Structural Decompositions and Algorithms for the Maximum Weight Independent Set Problem

Workshop on Graph Theory and Its Applications

Boğaziçi University, Istanbul, Turkey

October 24–25, 2025

Clément Dallard

University of Fribourg

UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

Maximum Weight Independent Set problem (MWIS): given a graph $G = (V, E)$ and a weight function $w : V \to \mathbb{Q}_+$, find an independent set I in G of maximum possible weight $w(I)$, where $w(I) = \sum_{x \in I} w(x)$.

MWIS is NP-hard on general graphs, and it remains NP-hard on many restricted graph classes, including cubic, triangle-free planar graphs.

Maximum Weight Independent Set: Computational hardness

Maximum Weight Independent Set problem (MWIS): given a graph $G = (V, E)$ and a weight function $w : V \to \mathbb{Q}_+$, find an independent set I in G of maximum possible weight $w(I)$, where $w(I) = \sum_{x \in I} w(x)$.

MWIS is NP-hard on general graphs, and it remains NP-hard on many restricted graph classes, including cubic, triangle-free planar graphs.

It is also NP-hard to approximate:

- within a factor of Δ^{ε}, for some $\varepsilon > 0$, on graphs with maximum degree Δ (Alon et al., 1995), and
- on cubic graphs within a factor of $94/95$ (Chlebík and Chlebíková, 2006).

Maximum Weight Independent Set: Computational hardness

Maximum Weight Independent Set problem (MWIS): given a graph $G = (V, E)$ and a weight function $w : V \to \mathbb{Q}_+$, find an independent set I in G of maximum possible weight $w(I)$, where $w(I) = \sum_{x \in I} w(x)$.

MWIS is NP-hard on general graphs, and it remains NP-hard on many restricted graph classes, including cubic, triangle-free planar graphs.

It is also NP-hard to approximate:

- within a factor of Δ^{ε}, for some $\varepsilon > 0$, on graphs with maximum degree Δ (Alon et al., 1995), and
- on cubic graphs within a factor of $94/95$ (Chlebík and Chlebíková, 2006).

One challenge is to identify graph classes on which MWIS can be solved efficiently.

Chordal graphs: Definition and warm-up proofs

Definition

A graph is **chordal** if every cycle of length at least four has a chord, i.e., an edge between two non-consecutive vertices of the cycle.

Definition

A graph is **chordal** if every cycle of length at least four has a chord, i.e., an edge between two non-consecutive vertices of the cycle.

From the definition above, one can derive several equivalent characterizations of chordal graphs:

1. A graph is chordal if and only if all its minimal separators are cliques.

Chordal graphs: Definition and warm-up proofs

Definition

A graph is **chordal** if every cycle of length at least four has a chord, i.e., an edge between two non-consecutive vertices of the cycle.

From the definition above, one can derive several equivalent characterizations of chordal graphs:

1. A graph is chordal if and only if all its minimal separators are cliques.
2. Every chordal graph (with at least one vertex) admits a simplicial vertex, i.e., a vertex whose neighborhood is a clique.

Definition

A graph is **chordal** if every cycle of length at least four has a chord, i.e., an edge between two non-consecutive vertices of the cycle.

From the definition above, one can derive several equivalent characterizations of chordal graphs:

1. A graph is chordal if and only if all its minimal separators are cliques.
2. Every chordal graph (with at least one vertex) admits a simplicial vertex, i.e., a vertex whose neighborhood is a clique.
3. A graph is chordal if and only if it is the intersection graph of subtrees of a tree.

Definition

A graph is **chordal** if every cycle of length at least four has a chord, i.e., an edge between two non-consecutive vertices of the cycle.

From the definition above, one can derive several equivalent characterizations of chordal graphs:

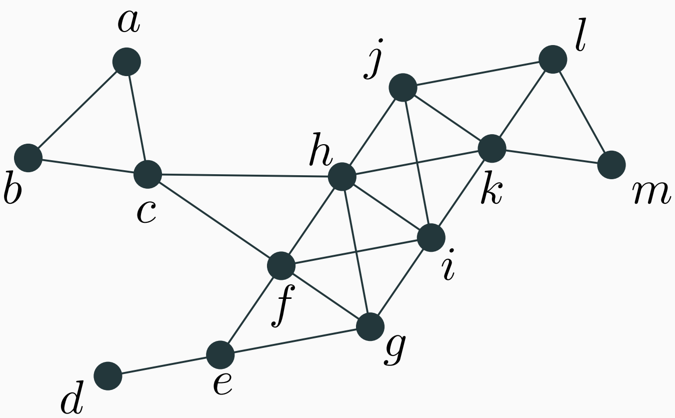
1. A graph is chordal if and only if all its minimal separators are cliques.
2. Every chordal graph (with at least one vertex) admits a simplicial vertex, i.e., a vertex whose neighborhood is a clique.
3. A graph is chordal if and only if it is the intersection graph of subtrees of a tree.
4. Every chordal graph admits a clique tree, i.e., a tree whose nodes correspond to the maximal cliques of the graph and satisfy the Helly property.

Perfect elimination orderings: Definition

A **perfect elimination ordering** (p.e.o. for short) of a graph G is an ordering $(v_1, \ldots, v_n)$ of its vertices such that, for every $i \in \{1, \ldots, n\}$, the set of neighbors of v_i among $\{v_1, \ldots, v_{i-1}\}$ is a clique.

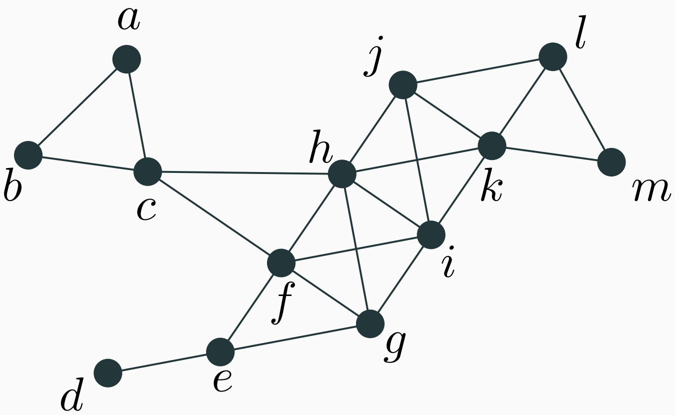
Perfect elimination orderings: Definition

A **perfect elimination ordering** (p.e.o. for short) of a graph G is an ordering $(v_1, \ldots, v_n)$ of its vertices such that, for every $i \in \{1, \ldots, n\}$, the set of neighbors of v_i among $\{v_1, \ldots, v_{i-1}\}$ is a clique.



Perfect elimination orderings: Definition

A **perfect elimination ordering** (p.e.o. for short) of a graph G is an ordering $(v_1, \ldots, v_n)$ of its vertices such that, for every $i \in \{1, \ldots, n\}$, the set of neighbors of v_i among $\{v_1, \ldots, v_{i-1}\}$ is a clique.



A graph is chordal if and only if it admits a p.e.o. (Rose, 1970).

Perfect elimination orderings: Solving MIS

Theorem

MIS can be solved in $\mathcal{O}(n+m)$ time on n-vertex m-edge chordal graphs using a perfect elimination ordering.

Perfect elimination orderings: Solving MIS

Theorem

MIS can be solved in $\mathcal{O}(n+m)$ time on n-vertex m-edge chordal graphs using a perfect elimination ordering.

Algorithm:

- find a p.e.o. $(v_1, \ldots, v_n)$ of G;
- $I := \emptyset$;
- for i from n to 1, do:
  - if v_i is not adjacent to any vertex in I, then add v_i to I;
- return I.

Theorem

MIS can be solved in $\mathcal{O}(n+m)$ time on n-vertex m-edge chordal graphs using a perfect elimination ordering.

Algorithm:

- find a p.e.o. $(v_1, \ldots, v_n)$ of G;
- $I := \emptyset$;
- for i from n to 1, do:
  - if v_i is not adjacent to any vertex in I, then add v_i to I;
- return I.

The correctness relies on the fact that at most one vertex in $N[v_i] \cap \{v_1, \ldots, v_i\}$ can be included in an independent set.

Perfect elimination orderings: Generalizing to independence degeneracy

The **independence degeneracy** (**α-degeneracy** for short) of a graph G is the minimum integer k for which there exists an ordering $(v_1, \ldots, v_n)$ of its vertices such that, for every $i \in \{1, \ldots, n\}$, the set of neighbors of v_i among $\{v_1, \ldots, v_{i-1}\}$ induces a subgraph with independence number at most k.

The **independence degeneracy** (α**-degeneracy** for short) of a graph G is the minimum integer k for which there exists an ordering $(v_1, \ldots, v_n)$ of its vertices such that, for every $i \in \{1, \ldots, n\}$, the set of neighbors of v_i among $\{v_1, \ldots, v_{i-1}\}$ induces a subgraph with independence number at most k.

This generalizes the notion of degeneracy, which considers the number of neighbors instead of the independence number of the neighborhood.

Perfect elimination orderings: Generalizing to independence degeneracy

The **independence degeneracy** (**α-degeneracy** for short) of a graph G is the minimum integer k for which there exists an ordering $(v_1, \ldots, v_n)$ of its vertices such that, for every $i \in \{1, \ldots, n\}$, the set of neighbors of v_i among $\{v_1, \ldots, v_{i-1}\}$ induces a subgraph with independence number at most k.

This generalizes the notion of degeneracy, which considers the number of neighbors instead of the independence number of the neighborhood.

There exists a $1/k$-approximation polynomial-time algorithm for MWIS on graphs with α-degeneracy at most k (Akcoglu et al., 2022).

Perfect elimination orderings: Generalizing to independence degeneracy

The independence degeneracy (α-degeneracy for short) of a graph G is the minimum integer k for which there exists an ordering $(v_1, \ldots, v_n)$ of its vertices such that, for every $i \in \{1, \ldots, n\}$, the set of neighbors of v_i among $\{v_1, \ldots, v_{i-1}\}$ induces a subgraph with independence number at most k.

This generalizes the notion of degeneracy, which considers the number of neighbors instead of the independence number of the neighborhood.

There exists a $1/k$-approximation polynomial-time algorithm for MWIS on graphs with α-degeneracy at most k (Akcoglu et al., 2022).

However, we cannot expect to solve MWIS exactly in polynomial time on graphs with bounded α-degeneracy since MWIS is NP-hard to approximate within Δ^{ε}, for some $\varepsilon > 0$.

Definition

A **clique tree** of a graph G is a pair $(T, \{X_t\}_{t \in V(T)})$ such that T is a tree, each $X_t \subseteq V(G)$ is a maximal clique of G, and the following property holds:

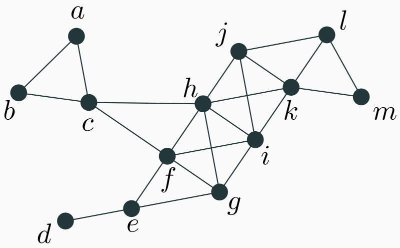
- (Helly property) for every vertex $v \in V(G)$, the set $\{t \in V(T) : v \in X_t\}$ induces a connected subtree of T.

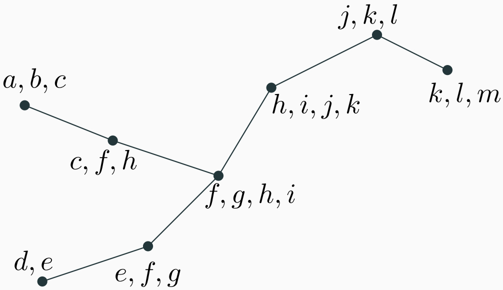
Definition

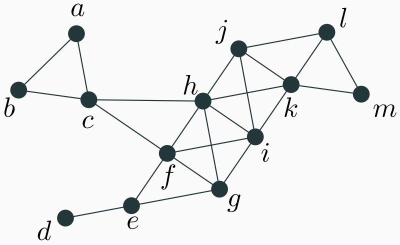
A **clique tree** of a graph G is a pair $(T, \{X_t\}_{t \in V(T)})$ such that T is a tree, each $X_t \subseteq V(G)$ is a maximal clique of G, and the following property holds:

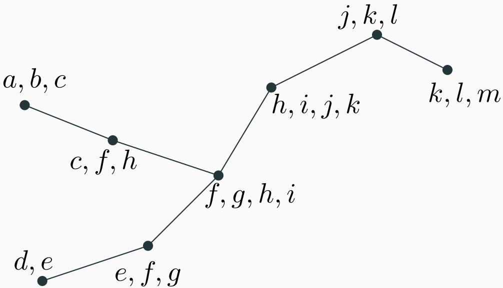
- (Helly property) for every vertex $v \in V(G)$, the set $\{t \in V(T) : v \in X_t\}$ induces a connected subtree of T.

Let's show that every chordal graph admits a clique tree.


a
b
c
d
e
f
g
h
i
j
k
l
m


j, k, l
a, b, c
h, i, j, k
k, l, m
c, f, h
f, g, h, i
d, e
e, f, g





One can construct a clique tree as follows.

Invariant: Each X_t, for $t \in V(T)$, corresponds to a maximal clique of $G[\{v_1, \ldots, v_i\}]$.

- We assume that G is connected; otherwise, build a clique tree for each connected component.
- Consider a p.e.o. $(v_1, \ldots, v_n)$ of a connected chordal graph G.
- Initialize $T = (V, E)$ with $V = E = \emptyset$.
- For each i from 1 to n, do:
  - If $N(v_i) \cap \{v_1, \ldots, v_{i-1}\} = X_{t'}$ for some $t' \in V(T)$, add v_i to $X_{t'}$.
  - Otherwise, create a new node t in T and set $X_t = N[v_i]$. Add the edge tt' to E, where t' is any node of T such that $X_t \setminus \{v_i\} \subseteq X_{t'}$.
- Return $(T, \{X_t\}_{t \in V(T)})$.

Consider a chordal graph G and a clique tree $(T, \{X_t\}_{t \in V(T)})$ of G. Root the clique tree at an arbitrary node r.

Consider a chordal graph G and a clique tree $(T, \{X_t\}_{t \in V(T)})$ of G. Root the clique tree at an arbitrary node r.

Observation

For any $tt' \in E(T)$, the set $X_t \cap X_{t'}$ is a separator of G.

Clique trees: Solving MWIS

Consider a chordal graph G and a clique tree $(T, \{X_t\}_{t \in V(T)})$ of G. Root the clique tree at an arbitrary node r.

Observation

For any $tt' \in E(T)$, the set $X_t \cap X_{t'}$ is a separator of G.

Let $DP_t(S)$ be the maximum weight of an independent set in G_t (the subgraph of G induced by the vertices in the sets $X_{t'}$, for t' either t or a descendant of t in T) that intersects X_t exactly in S.

Clique trees: Solving MWIS

Consider a chordal graph G and a clique tree $(T, \{X_t\}_{t \in V(T)})$ of G. Root the clique tree at an arbitrary node r.

Observation

For any $tt' \in E(T)$, the set $X_t \cap X_{t'}$ is a separator of G.

Let $DP_t(S)$ be the maximum weight of an independent set in G_t (the subgraph of G induced by the vertices in the sets $X_{t'}$, for t' either t or a descendant of t in T) that intersects X_t exactly in S.

Algo$((G, w), (T, \{X_t\}_{t \in V(T)}))$:

return $\max_{\substack{S \subseteq X_r \\ |S| \leq 1}} \mathrm{DP}_r(S)$

$\mathrm{DP}_t(S)$:

$w_t^S = w(S)$
forall $t' \in N^+(t)$ **do**
  $w_t^S += \max_{\substack{S' \subseteq X_{t'} \setminus N[S] \\ |S'| \leq 1}} \mathrm{DP}_{t'}(S')$
return w_t^S

Theorem

MWIS can be solved in $\mathcal{O}(n^2)$ time on n-vertex chordal graphs using a clique tree.

Tree decompositions: Definition

Definition

A **tree decomposition** of a graph G is a pair $(T, \{X_t\}_{t \in V(T)})$ such that T is a tree, each $X_t \subseteq V(G)$ (called a bag), and the following properties hold:

- $\bigcup_{t \in V(T)} X_t = V(G)$,
- for every edge $uv \in E(G)$, there exists $t \in V(T)$ such that $\{u, v\} \subseteq X_t$, and
- for every vertex $v \in V(G)$, the set $\{t \in V(T) : v \in X_t\}$ induces a connected subtree of T.

Tree decompositions: An interesting property

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Tree decompositions: An interesting property

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Lemma

For any graph G and any tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, there exists $u \in V(G)$ and $t \in V(T)$ such that $N[u] \subseteq X_t$.

Tree decompositions: An interesting property

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Lemma

For any graph G and any tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, there exists $u \in V(G)$ and $t \in V(T)$ such that $N[u] \subseteq X_t$.

Sketch of proof: Let G^* be the graph with vertex set $V(G)$ and such that two vertices are adjacent in G^* if and only if they are contained in a same bag of $\mathcal{T}$.

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Lemma

For any graph G and any tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, there exists $u \in V(G)$ and $t \in V(T)$ such that $N[u] \subseteq X_t$.

Sketch of proof: Let G^* be the graph with vertex set $V(G)$ and such that two vertices are adjacent in G^* if and only if they are contained in a same bag of $\mathcal{T}$.

Note that, for every $u \in V(G)$, $N_G[u] \subseteq N_{G^*}[u]$. Furthermore, $\mathcal{T}$ is also a tree decomposition of G^*.

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Lemma

For any graph G and any tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, there exists $u \in V(G)$ and $t \in V(T)$ such that $N[u] \subseteq X_t$.

Sketch of proof: Let G^* be the graph with vertex set $V(G)$ and such that two vertices are adjacent in G^* if and only if they are contained in a same bag of $\mathcal{T}$.

Note that, for every $u \in V(G)$, $N_G[u] \subseteq N_{G^*}[u]$. Furthermore, $\mathcal{T}$ is also a tree decomposition of G^*.

Recall that, for every vertex $u \in V(G)$, the set of nodes of T whose bags contain u induces a connected subtree of T. Hence, two vertices u and v are adjacent in G^* if and only if their corresponding subtrees in T intersect.

Tree decompositions: An interesting property

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Lemma

For any graph G and any tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, there exists $u \in V(G)$ and $t \in V(T)$ such that $N[u] \subseteq X_t$.

Sketch of proof: Let G^* be the graph with vertex set $V(G)$ and such that two vertices are adjacent in G^* if and only if they are contained in a same bag of $\mathcal{T}$.

Note that, for every $u \in V(G)$, $N_G[u] \subseteq N_{G^*}[u]$. Furthermore, $\mathcal{T}$ is also a tree decomposition of G^*.

Recall that, for every vertex $u \in V(G)$, the set of nodes of T whose bags contain u induces a connected subtree of T. Hence, two vertices u and v are adjacent in G^* if and only if their corresponding subtrees in T intersect.

This means that G^* is the intersection graph of subtrees in a tree, i.e., G^* is a chordal graph.

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Lemma

For any graph G and any tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, there exists $u \in V(G)$ and $t \in V(T)$ such that $N[u] \subseteq X_t$.

Sketch of proof: Let G^* be the graph with vertex set $V(G)$ and such that two vertices are adjacent in G^* if and only if they are contained in a same bag of $\mathcal{T}$.

Note that, for every $u \in V(G)$, $N_G[u] \subseteq N_{G^*}[u]$. Furthermore, $\mathcal{T}$ is also a tree decomposition of G^*.

Recall that, for every vertex $u \in V(G)$, the set of nodes of T whose bags contain u induces a connected subtree of T. Hence, two vertices u and v are adjacent in G^* if and only if their corresponding subtrees in T intersect.

This means that G^* is the intersection graph of subtrees in a tree, i.e., G^* is a chordal graph.

Since G^* is chordal, we can find a simplicial vertex $u \in V(G^*)$.

Tree decompositions: An interesting property

Lemma

Let G be a graph and $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ be a tree decomposition of G. Then, for every clique C of G, there exist $t \in V(T)$ such that $C \subseteq X_t$.

Lemma

For any graph G and any tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, there exists $u \in V(G)$ and $t \in V(T)$ such that $N[u] \subseteq X_t$.

Sketch of proof: Let G^* be the graph with vertex set $V(G)$ and such that two vertices are adjacent in G^* if and only if they are contained in a same bag of $\mathcal{T}$.

Note that, for every $u \in V(G)$, $N_G[u] \subseteq N_{G^*}[u]$. Furthermore, $\mathcal{T}$ is also a tree decomposition of G^*.

Recall that, for every vertex $u \in V(G)$, the set of nodes of T whose bags contain u induces a connected subtree of T. Hence, two vertices u and v are adjacent in G^* if and only if their corresponding subtrees in T intersect.

This means that G^* is the intersection graph of subtrees in a tree, i.e., G^* is a chordal graph.

Since G^* is chordal, we can find a simplicial vertex $u \in V(G^*)$.

Then, $N_{G^*}[u]$ is a clique in G^*, and thus there exists $t \in V(T)$ such that $N_{G^*}[u] \subseteq X_t$. $\square$

Tree decompositions: Solving MWIS

Let G be a graph and $(T, \{X_t\}_{t \in V(T)})$ a tree decomposition of G rooted at an arbitrary node r.

Let G be a graph and $(T, \{X_t\}_{t \in V(T)})$ a tree decomposition of G rooted at an arbitrary node r.

Let $DP_t(S)$ be the maximum weight of an independent set in G_t (the subgraph of G induced by the vertices in the sets $X_{t'}$, for t' either t or a descendant of t in T) that intersects X_t exactly in S.

Tree decompositions: Solving MWIS

Let G be a graph and $(T, \{X_t\}_{t \in V(T)})$ a tree decomposition of G rooted at an arbitrary node r.

Let $DP_t(S)$ be the maximum weight of an independent set in G_t (the subgraph of G induced by the vertices in the sets $X_{t'}$, for t' either t or a descendant of t in T) that intersects X_t exactly in S.

Algo$((G, w), (T, \{X_t\}_{t \in V(T)}))$:

return $\max\limits_{\substack{S \subseteq X_r \\ S \text{ is an ind. set}}} \mathrm{DP}_r(S)$

$\mathrm{DP}_t(S)$:

$w_t^S = w(S)$

forall $t' \in N^+(t)$ **do**

$\quad w_t^S += \max\limits_{\substack{S' \subseteq X_{t'} \setminus N[S] \\ S' \text{ is an ind. set}}} \mathrm{DP}_{t'}(S')$

return w_t^S

Let G be a graph and $(T, \{X_t\}_{t \in V(T)})$ a tree decomposition of G rooted at an arbitrary node r.

Let $DP_t(S)$ be the maximum weight of an independent set in G_t (the subgraph of G induced by the vertices in the sets $X_{t'}$, for t' either t or a descendant of t in T) that intersects X_t exactly in S.

Algo$((G, w), (T, \{X_t\}_{t \in V(T)}))$:

return $\max_{\substack{S \subseteq X_r \\ S \text{ is an ind. set}}} \mathrm{DP}_r(S)$

$\mathrm{DP}_t(S)$:

$w_t^S = w(S)$

forall $t' \in N^+(t)$ **do**

  $w_t^S += \max_{\substack{S' \subseteq X_{t'} \setminus N[S] \\ S' \text{ is an ind. set}}} \mathrm{DP}_{t'}(S')$

return w_t^S

Theorem

Given a graph G and a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of G, MWIS can be in time $O(2^w \cdot |V(T)|^2)$, where $w = \max_{t \in V(T)} |X_t|$.

Treewidth

The **width** of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ is $\max_{t \in V(T)} |X_t| - 1$, and the **treewidth** of G, denoted by $\mathrm{tw}(G)$, is the minimum width over all tree decompositions of G.

Treewidth

The **width** of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ is $\max_{t \in V(T)} |X_t| - 1$, and the **treewidth** of G, denoted by $\mathrm{tw}(G)$, is the minimum width over all tree decompositions of G.

While computing the treewidth of a graph G is NP-hard, there exists an exact FPT algorithm that either compute a tree decomposition of width k or report that $\mathrm{tw}(G) > k$, in time $2^{\mathcal{O}(k^2)} \cdot n^4$ (Korhonnen, Lokshtanov, 2023).

The **width** of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ is $\max_{t \in V(T)} |X_t| - 1$, and the **treewidth** of G, denoted by $\mathrm{tw}(G)$, is the minimum width over all tree decompositions of G.

While computing the treewidth of a graph G is NP-hard, there exists an exact FPT algorithm that either compute a tree decomposition of width k or report that $\mathrm{tw}(G) > k$, in time $2^{\mathcal{O}(k^2)} \cdot n^4$ (Korhonnen, Lokshtanov, 2023).

There also exist (faster) approximation algorithms that either compute a tree decomposition of width $\leq f(k)$ or report that $\mathrm{tw}(G) > k$.

The **width** of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ is $\max_{t \in V(T)} |X_t| - 1$, and the **treewidth** of G, denoted by $\mathrm{tw}(G)$, is the minimum width over all tree decompositions of G.

While computing the treewidth of a graph G is NP-hard, there exists an exact FPT algorithm that either compute a tree decomposition of width k or report that $\mathrm{tw}(G) > k$, in time $2^{\mathcal{O}(k^2)} \cdot n^4$ (Korhonnen, Lokshtanov, 2023).

There also exist (faster) approximation algorithms that either compute a tree decomposition of width $\leq f(k)$ or report that $\mathrm{tw}(G) > k$.

Corollary

MWIS can be solved in polynomial time on graphs with bounded treewidth.

Treewidth

The **width** of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ is $\max_{t \in V(T)} |X_t| - 1$, and the **treewidth** of G, denoted by $\mathrm{tw}(G)$, is the minimum width over all tree decompositions of G.

While computing the treewidth of a graph G is NP-hard, there exists an exact FPT algorithm that either compute a tree decomposition of width k or report that $\mathrm{tw}(G) > k$, in time $2^{\mathcal{O}(k^2)} \cdot n^4$ (Korhonnen, Lokshtanov, 2023).

There also exist (faster) approximation algorithms that either compute a tree decomposition of width $\leq f(k)$ or report that $\mathrm{tw}(G) > k$.

Corollary

MWIS can be solved in polynomial time on graphs with bounded treewidth.

The main shortcoming is that many graph classes have unbounded treewidth, including chordal graphs.

Tree-independence number

The **independence number** of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of a graph G, denoted by $\alpha(\mathcal{T})$, is $\max_{t \in V(T)} \alpha(G[X_t])$.

Tree-independence number

The **independence number** of a tree decomposition $(T, \{X_t\}_{t\in V(T)})$ of a graph G, denoted by $\alpha(\mathcal{T})$, is $\max_{t\in V(T)} \alpha(G[X_t])$.

The **tree-independence number** of a graph G, denoted by tree-$\alpha(G)$, is the minimum independence number over all tree decompositions of G (Yolov, 2018, and independently D., Milanič, Štorgel, 2025).

Tree-independence number

The **independence number** of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of a graph G, denoted by $\alpha(\mathcal{T})$, is $\max_{t \in V(T)} \alpha(G[X_t])$.

The **tree-independence number** of a graph G, denoted by tree-$\alpha(G)$, is the minimum independence number over all tree decompositions of G (Yolov, 2018, and independently D., Milanič, Štorgel, 2025).

Computing tree-$\alpha(G)$ is NP-hard (D., Milanič, Štorgel, 2025); even deciding whether tree-$\alpha(G) \leq 4$ is NP-complete (D., Fomin, Golovach, Korhonnen, Milanič, 2024).

Tree-independence number

The independence number of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of a graph G, denoted by $\alpha(\mathcal{T})$, is $\max_{t \in V(T)} \alpha(G[X_t])$.

The tree-independence number of a graph G, denoted by tree-$\alpha(G)$, is the minimum independence number over all tree decompositions of G (Yolov, 2018, and independently D., Milanič, Štorgel, 2025).

Computing tree-$\alpha(G)$ is NP-hard (D., Milanič, Štorgel, 2025); even deciding whether tree-$\alpha(G) \leq 4$ is NP-complete (D., Fomin, Golovach, Korhonnen, Milanič, 2024).

There exists a polynomial-time algorithm that, given a graph G and an integer k, either construct a tree decomposition of independence number $\leq 8k$ or report that tree-$\alpha(G) > k$ (D., Fomin, Golovach, Korhonnen, Milanič, 2025).

Tree-independence number

The independence number of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ of a graph G, denoted by $\alpha(\mathcal{T})$, is $\max_{t \in V(T)} \alpha(G[X_t])$.

The tree-independence number of a graph G, denoted by tree-$\alpha(G)$, is the minimum independence number over all tree decompositions of G (Yolov, 2018, and independently D., Milanič, Štorgel, 2025).

Computing tree-$\alpha(G)$ is NP-hard (D., Milanič, Štorgel, 2025); even deciding whether tree-$\alpha(G) \leq 4$ is NP-complete (D., Fomin, Golovach, Korhonnen, Milanič, 2024).

There exists a polynomial-time algorithm that, given a graph G and an integer k, either construct a tree decomposition of independence number $\leq 8k$ or report that tree-$\alpha(G) > k$ (D., Fomin, Golovach, Korhonnen, Milanič, 2025).

Theorem

MWIS can be solved in polynomial time on graphs with bounded tree-independence number.

From the definition of tree-independence number, it follows that chordal graphs are exactly the graphs with tree-$\alpha(G) = 1$.

Tree independence number: Indeed a generalization of chordal graphs

From the definition of tree-independence number, it follows that chordal graphs are exactly the graphs with tree-$\alpha(G) = 1$.

In particular, graphs with bounded tree-α generalize chordal graphs from the point of view of clique trees and tree decompositions.

Tree independence number: Indeed a generalization of chordal graphs

From the definition of tree-independence number, it follows that chordal graphs are exactly the graphs with tree-$\alpha(G) = 1$.

In particular, graphs with bounded tree-α generalize chordal graphs from the point of view of clique trees and tree decompositions.

For every graph G, it holds that $\omega(G) \leq \mathrm{tw}(G) - 1$ (because every clique must be contained in some bag of any tree decomposition of G).

If G is chordal, then $\omega(G) = \mathrm{tw}(G) - 1$.

From the definition of tree-independence number, it follows that chordal graphs are exactly the graphs with tree-$\alpha(G) = 1$.

In particular, graphs with bounded tree-α generalize chordal graphs from the point of view of clique trees and tree decompositions.

For every graph G, it holds that $\omega(G) \leq \mathrm{tw}(G) - 1$ (because every clique must be contained in some bag of any tree decomposition of G).

If G is chordal, then $\omega(G) = \mathrm{tw}(G) - 1$.

Lemma

For every graph G with tree-$\alpha(G) \leq k$, it holds $\mathrm{tw}(G) < R(k+1, \omega(G)+1)$, where $R(\cdot,\cdot)$ is the Ramsey number, which is upper-bounded by a polynomial in p of degree k.

Tree independence number: Indeed a generalization of chordal graphs

From the definition of tree-independence number, it follows that chordal graphs are exactly the graphs with tree-$\alpha(G) = 1$.

In particular, graphs with bounded tree-α generalize chordal graphs from the point of view of clique trees and tree decompositions.

For every graph G, it holds that $\omega(G) \leq \mathrm{tw}(G) - 1$ (because every clique must be contained in some bag of any tree decomposition of G).

If G is chordal, then $\omega(G) = \mathrm{tw}(G) - 1$.

Lemma

For every graph G with tree-$\alpha(G) \leq k$, it holds $\mathrm{tw}(G) < R(k+1, \omega(G)+1)$, where $R(\cdot,\cdot)$ is the Ramsey number, which is upper-bounded by a polynomial in p of degree k.

However, there exist graph classes whose graphs G have $\mathrm{tw}(G) \leq f(\omega(G)+1)$, for some polynomial function f, but for which tree-$\alpha(G)$ is unbounded.

Tree independence number: Indeed a generalization of chordal graphs

From the definition of tree-independence number, it follows that chordal graphs are exactly the graphs with tree-$\alpha(G) = 1$.

In particular, graphs with bounded tree-α generalize chordal graphs from the point of view of clique trees and tree decompositions.

For every graph G, it holds that $\omega(G) \leq \mathrm{tw}(G) - 1$ (because every clique must be contained in some bag of any tree decomposition of G).

If G is chordal, then $\omega(G) = \mathrm{tw}(G) - 1$.

Lemma

For every graph G with tree-$\alpha(G) \leq k$, it holds $\mathrm{tw}(G) < R(k+1, \omega(G)+1)$, where $R(\cdot,\cdot)$ is the Ramsey number, which is upper-bounded by a polynomial in p of degree k.

However, there exist graph classes whose graphs G have $\mathrm{tw}(G) \leq f(\omega(G)+1)$, for some polynomial function f, but for which tree-$\alpha(G)$ is unbounded.

Let $\mathcal{H}$ be a finite set of connected graphs.

Cameron and Hell construction

Let $\mathcal{H}$ be a finite set of connected graphs.

Given a graph G, denote by $\mathcal{H}(G)$ the following graph:

- the vertices are the subgraphs of G isomorphic to a member of $\mathcal{H}$,
- an edge between two vertices means that the corresponding subgraphs of G have a vertex in common or an edge between them.

Cameron and Hell construction

Let $\mathcal{H}$ be a finite set of connected graphs.

Given a graph G, denote by $\mathcal{H}(G)$ the following graph:

- the vertices are the subgraphs of G isomorphic to a member of $\mathcal{H}$,
- an edge between two vertices means that the corresponding subgraphs of G have a vertex in common or an edge between them.

Note that since $\mathcal{H}$ is finite, computing $\mathcal{H}(G)$ can be done in polynomial time.

Let $\mathcal{H}$ be a finite set of connected graphs.

Given a graph G, denote by $\mathcal{H}(G)$ the following graph:

- the vertices are the subgraphs of G isomorphic to a member of $\mathcal{H}$,
- an edge between two vertices means that the corresponding subgraphs of G have a vertex in common or an edge between them.

Note that since $\mathcal{H}$ is finite, computing $\mathcal{H}(G)$ can be done in polynomial time.

Special cases:

- if $\mathcal{H} = \{K_1\}$, then $\mathcal{H}(G) = G$

Let $\mathcal{H}$ be a finite set of connected graphs.

Given a graph G, denote by $\mathcal{H}(G)$ the following graph:

- the vertices are the subgraphs of G isomorphic to a member of $\mathcal{H}$,
- an edge between two vertices means that the corresponding subgraphs of G have a vertex in common or an edge between them.

Note that since $\mathcal{H}$ is finite, computing $\mathcal{H}(G)$ can be done in polynomial time.

Special cases:

- if $\mathcal{H} = \{K_1\}$, then $\mathcal{H}(G) = G$
- if $\mathcal{H} = \{K_2\}$, then $\mathcal{H}(G)$ is the square of the line graph of G

The Maximum Weight Independent $\mathcal{H}$-Packing problem on a graph G corresponds to the MWIS problem in the graph $\mathcal{H}(G)$.

The Maximum Weight Independent $\mathcal{H}$-Packing problem on a graph G corresponds to the MWIS problem in the graph $\mathcal{H}(G)$.

This is a common generalization of:

- the Independent $\mathcal{H}$-Packing problem (unweighted variant),
- the Maximum Weight Independent Set problem (when $\mathcal{H} = \{K_1\}$),
- the Maximum Weight Induced Matching problem (when $\mathcal{H} = \{K_2\}$),
- the Dissociation Set problem (when $\mathcal{H} = \{K_1, K_2\}$),
- the k-Separator problem (finding a minimum-weight set of vertices whose removal leaves components of size at most k).

One can show that, for every graph G, it holds $\text{tree-}\alpha(\mathcal{H}(G)) \leq \text{tree-}\alpha(G)$.

One can show that, for every graph G, it holds tree-$\alpha(\mathcal{H}(G)) \leq$ tree-$\alpha(G)$.

Sketch of proof: Let $\mathcal{T} = (T, (X_t)_{t \in V(T)})$ be a tree decomposition of G with independence number at most k. Define the tree decomposition $\mathcal{T}' = (T, (X'_t)_{t \in V(T)})$ of $\mathcal{H}(G)$ such that:

- for each $t \in V(T)$, X'_t is the set of vertices of $\mathcal{H}(G)$ whose corresponding subgraphs of G have at least one vertex in X_t.

One can show that, for every graph G, it holds tree-$\alpha(\mathcal{H}(G)) \leq$ tree-$\alpha(G)$.

Sketch of proof: Let $\mathcal{T} = (T, (X_t)_{t \in V(T)})$ be a tree decomposition of G with independence number at most k. Define the tree decomposition $\mathcal{T}' = (T, (X'_t)_{t \in V(T)})$ of $\mathcal{H}(G)$ such that:

- for each $t \in V(T)$, X'_t is the set of vertices of $\mathcal{H}(G)$ whose corresponding subgraphs of G have at least one vertex in X_t.

Since graphs in $\mathcal{H}$ are connected, $\mathcal{T}'$ is a valid tree decomposition of $\mathcal{H}(G)$.

One can show that, for every graph G, it holds tree-$\alpha(\mathcal{H}(G)) \leq$ tree-$\alpha(G)$.

Sketch of proof: Let $\mathcal{T} = (T, (X_t)_{t \in V(T)})$ be a tree decomposition of G with independence number at most k. Define the tree decomposition $\mathcal{T}' = (T, (X'_t)_{t \in V(T)})$ of $\mathcal{H}(G)$ such that:

- for each $t \in V(T)$, X'_t is the set of vertices of $\mathcal{H}(G)$ whose corresponding subgraphs of G have at least one vertex in X_t.

Since graphs in $\mathcal{H}$ are connected, $\mathcal{T}'$ is a valid tree decomposition of $\mathcal{H}(G)$.

Furthermore, an independent set in $\mathcal{H}(G)[X'_t]$, for some $t \in V(T)$, implies the existence of an independent set in $G[X_t]$ of the same size. Thus, $\alpha(G[X'_t]) \leq \alpha(G[X_t]) \leq k$. $\square$

One can show that, for every graph G, it holds tree-$\alpha(\mathcal{H}(G)) \leq$ tree-$\alpha(G)$.

Sketch of proof: Let $\mathcal{T} = (T, (X_t)_{t\in V(T)})$ be a tree decomposition of G with independence number at most k. Define the tree decomposition $\mathcal{T}' = (T, (X'_t)_{t\in V(T)})$ of $\mathcal{H}(G)$ such that:

- for each $t \in V(T)$, X'_t is the set of vertices of $\mathcal{H}(G)$ whose corresponding subgraphs of G have at least one vertex in X_t.

Since graphs in $\mathcal{H}$ are connected, $\mathcal{T}'$ is a valid tree decomposition of $\mathcal{H}(G)$.

Furthermore, an independent set in $\mathcal{H}(G)[X'_t]$, for some $t \in V(T)$, implies the existence of an independent set in $G[X_t]$ of the same size. Thus, $\alpha(G[X'_t]) \leq \alpha(G[X_t]) \leq k$. $\square$

Theorem

The Maximum Weight Independent $\mathcal{H}$-Packing problem is solvable in polynomial time on graphs with bounded tree-independence number.

Counting monadic second-order logic (*$CMSO_2$*) is an extension of monadic second-order logic (*MSO_2*) that allows counting the cardinality of sets modulo some fixed integer.

Counting monadic second-order logic ($CMSO_2$) is an extension of monadic second-order logic (MSO_2) that allows counting the cardinality of sets modulo some fixed integer.

Theorem

Let φ be a fixed $CMSO_2$ formula and let k and c be fixed integers. Then, given a graph G with tree-$\alpha(G) \leq k$, one can find in polynomial time a maximum-weight induced subgraph of G with clique number at most c that satisfies φ (if such a subgraph exists).

Operations that are monotone for tree-α

The following graph operations do not increase the tree-independence number:

- vertex deletion,

Operations that are monotone for tree-α

The following graph operations do not increase the tree-independence number:

- vertex deletion,
- edge contraction,

Operations that are monotone for tree-α

The following graph operations do not increase the tree-independence number:

- vertex deletion,
- edge contraction,
- gluing two graphs along a clique.

Operations that are monotone for tree-α

The following graph operations do not increase the tree-independence number:

- vertex deletion,
- edge contraction,
- gluing two graphs along a clique.

However, edge deletion does not necessarily preserve the tree-independence number.

Operations that are monotone for tree-α

The following graph operations do not increase the tree-independence number:

- vertex deletion,
- edge contraction,
- gluing two graphs along a clique.

However, edge deletion does not necessarily preserve the tree-independence number.

Definition

A graph H is an induced minor of a graph G if H can be obtained from G by a sequence of vertex deletions and edge contractions.

Operations that are monotone for tree-α

The following graph operations do not increase the tree-independence number:

- vertex deletion,
- edge contraction,
- gluing two graphs along a clique.

However, edge deletion does not necessarily preserve the tree-independence number.

Definition

A graph H is an **induced minor** of a graph G if H can be obtained from G by a sequence of vertex deletions and edge contractions.

Proposition

If H is an induced minor of G, then $\text{tree-}\alpha(H) \leq \text{tree-}\alpha(G)$.

Some graph classes with unbounded tree-α

The following families of graph classes all have unbounded tree-$\boldsymbol{\alpha}$:

- bipartite graphs (tree-$\alpha(K_{n,n}) = n$),

Some graph classes with unbounded tree-α

The following families of graph classes all have unbounded tree-$\boldsymbol{\alpha}$:

- bipartite graphs (tree-$\alpha(K_{n,n}) = n$),
- graphs with large treewidth and small chromatic number, which include planar graphs, and in particular walls and their line graphs.

Some graph classes with unbounded tree-α

The following families of graph classes all have unbounded tree-α:

- bipartite graphs (tree-$\alpha(K_{n,n}) = n$),
- graphs with large treewidth and small chromatic number, which include planar graphs, and in particular walls and their line graphs.

From the previous proposition, it follows that any graph class that contains large complete bipartite graphs, large walls, or the line graphs of large walls as induced minors has unbounded tree-α.

Some graph classes with bounded tree-α

The following families of graph classes all have bounded tree-α:

- graph classes of bounded independence number (tree-$\alpha(G) \leq \alpha(G)$);

The following families of graph classes all have bounded tree-α:

- graph classes of bounded independence number (tree-$\alpha(G) \leq \alpha(G)$);
- graph classes of bounded treewidth (tree-$\alpha(G) \leq \mathrm{tw}(G) + 1$);

The following families of graph classes all have bounded tree-α:

- graph classes of bounded independence number (tree-$\alpha(G) \leq \alpha(G)$);
- graph classes of bounded treewidth (tree-$\alpha(G) \leq \mathrm{tw}(G) + 1$);
- classes of graphs in which all minimal separators are of bounded size (following a result by Skodinis).

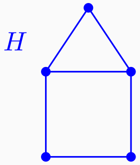
Graph containment relations

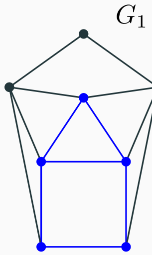
$H \subseteq_{itm} G_2$ $H \subseteq_{im} G_3$

induced subgraph

induced topological minor

induced minor





$H \subseteq_{is} G_1$

induced
subgraph

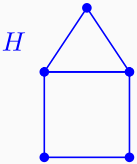
$H \subseteq_{itm} G_2$

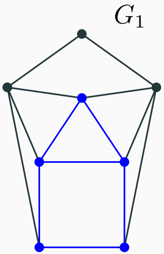
induced
topological minor

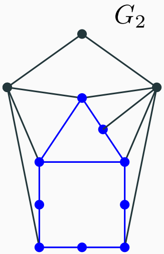
$H \subseteq_{im} G_3$

induced
minor

Graph containment relations





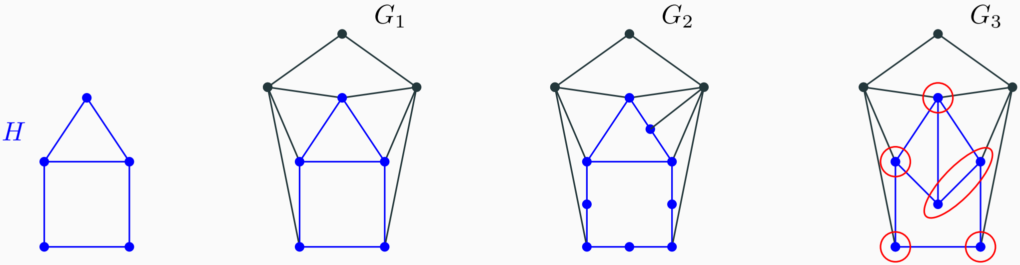


$H \subseteq_{is} G_1$

induced
subgraph

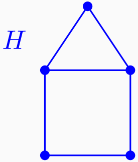
$H \subseteq_{itm} G_2$

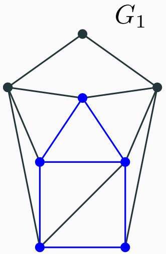
induced
topological minor



$H \subseteq_{is} G_1$	$H \subseteq_{itm} G_2$	$H \subseteq_{im} G_3$
induced subgraph	induced topological minor	induced minor

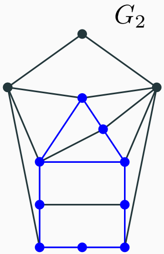
Graph containment relations





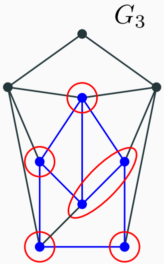
$H \subseteq_s G_1$

subgraph



$H \subseteq_{tm} G_2$

topological minor



$H \subseteq_m G_3$

minor

Forbidding a graph H

Graphs H for which the class of graphs excluding H has bounded tree-independence number (D., Milanič, Štorgel, 2024):

	Non-induced	Induced
Subgraph	$H \in \mathcal{S}$	P_3 or edgeless
Topological minor	H is subcubic and planar	C_4, K_4^-, or edgeless
Minor	H is planar	W_4, K_5^-, $K_{2,q}$ for some $q \in \mathbb{N}$

$\mathcal{S}$ is the class of graphs whose connected components are either paths or subdivisions of the claw ($K_{1,3}$).

In all the bounded cases above, we can efficiently compute a tree decomposition with bounded independence number.

Some recent works in the world of tree-α

- Treewidth versus clique number. IV. Tree-independence number of graphs excluding an induced star,
  C. Dallard, M. Krnc, O. Kwon, M. Milanič, A. Munaro, K. Štorgel, S. Wiederrecht, 2024.
- Polynomial-time approximation schemes for induced subgraph problems on fractionally tree-independence-number-fragile graphs,
  E. Galby, A. Munaro, S. Yang, 2024.
- Tree independence number II. Three-path-configurations,
  M. Chudnovsky, S. Hajebi, D. Lokshtanov, S. Spirkl, 2024.
- Tree independence number III. Thetas, prisms and stars,
  M. Chudnovsky, S. Hajebi, N. Trotignon, 2024.
- Tree independence number IV. Even-hole-free graphs,
  M. Chudnovsky, P. Gartland, S. Hajebi, D. Lokshtanov, S. Spirkl, 2024.
- Tree independence number V. Walls and claws,
  M. Chudnovsky, J. Codsi, D. Lokshtanov, M. Milanič, V. Sivashankar, 2025.
- Treewidth versus clique number. V. Further connections with tree-independence number,
  Claire Hilaire, Martin Milanič, Đorđe Vasić
- On the relation between treewidth, tree-independence number, and tree-chromatic number of graphs,
  K. Krause, M. Redzic, T. Ueckerdt, 2025.
- Excluding a Ladder as an Induced Minor in Graphs Without Induced Stars,
  M. Choi, S. Wiederrecht, 2025.
- Layered tree-independence number and clique-based separators,
  C. Dallard, M. Milanič, A. Munaro, S. Yang, 2025.
- Tree-independence number VI. Thetas and pyramids,
  M. Chudnovsky, J. Codsi, 2025.
- (Treewidth, Clique)-Boundedness and Poly-logarithmic Tree-Independence,
  M. Chudnovsky, A. E. S., D. Lokshtanov, 2025.

Beyond independence: Induced matching treewidth

For a graph G and a tree decomposition $\mathcal{T} = (T, \{X_t\}_{t\in V(T)})$ of G, let

$$\mu(\mathcal{T}) = \max_{t\in V(T)} \{M\colon M \text{ is an induced matching s.t. every edge of } M \text{ has at least one endpoint in } X_t\}.$$

Beyond independence: Induced matching treewidth

For a graph G and a tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ of G, let

$$\mu(\mathcal{T}) = \max_{t \in V(T)} \{M \colon M \text{ is an induced matching s.t. every edge of } M \text{ has at least one endpoint in } X_t\}.$$

The **induced matching treewidth** of G, denoted tree-$\mu(G)$, is the minimum of $\mu(\mathcal{T})$ over all tree decompositions $\mathcal{T}$ of G.

Beyond independence: Induced matching treewidth

For a graph G and a tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ of G, let

$$\mu(\mathcal{T}) = \max_{t \in V(T)} \{M \colon M \text{ is an induced matching s.t. every edge of } M \text{ has at least one endpoint in } X_t\}.$$

The **induced matching treewidth** of G, denoted tree-$\mu(G)$, is the minimum of $\mu(\mathcal{T})$ over all tree decompositions $\mathcal{T}$ of G.

Note that a graph class with bounded tree-α also has bounded tree-μ, but the converse is not true (e.g., complete bipartite graphs have tree-$\mu = 1$).

Beyond independence: Induced matching treewidth

For a graph G and a tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ of G, let

$$\mu(\mathcal{T}) = \max_{t \in V(T)} \{M \colon M \text{ is an induced matching s.t. every edge of } M \text{ has at least one endpoint in } X_t\}.$$

The **induced matching treewidth** of G, denoted tree-$\mu(G)$, is the minimum of $\mu(\mathcal{T})$ over all tree decompositions $\mathcal{T}$ of G.

Note that a graph class with bounded tree-α also has bounded tree-μ, but the converse is not true (e.g., complete bipartite graphs have tree-$\mu = 1$).

While determining tree-$\mu(G)$ is NP-hard, there exists a polynomial-time algorithm that, given a graph G and an integer k, either constructs a tree decomposition of G with induced matching number at most $\mathcal{O}(k^3)$ or reports that tree-$\mu(G) > k$ (Yolov, 2018).

Theorem (Lima et al., 2024)

Graphs with bounded tree-μ admit polynomial-time algorithms for MWIS, Max Weight Induced Forest, Max Weight Independent $\mathcal{H}$-Packing (for finite $\mathcal{H}$ of connected graphs), and more.