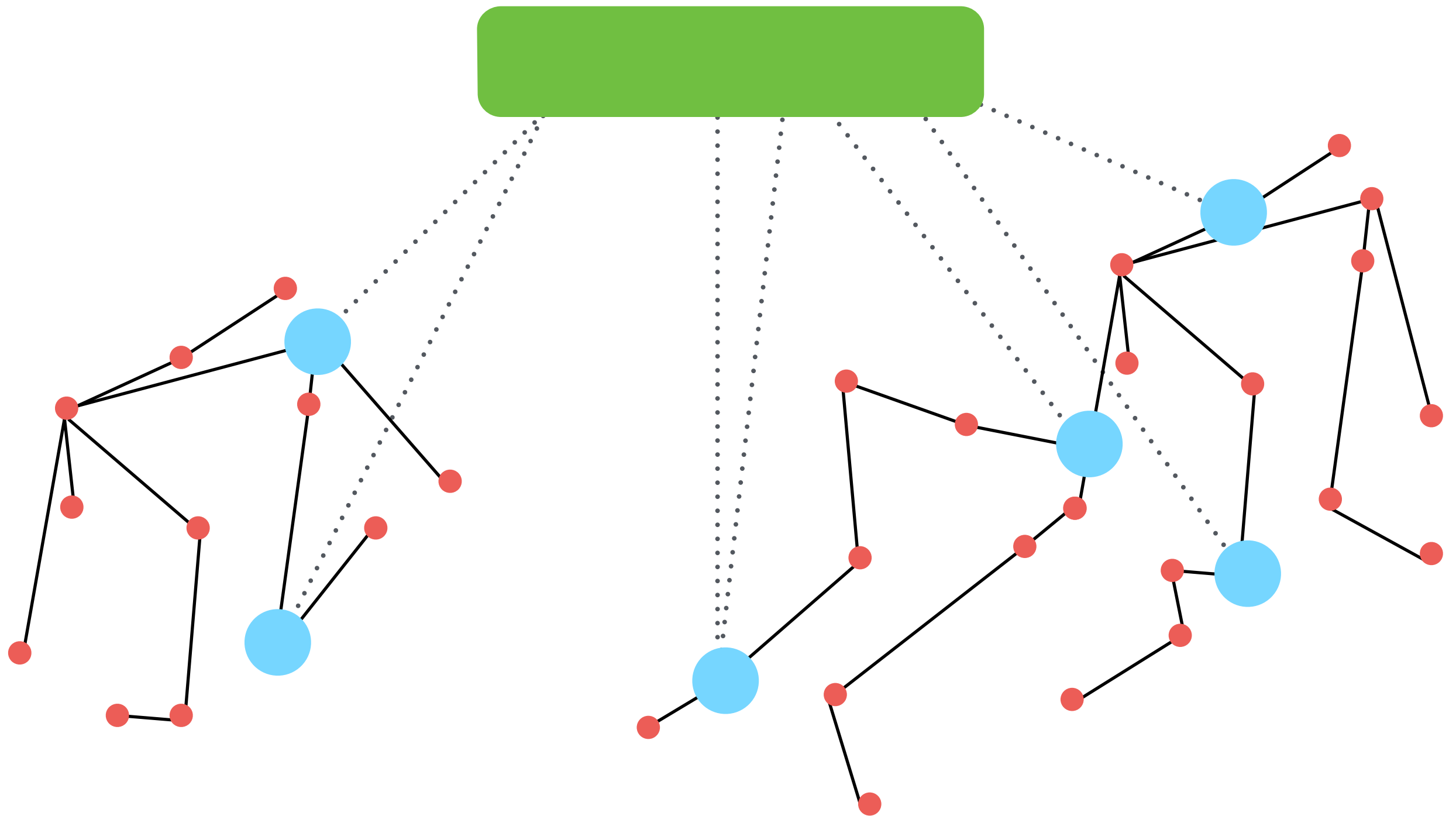
The total number of edges in $G \setminus S$ is at most $5ct$.

The number of **good edges is at least t** , and the number of **bad edges is at most $5ct$** .

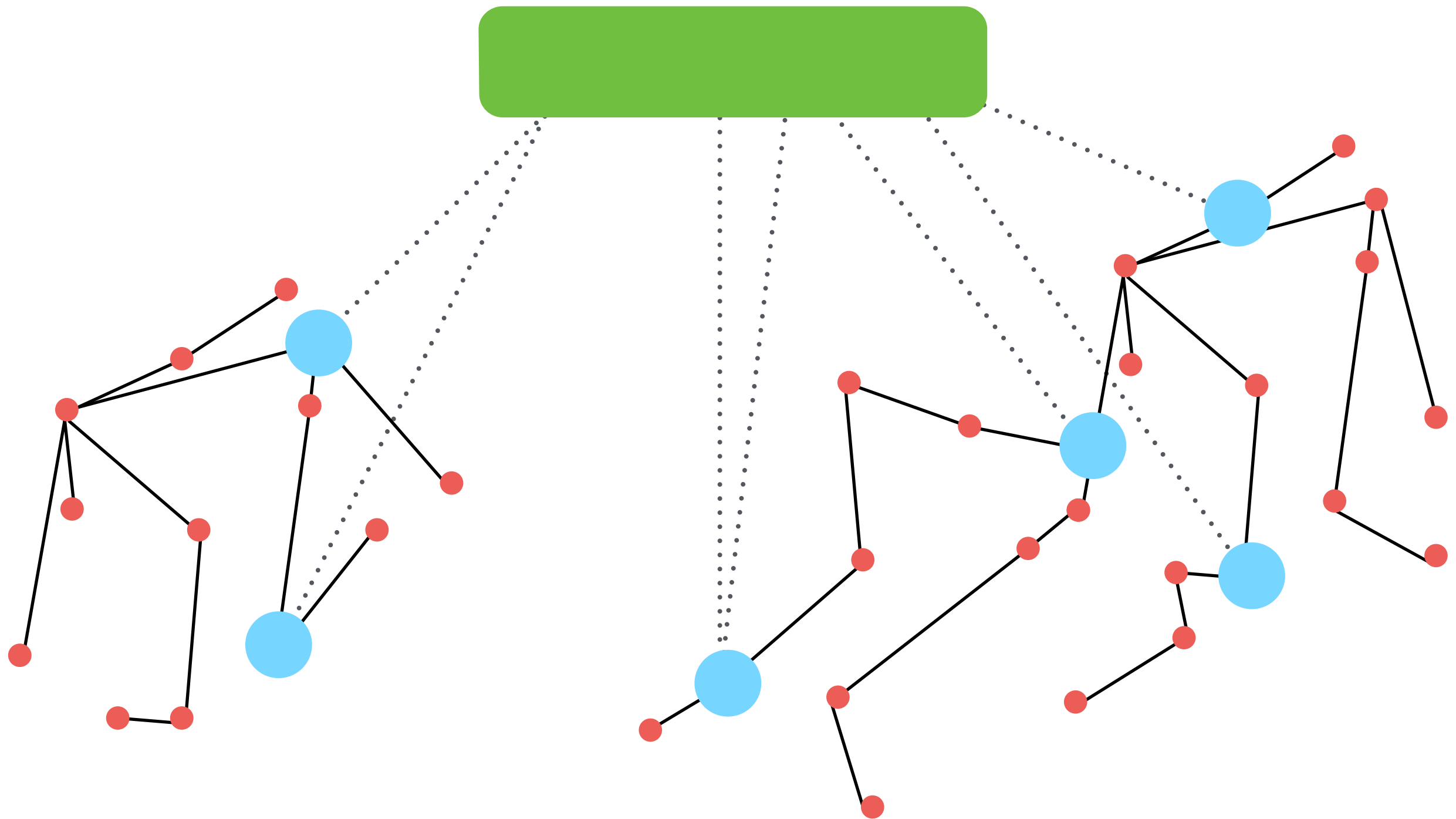
In this scenario, a randomly chosen edge will be **good with constant probability!**



Bad scenario: **one of the pieces is large.**

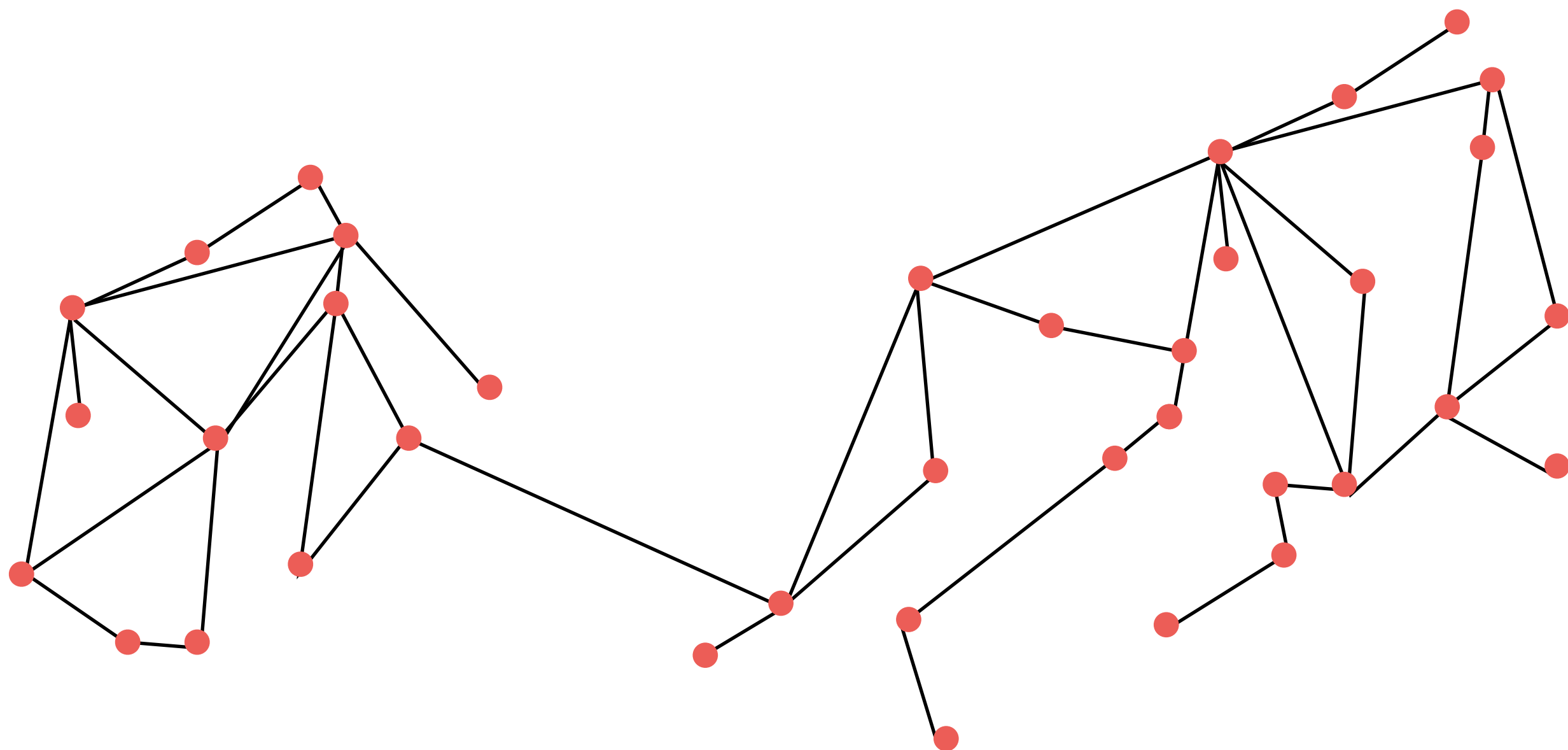


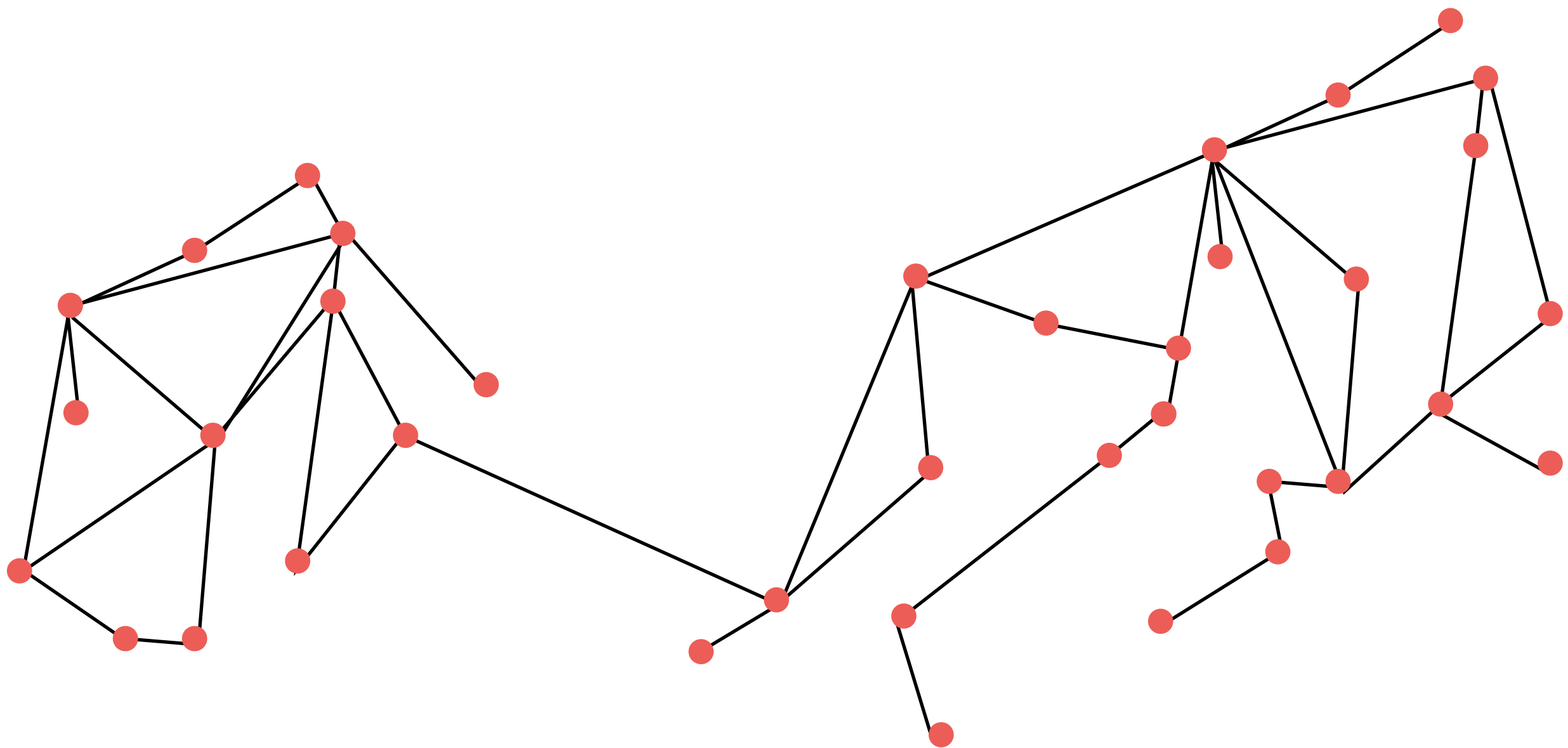
These pieces have **simple structure** and **bounded interaction** with the outer world.



These pieces have **simple structure** and **bounded interaction** with the outer world.

Such pieces are called **“protrusions”**.





Finding protrusions? What do we replace them with?

One Size Fits All?

A meta-algorithm for a class of graph-theoretic problems

The general notion of a “graph modification problem”.

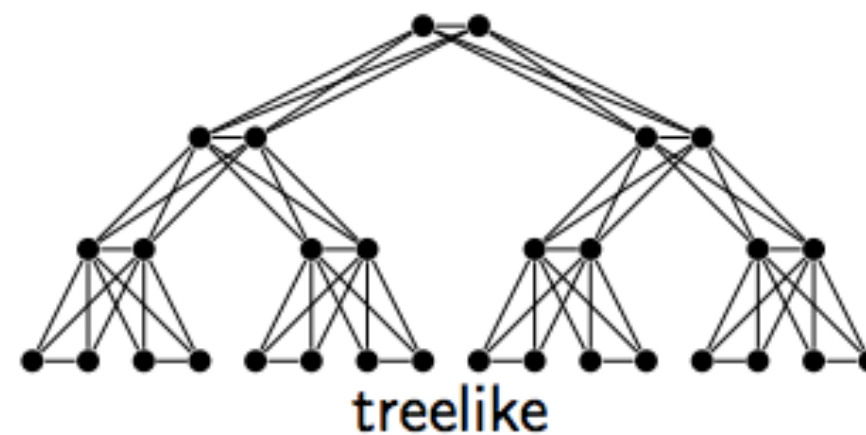
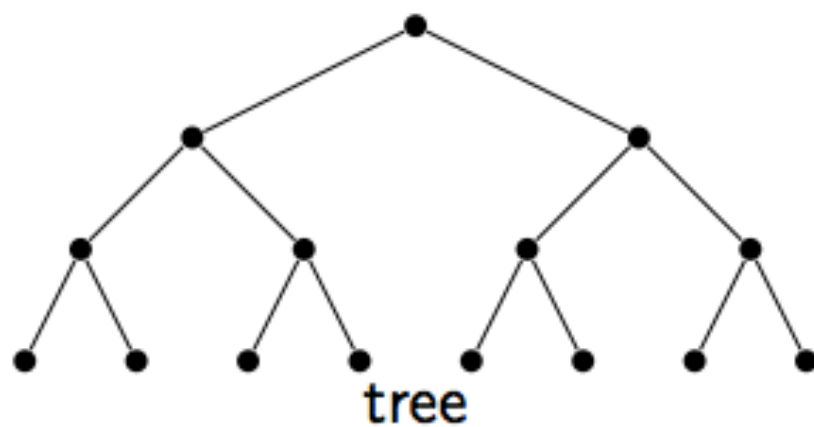
Implications of the Graph Minor Theorem.

Vertex Cover, Revisited.

Beyond Vertex Cover: Feedback Vertex Set
(on graphs of constant degree).

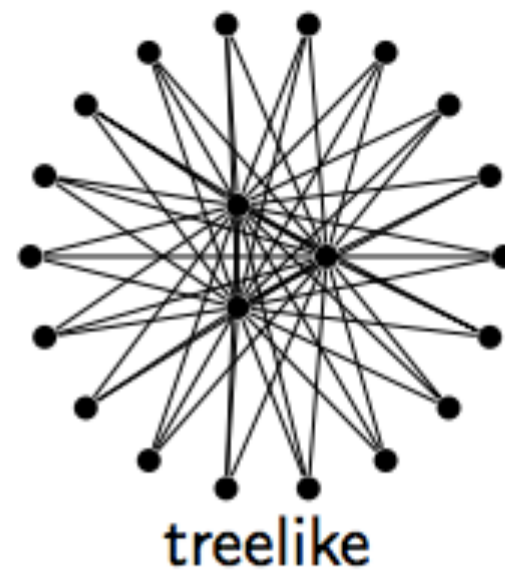
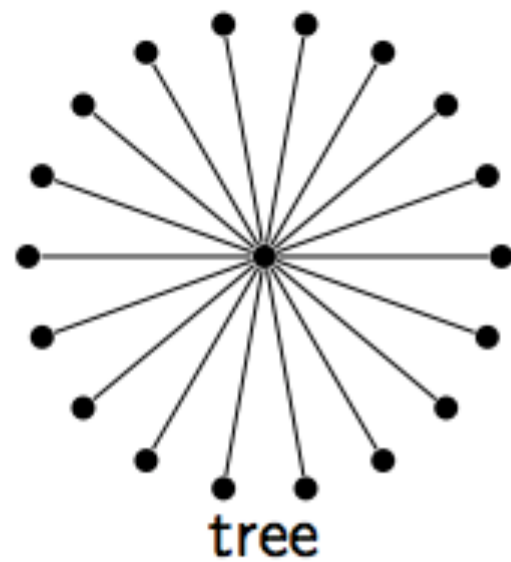
Planar F-Deletion.

The notion of treewidth – capturing “treelike” graphs



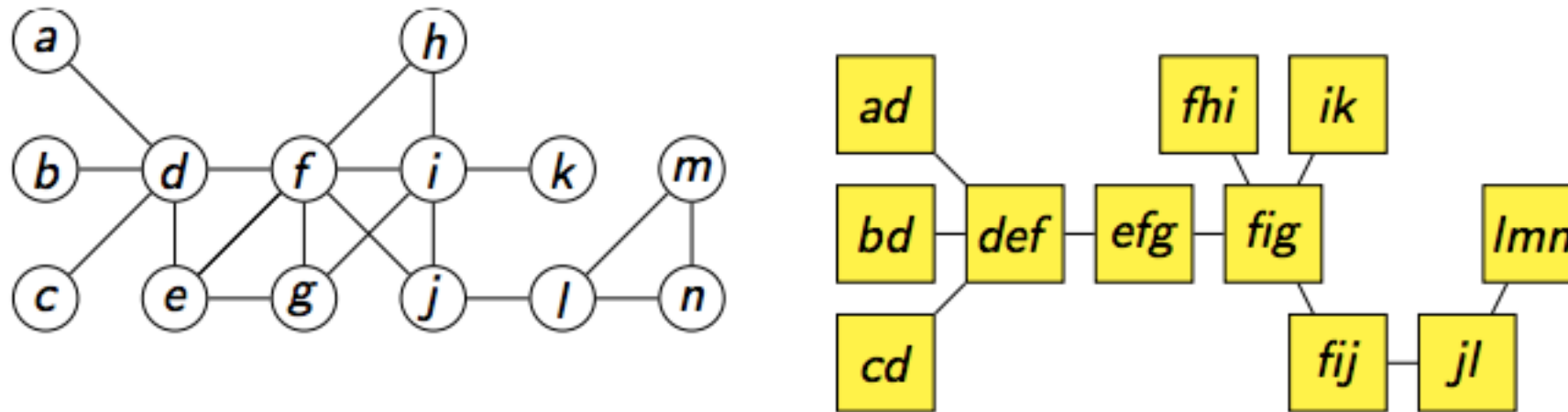
Pictures by Michal Pilipczuk

The notion of treewidth – capturing “treelike” graphs

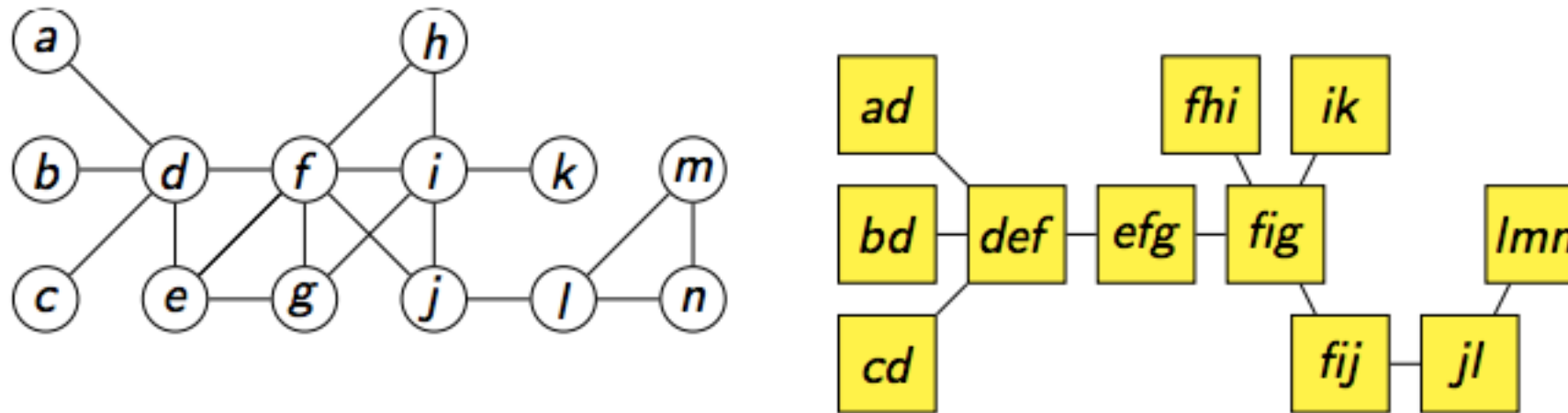


Pictures by Michal Pilipczuk

The notion of treewidth – capturing “treelike” graphs



The notion of treewidth – capturing “treelike” graphs



A *tree decomposition* of a graph G is a pair (T, β) where T is a tree and $\beta : V(T) \rightarrow 2^{V(G)}$ satisfying:

- ① $\{t : v \in \beta(t)\}$ is nonempty and connected for every $v \in V(G)$;
- ② for every $uv \in E(G)$ there exists $t \in V(T)$ such that $u, v \in \beta(t)$.

Width of the decomposition is $\max_{t \in V(T)} |\beta(t)| - 1$.

Treewidth of G is minimum possible width of tree decomposition of G .

Can G be made H -minor free, for all H in $\mathcal{F}$,
by the removal of at most k vertices?

Can G be made H -minor free, for all H in $\mathcal{F}$,
by the removal of at most k vertices?

Suppose G is a **YES-instance**, and further, suppose $\mathcal{F}$ contains **at least one planar graph**.

Can G be made H -minor free, for all H in $\mathcal{F}$,
by the removal of at most k vertices?

Suppose G is a **YES-instance**, and further, suppose $\mathcal{F}$ contains **at least one planar graph**.

Then $G \setminus S$ cannot have large grids.

Can G be made H -minor free, for all H in $\mathcal{F}$,
by the removal of at most k vertices?

Suppose G is a **YES-instance**, and further, suppose $\mathcal{F}$ contains **at least one planar graph**.

Then $G \setminus S$ cannot have large grids.

Thus the treewidth of $G \setminus S$ is **bounded by a constant** that depends only on $\mathcal{F}$.

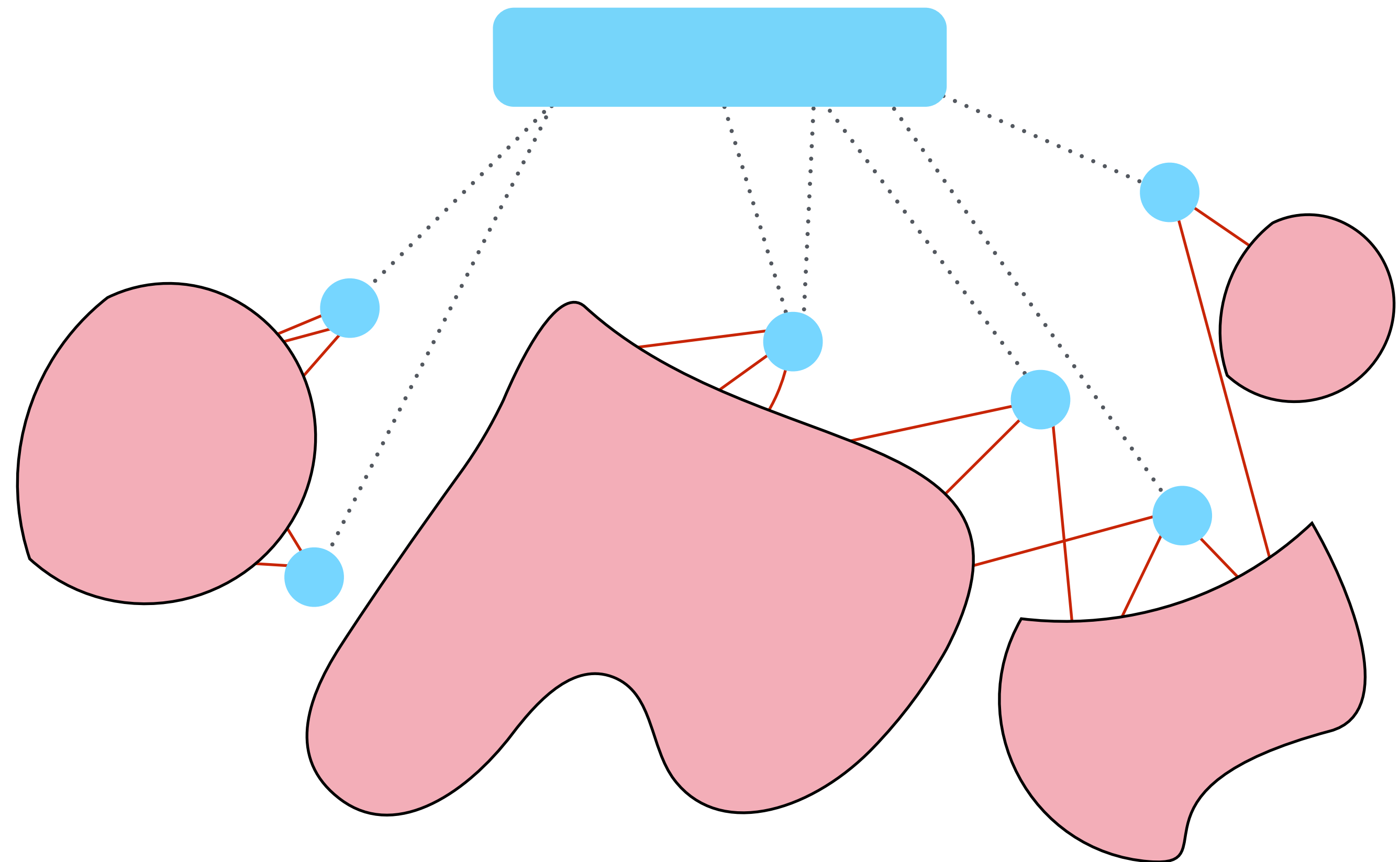
Can G be made H -minor free, for all H in $\mathcal{F}$,
by the removal of at most k vertices?

Suppose G is a **YES-instance**, and further, suppose $\mathcal{F}$ contains **at least one planar graph**.

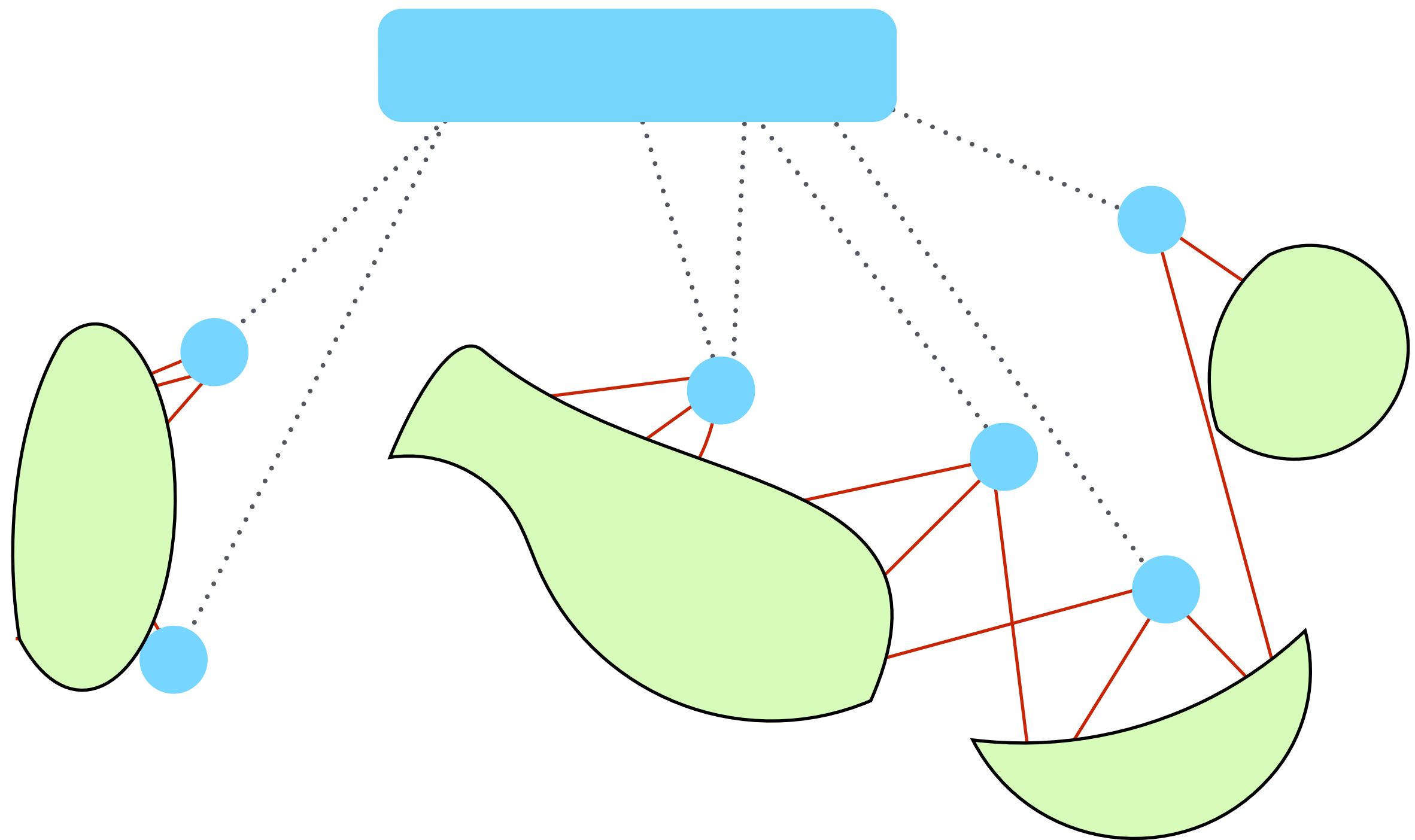
Then $G \setminus S$ cannot have large grids.

Thus the treewidth of $G \setminus S$ is **bounded by a constant** that depends only on $\mathcal{F}$.

The Planar $\mathcal{F}$ -Deletion Problem



$G \setminus S$: no copies of graphs from $\mathcal{F}$ here.



$G \setminus S$: no copies of graphs from $\mathcal{F}$ here.

Planar $\mathcal{F}$ -Deletion: The Algorithm

Planar $\mathcal{F}$ -Deletion: The Algorithm

Ensure that protrusions are removed.

Planar $\mathcal{F}$ -Deletion: The Algorithm

Ensure that protrusions are removed.

Pick an edge uniformly at random.

Planar $\mathcal{F}$ -Deletion: The Algorithm

Ensure that protrusions are removed.

Pick an edge uniformly at random.

Include both endpoints.

Planar $\mathcal{F}$ -Deletion: The Algorithm

Ensure that protrusions are removed.

Pick an edge uniformly at random.

Include both endpoints.

Pick an endpoint u.a.r.

Planar $\mathcal{F}$ -Deletion: The Algorithm

Ensure that protrusions are removed.

Pick an edge uniformly at random.

Include both endpoints.

Pick an endpoint u.a.r.

Repeat till G is $\mathcal{F}$ -free.

Planar $\mathcal{F}$ -Deletion: The Algorithm

Ensure that protrusions are removed.

Pick an edge uniformly at random.

Include both endpoints.

Pick an endpoint u.a.r.

(Approximation algorithm.)

Repeat till G is $\mathcal{F}$ -free.

Planar $\mathcal{F}$ -Deletion: The Algorithm

Ensure that protrusions are removed.

Pick an edge uniformly at random.

Include both endpoints.

(Approximation algorithm.)

Pick an endpoint u.a.r.

(Randomized algorithm.)

Repeat till G is $\mathcal{F}$ -free.

The Planar $\mathcal{F}$ -Deletion Problem
accounts for several specific problems...

The Planar $\mathcal{F}$ -Deletion Problem accounts for several specific problems...

$\mathcal{F}$	Corresponding Deletion Problem
K_2	Vertex Cover
K_3	Feedback Vertex Set
$((k+1)\text{-by-}(k+1))$ Grids	Treewidth k
$K_{2,3}, K_4$	Outerplanar
K_4	Series-Parallel
K_{3,T_2}	Pathwidth-One

Thanks to **Graph Minors**, we have a $f(k)n^2$ algorithm for the $\mathcal{F}$ -Deletion problem.

Thanks to **Graph Minors**, we have a $f(k)n^2$ algorithm for the $\mathcal{F}$ -Deletion problem.

For **Planar** $\mathcal{F}$ -Deletion, the starting point was a double-exponential algorithm.
[Bodlaender, 1997]

Thanks to **Graph Minors**, we have a $f(k)n^2$ algorithm for the $\mathcal{F}$ -Deletion problem.

For **Planar** $\mathcal{F}$ -Deletion, the starting point was a double-exponential algorithm.
[Bodlaender, 1997]

Many single-exponential algorithms are known for special cases of $\mathcal{F}$.

Thanks to **Graph Minors**, we have a $f(k)n^2$ algorithm for the $\mathcal{F}$ -Deletion problem.

For **Planar** $\mathcal{F}$ -Deletion, the starting point was a double-exponential algorithm.
[Bodlaender, 1997]

Many single-exponential algorithms are known for special cases of $\mathcal{F}$.

Cao, Chen & Liu
[2010]

Feedback Vertex Set has a $O^*(3.83^k)$ algorithm.

Thanks to **Graph Minors**, we have a $f(k)n^2$ algorithm for the $\mathcal{F}$ -Deletion problem.

For **Planar** $\mathcal{F}$ -Deletion, the starting point was a double-exponential algorithm.
[Bodlaender, 1997]

Many single-exponential algorithms are known for special cases of $\mathcal{F}$.

Cao, Chen & Liu
[2010]

Feedback Vertex Set has a $O^*(3.83^k)$ algorithm.

Kim, Paul & Philip
[2012]

$\{K_4\}$ -Deletion has a $O^*(2^{O(k)})$ algorithm.

Thanks to **Graph Minors**, we have a $f(k)n^2$ algorithm for the $\mathcal{F}$ -Deletion problem.

For **Planar** $\mathcal{F}$ -Deletion, the starting point was a double-exponential algorithm.
[Bodlaender, 1997]

Many single-exponential algorithms are known for special cases of $\mathcal{F}$.

Cao, Chen & Liu
[2010]

Feedback Vertex Set has a $O^*(3.83^k)$ algorithm.

Kim, Paul & Philip
[2012]

$\{K_4\}$ -Deletion has a $O^*(2^{O(k)})$ algorithm.

Cygan, Pilipczuk,
Pilipczuk & Wojtaszczyk
[2010]

Pathwidth-1-Deletion has a $O^*(4.65^k)$ algorithm.

Fomin, Lokshantov,
M. & Saurabh
[FOCS; 2012]

Planar $\mathcal{F}$ -Deletion admits a randomized $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Fomin, Lokshantov,
M. & Saurabh
[FOCS; 2012]

Planar $\mathcal{F}$ -Deletion admits a randomized $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Best possible under ETH.

Fomin, Lokshantov,
M. & Saurabh
[FOCS; 2012]

Planar $\mathcal{F}$ -Deletion admits a randomized $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Best possible under ETH.

Planar $\mathcal{F}$ -Deletion admits a deterministic $(2^{O(k)}n \log^2 n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Fomin, Lokshantov,
M. & Saurabh
[FOCS; 2012]

Planar $\mathcal{F}$ -Deletion admits a randomized $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Best possible under ETH.

Planar $\mathcal{F}$ -Deletion admits a deterministic $(2^{O(k)}n \log^2 n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Planar $\mathcal{F}$ -Deletion admits an $O(nm)$ randomized algorithm
that leads us to a **constant-factor approximation**.

*Can we get a deterministic
constant-factor approximation algorithm?*

Fomin, Lokshtanov,
M. & Saurabh
[FOCS; 2012]

Planar $\mathcal{F}$ -Deletion admits a randomized $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Best possible under ETH.

Planar $\mathcal{F}$ -Deletion admits a deterministic $(2^{O(k)}n \log^2 n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Planar $\mathcal{F}$ -Deletion admits an $O(nm)$ randomized algorithm
that leads us to a **constant-factor approximation**.

*Can we get a deterministic
constant-factor approximation algorithm?*

Fomin, Lokshtanov,
M., Philip & Saurabh
[STACS; 2011]

Planar $\mathcal{F}$ -Deletion admits an deterministic algorithm
that leads us to a **$O(\log^{3/2}(\text{OPT}))$** approximation.

Fomin, Lokshtanov,
M. & Saurabh
[FOCS; 2012]

Planar $\mathcal{F}$ -Deletion admits a randomized $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Best possible under ETH.

Planar $\mathcal{F}$ -Deletion admits a deterministic $(2^{O(k)}n \log^2 n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Planar $\mathcal{F}$ -Deletion admits an $O(nm)$ randomized algorithm
that leads us to a **constant-factor approximation**.

*Can we get a deterministic
constant-factor approximation algorithm?*

Fomin, Lokshtanov,
M., Philip & Saurabh
[STACS; 2011]

Planar $\mathcal{F}$ -Deletion admits a deterministic algorithm
that leads us to a $O(\log^{3/2}(\text{OPT}))$ approximation.

Kim, Langer, Paul, Reidl,
Rossmann, Sau, & Sikdar
[ICALP; 2013]

Planar $\mathcal{F}$ -Deletion admits a deterministic $(2^{O(k)}n^2)$ algorithm.

Fomin, Lokshtanov,
M. & Saurabh
[FOCS; 2012]

Planar $\mathcal{F}$ -Deletion admits a randomized $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Best possible under ETH.

Fomin, Lokshtanov,
M., Ramanujan & Saurabh
[SODA; 2015]

Planar $\mathcal{F}$ -Deletion admits a deterministic $(2^{O(k)}n)$ algorithm
when every graph in $\mathcal{F}$ is connected.

Planar $\mathcal{F}$ -Deletion admits an $O(nm)$ randomized algorithm
that leads us to a **constant-factor approximation**.

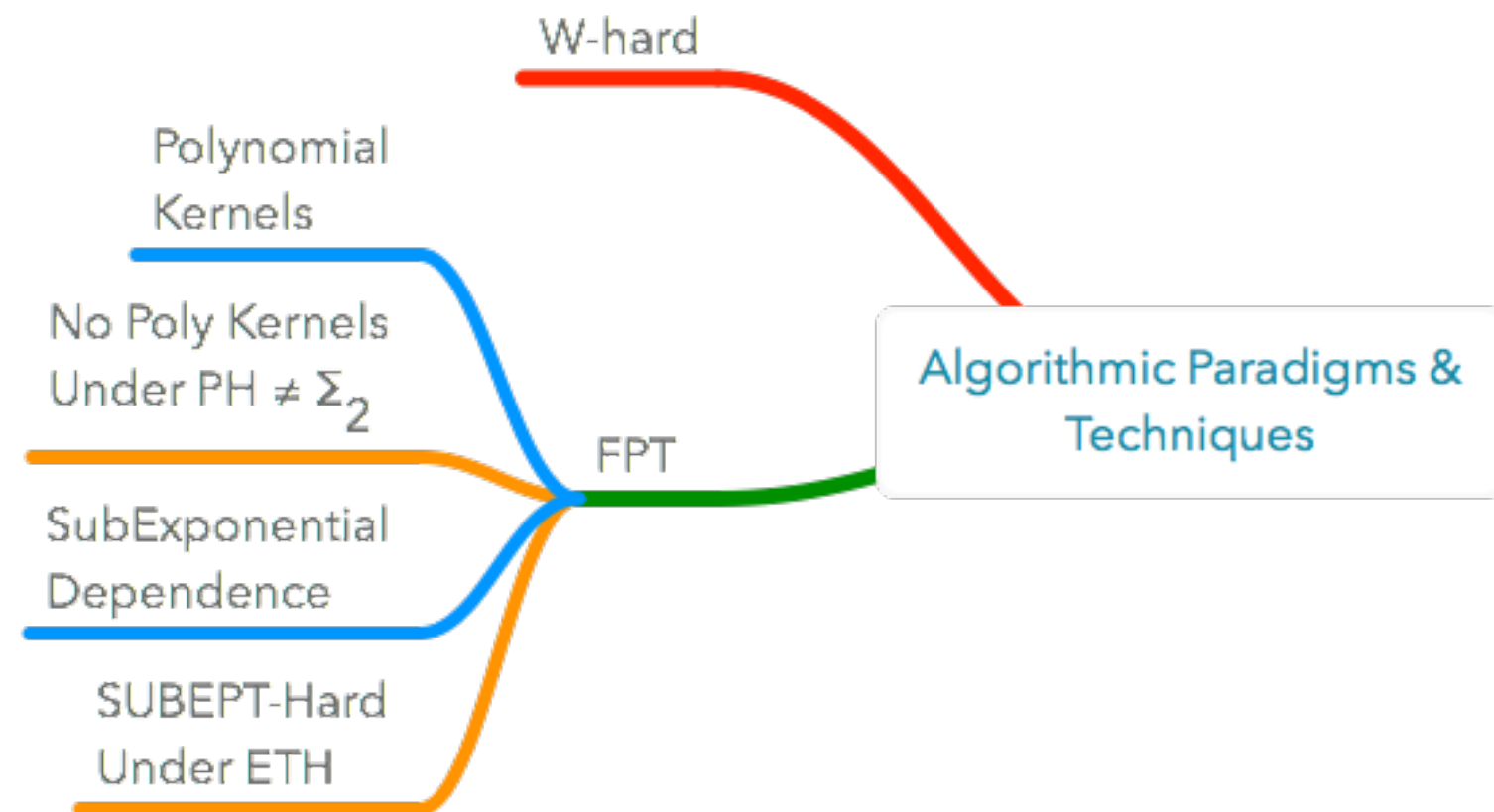
*Can we get a deterministic
constant-factor approximation algorithm?*

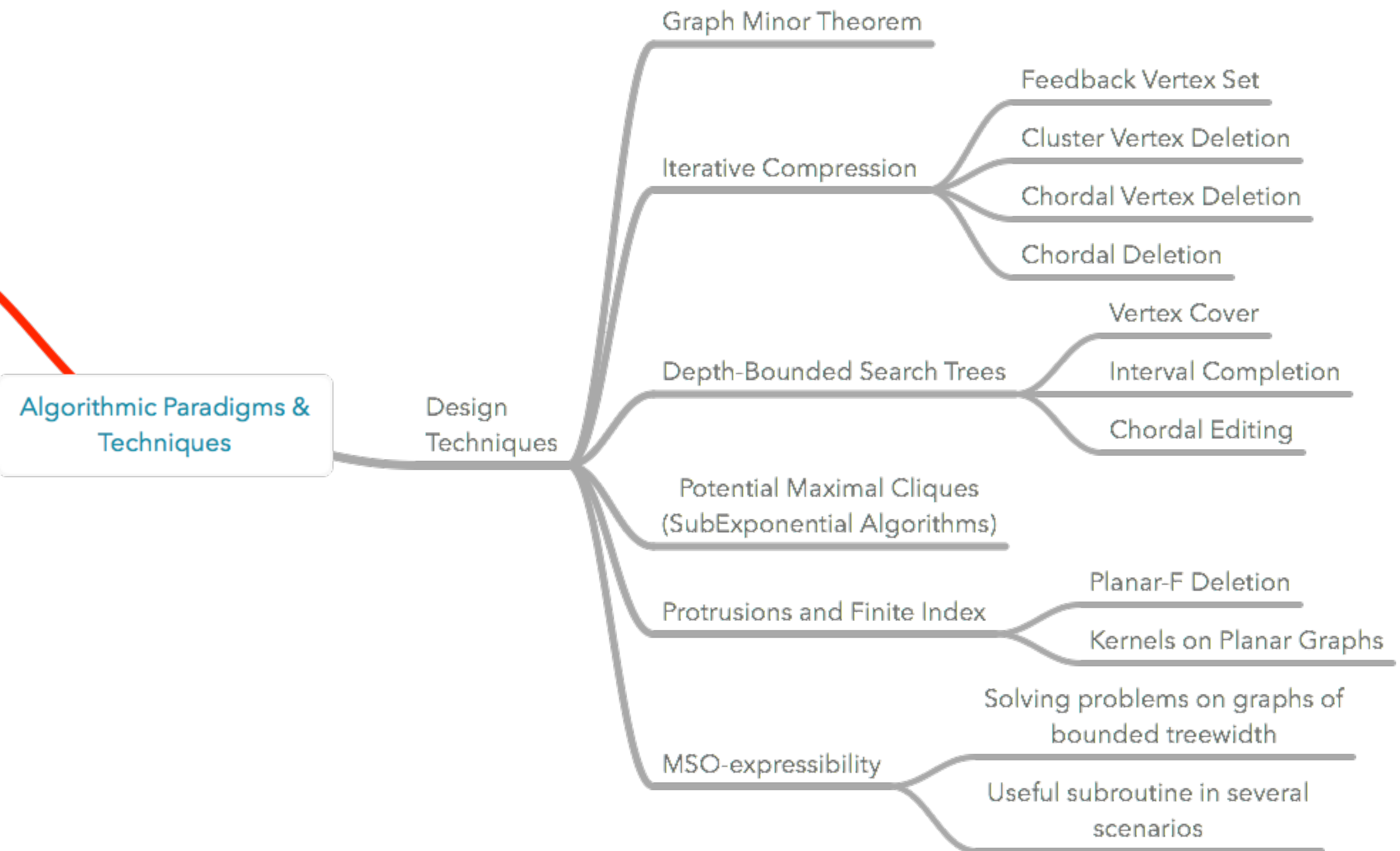
Fomin, Lokshtanov,
M., Philip & Saurabh
[STACS; 2011]

Planar $\mathcal{F}$ -Deletion admits an deterministic algorithm
that leads us to a **$O(\log^{3/2}(\text{OPT}))$** approximation.

Kim, Langer, Paul, Reidl,
Rossmanith, Sau, & Sikdar
[ICALP; 2013]

Planar $\mathcal{F}$ -Deletion admits a deterministic $(2^{O(k)}n^2)$ algorithm.





Parameterized Graph Modification: A Modern Perspective

Parameterized Graph Modification: A Modern Perspective

