List matrix partition problems on chordal graphs parameterized by leafage

Flavia Bonomo

Universidad de Buenos Aires, Argentina

XVII Congreso Dr. Antonio Monteiro
UNS, Bahía Blanca, 9 de junio de 2023

Outline

1. Generalized list matrix partition problems
   - Matrix partition problems
   - With lists
   - On chordal graphs
   - With objective functions
   - With bounds
   - Problem definition
2. GLMP on graphs of bounded thinness
   - Thinness
   - The algorithm
   - Complexity boundaries
3. Algorithmic applications to chordal and co-comparability graphs
   - Thinness and leafage of chordal graphs
   - Thinness and chromatic number of co-comparability graphs

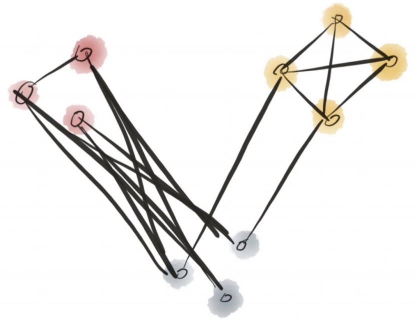
Outline

Matrix partition problems

(Feder, Hell, Klein and Motwani, 1999)

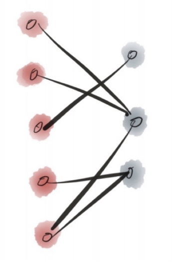
For each symmetric matrix M over $\{0, 1, *\}$, the M-Partition Problem seeks a partition of the input graph into independent sets, cliques, or arbitrary sets, with certain pairs of sets being required to have no edges, or to have all edges joining them, as encoded in the matrix M.

$$\begin{bmatrix} * & 0 & 1 \\ 0 & 1 & * \\ 1 & * & 0 \end{bmatrix}$$

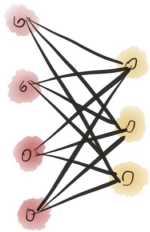


Example: bipartition and complete bipartition

$$\begin{bmatrix} 0 & * \\ * & 0 \end{bmatrix}$$

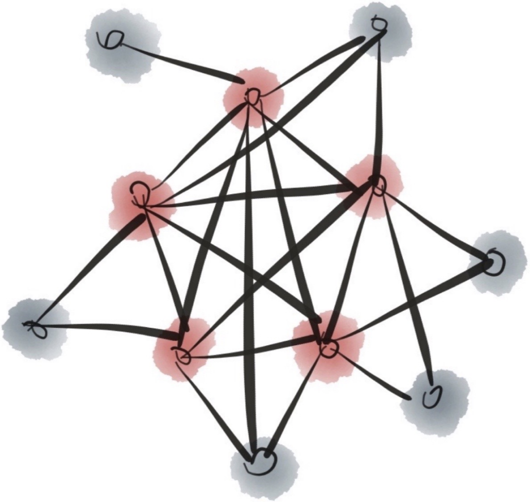


$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$



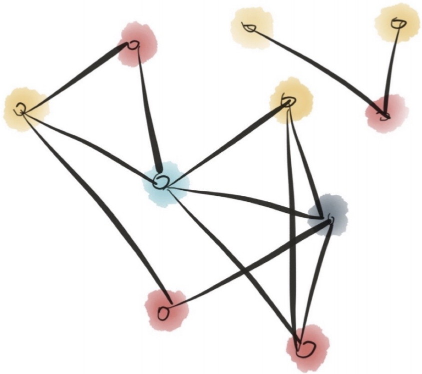
Example: Split partition

$$\begin{bmatrix} 0 & * \\ * & 1 \end{bmatrix}$$



Example: r-coloring

$$\begin{bmatrix} 0 & * & * & \cdots & * \\ * & 0 & * & \cdots & * \\ \vdots & & \ddots & & \vdots \\ * & * & * & \cdots & 0 \end{bmatrix}$$



Adding lists

In the LIST M-PARTITION PROBLEM, additionally, each vertex v has a list $L(v)$ of the sets $S_1, \ldots, S_r$ to which it can belong. That is, we have a fixed $r \times r$ matrix M, an input graph $G = (V, E)$ and an input function $L : V \to \mathcal{P}(\{1, \ldots, r\})$.

In the r-coloring case, the problem is known as LIST r-COLORING.

Adding lists

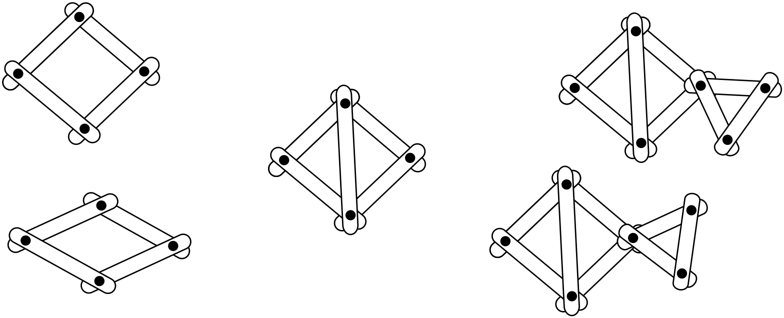
In the LIST M-PARTITION PROBLEM, additionally, each vertex v has a list $L(v)$ of the sets $S_1, \dots, S_r$ to which it can belong. That is, we have a fixed $r \times r$ matrix M, an input graph $G = (V, E)$ and an input function $L : V \to \mathcal{P}(\{1, \dots, r\})$.

In the r-coloring case, the problem is known as LIST r-COLORING.

In a more general setting, that is not necessarily a partition, each vertex v has a list $L(v)$ of combinations of the sets $S_1, \dots, S_r$ to which it can belong (that may include the empty combination). That is, we have a fixed $r \times r$ matrix M, an input graph $G = (V, E)$ and an input function $L : V \to \mathcal{P}(\mathcal{P}(\{1, \dots, r\}))$.

Chordal graphs

A graph is chordal if it does not contain a chordless cycle of length at least 4 (C_n, $n \geq 4$).
They are also called triangulated or rigid circuit.



Matrix partition problems on chordal graphs

Feder, Hell, Klein, Nogueira and Protti identified several cases of $r \times r$ matrices M for which the list M-partition problem is polynomial-time solvable. For instance,

Theorem (Feder, Hell, Klein, Nogueira and Protti, 2005)

If all diagonal entries of M are zero, then the CHORDAL LIST M-PARTITION problem can be solved in time $O(nr(2r)^r)$, linear in n.

Matrix partition problems on chordal graphs

Feder, Hell, Klein, Nogueira and Protti identified several cases of $r \times r$ matrices M for which the list M-partition problem is polynomial-time solvable. For instance,

Theorem (Feder, Hell, Klein, Nogueira and Protti, 2005)

If all diagonal entries of M are zero, then the CHORDAL LIST M-PARTITION problem can be solved in time $O(nr(2r)^r)$, linear in n.

Idea: A solution is, in particular, a r-coloring. Chordal graphs are either not r-colorable or have treewidth at most $r-1$. The algorithm then works with a tree-decomposition of the graph.

Matrix partition problems on chordal graphs

However,

Theorem (Feder, Hell, Klein, Nogueira and Protti, 2005)

There are matrices M such that the M-PARTITION problem is NP-complete for chordal graphs.

Adding weights and an objective function

We still have an $r \times r$ matrix M, an input graph $G = (V, E)$, and input arbitrary nonnegative weight functions $w_1, \ldots, w_t$ on V.
The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$.

Examples:

Max Weight Independent Set

$$M = \begin{bmatrix} 0 & * \\ * & * \end{bmatrix}$$

$t = 1$, $w_1 : V \to \mathbb{N}_0$

$c_{11} = 1$; $c_{12} = 0$;

Max Weight Clique

$$M = \begin{bmatrix} 1 & * \\ * & * \end{bmatrix}$$

$t = 1$, $w_1 : V \to \mathbb{N}_0$

$c_{11} = 1$; $c_{12} = 0$;

Adding bounds on the sizes of the sets

In this case, we can add to the M-partition problem, constraints about the number (or sum of weights) of elements that can be placed on each of the sets.

In the coloring setting, this is called in the literature bounded coloring (the size of any color class does not exceed a given bound), capacitated coloring (different upper bounds for each color), equitable coloring (some of the upper bounds are $\lceil \frac{n}{r} \rceil$ and some $\lfloor \frac{n}{r} \rfloor$, in order to make the difference of color class sizes at most one).

In combination with lists, they are called list coloring problem with cardinalities (exact expected number), list coloring problem with bounded cardinalities (upper bounds on the expected number), weighted locally bounded list coloring (more general, with different weight functions and bounds).

Generalized list matrix partition problems

Generalized (p, q, r)-List Matrix Partition ((p, q, r)-GLMP)
For a graph $G = (V, E)$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.

Generalized list matrix partition problems

GENERALIZED (p, q, r)-LIST MATRIX PARTITION ((p, q, r)-GLMP)
For a graph $G = (V, E)$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).

Generalized list matrix partition problems

Generalized (p, q, r)-List Matrix Partition ((p, q, r)-GLMP)

For a graph $G = (V, E)$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).
- There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.

Generalized list matrix partition problems

Generalized (p, q, r)-List Matrix Partition ((p, q, r)-GLMP)
For a graph $G = (V, E)$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).
- There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.
- There is a family of nonnegative weights $b_1, \ldots, b_p$ on V, p fixed and each b_i bounded by a fixed polynomial $q(n)$, and restrictions on the weight of intersections and unions of the output sets, expressed as:

Generalized list matrix partition problems

GENERALIZED (p, q, r)-LIST MATRIX PARTITION ((p, q, r)-GLMP)
For a graph $G = (V, E)$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \le i \le t; 1 \le j \le r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).
- There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.
- There is a family of nonnegative weights $b_1, \ldots, b_p$ on V, p fixed and each b_i bounded by a fixed polynomial $q(n)$, and restrictions on the weight of intersections and unions of the output sets, expressed as:
  - $0 \le \ell_{iJ\cap} \le b_i(\bigcap_{j \in J} S_j) \le u_{iJ\cap}$, such that $1 \le i \le p$, $J \subseteq \{1, \ldots, r\}$.

Generalized list matrix partition problems

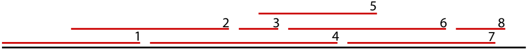
GENERALIZED (p, q, r)-LIST MATRIX PARTITION ((p, q, r)-GLMP)
For a graph $G = (V, E)$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).
- There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.
- There is a family of nonnegative weights $b_1, \ldots, b_p$ on V, p fixed and each b_i bounded by a fixed polynomial $q(n)$, and restrictions on the weight of intersections and unions of the output sets, expressed as:
  - $0 \leq \ell_{iJ\cap} \leq b_i(\bigcap_{j \in J} S_j) \leq u_{iJ\cap}$, such that $1 \leq i \leq p$, $J \subseteq \{1, \ldots, r\}$.
  - $0 \leq \ell_{iJ\cup} \leq b_i(\bigcup_{j \in J} S_j) \leq u_{iJ\cup}$, such that $1 \leq i \leq p$, $J \subseteq \{1, \ldots, r\}$.

Outline

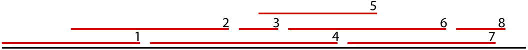
Interval graphs

- The right-end ordering of the vertices of an interval graph satisfies the following property: for each triple (r, s, t) with $r < s < t$, if $v_t v_r \in E$, then $v_t v_s \in E$.



Interval graphs

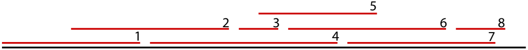
- The right-end ordering of the vertices of an interval graph satisfies the following property: for each triple (r, s, t) with $r < s < t$, if $v_t v_r \in E$, then $v_t v_s \in E$.



- In other words, the neighbours of vertex t with index less than t are $t-1, t-2, \ldots, t-d$.

Interval graphs

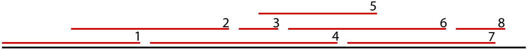
- The right-end ordering of the vertices of an interval graph satisfies the following property: for each triple (r, s, t) with $r < s < t$, if $v_t v_r \in E$, then $v_t v_s \in E$.



- In other words, the neighbours of vertex t with index less than t are $t-1, t-2, \ldots, t-d$.
- Moreover, G is an interval graph if and only if there exists an ordering of its vertices satisfying the property above (Ramalingam and Pandu Rangan, 1988 / Olariu, 1991).

Interval graphs

- The right-end ordering of the vertices of an interval graph satisfies the following property: for each triple (r, s, t) with $r < s < t$, if $v_t v_r \in E$, then $v_t v_s \in E$.

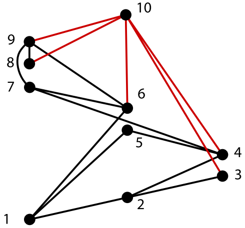


- In other words, the neighbours of vertex t with index less than t are $t-1, t-2, \ldots, t-d$.
- Moreover, G is an interval graph if and only if there exists an ordering of its vertices satisfying the property above (Ramalingam and Pandu Rangan, 1988 / Olariu, 1991).
- The thinness of a graph was introduced by Mannino, Oriolo, Ricci, and Chandran in 2007, such that graphs with bounded thinness are a generalization of interval graphs (that are exactly the graphs with thinness 1) and capture some of their algorithmic properties.

k-thin graphs

Definition

A graph $G = (V, E)$ is k-thin if there exist an ordering $v_1, \ldots, v_n$ of V and a partition of V into k classes such that, for each triple (r, s, t) with $r < s < t$, if v_r, v_s belong to the same class and $v_t v_r \in E$, then $v_t v_s \in E$.

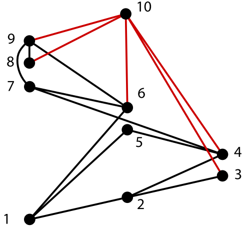


In other words, the neighbours of vertex t with index less than t on each class are the greatest of each class.

k-thin graphs

Definition

A graph $G = (V, E)$ is k-thin if there exist an ordering $v_1, \ldots, v_n$ of V and a partition of V into k classes such that, for each triple (r, s, t) with $r < s < t$, if v_r, v_s belong to the same class and $v_t v_r \in E$, then $v_t v_s \in E$.



In other words, the neighbours of vertex t with index less than t on each class are the greatest of each class.

The minimum k such that G is k-thin is called the thinness of G, $thin(G)$.

Graphs with high and low thinness

Trees, co-bipartite graphs, split graphs, permutation graphs, circular-arc graphs and cographs have unbounded thinness.

Interval bigraphs have thinness at most 2 (B. and Brito, 2022).

Deciding if the thinness of a graph is at most k is NP-complete (Shitov, 2021).

The problem is open for fixed $k \geq 2$.

(p, q, r)-GLMP problems on graphs with bounded thinness

We are given the (p, q, r)-GLMP instance and a k-thin representation of $G = (V, E)$, with ordering $<$ of V, namely $v_1 < \cdots < v_n$, and partition of V into k classes $V^1, \ldots, V^k$.

We will solve such a problem by dynamic programming, as a shortest or longest path problem (according to minimization or maximization of the objective function) in an auxiliary acyclic digraph $D = (X, A)$ whose nodes correspond to *states* and whose arcs are weighted and labeled. The total weight of the path is the value of the objective function in the solution that can be built by using the arc labels.

GL~~M~~P resembles knapsack (the graph becomes irrelevant)

Generalized (p, q, r)-List ~~Matrix~~ Partition ((p, q, r)-GLMP)
For a ~~graph~~ set $\cancel{G = (}V\cancel{, E)}$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.

GL~~M~~P resembles knapsack (the graph becomes irrelevant)

Generalized (p, q, r)-List ~~Matrix~~ Partition ((p, q, r)-GLMP)
For a ~~graph~~ set $\cancel{G = (}V\cancel{, E)}$, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).

GL~~M~~P resembles knapsack (the graph becomes irrelevant)

GENERALIZED (p, q, r)-LIST ~~MATRIX~~ PARTITION ((p, q, r)-GLMP)
For a ~~graph~~ set $\cancel{G = (}V\cancel{, E)}$, the aim is to find subsets of vertices $S_1, \dots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \dots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \dots, S_r$ to which it can belong (that may include the empty combination).
- ~~There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.~~

GL~~M~~P resembles knapsack (the graph becomes irrelevant)

GENERALIZED (p, q, r)-LIST ~~MATRIX~~ PARTITION ((p, q, r)-GLMP)
For a ~~graph~~ set $\cancel{G = (}V\cancel{, E)}$, the aim is to find subsets of vertices $S_1, \dots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \dots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \dots, S_r$ to which it can belong (that may include the empty combination).
- ~~There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.~~
- There is a family of nonnegative weights $b_1, \dots, b_p$ on V, p fixed and each b_i bounded by a fixed polynomial $q(n)$, and restrictions on the weight of intersections and unions of the output sets, expressed as:

GL~~M~~P resembles knapsack (the graph becomes irrelevant)

GENERALIZED (p, q, r)-LIST ~~MATRIX~~ PARTITION ((p, q, r)-GLMP)
For a ~~graph~~ set ~~$G = (V, E)$~~, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).
- ~~There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.~~
- There is a family of nonnegative weights $b_1, \ldots, b_p$ on V, p fixed and each b_i bounded by a fixed polynomial $q(n)$, and restrictions on the weight of intersections and unions of the output sets, expressed as:
  - $0 \leq \ell_{iJ\cap} \leq b_i(\bigcap_{j \in J} S_j) \leq u_{iJ\cap}$, such that $1 \leq i \leq p$, $J \subseteq \{1, \ldots, r\}$.

GL~~M~~P resembles knapsack (the graph becomes irrelevant)

GENERALIZED (p, q, r)-LIST ~~MATRIX~~ PARTITION ((p, q, r)-GLMP)
For a ~~graph~~ set ~~$G = (V, E)$~~, the aim is to find subsets of vertices $S_1, \ldots, S_r$ (r fixed, not necessarily disjoint), such that:

- The objective is to minimize or maximize a linear function $\sum_{1 \leq i \leq t; 1 \leq j \leq r} c_{ij} w_i(S_j)$, for a family of arbitrary nonnegative weights $w_1, \ldots, w_t$ on V.
- Each vertex v has a list $L(v)$ of combinations of the sets $S_1, \ldots, S_r$ to which it can belong (that may include the empty combination).
- ~~There is an $r \times r$ symmetric matrix M over $\{0, 1, *\}$, stating the nature and adjacency conditions on the sets S_j.~~
- There is a family of nonnegative weights $b_1, \ldots, b_p$ on V, p fixed and each b_i bounded by a fixed polynomial $q(n)$, and restrictions on the weight of intersections and unions of the output sets, expressed as:
  - $0 \leq \ell_{iJ\cap} \leq b_i(\bigcap_{j \in J} S_j) \leq u_{iJ\cap}$, such that $1 \leq i \leq p$, $J \subseteq \{1, \ldots, r\}$.
  - $0 \leq \ell_{iJ\cup} \leq b_i(\bigcup_{j \in J} S_j) \leq u_{iJ\cup}$, such that $1 \leq i \leq p$, $J \subseteq \{1, \ldots, r\}$.

Sketch of the algorithm

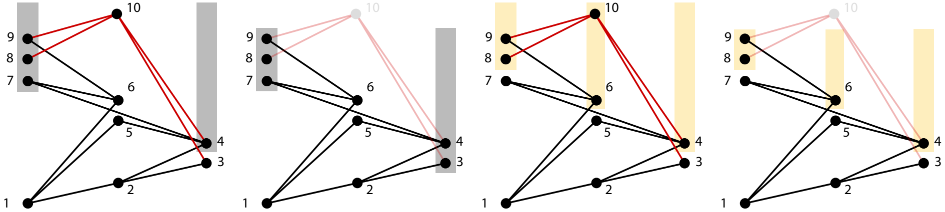
A state is a tuple, containing:

- a number $s \leq n$ indicating we are considering $G_s = G[\{v_1, \ldots, v_s\}]$.
- nonnegative parameters $\ell_{iJ\cap}, u_{iJ\cap}, \ell_{iJ\cup}, u_{iJ\cup}$, for $1 \leq i \leq p$, $J \subseteq \{1, \ldots, r\}$; they are at most $2^{r+2}p$, and each of them may take a nonnegative value at most $nq(n)$, which is an upper bound for $b_i(V)$, for every $1 \leq i \leq p$.
- a family of nonnegative parameters $\{\alpha_{ij}\}_{1 \leq i \leq k; 1 \leq j \leq r}$, meaning that we cannot pick for S_j a vertex of the first α_{ij} vertices of the set V^i of the partition; there are kr such parameters and each of them may take a nonnegative value at most $n - 1$.
- a family of nonnegative parameters $\{\beta_{ij}\}_{1 \leq i \leq k; 1 \leq j \leq r}$, meaning that we cannot pick for S_j a vertex on the last β_{ij} vertices of the set V^i of the partition; there are kr such parameters and each of them may take a nonnegative value at most $n - 1$.

The total number of states is then at most $n^{2kr+1}(nq(n))^{2^{r+2}p}$, that is polynomial in n, since k, r, and p are constant and $q(n)$ is polynomial in n.

Sketch of the algorithm: how do we use the thinness

Example of a state of G_{10} and the computation of the parameters $\{\beta_{ij}\}_{i,j}$ for a predecessor state of G_9, assuming that $\beta_{2,1} = 0$, $\{1\} \in L(v_{10})$, all the weight constraints are satisfied, and we are assigning, as one of the possibilities, $v_{10} \in V^2$ to the set S_1.



$\{\beta_{ij}\}_i$ $\quad$ $\{\beta_{ij}\}_i$, j s.t. $M_{1j} = 0$ $\quad$ $\{\beta_{ij}\}_i$ $\quad$ $\{\beta_{ij}\}_i$, j s.t. $M_{1j} \neq 0$

(p, q, r)-GLMP on k-thin graphs

Theorem (B., De Estrada, 2019)

Given as input a k-thin representation of a graph G, (p, q, r)-Generalized List Matrix Partition Problem can be solved in $O(n^{4kr+2}(nq(n))^{2^{r+3}p})$ time, that is, XP with respect to the fixed parameters r, p, and k.

Tightness of the algorithmic results

All the conditions in blue (number r of sets fixed, number p of weights b_i fixed, weights b_i polynomially bounded by $q(n)$) are necessary for the existence of a polynomial time algorithm in graphs of bounded thinness, unless $P = NP$, even in the feasibility case (without objective function).

- If $p = 1$ but the only weight b_1 is not polynomially bounded, GLMP is NP-complete in edgeless graphs, even if $r = 2$, $M = \begin{pmatrix} * & * \\ * & * \end{pmatrix}$ and $L(v) = \mathcal{P}(\{1, 2\})$ for every v (Bentz, 2019).

Tightness of the algorithmic results

All the conditions in blue (number r of sets fixed, number p of weights b_i fixed, weights b_i polynomially bounded by $q(n)$) are necessary for the existence of a polynomial time algorithm in graphs of bounded thinness, unless $P = NP$, even in the feasibility case (without objective function).

- If $p = 1$ but the only weight b_1 is not polynomially bounded, GLMP is NP-complete in edgeless graphs, even if $r = 2$, $M = \begin{pmatrix} * & * \\ * & * \end{pmatrix}$ and $L(v) = \mathcal{P}(\{1, 2\})$ for every v (Bentz, 2019).
- If p is not bounded, GLMP is NP-complete in star forests and linear forests, even if $r = 2$, $M = \begin{pmatrix} 0 & * \\ * & 0 \end{pmatrix}$ (coloring problem), $L(v) = \{\{1\}, \{2\}\}$, and $b_i(v) \in \{0, 1\}$ for every v and every i (Bentz, 2019).

Tightness of the algorithmic results

All the conditions in blue (number r of sets fixed, number p of weights b_i fixed, weights b_i polynomially bounded by $q(n)$) are necessary for the existence of a polynomial time algorithm in graphs of bounded thinness, unless $P = NP$, even in the feasibility case (without objective function).

- If $p = 1$ but the only weight b_1 is not polynomially bounded, GLMP is NP-complete in edgeless graphs, even if $r = 2$, $M = \begin{pmatrix} * & * \\ * & * \end{pmatrix}$ and $L(v) = \mathcal{P}(\{1, 2\})$ for every v (Bentz, 2019).
- If p is not bounded, GLMP is NP-complete in star forests and linear forests, even if $r = 2$, $M = \begin{pmatrix} 0 & * \\ * & 0 \end{pmatrix}$ (coloring problem), $L(v) = \{\{1\}, \{2\}\}$, and $b_i(v) \in \{0, 1\}$ for every v and every i (Bentz, 2019).
- In both cases, the instance can be made connected (a path) with $r = 3$ and using lists.

When the number of output sets is not fixed

- The coloring problem is polynomial time solvable in chordal graphs (Golumbic, 1980), but it is NP-complete for subclasses of 2-thin graphs (B., Brandwein, Oliveira, Sampaio, Sansone and Szwarcfiter, 2023).

When the number of output sets is not fixed

- The coloring problem is polynomial time solvable in chordal graphs (Golumbic, 1980), but it is NP-complete for subclasses of 2-thin graphs (B., Brandwein, Oliveira, Sampaio, Sansone and Szwarcfiter, 2023).
- The list-coloring problem is NP-complete for interval graphs (Biro, Hujter and Tuza, 1992).

When the number of output sets is not fixed

- The coloring problem is polynomial time solvable in chordal graphs (Golumbic, 1980), but it is NP-complete for subclasses of 2-thin graphs (B., Brandwein, Oliveira, Sampaio, Sansone and Szwarcfiter, 2023).
- The list-coloring problem is NP-complete for interval graphs (Biro, Hujter and Tuza, 1992).
- The capacitated coloring problem is NP-complete for interval graphs (Bodlaender and Jansen, 1995).

When the number of output sets is not fixed

- The coloring problem is polynomial time solvable in chordal graphs (Golumbic, 1980), but it is NP-complete for subclasses of 2-thin graphs (B., Brandwein, Oliveira, Sampaio, Sansone and Szwarcfiter, 2023).
- The list-coloring problem is NP-complete for interval graphs (Biro, Hujter and Tuza, 1992).
- The capacitated coloring problem is NP-complete for interval graphs (Bodlaender and Jansen, 1995).
- The coloring problem with lists and capacities (De Werra, 1997), is NP-complete on paths (Dror, Finke, Gravier and Kubiak, 1999), even for lists of size at most 2, and in star forests (Bentz, 2019).

When the number of output sets is not fixed

- The coloring problem is polynomial time solvable in chordal graphs (Golumbic, 1980), but it is NP-complete for subclasses of 2-thin graphs (B., Brandwein, Oliveira, Sampaio, Sansone and Szwarcfiter, 2023).
- The list-coloring problem is NP-complete for interval graphs (Biro, Hujter and Tuza, 1992).
- The capacitated coloring problem is NP-complete for interval graphs (Bodlaender and Jansen, 1995).
- The coloring problem with lists and capacities (De Werra, 1997), is NP-complete on paths (Dror, Finke, Gravier and Kubiak, 1999), even for lists of size at most 2, and in star forests (Bentz, 2019).
- The coloring problem with weights and capacities is NP-complete in edgeless graphs, even if $p = 1$, b_1 is polynomially bounded, and each vertex can take any color (Bentz, 2019).

Outline

Thinness of chordal graphs

- Recall that interval graphs are equivalent to graphs of thinness 1.

Thinness of chordal graphs

- Recall that interval graphs are equivalent to graphs of thinness 1.
- Trees can have logarithmic thinness:

$$\text{thin}(T) \leq \log_3(n+2)$$

and the thinness of the complete binary tree of height h is $\lceil \frac{h+1}{3} \rceil$.

Moreover, the thinness of a tree can be computed in $O(n \log n)$ time (Brandwein and Sansone, 2021).

Thinness of chordal graphs

- Recall that interval graphs are equivalent to graphs of thinness 1.
- Trees can have logarithmic thinness:

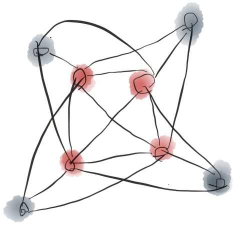
$$\text{thin}(T) \leq \log_3(n+2)$$

and the thinness of the complete binary tree of height h is $\lceil \frac{h+1}{3} \rceil$.

Moreover, the thinness of a tree can be computed in $O(n \log n)$ time (Brandwein and Sansone, 2021).

- Split graphs can have linear thinness, $\text{thin}(\overline{K_n \boxminus S_n}) \geq \frac{n}{2}$ (B., Gonzalez, Oliveira, Sampaio and Szwarcfiter, 2020).

$\overline{K_4 \boxminus S_4}$

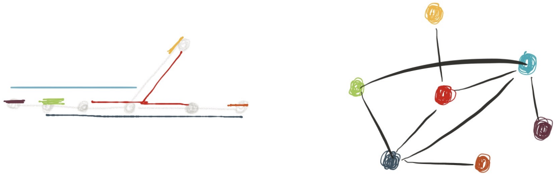


Leafage of a chordal graph

A graph is chordal if and only if it is the intersection graph of subtrees of a tree (Gavril, 1974), called *host tree*.

Leafage of a chordal graph

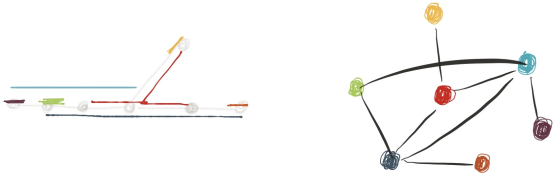
A graph is chordal if and only if it is the intersection graph of subtrees of a tree (Gavril, 1974), called *host tree*.



The *leafage* $\ell(G)$ of a chordal graph G (Lin, McKee and West, 1998), is the minimum number of leaves of a host tree.

Leafage of a chordal graph

A graph is chordal if and only if it is the intersection graph of subtrees of a tree (Gavril, 1974), called *host tree*.



The *leafage* $\ell(G)$ of a chordal graph G (Lin, McKee and West, 1998), is the minimum number of leaves of a host tree.

chordal graphs of leafage 2 = interval graphs

The leafage of a graph G and an optimal representation can be computed in $O(n^3)$ time (Habib and Stacho, 2009).

Bound of thinness in terms of leafage

Theorem (B., Brettell, Munaro and Paulusma, 2022)

The thinness of a chordal graph can be bounded in terms of its leafage:

$$thin(G) \leq \max\{1, \ell(G) - 1\}.$$

Moreover, an $(\ell(G) - 1)$-thin representation can be efficiently computed.

Bound of thinness in terms of leafage

Theorem (B., Brettell, Munaro and Paulusma, 2022)

The thinness of a chordal graph can be bounded in terms of its leafage:

$$thin(G) \leq \max\{1, \ell(G) - 1\}.$$

Moreover, an $(\ell(G) - 1)$-thin representation can be efficiently computed.

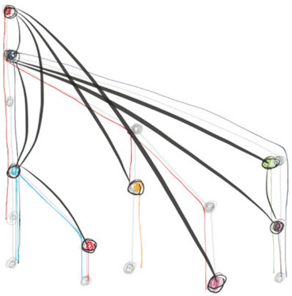
Bound of thinness in terms of leafage

Theorem (B., Brettell, Munaro and Paulusma, 2022)

The thinness of a chordal graph can be bounded in terms of its leafage:

$$thin(G) \leq \max\{1, \ell(G) - 1\}.$$

Moreover, an $(\ell(G) - 1)$-thin representation can be efficiently computed.



Algorithmic consequences of the bound

Corollary

GENERALIZED LIST MATRIX PARTITION PROBLEMS is in XP for chordal graphs parameterized by leafage, thus polynomially solvable for chordal graphs with bounded leafage.

Algorithmic consequences of the bound

Corollary

GENERALIZED LIST MATRIX PARTITION PROBLEMS is in XP for chordal graphs parameterized by leafage, thus polynomially solvable for chordal graphs with bounded leafage.

Recall that

Theorem (Feder, Hell, Klein, Nogueira and Protti, 2005)

There are matrices M such that the M-PARTITION problem is NP-complete for chordal graphs.

(p, q, r)-GLMP on co-comparability graphs

Theorem (B., Mattia and Oriolo, 2011)

The thinness of a co-comparability graph G can be bounded in terms of its chromatic number:

$$thin(G) \leq \chi(G).$$

Moreover, a $\chi(G)$-thin representation can be efficiently computed.

(p, q, r)-GLMP on co-comparability graphs

Theorem (B., Mattia and Oriolo, 2011)

The thinness of a co-comparability graph G can be bounded in terms of its chromatic number:

$$thin(G) \leq \chi(G).$$

Moreover, a $\chi(G)$-thin representation can be efficiently computed.

The bound does not hold for general graphs, not even for comparability graphs, since bipartite graphs can have arbitrarily large thinness (Chandran, Mannino and Oriolo, 2007). Indeed, List 3-Coloring is NP-complete for bipartite graphs (Kubale, 1992).

Algorithmic consequences of the bound

Corollary

If all diagonal entries of the input $r \times r$ matrix M are zero, and no vertex has the empty combination in its list, then the co-comparability (p, q, r)-GLMP problem can be solved in polynomial time.

Algorithmic consequences of the bound

Corollary

If all diagonal entries of the input $r \times r$ matrix M are zero, and no vertex has the empty combination in its list, then the co-comparability (p, q, r)-GLMP problem can be solved in polynomial time.

Idea: A solution is, in particular, an r-coloring. Co-comparability graphs are either not r-colorable or have thinness at most r, and an r-thin representation can be efficiently computed. Then we use the algorithm for graphs of bounded thinness.

Algorithmic consequences of the bound

Corollary

If all diagonal entries of the input $r \times r$ matrix M are zero, and no vertex has the empty combination in its list, then the co-comparability (p, q, r)-GLMP problem can be solved in polynomial time.

Idea: A solution is, in particular, an r-coloring. Co-comparability graphs are either not r-colorable or have thinness at most r, and an r-thin representation can be efficiently computed. Then we use the algorithm for graphs of bounded thinness.

There are matrices M such that List M-Partition is NP-complete for co-comparability graphs. For instance, when $M = \begin{pmatrix} 1 & * & * \\ * & 1 & * \\ * & * & 1 \end{pmatrix}$, co-comparability List M-Partition is equivalent to comparability List 3-Coloring, which is NP-complete (Kubale, 1992).

Muchas gracias!!