# Submodular partition functions and duality treewidth/bramble

Omid Amini[1] Frédéric Mazoit[2] Nicolas Nisse[3]
Stéphan Thomassé[4]

[1]Projet Mascotte, INRIA Sophia Antipolis.

[2]LABRI, Université Bordeaux.

[3]LRI, Université Paris-Sud.

[4]LIRMM, Université Montpellier II.

Séminaire équipe GraphComb, LRI, 25 mai 2007

# Plan

# Min-Max Theorem for several width parameters

## Our goal

Duality treewidth/bramble [Seymour and Thomas 93]
New proof of the min-max theorem for treewidth

## Our tool

Submodular partition functions

## Generalization

Interpretation of several width-parameters (treewidth, pathwidth, branchwidth, rankwidth, treewidth of matroid) in terms of submodular partition functions.

# Min-Max Theorem for several width parameters

## Our goal

Duality treewidth/bramble [Seymour and Thomas 93]
New proof of the min-max theorem for treewidth

## Our tool

Submodular partition functions

## Generalization

Interpretation of several width-parameters (treewidth, pathwidth, branchwidth, rankwidth, treewidth of matroid) in terms of submodular partition functions.

# Min-Max Theorem for several width parameters

## Our goal

Duality treewidth/bramble [Seymour and Thomas 93]
New proof of the min-max theorem for treewidth

## Our tool

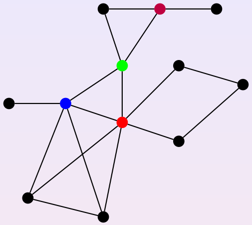
Submodular partition functions

## Generalization

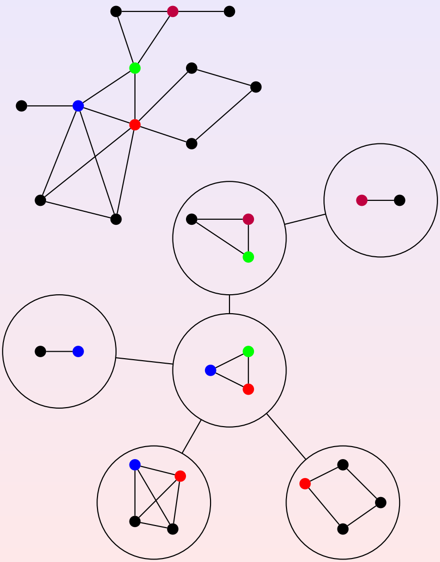
Interpretation of several width-parameters (treewidth, pathwidth, branchwidth, rankwidth, treewidth of matroid) in terms of submodular partition functions.

# Plan

# Tree decomposition and treewidth

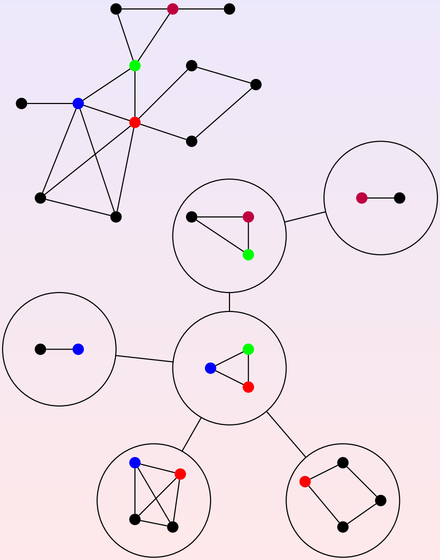


# Tree decomposition and treewidth



a tree $T$ and bags $(X_t)_{t \in V(T)}$

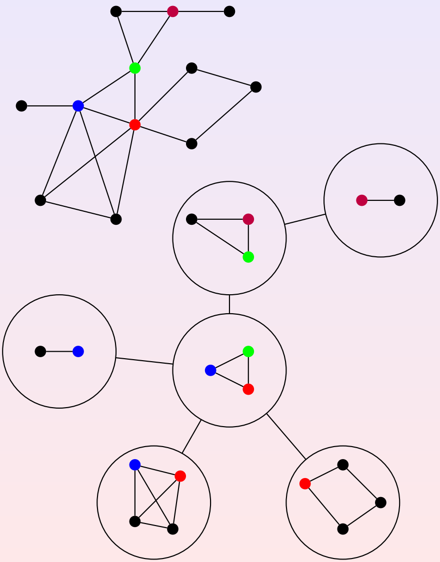
# Tree decomposition and treewidth



**a tree $T$ and bags $(X_t)_{t \in V(T)}$**

- every vertex of $G$ is in at least one bag;

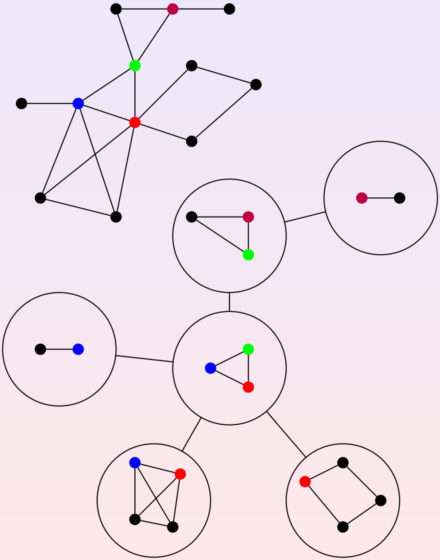
# Tree decomposition and treewidth



## a tree $T$ and bags $(X_t)_{t \in V(T)}$

- every **vertex** of $G$ is in at least one bag;
- both ends of an edge of $G$ are in at least one same bag;

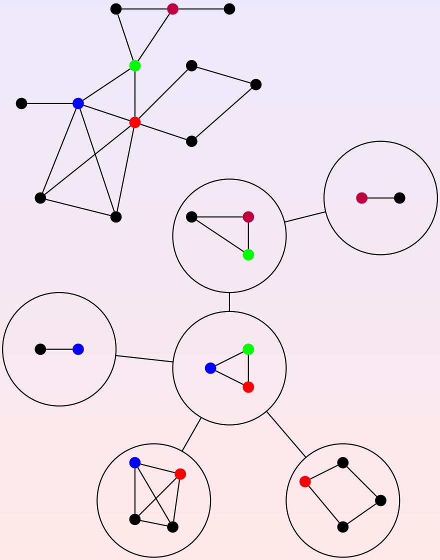
# Tree decomposition and treewidth



**a tree $T$ and bags $(X_t)_{t \in V(T)}$**

- every **vertex** of $G$ is in at least one bag;
- both ends of an **edge** of $G$ are in at least one same bag;
- for any vertex of $G$, all bags that contain it form a subtree.

# Tree decomposition and treewidth

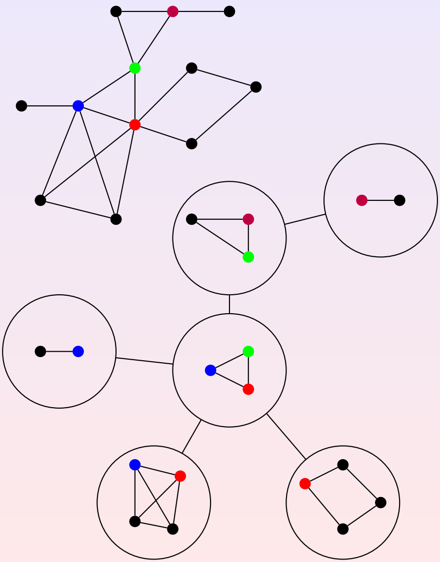


**a tree $T$ and bags $(X_t)_{t \in V(T)}$**

- every **vertex** of $G$ is in at least one bag;
- both ends of an **edge** of $G$ are in at least one same bag;
- for any vertex of $G$, all bags that contain it form a **subtree**.

**width = Size of largest Bag -1**

# Tree decomposition and treewidth



## a tree $T$ and bags $(X_t)_{t \in V(T)}$

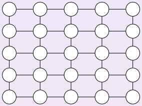
- every **vertex** of $G$ is in at least one bag;
- both ends of an **edge** of $G$ are in at least one same bag;
- for any vertex of $G$, all bags that contain it form a **subtree**.

**width = Size of largest Bag -1**

## treewidth of $G$

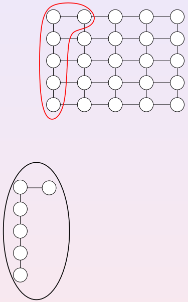
**tw**$(G)$, minimum width among all tree-decompositions.

# Example of the Grid $G_{k*k}$



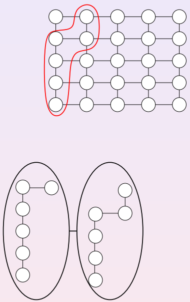
It is easy to find a tree-decomposition,

# Example of the Grid $G_{k*k}$



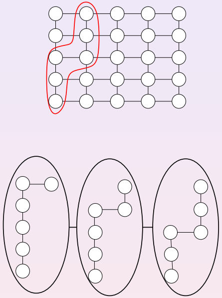
It is easy to find a tree-decomposition,

# Example of the Grid $G_{k*k}$



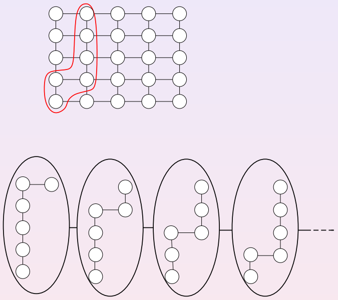
It is easy to find a tree-decomposition,

# Example of the Grid $G_{k*k}$



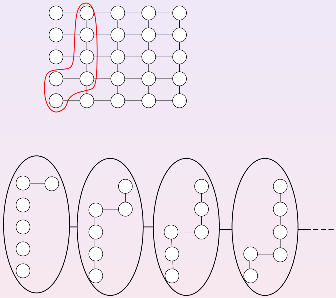
It is easy to find a tree-decomposition,

# Example of the Grid $G_{k*k}$



It is easy to find a tree-decomposition, $\mathbf{tw}(G_{k*k}) \leq k$

# Example of the Grid $G_{k*k}$



It is easy to find a tree-decomposition, $\mathbf{tw}(G_{k*k}) \leq k$
How to prove that it is an optimal tree-decomposition?

# An application: the graph searching problem

## The game

A team of searchers is aiming at catching a visible fugitive on the vertices of a graph $G$.

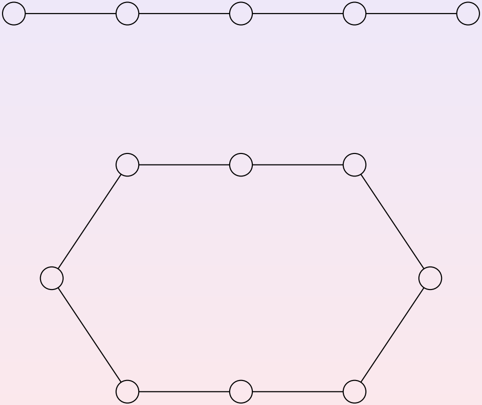
- a searcher can be placed at or removed from a vertex;
- the fugitive can move arbitrary fast along the paths of the graph, if it does not meet any searcher.

The fugitive is caught if it stands at a vertex occupied by a searcher.

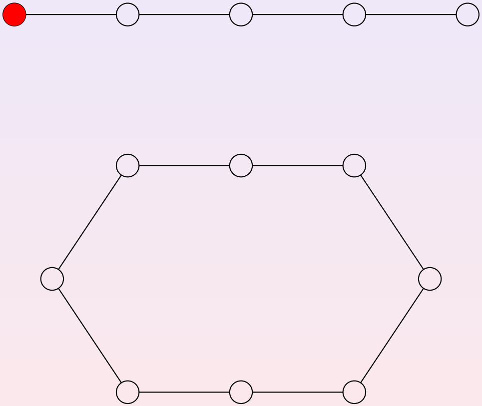
## The graph searching problem

We want to minimize the number of searchers required to catch any fugitive in $G$.

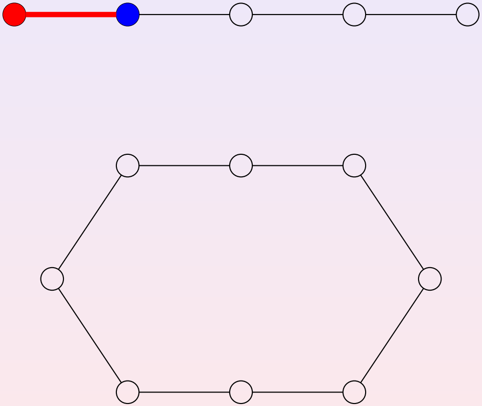
# Simple Examples : Path and Ring



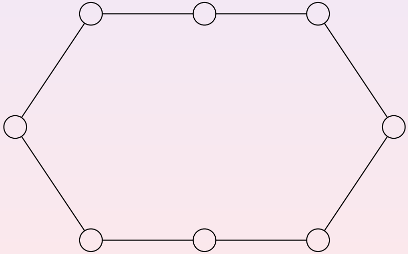
# Simple Examples : Path and Ring



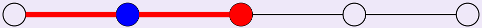
# Simple Examples : Path and Ring

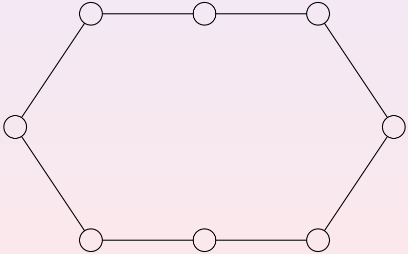


# Simple Examples : Path and Ring

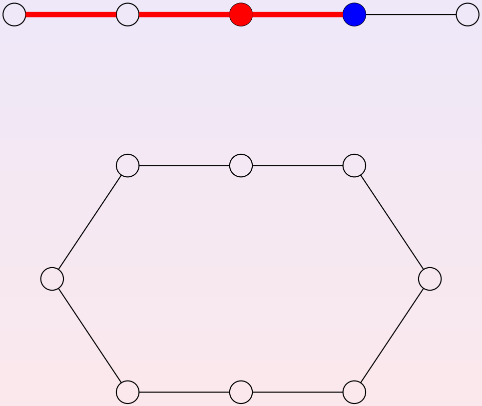


# Simple Examples : Path and Ring



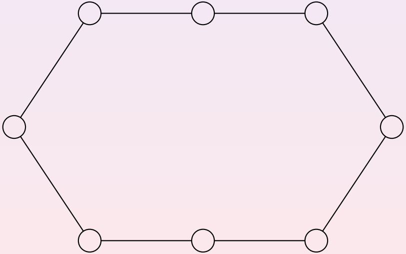


# Simple Examples : Path and Ring

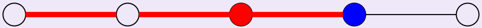


# Simple Examples : Path and Ring

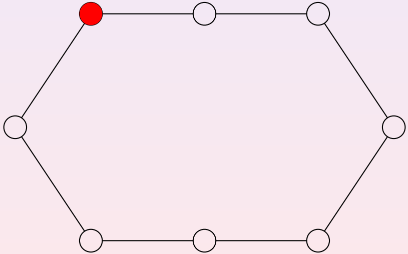
#searchers(Path)=2



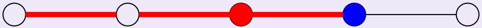
# Simple Examples : Path and Ring



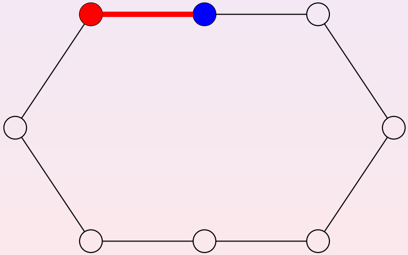
**#searchers(Path)=2**



# Simple Examples : Path and Ring

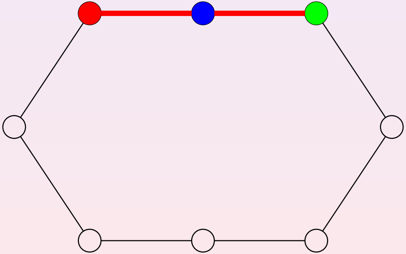


**#searchers(Path)=2**

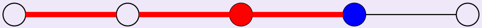


# Simple Examples : Path and Ring

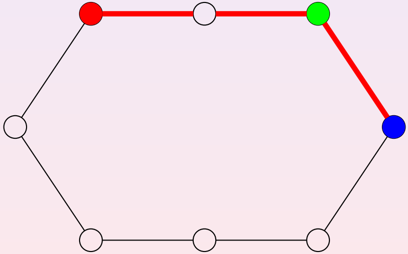
**#searchers(Path)=2**



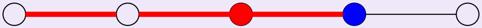
# Simple Examples : Path and Ring



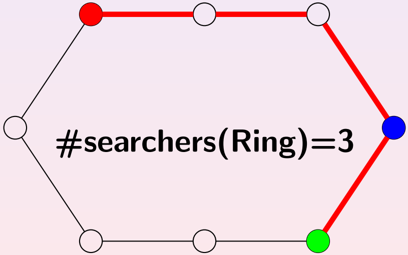
**#searchers(Path)=2**



# Simple Examples : Path and Ring



#searchers(Path)=2



# Monotone Search Number / Treewidth

## Monotonicity

A search strategy is monotone if no vertices are occupied twice by a searcher.
ms(G): the smallest number of searchers such that it exists a monotone search strategy that catch any fugitive in $G$.

## Link with treewidth

For any graph $G$, $ms(G) = tw(G) + 1$

# Monotone Search Number / Treewidth

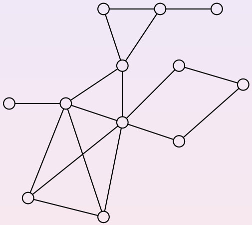
## Monotonicity

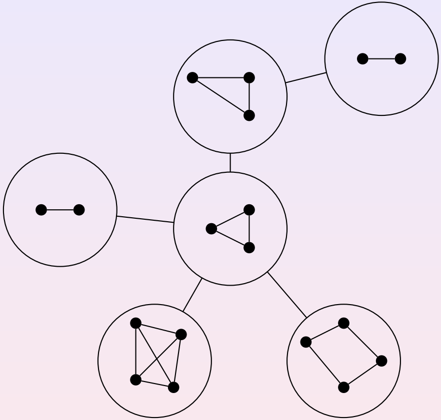
A search strategy is monotone if no vertices are occupied twice by a searcher.
ms(G): the smallest number of searchers such that it exists a monotone search strategy that catch any fugitive in $G$.

## Link with treewidth

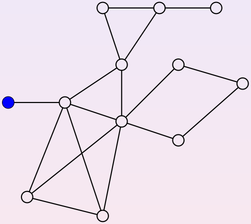
For any graph $G$, $ms(G) = tw(G) + 1$

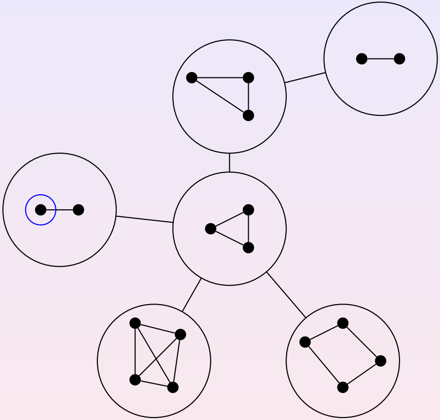
# Monotone Search Number / Treewidth



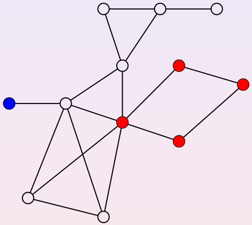


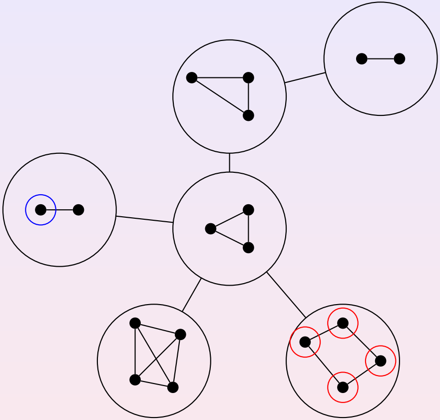
# Monotone Search Number / Treewidth



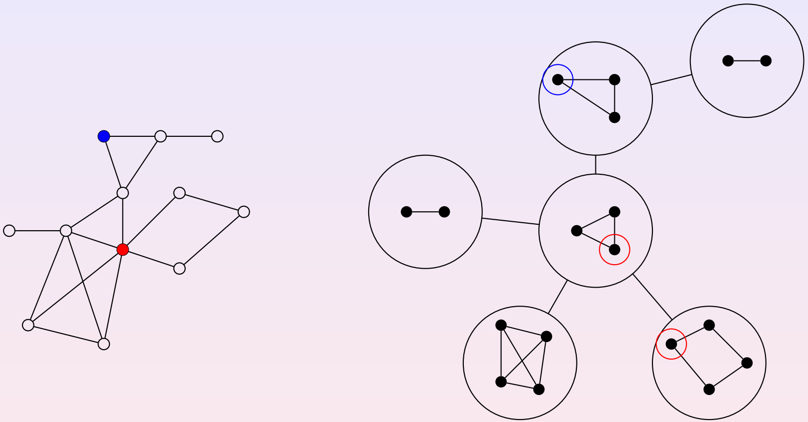


# Monotone Search Number / Treewidth

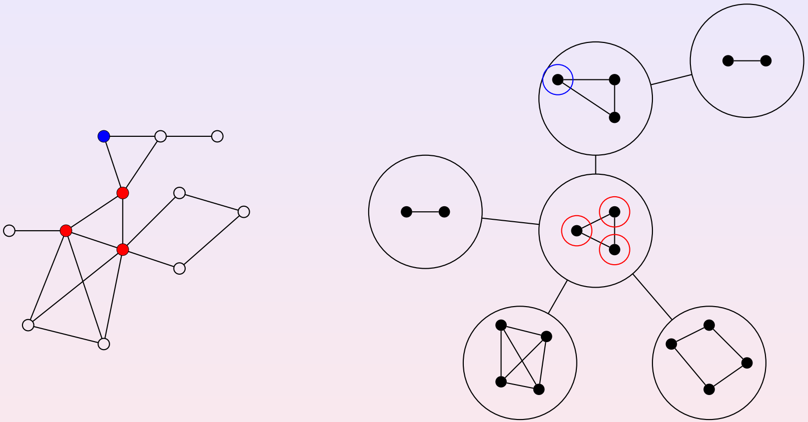




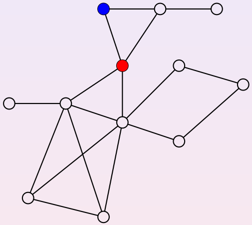
# Monotone Search Number / Treewidth

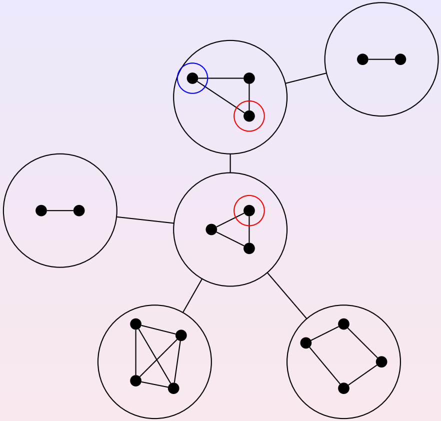


# Monotone Search Number / Treewidth

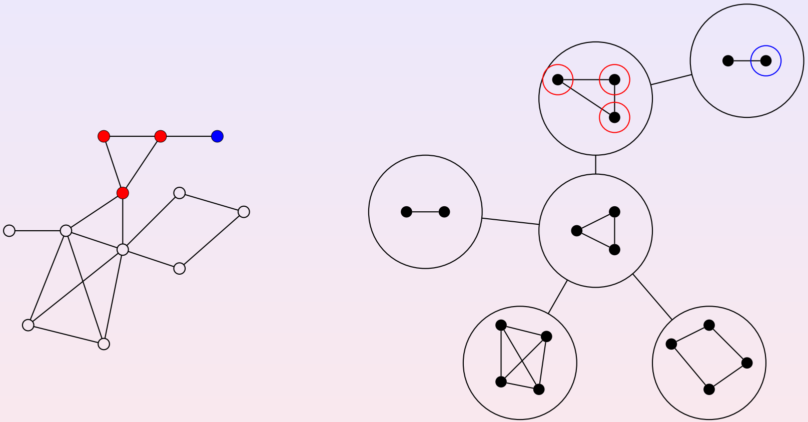


# Monotone Search Number / Treewidth





# Monotone Search Number / Treewidth



Thus, $\mathbf{ms(G)} \leq \mathbf{tw(G)} + \mathbf{1}$

# A winning strategy for the fugitive

How to prove that a search strategy uses the smallest number of searcher?

## A fugitive winning against $k$ searchers

A set $\mathcal{B}$ of subsets of $V(G)$

- for any position of $k$ searchers $X \subset V(G)$, a connected component $\beta(X)$ of $G \setminus X$ belongs to $\mathcal{B}$;
- for any $X, Y \subset V(G)$, $|X|, |Y| \leq k$, $\beta(X)$ touches $\beta(Y)$.

## Alternative definition

A set $\mathcal{B}$ of connected subsets of $V(G)$ pairwise touching such that, for any $X \subset V(G)$, $|X| \leq k$, it exists $A \in \mathcal{B}$, $A \cap X = \emptyset$.

If such a set $\mathcal{B}$ exists, $tw(G) + 1 = ms(G) \geq k + 1$.

# A winning strategy for the fugitive

How to prove that a search strategy uses the smallest number of searcher?

## A fugitive winning against $k$ searchers

A set $\mathcal{B}$ of subsets of $V(G)$

- for any position of $k$ searchers $X \subset V(G)$, a connected component $\beta(X)$ of $G \setminus X$ belongs to $\mathcal{B}$;
- for any $X, Y \subset V(G)$, $|X|, |Y| \leq k$, $\beta(X)$ touches $\beta(Y)$.

## Alternative definition

A set $\mathcal{B}$ of connected subsets of $V(G)$ pairwise touching such that, for any $X \subset V(G)$, $|X| \leq k$, it exists $A \in \mathcal{B}$, $A \cap X = \emptyset$.

If such a set $\mathcal{B}$ exists, $tw(G) + 1 = ms(G) \geq k + 1$.

# Bramble and bramble-number

## Definition

Bramble $\mathcal{B}$: set of connected subsets of $V(G)$, pairwise touching.

- for any $B \in \mathcal{B}$, $B \subseteq V(G)$;
- for any $B_i, B_j \in \mathcal{B}$, $B_i \cup B_j$ connected.

A transversal is a subset $\mathcal{T} \subseteq V(G)$ such that:

For all $B_i \in \mathcal{B}$, $B_i \cap \mathcal{T} \neq \emptyset$

## Order of a bramble

$Order(\mathcal{B})$: Minimum size of a transversal of $\mathcal{B}$.

# Bramble and bramble-number

## Definition

Bramble $\mathcal{B}$: set of connected subsets of $V(G)$, pairwise touching.

- for any $B \in \mathcal{B}$, $B \subseteq V(G)$;
- for any $B_i, B_j \in \mathcal{B}$, $B_i \cup B_j$ connected.

A transversal is a subset $\mathcal{T} \subseteq V(G)$ such that:

$$\text{For all } B_i \in \mathcal{B},\ B_i \cap \mathcal{T} \neq \emptyset$$

## Order of a bramble

$Order(\mathcal{B})$: Minimum size of a transversal of $\mathcal{B}$.

# Bramble and bramble-number

## Definition

Bramble $\mathcal{B}$: set of connected subsets of $V(G)$, pairwise touching.

- for any $B \in \mathcal{B}$, $B \subseteq V(G)$;
- for any $B_i, B_j \in \mathcal{B}$, $B_i \cup B_j$ connected.

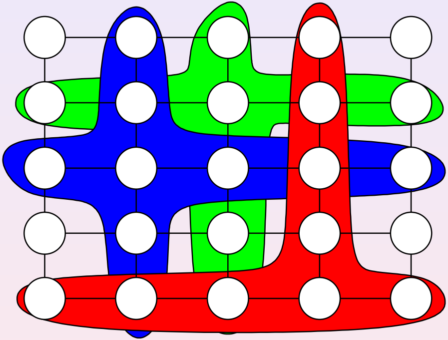
## Order of a bramble

$Order(\mathcal{B})$: Minimum size of a transversal of $\mathcal{B}$.

## Bramble-number $\mathbf{bn}(G)$

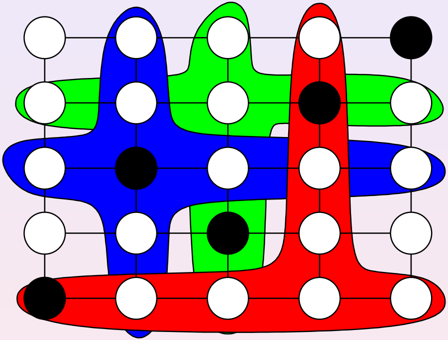
$\mathbf{bn}(G)$: maximum order among all brambles of $G$.

# Bramble of the Grid $G_{k*k}$



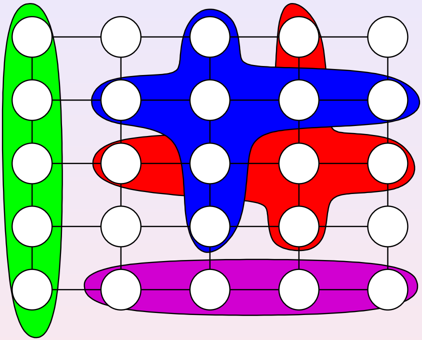
$\mathcal{B}_1$ set of all crosses (one row + one column)

# Bramble of the Grid $G_{k*k}$



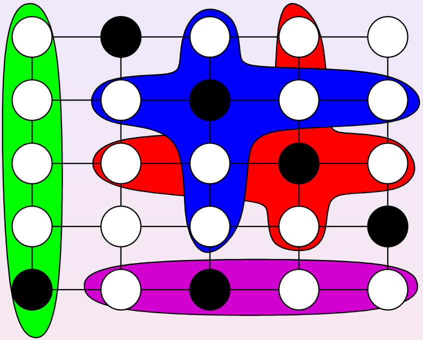
$\mathcal{B}_1$ set of all crosses (one row + one column)
$Order(\mathcal{B}_1) = k$, therefore $\mathbf{bn}(G_{k*k}) \geq k$

# Bramble of the Grid $G_{k*k}$



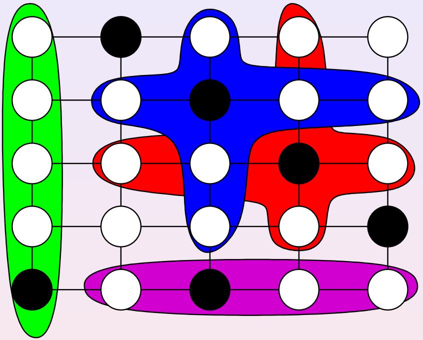
$\mathcal{B}_2$ first column + last row minus its first vertex + set of all crosses of $G_{(k-1)*(k-1)}$

# Bramble of the Grid $G_{k*k}$



$\mathcal{B}_2$ first column + last row minus its first vertex + set of all crosses of $G_{(k-1)*(k-1)}$
$Order(\mathcal{B}_2) = k + 1$, therefore $\mathbf{bn}(G_{k*k}) \geq k + 1$

# Bramble of the Grid $G_{k*k}$



$\mathcal{B}_2$ first column + last row minus its first vertex + set of all crosses of $G_{(k-1)*(k-1)}$

$Order(\mathcal{B}_2) = k+1$, therefore $\mathbf{bn}(G_{k*k}) \geq k+1$

How to prove that it is a maximal bramble?

# Min-Max Theorem

**For any graph $G$, $\mathbf{tw}(G) + 1 = \mathbf{bn}(G)$**

**Seymour and Thomas**, J. of Comb. Th., 1993.
Graph searching and a min-max theorem for tree-width

$$\min_{\text{(T,X) tree-dec. of G}} \; \max_{t \in V(T)} |X_t| = \max_{\mathcal{B} \text{ bramble of G}} \; \min_{\text{Y transv. of } \mathcal{B}} |Y|$$

**Example of the grid**

$\mathbf{tw}(G_{k*k}) + 1 = \mathbf{bn}(G_{k*k}) = k + 1$

# Min-Max Theorem

**For any graph $G$, $\mathbf{tw}(G) + 1 = \mathbf{bn}(G)$**

**Seymour and Thomas**, J. of Comb. Th., 1993.
Graph searching and a min-max theorem for tree-width

$$\min_{\text{(T,X) tree-dec. of G}} \max_{t \in V(T)} |X_t| = \max_{\mathcal{B} \text{ bramble of G}} \min_{\text{Y transv. of } \mathcal{B}} |Y|$$

**Example of the grid**

$\mathbf{tw}(G_{k*k}) + 1 = \mathbf{bn}(G_{k*k}) = k + 1$

# Min-Max Theorem

For any graph $G$, $\mathbf{tw}(G) + 1 = \mathbf{bn}(G)$

**Seymour and Thomas**, J. of Comb. Th., 1993.
Graph searching and a min-max theorem for tree-width

$$\min_{(T,X)\text{ tree-dec. of G}} \max_{t \in V(T)} |X_t| = \max_{\mathcal{B}\text{ bramble of G}} \min_{Y\text{ transv. of }\mathcal{B}} |Y|$$

## In terms of graph searching

- *Bramble of order* $k+1$ = winning strategy for a visible fugitive against $k$ searchers.
- *Tree-decomposition of width* $k$ = winning strategy for $k+1$ searchers against any visible fugitive.

# Other Min-Max Theorems

| Decomposition | Obstruction |
|---|---|
| *Tree-decomposition* | *Bramble [Seymour and Thomas]* |
| Path-decomposition | Blockage [Robertson and Seymour] |
| Branch-decomposition | Tangle [Robertson and Seymour] |
| Rank-decomposition | Tangle [Oum and Seymour] |
| Tree-decomposition of matroids | ? |

# Plan

# Partitioning-tree

## Definition

A set $E$. A partitioning-tree $T$ on $E$

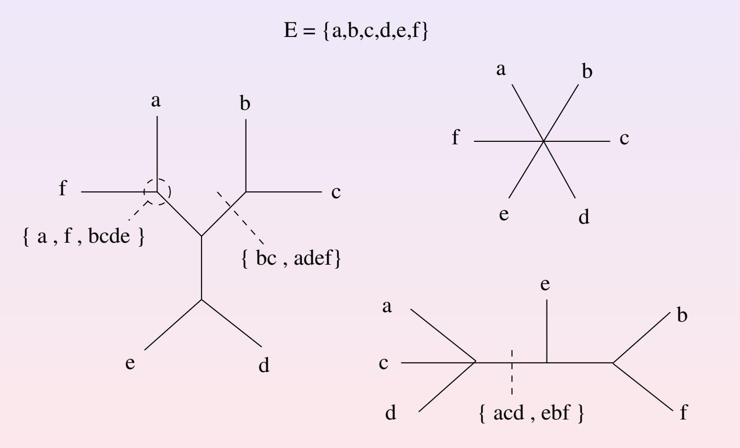
- $T$ a tree;
- a bijection between $E$ and the set of leaves of $T$.

## $T$-partitions

$T$ defines a set of partitions of $E$.

- any edge $e \in E(T) \Rightarrow$ a bipartition $T_e$ of $E$;
- any vertex $v \in V(T) \Rightarrow$ a partition $T_v$ of $E$.

# Partitioning-tree



# General Problem

Let $\mathcal{F}$ be a set of partitions of a set $E$
(but the trivial one $\{E\}$)
$\mathcal{F}$ is a set of admissible partitions of $E$.

**Question**

Is there an admissible partitioning-tree for $\mathcal{F}$,
i.e. a partitioning-tree $T$ such that $\{T\text{-}partitions\} \subseteq \mathcal{F}$ ?

# Bramble and principal bramble

Let $E$ be a set, and $\mathcal{F}$ be a set of admissible partitions of $E$.

## $\mathcal{F}$-bramble

A $\mathcal{F}$-bramble is a set $\mathcal{B}$ of subsets of $E$
$\mathcal{B} = \{X_i \mid X_i \subseteq E\}$

- for any $X_i, X_j \in \mathcal{B}$, $X_i \cap X_j \neq \emptyset$;
- for any $\{E_1, \cdots, E_k\} \in \mathcal{F}$, there is $E_i \in \mathcal{B}$.

## principal $\mathcal{F}$-bramble

$\mathcal{B}$ is principal if $\bigcap_{X_i \in \mathcal{B}} X_i \neq \emptyset$.

It is easy to compute a principal $\mathcal{F}$-bramble $\mathcal{B}$:

- pick an element $e \in E$;
- for any partition $Y \in \mathcal{F}$, put in $\mathcal{B}$ the element of $Y$ that contains $e$.

# Bramble and principal bramble

Let $E$ be a set, and $\mathcal{F}$ be a set of admissible partitions of $E$.

## $\mathcal{F}$-bramble

A $\mathcal{F}$-bramble is a set $\mathcal{B}$ of subsets of $E$
$\mathcal{B} = \{X_i \mid X_i \subseteq E\}$

- for any $X_i, X_j \in \mathcal{B}$, $X_i \cap X_j \neq \emptyset$;
- for any $\{E_1, \cdots, E_k\} \in \mathcal{F}$, there is $E_i \in \mathcal{B}$.

## principal $\mathcal{F}$-bramble

$\mathcal{B}$ is principal if $\bigcap_{X_i \in \mathcal{B}} X_i \neq \emptyset$.

It is easy to compute a principal $\mathcal{F}$-bramble $\mathcal{B}$:

- pick an element $e \in E$;
- for any partition $Y \in \mathcal{F}$, put in $\mathcal{B}$ the element of $Y$ that contains $e$.

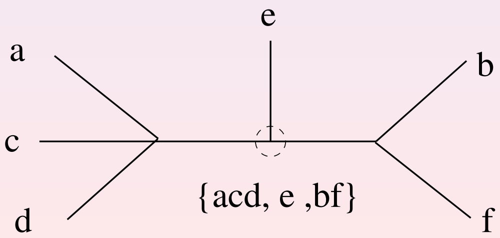
# An obstruction to the existence of a partitioning-tree

**Lemma**

If there is a non-principal $\mathcal{F}$-bramble, then there is no admissible partitioning-tree for $\mathcal{F}$.

$\mathcal{B}$ non-principal $\mathcal{F}$-bramble, $T$ admissible partitioning-tree.

- for any internal vertex $u$ of $T$, $T_u \in \mathcal{F}$;



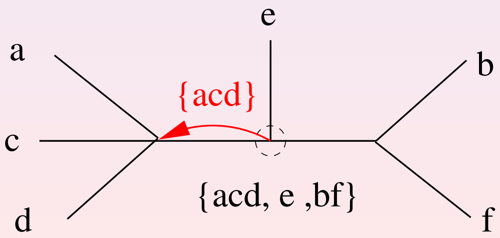
# An obstruction to the existence of a partitioning-tree

**Lemma**

If there is a non-principal $\mathcal{F}$-bramble, then there is no admissible partitioning-tree for $\mathcal{F}$.

$\mathcal{B}$ non-principal $\mathcal{F}$-bramble, $T$ admissible partitioning-tree.

- there is an element $X$ of $T_u$, $X \in \mathcal{B}$;



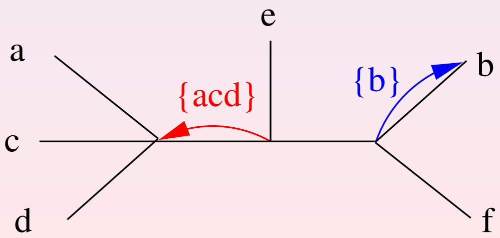
# An obstruction to the existence of a partitioning-tree

**Lemma**

If there is a non-principal $\mathcal{F}$-bramble, then there is no admissible partitioning-tree for $\mathcal{F}$.

$\mathcal{B}$ non-principal $\mathcal{F}$-bramble, $T$ admissible partitioning-tree.

- since $\mathcal{B}$ is not principal, no edge is oriented toward a leaf;



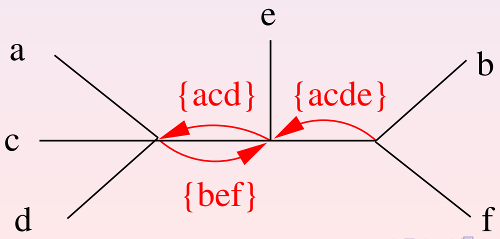
# An obstruction to the existence of a partitioning-tree

**Lemma**

If there is a non-principal $\mathcal{F}$-bramble, then there is no admissible partitioning-tree for $\mathcal{F}$.

$\mathcal{B}$ non-principal $\mathcal{F}$-bramble, $T$ admissible partitioning-tree.

- thus, an edge gets two orientations, $\{acd\} \cap \{bcf\} = \emptyset$, a contradiction.



# A new question

We know:

non-principal $\mathcal{F}$-bramble $\Rightarrow$ no admissible partitioning-tree

How to characterize the families $\mathcal{F}$ of partitions of $E$, s.t. it is an equivalence?

Good family $\mathcal{F}$ of admissible partitions

- either there is a non-principal $\mathcal{F}$-bramble,
- or there is an admissible partitioning-tree for $\mathcal{F}$?

# A new question

We know:
non-principal $\mathcal{F}$-bramble $\Rightarrow$ no admissible partitioning-tree

How to characterize the families $\mathcal{F}$ of partitions of $E$, s.t. it is an equivalence?

**Good family $\mathcal{F}$ of admissible partitions**

- either there is a non-principal $\mathcal{F}$-bramble,
- or there is an admissible partitioning-tree for $\mathcal{F}$?

# Partition Functions

## Partition functions

$\Phi : \{partitions\ of\ E\} \to N.$

Let $\Phi$ be a partition function and let $k \geq 1$.
Let $\mathcal{F}_{\Phi,k}$ be the family of the partitions $P$, with $\Phi(P) \leq k$.

A $k$-partitioning-tree $T$ for $\Phi$ is an admissible partitioning-tree for $\mathcal{F}_{\Phi,k}$.

A $k$-bramble for $\Phi$ is a $\mathcal{F}_{\Phi,k}$-bramble.

# Partition Functions

**Partition functions**

$\Phi : \{partitions\ of\ E\} \rightarrow N.$

Let $\Phi$ be a partition function and let $k \geq 1$.
Let $\mathcal{F}_{\Phi,k}$ be the family of the partitions $P$, with $\Phi(P) \leq k$.

A $k$-partitioning-tree $T$ for $\Phi$ is an admissible partitioning-tree for $\mathcal{F}_{\Phi,k}$.

A $k$-bramble for $\Phi$ is a $\mathcal{F}_{\Phi,k}$-bramble.

# Our Problem

How to characterize the partition functions $\Phi$ such that, for any $k$, $\mathcal{F}_{\Phi,k} = \{partition\ P \mid \Phi(P) \leq k\}$ is a family of good admissible partitions?

**In other words**

How to characterize the partition functions $\Phi$ such that, for any $k$,

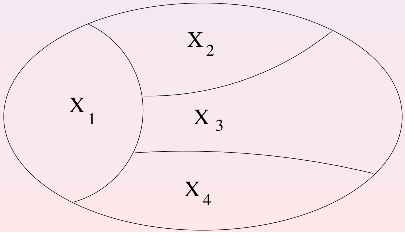
- either there is a non-principal $k$-bramble for $\Phi$,
- or there is a $k$-partitioning-tree for $\Phi$?

# An operation on partitions

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ be a partition of $E$, and $Y \subset E$ such that $X_1 \cap Y = \emptyset$.

## To *push* a partition

By pushing $X_1$ to $Y$ in $\mathcal{X}$, we get the new partition: $\mathcal{X}_{X_1 \to Y} = \{Y^c, X_2 \cap Y, \cdots, X_n \cap Y\}$

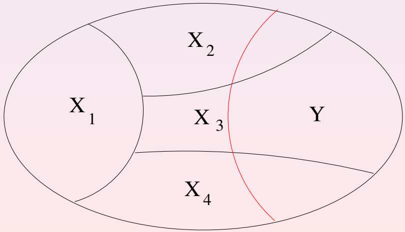


# An operation on partitions

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ be a partition of $E$, and $Y \subset E$ such that $X_1 \cap Y = \emptyset$.

## To *push* a partition

By pushing $X_1$ to $Y$ in $\mathcal{X}$, we get the new partition:
$\mathcal{X}_{X_1 \to Y} = \{Y^c, X_2 \cap Y, \cdots, X_n \cap Y\}$

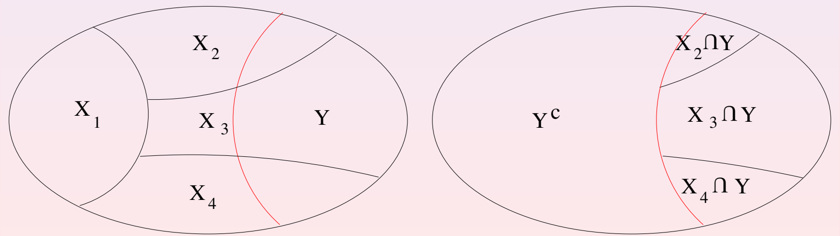


# An operation on partitions

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ be a partition of $E$, and $Y \subset E$ such that $X_1 \cap Y = \emptyset$.

## To *push* a partition

By pushing $X_1$ to $Y$ in $\mathcal{X}$, we get the new partition: $\mathcal{X}_{X_1 \to Y} = \{Y^c, X_2 \cap Y, \cdots, X_n \cap Y\}$



# Submodular partition functions

## Definition

A partition function $\Phi$ is submodular if, for any partition $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$, $\mathcal{Y} = \{Y_1, Y_2 \cdots, Y_m\}$ s.t. $X_i \cap Y_j = \emptyset$.

$$\Phi(\mathcal{X}) + \Phi(\mathcal{Y}) \geq \Phi(\mathcal{X}_{X_i \to Y_j}) + \Phi(\mathcal{Y}_{Y_j \to X_i})$$

## Weakly submodular partition functions

A partition function $\Phi$ is weakly submodular if, for any partition $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$, $\mathcal{Y} = \{Y_1, Y_2 \cdots, Y_m\}$ s.t. $X_i \cap Y_j = \emptyset$.

- either there exists $F$ with $X_i \subseteq F \subseteq Y_j^c$ such that $\Phi(\mathcal{X}) > \Phi(\mathcal{X}_{X_i \to F^c})$
- or $\Phi(\mathcal{Y}) \geq \Phi(\mathcal{Y}_{Y_j \to X_i})$

# Submodular partition functions

## Definition

A partition function $\Phi$ is submodular if, for any partition $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$, $\mathcal{Y} = \{Y_1, Y_2 \cdots, Y_m\}$ s.t. $X_i \cap Y_j = \emptyset$.

$$\Phi(\mathcal{X}) + \Phi(\mathcal{Y}) \geq \Phi(\mathcal{X}_{X_i \to Y_j}) + \Phi(\mathcal{Y}_{Y_j \to X_i})$$

## Weakly submodular partition functions

A partition function $\Phi$ is weakly submodular if, for any partition $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$, $\mathcal{Y} = \{Y_1, Y_2 \cdots, Y_m\}$ s.t. $X_i \cap Y_j = \emptyset$.

- either there exists $F$ with $X_i \subseteq F \subseteq Y_j^c$ such that $\Phi(\mathcal{X}) > \Phi(\mathcal{X}_{X_i \to F^c})$
- or $\Phi(\mathcal{Y}) \geq \Phi(\mathcal{Y}_{Y_j \to X_i})$

# Main Theorem

**Theorem: Duality partitioning-tree/bramble**

Let $\Phi$ be a weakly submodular partition function on a set $E$, and let $k \geq 1$.

- either there is a non-principal $k$-bramble for $\Phi$,
- or there is a $k$-partitioning-tree for $\Phi$.

$\mathcal{F}_{\Phi,k}$ is a good family of admissible partitions

# Plan

# Duality Treewidth / Bramble

Let $G = (V, E)$ be a graph, and $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ a partition of $E$. The border function $\delta$ is defined by:
$\delta(\mathcal{X})$ is the set of vertices incident to an edge in $X_i$ and in $X_j$.

**Lemma**

$|\delta|$ is a submodular partition function.

**Duality treewidth/bramble**

- If $T$ is a $k$-partitioning-tree for $|\delta|$, then $(T, (\delta(T_t))_{t \in V(T)})$ is a tree-decomposition of width at most $k-1$.
- We can compute a bramble (in usual sense) of order at least $k$ from any non-principal $k$-bramble for $|\delta|$.

# Duality Treewidth / Bramble

Let $G = (V, E)$ be a graph, and $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ a partition of $E$. The border function $\delta$ is defined by:
$\delta(\mathcal{X})$ is the set of vertices incident to an edge in $X_i$ and in $X_j$.

**Lemma**

$|\delta|$ is a submodular partition function.

**Duality treewidth/bramble**

- If $T$ is a $k$-partitioning-tree for $|\delta|$, then $(T, (\delta(T_t))_{t \in V(T)})$ is a tree-decomposition of width at most $k-1$.
- We can compute a bramble (in usual sense) of order at least $k$ from any non-principal $k$-bramble for $|\delta|$.

# Other Min-Max Theorems

| Decomposition | Partition Function |
| --- | --- |
| Tree-decomposition | $\|\delta\|$ (submodular) |
| Path-decomposition | $(\|\delta\|)'_2$ (weakly submodular) |
| Branch-decomposition | $(\max_{\|\delta\|})_3$ (weakly submodular) |
| Rank-decomposition | $(\max_{rk})_3$ (weakly submodular) |
| Tree-decomposition of matroids | $\Phi$ (submodular) |

# To build new weakly submodular functions (1)

Let $\Phi$ be a weakly submodular partition function

**weakly submodular partition function $\Phi_p$**

Let $p \geq 2$. Let $\Phi_p$ be defined such that $\Phi_p(\mathcal{X}) = \Phi(\mathcal{X})$ if $\mathcal{X}$ consists of at most $p$ non empty parts, and $\infty$ otherwise.
Then, $\Phi_p$ is weakly submodular.

**weakly submodular partition function $\Phi'_p$**

Let $p \geq 2$. Let $\Phi'_p$ be defined such that $\Phi'_p(\mathcal{X}) = \Phi(\mathcal{X})$ if $\mathcal{X}$ consists of at most $p$ parts with at least 2 elements, and $\infty$ otherwise.
Then, $\Phi'_p$ is weakly submodular if we cannot push a singleton.

The main theorem holds for such a function

# To build new weakly submodular functions (1)

Let $\Phi$ be a weakly submodular partition function

**weakly submodular partition function $\Phi_p$**

Let $p \geq 2$. Let $\Phi_p$ be defined such that $\Phi_p(\mathcal{X}) = \Phi(\mathcal{X})$ if $\mathcal{X}$ consists of at most $p$ non empty parts, and $\infty$ otherwise.
Then, $\Phi_p$ is weakly submodular.

**weakly submodular partition function $\Phi'_p$**

Let $p \geq 2$. Let $\Phi'_p$ be defined such that $\Phi'_p(\mathcal{X}) = \Phi(\mathcal{X})$ if $\mathcal{X}$ consists of at most $p$ parts with at least 2 elements, and $\infty$ otherwise.
Then, $\Phi'_p$ is weakly submodular if we cannot push a singleton.

The main theorem holds for such a function

# To build new weakly submodular functions (2)

Let $f$ be a submodular function on $E$, i.e., that satisfies $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$

**submodular partition function $\sum_f$**

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ be a partition. Let us define:
$\sum_f(\mathcal{X}) = \sum_{i \leq n} f(X_i)$.
Then, $\sum_f$ is submodular.

Let $f$ be a symmetric submodular function on $E$, i.e., satisfying moreover $f(X) = f(X^c)$.

**submodular partition function $\max_f$**

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ be a partition. Let us define:
$\max_f(\mathcal{X}) = \max_{i \leq n} f(X_i)$.
$\max_f + \epsilon \sum_f$ is submodular for some arbitrary small $\epsilon > 0$.

# To build new weakly submodular functions (2)

Let $f$ be a submodular function on $E$, i.e., that satisfies $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$

**submodular partition function $\sum_f$**

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ be a partition. Let us define:
$\sum_f(\mathcal{X}) = \sum_{i \leq n} f(X_i)$.
Then, $\sum_f$ is submodular.

Let $f$ be a symmetric submodular function on $E$, i.e., satisfying moreover $f(X) = f(X^c)$.

**submodular partition function $\max_f$**

Let $\mathcal{X} = \{X_1, X_2, \cdots, X_n\}$ be a partition. Let us define:
$\max_f(\mathcal{X}) = \max_{i \leq n} f(X_i)$.
$\max_f + \epsilon \sum_f$ is submodular for some arbitrary small $\epsilon > 0$.

# Other width parameters (1)

Let $G = (V, E)$ be a graph,

**Duality branchwidth / tangle [Graph Minors X]**

A $k$-partitioning-tree for the partition function $(\max_{|\delta|})_3$ is a branch decomposition of width at most $k$.
We can compute a tangle of order at least $k$ from any non-principal $k$-bramble for $(\max_{|\delta|})_3$.

**Duality pathwidth / blockage [Graph Minors I]**

A $k$-partitioning-tree for the partition function $(|\delta|)'_2$ is a path decomposition of width at most $k-1$.
We can compute a blockage of order at least $k$ from any non-principal $k$-bramble for $(|\delta|)'_2$.

# Other width parameters (1)

Let $G = (V, E)$ be a graph,

## Duality branchwidth / tangle [Graph Minors X]

A $k$-partitioning-tree for the partition function $(\max_{|\delta|})_3$ is a branch decomposition of width at most $k$.
We can compute a tangle of order at least $k$ from any non-principal $k$-bramble for $(\max_{|\delta|})_3$.

## Duality pathwidth / blockage [Graph Minors I]

A $k$-partitioning-tree for the partition function $(|\delta|)'_2$ is a path decomposition of width at most $k - 1$.
We can compute a blockage of order at least $k$ from any non-principal $k$-bramble for $(|\delta|)'_2$.

# Other width parameters (2)

Let $G = (V, E)$ be a graph,

**Rankwidth [Oum et Seymour 06]**

- consider the set $V$;
- based on the symmetric submodular function $rk$;
- submodular partition function $(\max_{rk})_3$.

**Treewidth of matroid [Hlileny et Whittle 06]**

- $M$ a matroid on groud set $E$ with rank function $r$;
- based on the submodular function $r^c$ such that $r^c(F) = r(F^c)$;
- $\mathcal{X} = \{X_1, \cdots, X_\ell\}$, $\Phi(\mathcal{X}) = \sum_{r^c}(\mathcal{X}) - (\ell - 1)r(E)$.

# Other width parameters (2)

Let $G = (V, E)$ be a graph,

## Rankwidth [Oum et Seymour 06]

- consider the set $V$;
- based on the symmetric submodular function $rk$;
- submodular partition function $(\max_{rk})_3$.

## Treewidth of matroid [Hlileny et Whittle 06]

- $M$ a matroid on groud set $E$ with rank function $r$;
- based on the submodular function $r^c$ such that $r^c(F) = r(F^c)$;
- $\mathcal{X} = \{X_1, \cdots, X_\ell\}$, $\Phi(\mathcal{X}) = \sum_{r^c}(\mathcal{X}) - (\ell - 1)r(E)$.