Explicit Modular Decomposition

Marc Hellmuth

Department of Mathematics
Faculty of Science
Stockholm University

WU Wien, 2024

Research

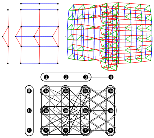
Research

Discrete Mathematics

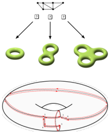
Research

Discrete Mathematics

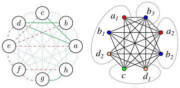
- Graph and Hypergraph Theory
- Discrete Optimization and Matroid Theory
- Combinatorics and Geometric Graph Theory



Products of Graphs and Hypergraphs



Planarity and 1-Face Embeddings (= particular planar drawing on orientable manifolds)



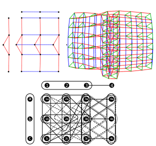
Packings, Edge- and Vertex-Colorings

Research

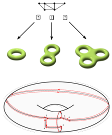
Discrete Mathematics

- Graph and Hypergraph Theory
- Discrete Optimization and Matroid Theory
- Combinatorics and Geometric Graph Theory

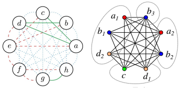
Computer Science



Products of Graphs and Hypergraphs



Planarity and 1-Face Embeddings
(= particular planar drawing
on orientable manifolds)



Packings, Edge- and Vertex-Colorings

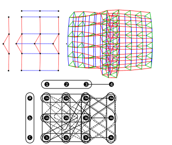
Research

Discrete Mathematics

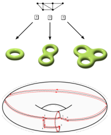
- Graph and Hypergraph Theory
- Discrete Optimization and Matroid Theory
- Combinatorics and Geometric Graph Theory

Computer Science

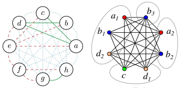
- Complexity Theory
  (Co)NP-completeness, Fixed-Parameter Tractability (FPT)
- Algorithm Design
  Exact algorithms, Heuristics, Integer Linear Programming (ILP), Approximation algorithms



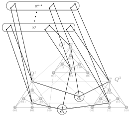
Products of Graphs and Hypergraphs



Planarity and 1-Face Embeddings
(= particular planar drawing
on orientable manifolds)



Packings, Edge- and Vertex-Colorings



Reductions, NP-hardness,
ILP, FPT, approx. algorithms

Research

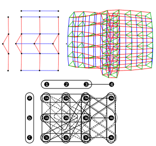
Discrete Mathematics

- Graph and Hypergraph Theory
- Discrete Optimization and Matroid Theory
- Combinatorics and Geometric Graph Theory

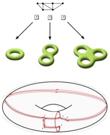
Computer Science

- Complexity Theory
  (Co)NP-completeness, Fixed-Parameter Tractability (FPT)
- Algorithm Design
  Exact algorithms, Heuristics, Integer Linear Programming (ILP), Approximation algorithms

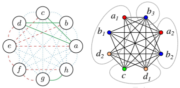
Applied Mathematics & Math. Data Science



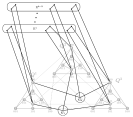
Products of Graphs and Hypergraphs



Planarity and 1-Face Embeddings (= particular planar drawing on orientable manifolds)



Packings, Edge- and Vertex-Colorings



Reductions, NP-hardness, ILP, FPT, approx. algorithms

Research

Discrete Mathematics

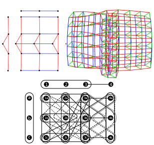
- Graph and Hypergraph Theory
- Discrete Optimization and Matroid Theory
- Combinatorics and Geometric Graph Theory

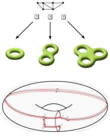
Computer Science

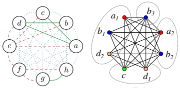
- Complexity Theory
  (Co)NP-completeness, Fixed-Parameter Tractability (FPT)
- Algorithm Design
  Exact algorithms, Heuristics, Integer Linear Programming (ILP), Approximation algorithms

Applied Mathematics & Math. Data Science

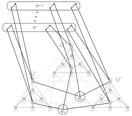
- General Focus: mathematical characterization of discrete data and structures that naturally occur in Life Sciences or Engineering

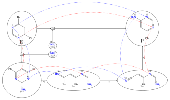

Products of Graphs and Hypergraphs

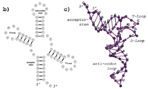

Planarity and 1-Face Embeddings
(= particular planar drawing
on orientable manifolds)



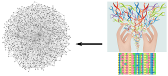
Packings, Edge- and Vertex-Colorings


Reductions, NP-hardness,
ILP, FPT, approx. algorithms


Graph Grammars, Cayley Graphs,
Chemical Reaction Networks

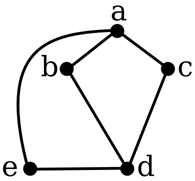


Combinatorics of RNA structures


Genomics and Evolutionary Biology

Basics: Graphs and their Representation

undirected, simple graph G



Aim:

find efficient representation

that provides deep structural insights

and helps to understand complexity of certain problems

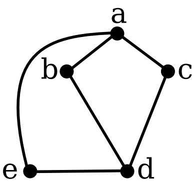
Basics: Graphs and their Representation

Representation

$$A(G) = \begin{array}{c} \begin{array}{ccccc} \mathbf{a} & \mathbf{b} & \mathbf{c} & \mathbf{d} & \mathbf{e} \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \begin{array}{c} \\ \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \\ \mathbf{e} \end{array}$$

$$A_{\text{list}}(G) = \begin{array}{c|ccc} a & b & c & e \\ b & a & d & \\ c & a & d & \\ d & b & c & e \\ e & a & d & \end{array}$$

undirected, simple graph G



Aim:

find efficient representation

that provides deep structural insights

and helps to understand complexity of certain problems

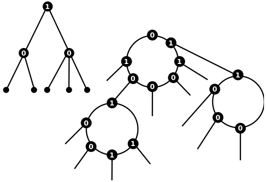
Basics: Graphs and their Representation

Representation

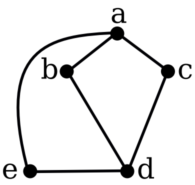
$$A(G) = \begin{array}{c} \\ \\ \end{array}\begin{array}{cc} \begin{matrix} \mathbf{a} & \mathbf{b} & \mathbf{c} & \mathbf{d} & \mathbf{e} \end{matrix} & \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} & \begin{matrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ \mathbf{d} \\ \mathbf{e} \end{matrix} \end{array}$$

$$A_{\text{list}}(G) = \begin{array}{c|ccc} a & b & c & e \\ b & a & d & \\ c & a & d & \\ d & b & c & e \\ e & a & d & \end{array}$$

Use other discrete structures as a way to present graphs



undirected, simple graph G



Aim:

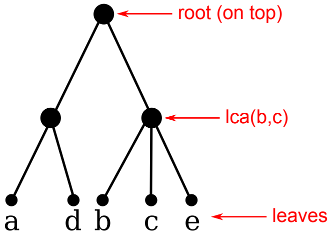
find efficient representation

that provides deep structural insights

and helps to understand complexity of certain problems

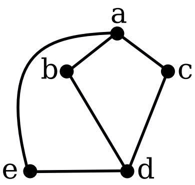
Basics: Graphs and their Representation

Representation via (T, t)



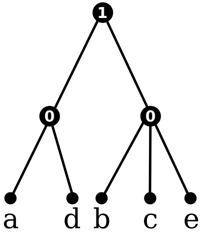
lca = last common ancestor

undirected, simple graph G



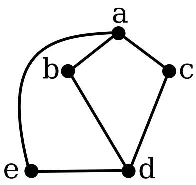
Basics: Graphs and their Representation

Representation via (T,t)



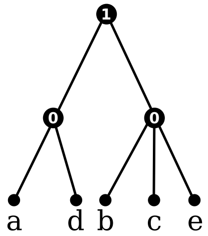
lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G



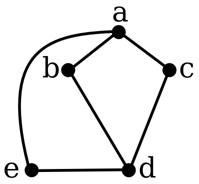
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

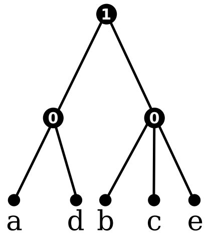


In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

G is explained by (T,t): $t(\mathrm{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$

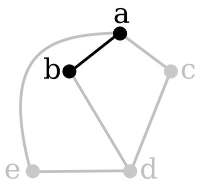
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

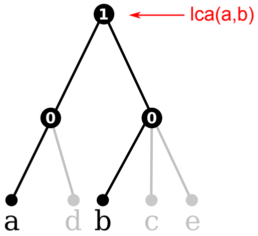


In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

G is explained by (T,t): $t(\mathrm{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$

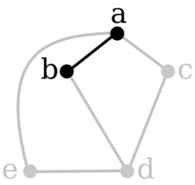
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

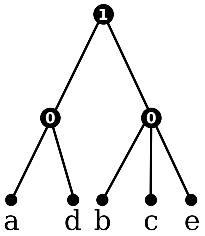


In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

$$G \text{ is explained by } (T,t)\text{: } t(\mathrm{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$$

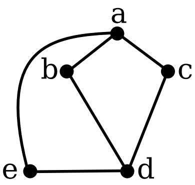
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

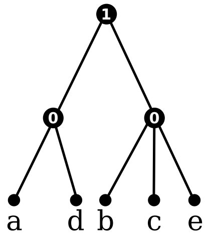


In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

G is explained by (T,t): $t(\mathsf{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$

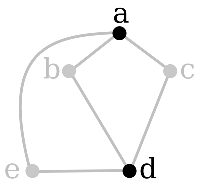
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

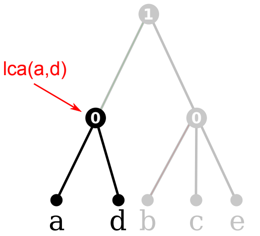


In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

G is explained by (T,t): $t(\text{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$

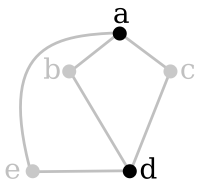
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

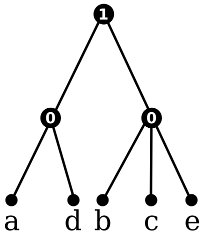


In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

G is explained by (T,t): $t(\mathrm{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$

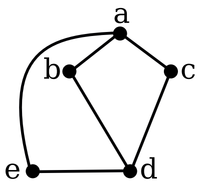
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

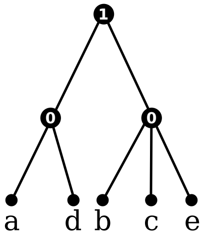


In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

$$G \text{ is explained by } (T,t)\text{: } t(\mathrm{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$$

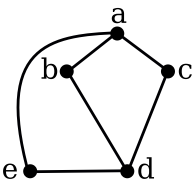
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G



In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

G is explained by (T,t): $t(\mathrm{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$

The subclass of graphs that can be explained by (T,t) are precisely the **cographs**

Basics: Graphs and their Representation

Cographs ?

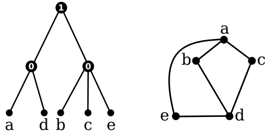
In this example, (T,t) contains all structural information of G and can be used to recover G, i.e.,

G is explained by (T,t): $t(\mathsf{lca}_T(x,y)) = 1 \iff \{x,y\} \in E$

The subclass of graphs that can be explained by (T,t) are precisely the **cographs**

Cographs

Cographs ...



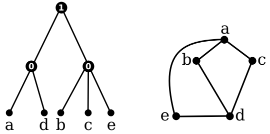
Cotree (T,t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

Cographs

Cographs . . .

- . . . form an extremely well-known graph class



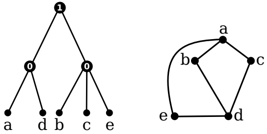
Cotree (T, t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

Cographs

Cographs ...

- ... form an extremely well-known graph class
- ... are recursively defined *(omitted)*



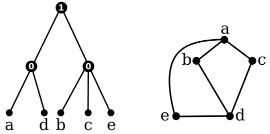
Cotree (T, t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: • – • – • – •



Cotree (T, t) explains cograph G

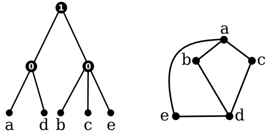
---

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain
  induced paths on 4 vertices: ● – ● – ● – ●



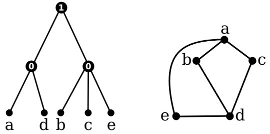
Cotree (T, t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

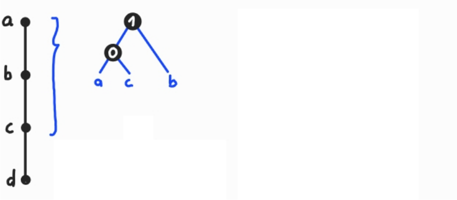
Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: • – • – • – •



Cotree (T, t) explains cograph G

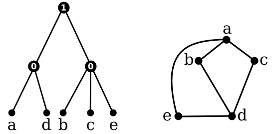


discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

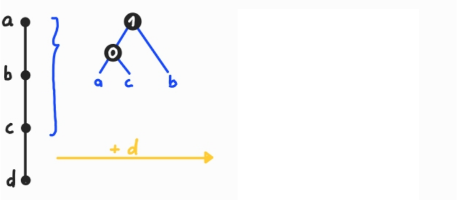
Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: • – • – • – •



Cotree (T, t) explains cograph G

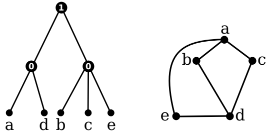


discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

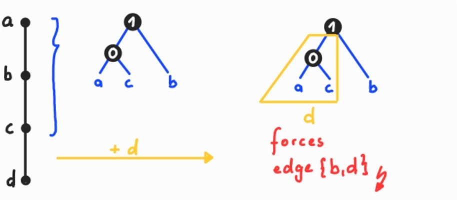
Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: • – • – • – •



Cotree (T, t) explains cograph G

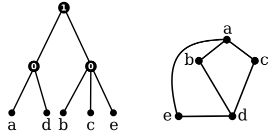


discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

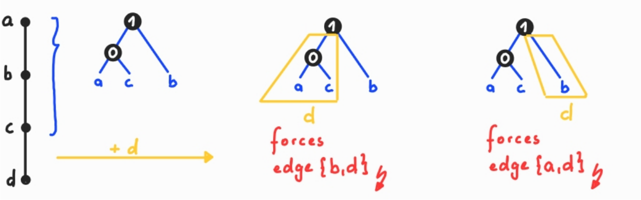
Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: • – • – • – •



Cotree (T, t) explains cograph G

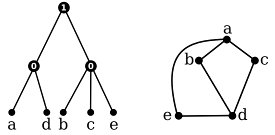


discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

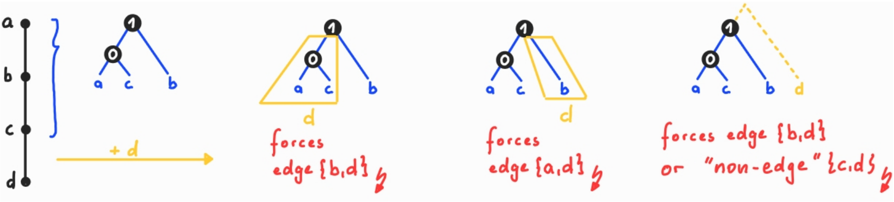
Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: • – • – • – •



Cotree (T, t) explains cograph G

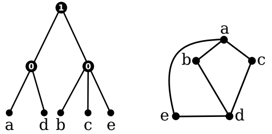


discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

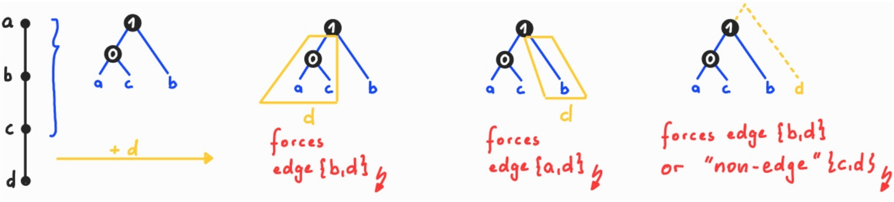
Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: • – • – • – •
- . . . have appealing properties (GI - easy, many NP-hard problems become polynomial-time solvable)



Cotree (T, t) explains cograph G

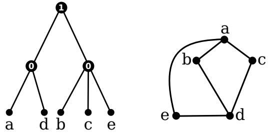


discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

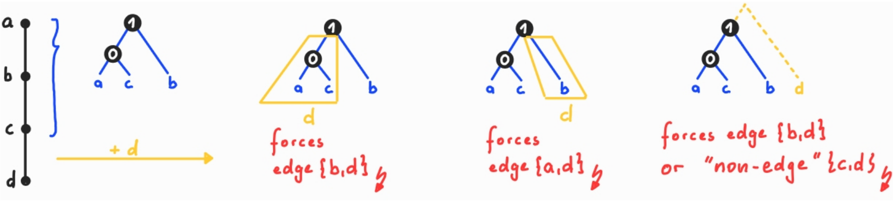
Cographs

Cographs . . .

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are presicely the graphs that do not contain induced paths on 4 vertices: ● – ● – ● – ●
- . . . have appealing properties (GI - easy, many NP-hard problems become polynomial-time solvable)
- . . . data-storage $O(|V(G)|)$



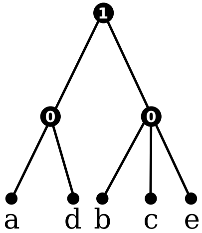
Cotree (T,t) explains cograph G



discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

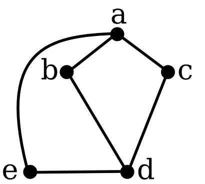
Basics: Graphs and their Representation

Representation via (T,t)



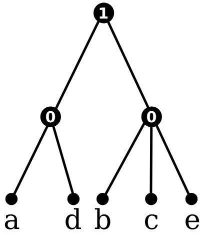
lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G



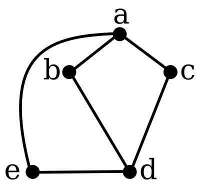
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

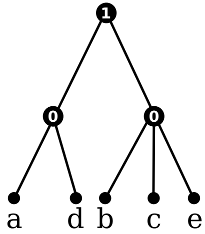
undirected, simple graph G



BAD news: Not all graphs can be explained by such 0/1-labeled trees (T,t)

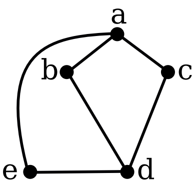
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G

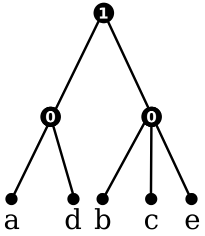


BAD news: Not all graphs can be explained by such 0/1-labeled trees (T,t)

The subclass of graphs that can be explained by (T,t) are *precisely* the **cographs** (= extremely well-studied graph class).

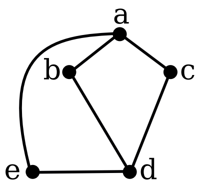
Basics: Graphs and their Representation

Representation via (T,t)



lca = last common ancestor
rooted tree T with 0/1-labeling t

undirected, simple graph G



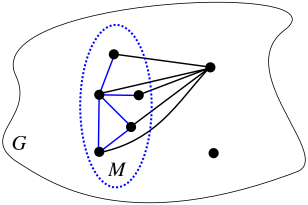
BAD news: Not all graphs can be explained by such 0/1-labeled trees (T,t)

The subclass of graphs that can be explained by (T,t) are *precisely* the **cographs** (= extremely well-studied graph class).

GOOD news: Every graph G has a unique **Modular Decomposition Tree** that at least to some extent provides structural information of G

Basics: Modular Decomposition (MD)*

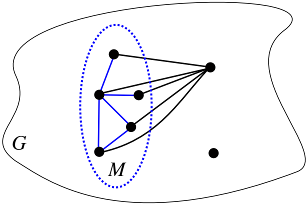
$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



*MD is a standard technique to understand discrete structure by decomposing them into smaller "building blocks".
MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)*

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



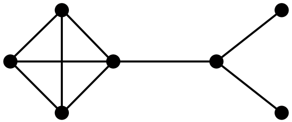
Every graph has a **unique decomposition into non-overlapping modules** that can be computed in *linear-time*.

MD = set of *all* non-overlapping modules

*MD is a standard technique to understand discrete structure by decomposing them into smaller "building blocks".
MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)*

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



Every graph has a **unique decomposition into non-overlapping modules** that can be computed in *linear-time*.

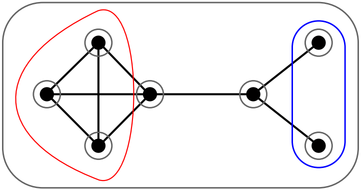
MD = set of *all* non-overlapping modules

*MD is a standard technique to understand discrete structure by decomposing them into smaller “building blocks”.

MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)*

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



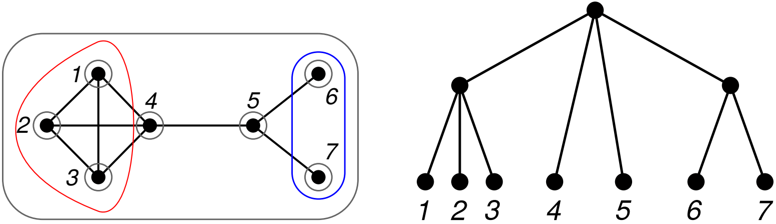
Every graph has a **unique decomposition into non-overlapping modules** that can be computed in *linear-time*.

MD = set of *all* non-overlapping modules

*MD is a standard technique to understand discrete structure by decomposing them into smaller "building blocks".
MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)*

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



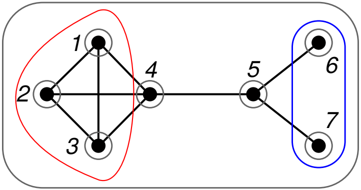
Every graph has a **unique decomposition into non-overlapping modules** that can be computed in *linear-time*.

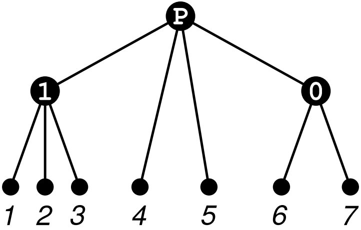
MD = set of *all* non-overlapping modules = hierarchy = rooted tree

*MD is a standard technique to understand discrete structure by decomposing them into smaller “building blocks”.
MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)*

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$





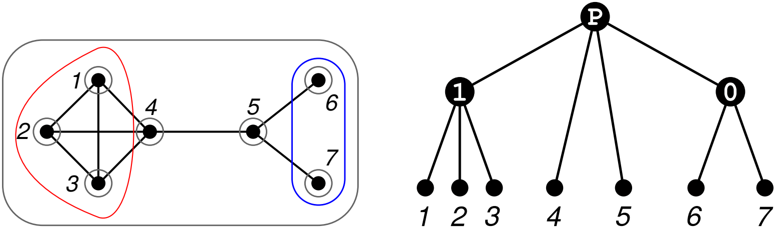
Every graph has a **unique decomposition into non-overlapping modules** that can be computed in *linear-time*.

MD = set of *all* non-overlapping modules = hierarchy = rooted tree

*MD is a standard technique to understand discrete structure by decomposing them into smaller "building blocks".
MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)*

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



Every graph has a **unique decomposition into non-overlapping modules** that can be computed in *linear-time*.

MD = set of *all* non-overlapping modules = hierarchy = rooted tree

There are different type of modules:

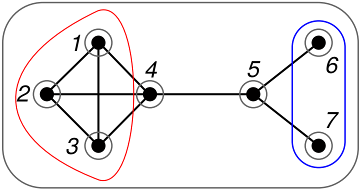
> 0, 1 and P (Prime) modules
>
> *defined in terms of connectedness conditions (omitted)*.

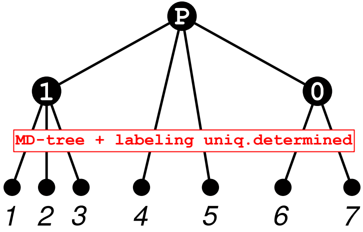
---

*MD is a standard technique to understand discrete structure by decomposing them into smaller “building blocks”.
MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)*

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$





Every graph has a **unique decomposition into non-overlapping modules** that can be computed in *linear-time*.

MD = set of *all* non-overlapping modules = hierarchy = rooted tree

There are different type of modules:

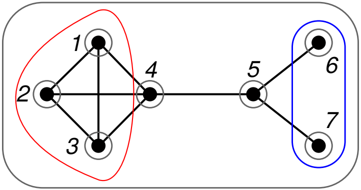
0, 1 and P (Prime) modules

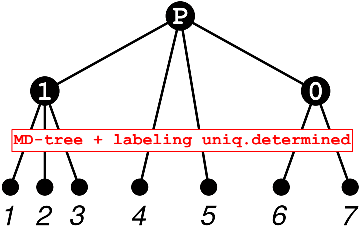
defined in terms of connectedness conditions (omitted).

*MD is a standard technique to understand discrete structure by decomposing them into smaller "building blocks".
MD is of such basic importance that it was rediscovered several times under several names, e.g., Gallai (1967), Habib&Maurer (1979), Möhring&Rademacher (1979), Sumner (1971)

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



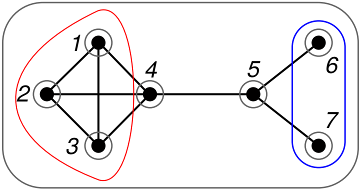


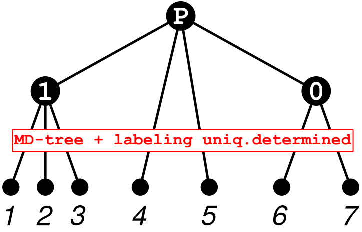
0- and 1-vertices in MD-tree are good:

$$t(\mathrm{lca}(x,y)) = 0 \implies \{x,y\} \notin E(G) \quad \text{and} \quad t(\mathrm{lca}(x,y)) = 1 \implies \{x,y\} \in E(G)$$

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$





0- and 1-vertices in MD-tree are good:

$$t(\text{lca}(x,y)) = 0 \implies \{x,y\} \notin E(G) \quad \text{and} \quad t(\text{lca}(x,y)) = 1 \implies \{x,y\} \in E(G)$$

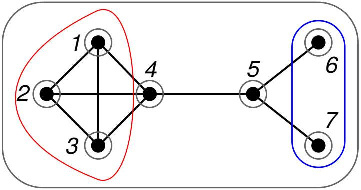
BUT, not all structural information of G is provided by the MD-tree if it contains P-vertices.

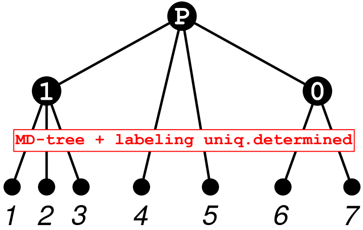
$t(\text{lca}(3,4)) = P$ and $\{3,4\} \in E(G)$

$t(\text{lca}(3,5)) = P$ and $\{3,5\} \notin E(G)$

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x,y \in M$





0- and 1-vertices in MD-tree are good:

$$t(\mathrm{lca}(x,y)) = 0 \implies \{x,y\} \notin E(G) \quad \text{and} \quad t(\mathrm{lca}(x,y)) = 1 \implies \{x,y\} \in E(G)$$

BUT, not all structural information of G is provided by the MD-tree if it contains P-vertices.

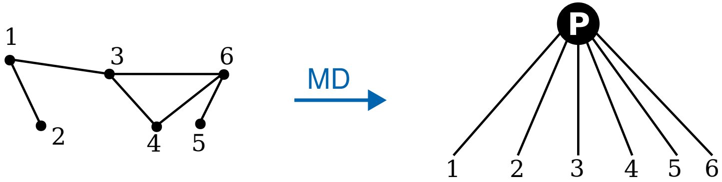
$t(\mathrm{lca}(3,4)) = P$ and $\{3,4\} \in E(G)$

$t(\mathrm{lca}(3,5)) = P$ and $\{3,5\} \notin E(G)$

Full information of G is provided only if MD-tree does not contain P-vertices (**cographs**)

Basics: Modular Decomposition (MD)

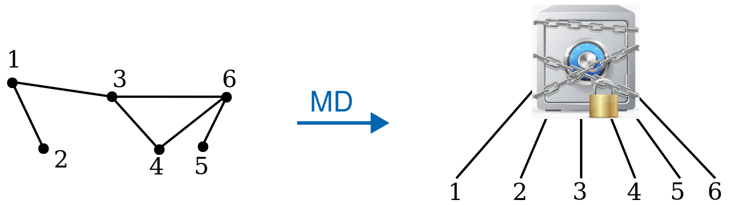
$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



The MD-tree cannot be used to recover G (structural information gets lost on P-vertices)

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$

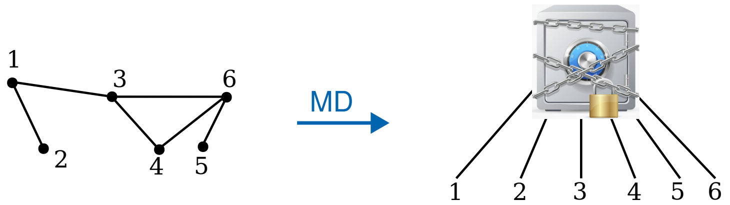


The MD-tree cannot be used to recover G (structural information gets lost on P-vertices)

P-vertices $\widehat{=}$ extremely secured closed box hiding structural information.

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



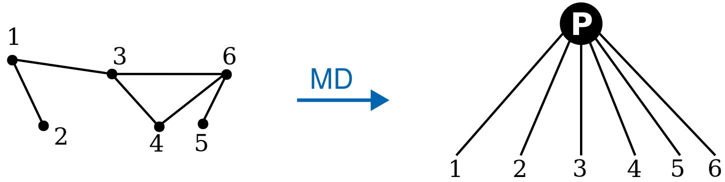
The MD-tree cannot be used to recover G (structural information gets lost on P-vertices)

P-vertices $\widehat{=}$ extremely secured closed box hiding structural information.

Can we break and unbox the P-vertices to obtain 0/1-labeled rooted networks that provides the missing structural information of G ?

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



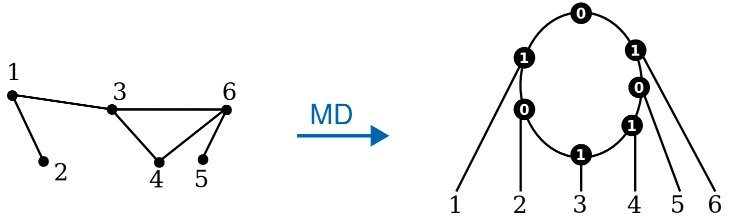
The MD-tree cannot be used to recover G (structural information gets lost on P-vertices)

P-vertices $\widehat{=}$ extremely secured closed box hiding structural information.

Can we break and unbox the P-vertices to obtain 0/1-labeled rooted networks that provides the missing structural information of G ?

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



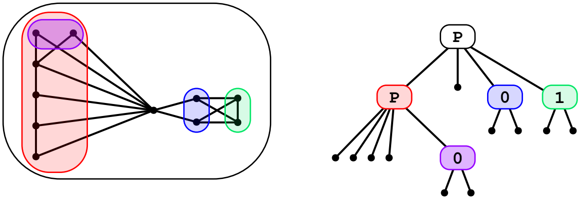
The MD-tree cannot be used to recover G (structural information gets lost on P-vertices)

P-vertices $\widehat{=}$ extremely secured closed box hiding structural information.

Can we break and unbox the P-vertices to obtain 0/1-labeled rooted networks that provides the missing structural information of G ?

$\Longrightarrow$ **Explicit Modular Decomposition**

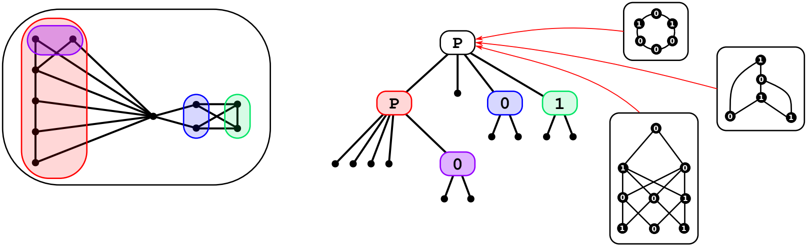
Explicit Modular Decomposition



General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

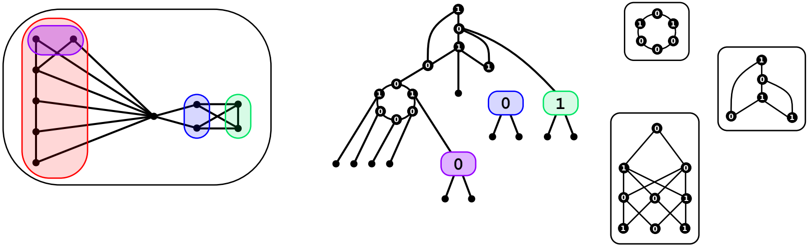
Explicit Modular Decomposition



General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

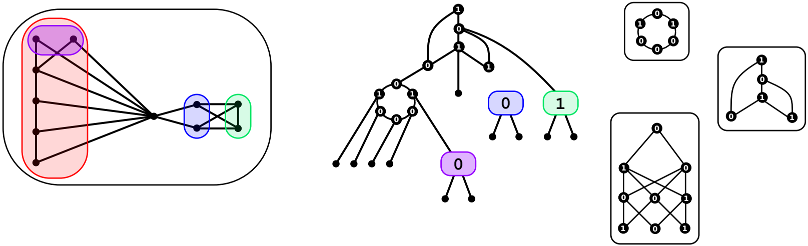
Explicit Modular Decomposition



General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

Explicit Modular Decomposition

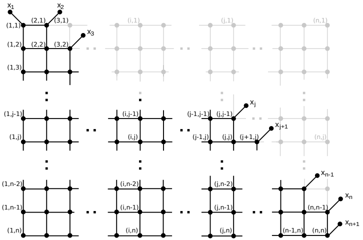


General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

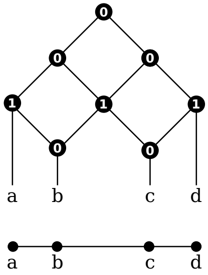
Proof of Concept: Half-grid networks.

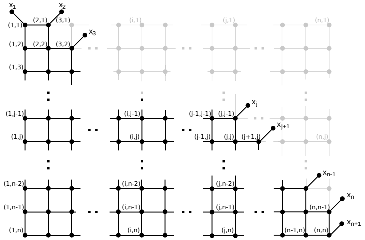
Explicit Modular Decomposition: Half-grids



Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

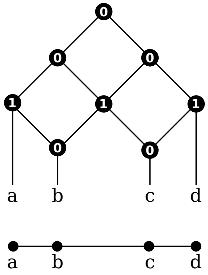
Explicit Modular Decomposition: Half-grids

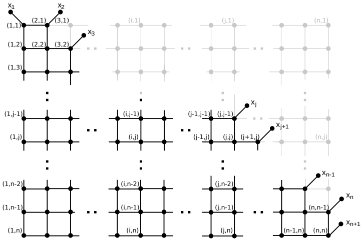




Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids





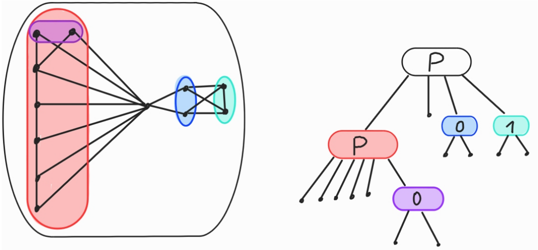
Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from MDT by locally replacing "P"-vertices by half-grids = median graphs.

Proof: In half-grids we have $\text{lca}(x,y) \neq \text{lca}(x',y')$ for distinct pairs x,y and x',y'.

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids



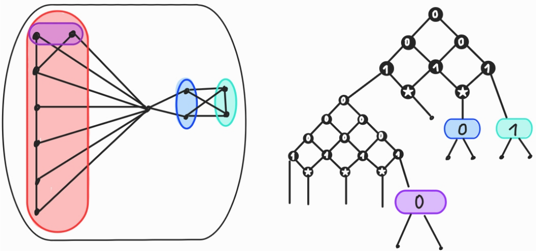
Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from MDT by locally replacing "P"-vertices by half-grids = median graphs.

Proof: In half-grids we have $\mathrm{lca}(x,y) \neq \mathrm{lca}(x',y')$ for distinct pairs x,y and x',y'.

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids



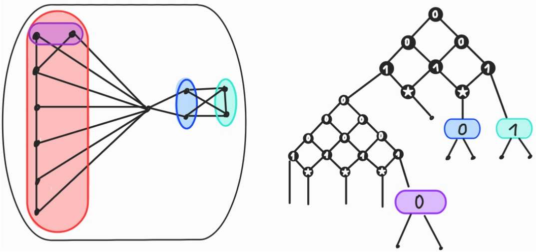
Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from MDT by locally replacing "P"-vertices by half-grids = median graphs.

Proof: In half-grids we have $\mathrm{lca}(x,y) \neq \mathrm{lca}(x',y')$ for distinct pairs x,y and x',y'.

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids



Theorem (2021)

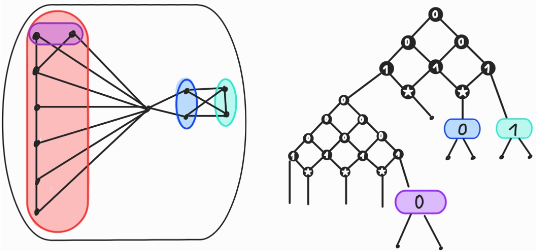
Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from MDT by locally replacing "P"-vertices by half-grids = median graphs.

Proof: In half-grids we have $\mathrm{lca}(x,y) \neq \mathrm{lca}(x',y')$ for distinct pairs x,y and x',y'.

$\Longrightarrow$ The idea of explicit modular decomposition is feasible!

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids



Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from MDT by locally replacing "P"-vertices by half-grids = median graphs.

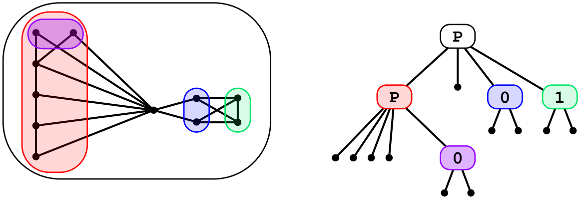
Proof: In half-grids we have $\mathrm{lca}(x,y) \neq \mathrm{lca}(x',y')$ for distinct pairs x,y and x',y'.

$\Longrightarrow$ The idea of explicit modular decomposition is feasible!

But half-grids are only as suitable as representing G with its adjacency matrix (no deep structural insights)

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

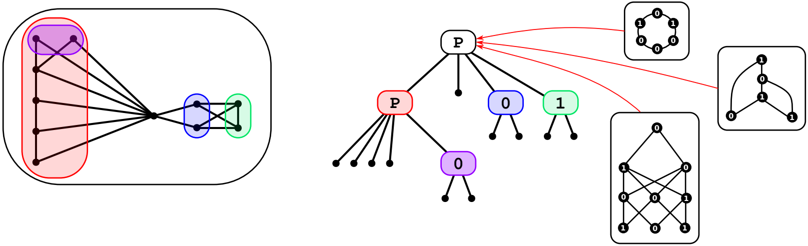
Explicit Modular Decomposition



General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

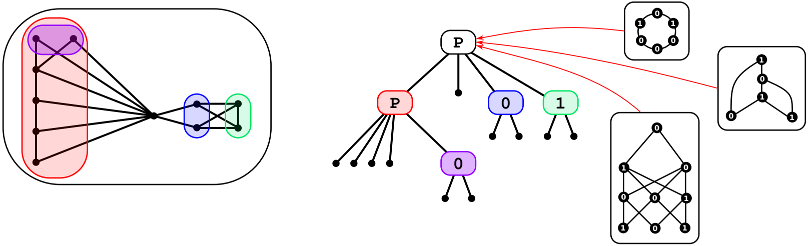
Explicit Modular Decomposition



General Aim:

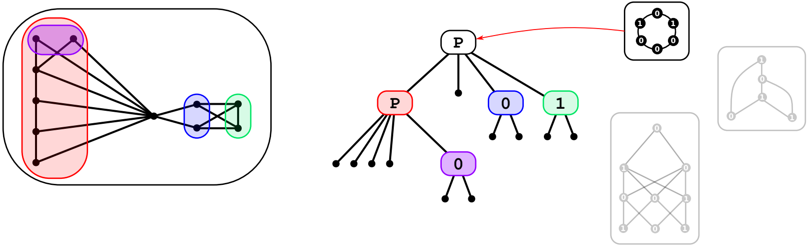
- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

Explicit Modular Decomposition



Instead of using ”dense” half-grids let’s look at the other extreme:
Use the most-simplest non-tree structure to replace P-vertices.

Explicit Modular Decomposition

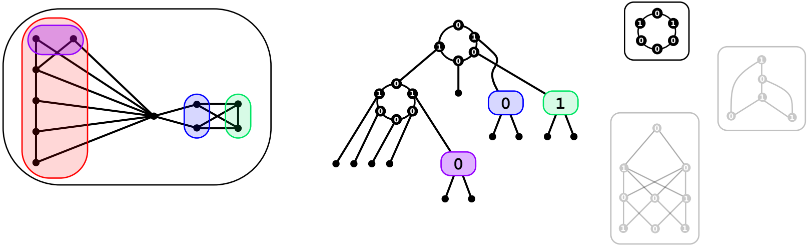


Instead of using ”dense” half-grids let’s look at the other extreme:
Use the most-simplest non-tree structure to replace P-vertices.

From here one, we focus on the special case:

replacing P-vertices by rooted 0/1-labeled cycles

Explicit Modular Decomposition



Instead of using ”dense” half-grids let’s look at the other extreme:
Use the most-simplest non-tree structure to replace P-vertices.

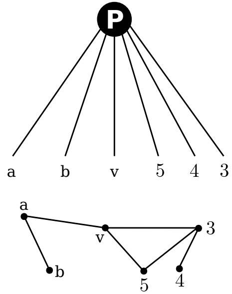
From here one, we focus on the special case:

replacing P-vertices by rooted 0/1-labeled cycles

Question: Which type of graphs can be explained by such networks (N, t)?

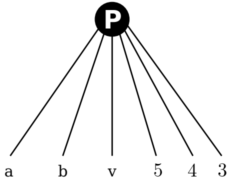
Simple case: Single P-vertex

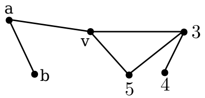
Consider graphs G whose MD-tree is a star-tree with single P-vertex



Simple case: Single P-vertex

Consider graphs G whose MD-tree is a star-tree with single P-vertex



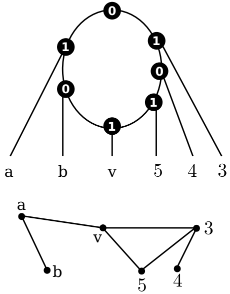


1st Task: Characterize graphs G where

- MD-tree is star with single P-vertex and
- the resulting network (N, t) explains G?

Simple case: Single P-vertex

Consider graphs G whose MD-tree is a star-tree with single P-vertex



1st Task: Characterize graphs G where

- MD-tree is star with single P-vertex and
- the resulting network (N,t) explains G?

Simple case: Single P-vertex and Pseudo-cographs

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

---

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$,
$V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

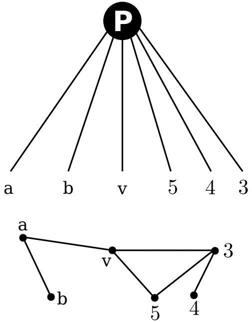
Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$,
$V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.



Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

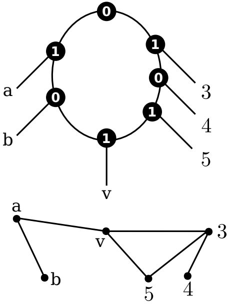
Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$, $V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.



Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

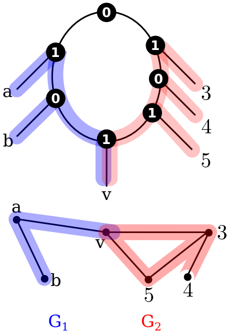
Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$, $V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.



Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

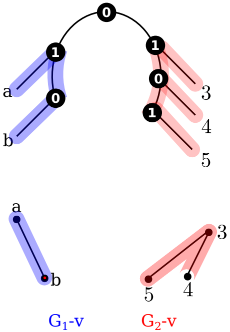
Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$,
$V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.



Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

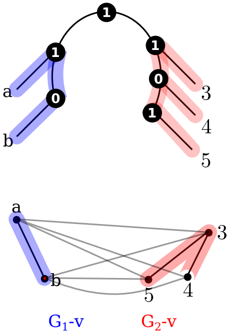
Simple case: Single P-vertex and Pseudo-cographs

A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$,
$V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.



Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

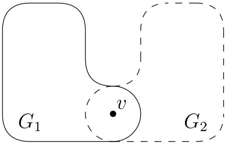
Simple case: Single P-vertex and Pseudo-cographs

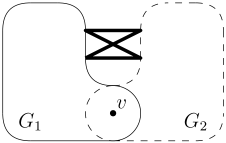
A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$, $V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.





Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

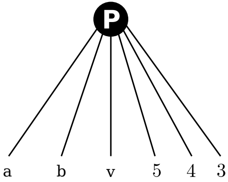
Simple case: Single P-vertex and Pseudo-cographs

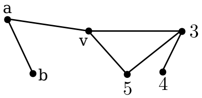
A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$,
$V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.





1st Task: Characterize graphs G where

- MD-tree is star with single P-vertex and
- the resulting network (N, t) explains G?

Solution: Pseudo-cographs

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

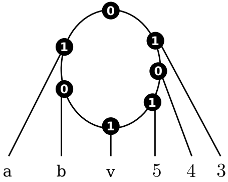
Simple case: Single P-vertex and Pseudo-cographs

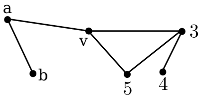
A graph G is a pseudo-cograph if $|V(G)| \leq 2$ or if there is a vertex $v \in V(G)$ and induced subgraphs $G_1, G_2 \subseteq G$, both with at least two vertices such that

(P1) $V(G_1) \cup V(G_2) = V(G)$,
$V(G_1) \cap V(G_2) = \{v\}$;

(P2) G_1 and G_2 are cographs;

(P3) $G - v$ is either the join or the disjoint union of $G_1 - v$ and $G_2 - v$.





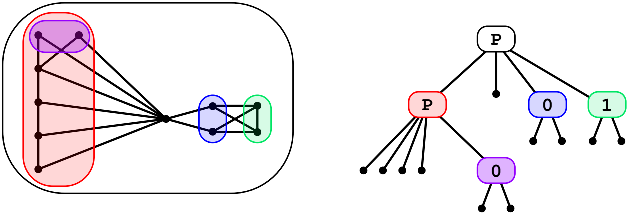
1st Task: Characterize graphs G where

- MD-tree is star with single P-vertex and
- the resulting network (N, t) explains G?

Solution: Pseudo-cographs

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

The general case: GaTEx graphs



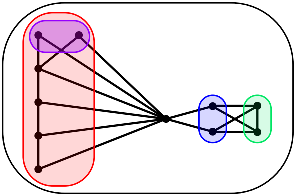
General Aim: Characterize graphs G where

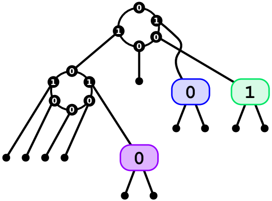
- MD-tree is *any tree* with *several* P-vertices and
- the resulting network (N,t) explains G?

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GaTEx graphs





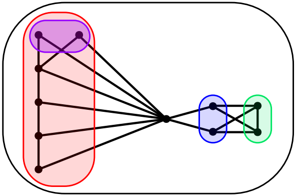
General Aim: Characterize graphs G where

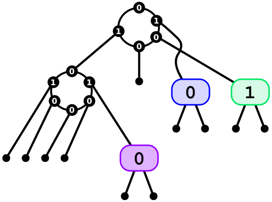
- MD-tree is *any tree* with *several* P-vertices and
- the resulting network (N,t) explains G?

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GaTEx graphs





General Aim: Characterize graphs G where

- MD-tree is *any tree* with *several* P-vertices and
- the resulting network (N,t) explains G?

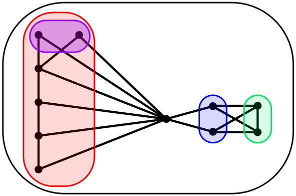
Graphs for which this approach works are called

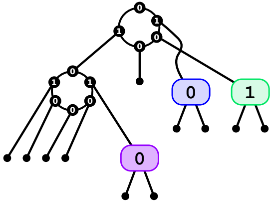
GaTEx $:=$ **Ga**lled-**T**ree **Ex**plainable (*graphs that can be explained by such networks*)

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GATEX graphs





Galled-tree := 0/1-labeled rooted network where all cycles are edge-disjoint

General Aim: Characterize graphs G where

- MD-tree is *any tree* with *several* P-vertices and
- the resulting network (N,t) explains G?

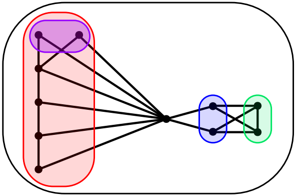
Graphs for which this approach works are called

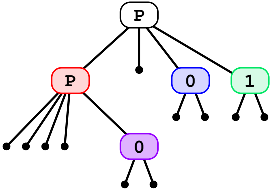
GATEX := **Ga**lled-**T**ree **Ex**plainable (*graphs that can be explained by such networks*)

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GATEX graphs





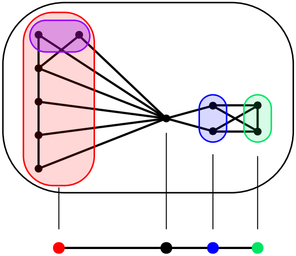
Theorem (2022, 2023)

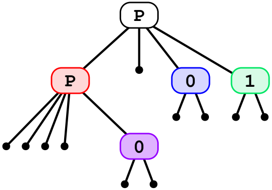
A graph is GATEX $\iff$ *for all P-modules M the "quotient of $G[M]$" is a pseudo-cograph.*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GATEX graphs





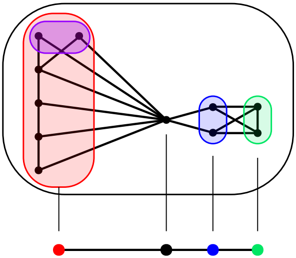
Theorem (2022, 2023)

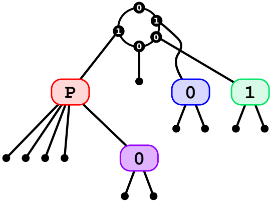
A graph is GATEX $\iff$ *for all* P*-modules* M *the "quotient of* $G[M]$*" is a pseudo-cograph.*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GaTEx graphs





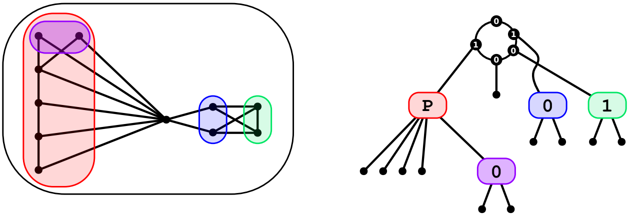
Theorem (2022, 2023)

A graph is GaTEx $\iff$ *for all P-modules M the "quotient of $G[M]$" is a pseudo-cograph.*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GATEX graphs



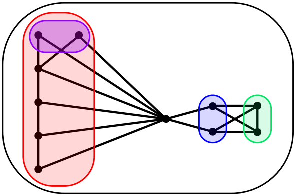
Theorem (2022, 2023)

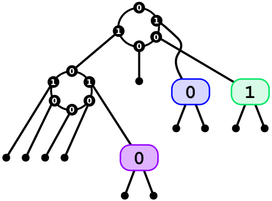
A graph is GATEX $\iff$ *for all P-modules M the "quotient of $G[M]$" is a pseudo-cograph.*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GATEX graphs





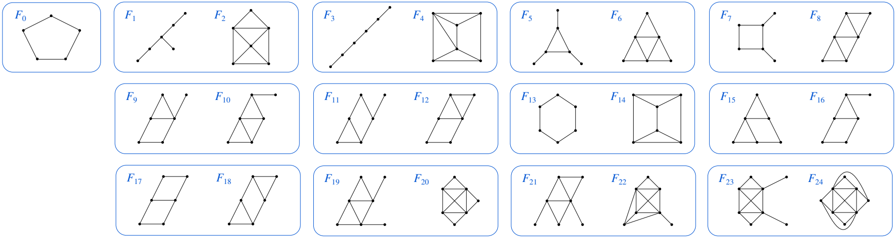
Theorem (2022, 2023)

A graph is GATEX $\iff$ *for all P-modules M the "quotient of $G[M]$" is a pseudo-cograph.*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GaTEx graphs



Theorem (2022, 2023)

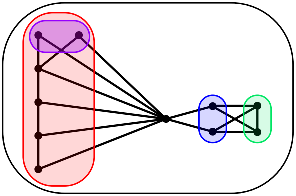
A graph is GaTEx $\iff$ *for all P-modules M the "quotient of $G[M]$" is a pseudo-cograph.*

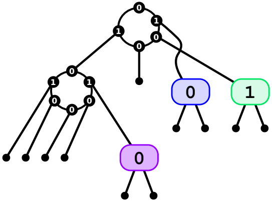
$\iff$ *G is $\mathfrak{F}$-free* *(leads to a brute-force $O(n^8)$-time recognition algorithm).*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GaTEx graphs





Theorem (2022, 2023)

A graph is GaTEx $\iff$ *for all* P*-modules* M *the "quotient of* $G[M]$*" is a pseudo-cograph.*

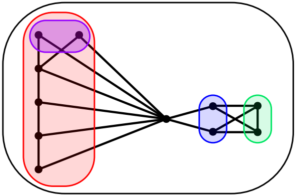
$\iff$ G *is* $\mathcal{F}$*-free* *(leads to a brute-force* $O(n^8)$*-time recognition algorithm).*

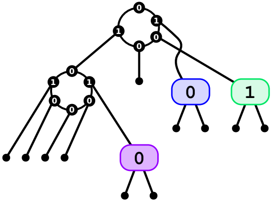
GaTEx *graphs can be recognized and construction of* (N,t) *can be done in* ***linear time****.*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

The general case: GaTEx graphs





Theorem (2022, 2023)

A graph is GaTEx $\iff$ *for all* P*-modules* M *the "quotient of* $G[M]$*" is a pseudo-cograph.*

$\iff$ G *is* $\mathcal{F}$*-free (leads to a brute-force* $O(n^8)$*-time recognition algorithm).*

GaTEx *graphs can be recognized and construction of* (N,t) *can be done in* ***linear time****.*

GaTEx *graphs can have* $O(|n|^2)$ *edges, but can be stored using only* ***linear space****.*

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

Consequences

GATEX graphs form a novel and interesting class of graphs that is closely related to many other well-known and famous graph classes.

Among other results, GATEX graphs are:

- perfect graphs

  Chromatic number of every induced subgraph = size of the largest clique of that subgraph

- comparability (=transitive orientable) graphs

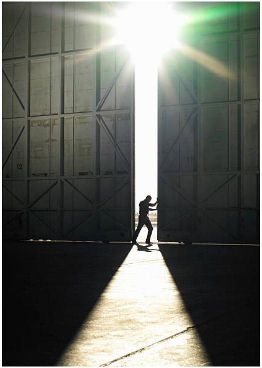
  There exists a transitive orientation on this graphs = represents partial orders in where two elements are connected by an edge if they are comparable to each other in the partial order.

- permutation graphs

  used to represent permutations

- perfectly orderable

  there is an ordering of the vertices of G such that a greedy coloring algorithm with that ordering optimally colors every induced subgraph of the given graph



Every cograph and every graph whose vertices are contained in at one most induced P_4 are GATEX

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

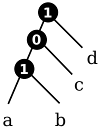
NP-hard Problems that become easy . . .

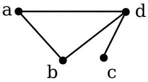
Example: Find optimal vertex coloring.

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

Example: Find optimal vertex coloring.

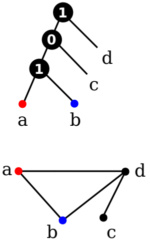




Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

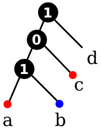
Example: Find optimal vertex coloring.

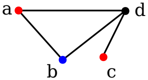


Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

Example: Find optimal vertex coloring.

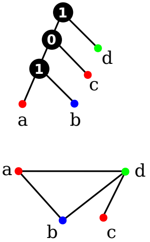




Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

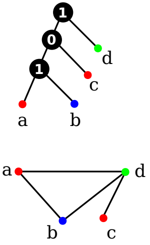
Example: Find optimal vertex coloring.

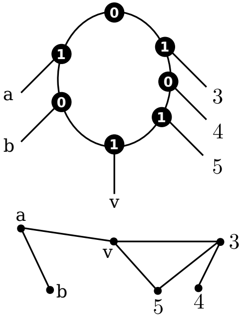


Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

Example: Find optimal vertex coloring.

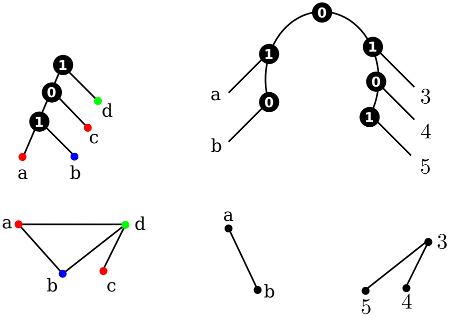




Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

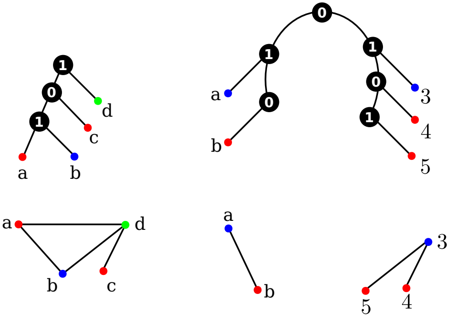
Example: Find optimal vertex coloring.



Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

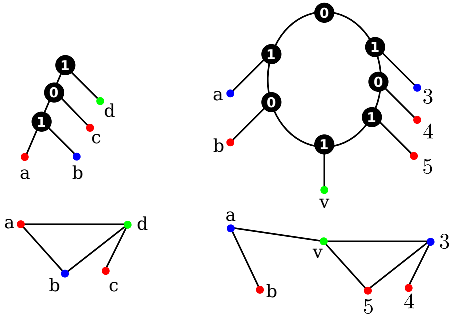
Example: Find optimal vertex coloring.



Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

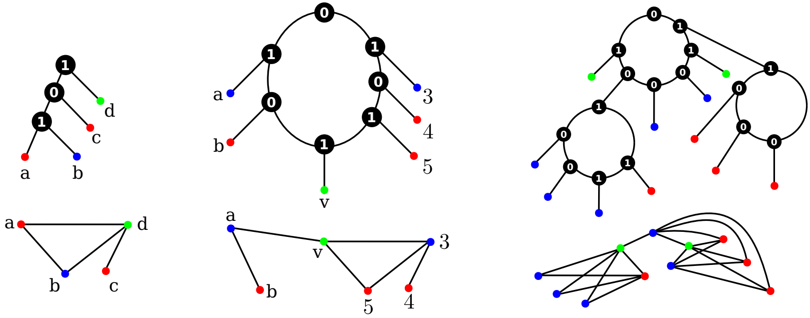
Example: Find optimal vertex coloring.



Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

Example: Find optimal vertex coloring.



Optimal vertex coloring can be solved in linear-time on GATEX graphs.

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy . . .

Theorem (2023)

The following NP-hard problems can be solved in linear-time on GaTEx *graphs.*

- *Finding a minimum vertex coloring*
- *Finding a perfect order*
- *Finding a maximum clique*
- *Finding a maximum independent set*

Scholz & Hellmuth, **Linear Time Algorithms for NP-hard Problems restricted to GaTEx Graphs**, *29th International Computing and Combinatorics Conference (COCOON 23)*, 2024

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy ...

Theorem (2023)

The following NP-hard problems can be solved in linear-time on GaTEx *graphs.*

- *Finding a minimum vertex coloring*
- *Finding a perfect order*
- *Finding a maximum clique*
- *Finding a maximum independent set*

Moreover, GaTEx *graphs have bounded twin-width, i.e., many complexity results established for those graphs (e.g. FPT or approximation results) become applicable for* GaTEx *graphs.*

Scholz & Hellmuth, **Linear Time Algorithms for NP-hard Problems restricted to GaTEx Graphs**, *29th International Computing and Combinatorics Conference (COCOON 23)*, 2024

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy ...

Theorem (2023)

The following NP-hard problems can be solved in linear-time on GaTEx *graphs.*

- *Finding a minimum vertex coloring*
- *Finding a perfect order*
- *Finding a maximum clique*
- *Finding a maximum independent set*

Moreover, GaTEx *graphs have bounded twin-width, i.e., many complexity results established for those graphs (e.g. FPT or approximation results) become applicable for* GaTEx *graphs.*

We conjecture that also the graph-isomorphism problem has a linear-time solution for GaTEx graphs (work in progress).

Scholz & Hellmuth, **Linear Time Algorithms for NP-hard Problems restricted to GaTEx Graphs**, *29th International Computing and Combinatorics Conference (COCOON 23)*, 2024

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy ...

Theorem (2023)

The following NP-hard problems can be solved in linear-time on GATEX *graphs.*

- *Finding a minimum vertex coloring*
- *Finding a perfect order*
- *Finding a maximum clique*
- *Finding a maximum independent set*

Moreover, GATEX *graphs have bounded twin-width, i.e., many complexity results established for those graphs (e.g. FPT or approximation results) become applicable for* GATEX *graphs.*

We conjecture that also the graph-isomorphism problem has a linear-time solution for GATEX graphs (work in progress).

Basic idea: Use the network (N,t) as a guide for the optimization algorithms

Scholz & Hellmuth, **Linear Time Algorithms for NP-hard Problems restricted to GaTEx Graphs**, *29th International Computing and Combinatorics Conference (COCOON 23)*, 2024

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

NP-hard Problems that become easy ...

Theorem (2023)

The following NP-hard problems can be solved in linear-time on GaTEx *graphs.*

- *Finding a minimum vertex coloring*
- *Finding a perfect order*
- *Finding a maximum clique*
- *Finding a maximum independent set*

Moreover, GaTEx *graphs have bounded twin-width, i.e., many complexity results established for those graphs (e.g. FPT or approximation results) become applicable for* GaTEx *graphs.*

We conjecture that also the graph-isomorphism problem has a linear-time solution for GaTEx graphs (work in progress).

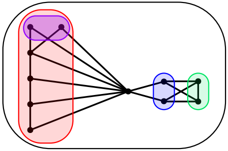
Basic idea: Use the network (N, t) as a guide for the optimization algorithms

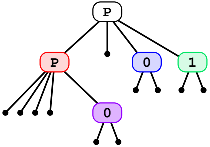
Explicit modular decomposition provides an essential tool-box for solving discrete optimization problems!

Scholz & Hellmuth, **Linear Time Algorithms for NP-hard Problems restricted to GaTEx Graphs**, *29th International Computing and Combinatorics Conference (COCOON 23)*, 2024

Scholz & Hellmuth, **Resolving Prime Modules: The Structure of Pseudo-cographs and Galled-Tree Explainable Graphs**, *Discr. Appl. Math*, 2024

Summary - Explicit Modular Decomposition

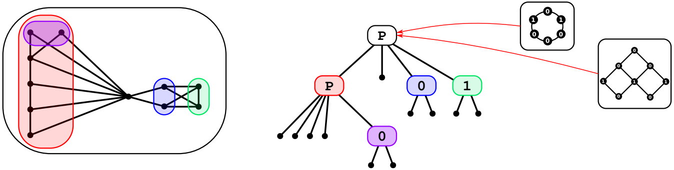




General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

Summary - Explicit Modular Decomposition

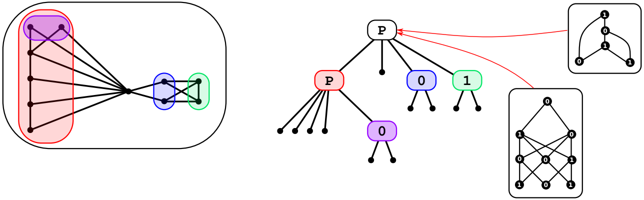


General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

- **Here:** replace "P" vertices in MD tree by simple cycles or half-grids
  This is only a snapshot of what is possible!

Summary - Explicit Modular Decomposition



General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the P-vertices
  to obtain 0/1-labeled rooted networks to explain graphs

- **Here:** replace "P" vertices in MD tree by simple cycles or half-grids
  This is only a snapshot of what is possible!

- we are not restricted to resolving P-vertices in the MD-tree by simple cycles or half-grids only
  For generalizations, we can draw on nearly unlimited resources from phylogenetic networks.

Summary - Explicit Modular Decomposition

- Every graph can be explained by half-grid networks
  = proof of concept

Summary - Explicit Modular Decomposition

- Every graph can be explained by half-grid networks
  = proof of concept
- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)

Summary - Explicit Modular Decomposition

- Every graph can be explained by half-grid networks
  = proof of concept

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
  GATEX graphs are closely related to other famous graph classes

Summary - Explicit Modular Decomposition

- Every graph can be explained by half-grid networks
  = proof of concept
- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
  GATEX graphs are closely related to other famous graph classes
  several NP-hard problems become easy on GATEX graphs

Summary - Explicit Modular Decomposition

- Every graph can be explained by half-grid networks
  = proof of concept

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
  GATEX graphs are closely related to other famous graph classes
  several NP-hard problems become easy on GATEX graphs

- Explicit Modular Decomposition is a very novel concept and a great playground (generalizations e.g. edge-colored di-graphs or matroids are coming)!

Applications

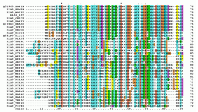
The initial motivation to investigate ”explicit modular decomposition” is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)

Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

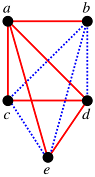
Applications

The initial motivation to investigate "explicit modular decomposition" is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)

genomic sequence data + similarity information



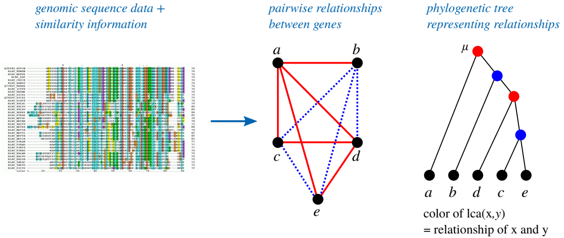
pairwise relationships between genes



Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

Applications

The initial motivation to investigate ”explicit modular decomposition” is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)

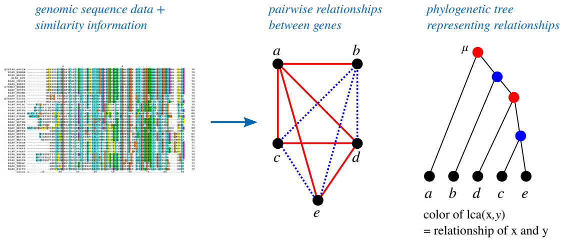


Given a relationship R that can be represented by trees T

Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

Applications

The initial motivation to investigate ”explicit modular decomposition” is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)



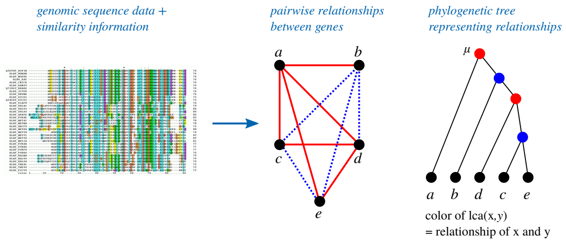
Given a relationship R that can be represented by trees T

- Colors of pairwise lca’s determine relationship

Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

Applications

The initial motivation to investigate ”explicit modular decomposition” is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)



Given a relationship R that can be represented by trees T

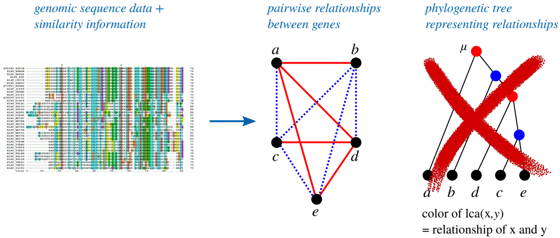
- Colors of pairwise lca's determine relationship

Noise in the data or NON-tree-like evolution $\Longrightarrow$ cannot expect trees!

Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

Applications

The initial motivation to investigate ”explicit modular decomposition” is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)



Given a relationship R that can be represented by trees T

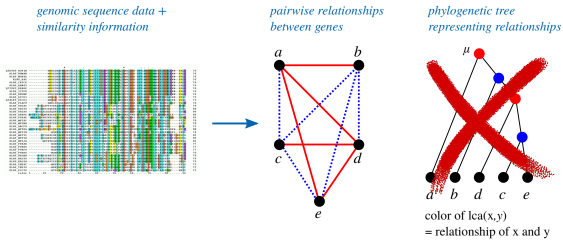
- Colors of pairwise lca’s determine relationship

Noise in the data or NON-tree-like evolution $\Longrightarrow$ cannot expect trees!

Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

Applications

The initial motivation to investigate "explicit modular decomposition" is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)



Given a relationship R that can be represented by trees T

- Colors of pairwise lca's determine relationship

Noise in the data or NON-tree-like evolution $\Longrightarrow$ cannot expect trees!

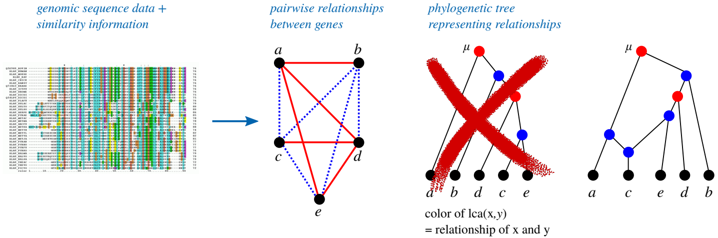
- How are networks characterized with pairwise-lca-properties?

  $\Longrightarrow$ explicit modular decomposition

Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

Applications

The initial motivation to investigate ”explicit modular decomposition” is based on the fact that we want understand in more detail pairwise relationships between genes (e.g horizontal gene transfer and orthology)



Given a relationship R that can be represented by trees T

- Colors of pairwise lca’s determine relationship

Noise in the data or NON-tree-like evolution $\Longrightarrow$ cannot expect trees!

- How are networks characterized with pairwise-lca-properties?

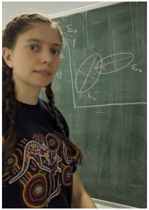
  $\Longrightarrow$ explicit modular decomposition

Hellmuth & Wieseke et al., **Phylogenomics with Paralogs**, *Proceedings of the National Academy of Sciences (PNAS)*, 2015

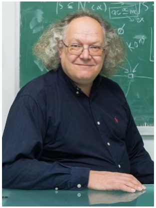
Collaborations

Anna Lindeberg

Guillaume Scholz



Carmen Bruckmann

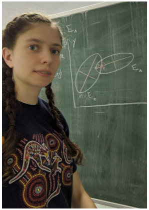


Peter F. Stadler

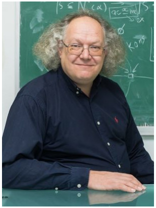
Collaborations

Anna Lindeberg

Guillaume Scholz



Carmen Bruckmann



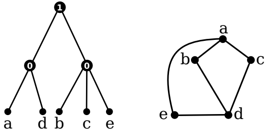
Peter F. Stadler

THANK YOU!

Appendix

Cographs : Graphs without prime modules

Cographs . . .



Cotree (T, t) explains cograph G

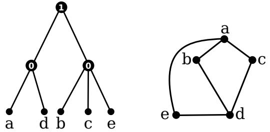
discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

Cographs : Graphs without prime modules

Cographs . . .

- . . . are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathsf{lca}(x,y)) = 1$$



Cotree (T,t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

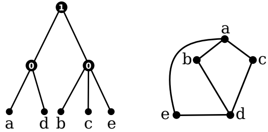
Cographs : Graphs without prime modules

Cographs ...

- ... are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathsf{lca}(x,y)) = 1$$

- ... form an extremely well-known graph class



Cotree (T,t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

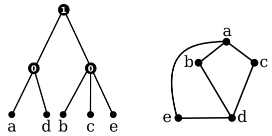
Cographs : Graphs without prime modules

Cographs ...

- ... are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathsf{lca}(x,y)) = 1$$

- ... form an extremely well-known graph class
- ... are recursively defined *(omitted)*



Cotree (T,t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

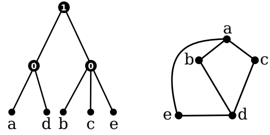
Cographs : Graphs without prime modules

Cographs . . .

- . . . are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathsf{lca}(x,y)) = 1$$

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are precisely the graphs that do not contain induced paths on 4 vertices: $\bullet - \bullet - \bullet - \bullet$



Cotree (T,t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

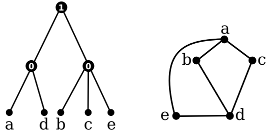
Cographs : Graphs without prime modules

Cographs . . .

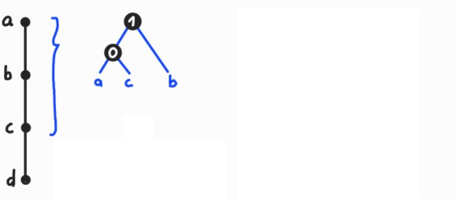
- . . . are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathrm{lca}(x,y)) = 1$$

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are precisely the graphs that do not contain induced paths on 4 vertices: $\bullet - \bullet - \bullet - \bullet$



Cotree (T,t) explains cograph G



discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

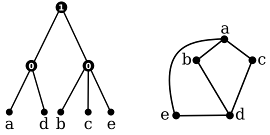
Cographs : Graphs without prime modules

Cographs . . .

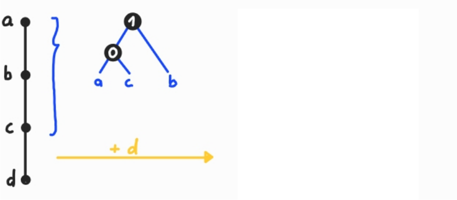
- . . . are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathsf{lca}(x,y)) = 1$$

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are precisely the graphs that do not contain induced paths on 4 vertices: $\bullet - \bullet - \bullet - \bullet$



Cotree (T,t) explains cograph G



discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

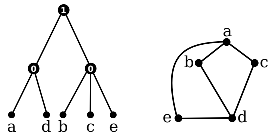
Cographs : Graphs without prime modules

Cographs ...

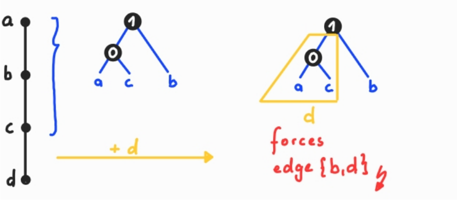
- ...are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathsf{lca}(x,y)) = 1$$

- ...form an extremely well-known graph class
- ...are recursively defined *(omitted)*
- ...are precisely the graphs that do not contain induced paths on 4 vertices: $\bullet - \bullet - \bullet - \bullet$



Cotree (T,t) explains cograph G



discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

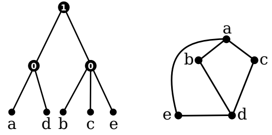
Cographs : Graphs without prime modules

Cographs ...

- ...are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathrm{lca}(x,y)) = 1$$

- ...form an extremely well-known graph class
- ...are recursively defined *(omitted)*
- ...are precisely the graphs that do not contain induced paths on 4 vertices: $\bullet - \bullet - \bullet - \bullet$



Cotree (T,t) explains cograph G

discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

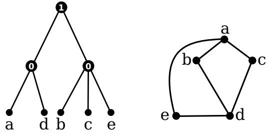
Cographs : Graphs without prime modules

Cographs . . .

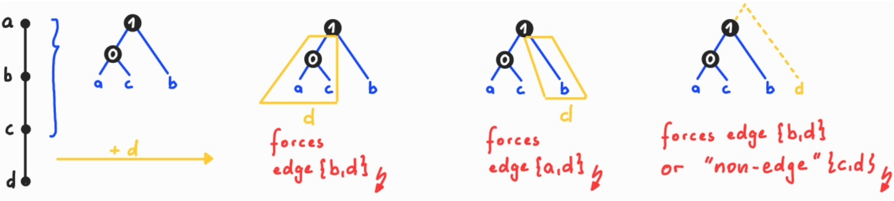
- . . . are explained by a 0/1-labeled tree (T,t):

$$\{x,y\} \in E \iff t(\mathrm{lca}(x,y)) = 1$$

- . . . form an extremely well-known graph class
- . . . are recursively defined *(omitted)*
- . . . are precisely the graphs that do not contain induced paths on 4 vertices: • – • – • – •



Cotree (T,t) explains cograph G



discovered independently by several authors since the 1970s; Jung (1978), Lerchs (1971), Seinsche (1974), and Sumner (1974).

Pseudo-cographs: Algorithms

Recognition - The Basic Idea

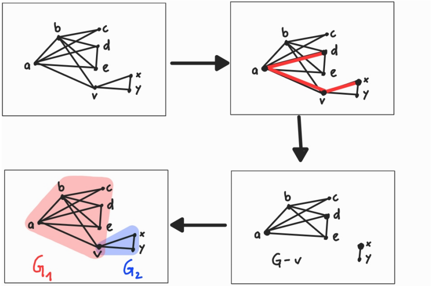
If G cograph - done

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Pseudo-cographs: Algorithms

Recognition - The Basic Idea

If G cograph - done
Else find one P_4
  For all $v \in P_4$
    If $G - v$ or $\overline{G - v}$ disconnected
      take one connected component C
      Put $V(G_1) = C \cup \{v\}$
      Put $V(G_2) = (V \setminus C) \cup \{v\}$
      If G_1 and G_2 are cographs
        return (v, G_1, G_2)

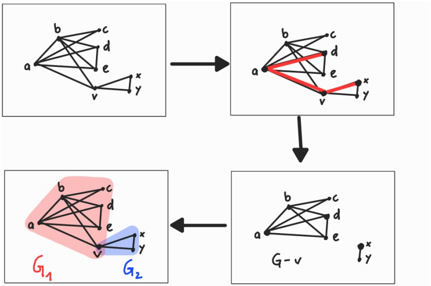


Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Pseudo-cographs: Algorithms

Recognition - The Basic Idea

If G cograph - done
Else find one P_4
  For all $v \in P_4$
    If $G - v$ or $\overline{G - v}$ disconnected
      take one connected component C
      Put $V(G_1) = C \cup \{v\}$
      Put $V(G_2) = (V \setminus C) \cup \{v\}$
      If G_1 and G_2 are cographs
        return (v, G_1, G_2)



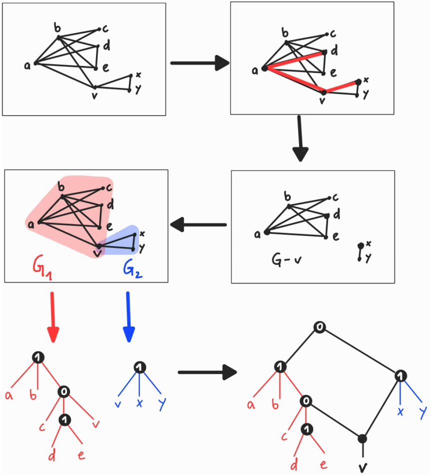
Theorem (2022)
Pseudo-cographs can be recognized in linear time and a corresponding 0/1-labeled network can be constructed within the same time complexity.

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Pseudo-cographs: Algorithms

Recognition - The Basic Idea

If G cograph - done
Else find one P_4
  For all $v \in P_4$
    If $G - v$ or $\overline{G - v}$ disconnected
      take one connected component C
      Put $V(G_1) = C \cup \{v\}$
      Put $V(G_2) = (V \setminus C) \cup \{v\}$
      If G_1 and G_2 are cographs
        return (v, G_1, G_2)



Theorem (2022)
Pseudo-cographs can be recognized in linear time and a corresponding 0/1-labeled network can be constructed within the same time complexity.

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Pseudo-cographs and galled-tree explainable graphs

Basic Properties

Pseudo-cographs and galled-tree explainable graphs

Basic Properties

- Every cograph is a pseudo-cograph and, in particular, galled-tree explainable.

Pseudo-cographs and galled-tree explainable graphs

Basic Properties

- Every cograph is a pseudo-cograph and, in particular, galled-tree explainable.
- G is galled-tree explainable $\iff$ $\overline{G}$ is galled-tree explainable.
  [closed under complementation]

Pseudo-cographs and galled-tree explainable graphs

Basic Properties

- Every cograph is a pseudo-cograph and, in particular, galled-tree explainable.
- G is galled-tree explainable $\iff$ $\overline{G}$ is galled-tree explainable.
  [closed under complementation]
- G is galled-tree explainable $\iff$ every induced subgraph of G is galled-tree explainable.
  [heritable]

Pseudo-cographs and galled-tree explainable graphs

Basic Properties

- Every cograph is a pseudo-cograph and, in particular, galled-tree explainable.
- G is galled-tree explainable $\iff$ $\overline{G}$ is galled-tree explainable.
  [closed under complementation]
- G is galled-tree explainable $\iff$ every induced subgraph of G is galled-tree explainable.
  [heritable]
- Pseudo-cographs as well as galled-tree explainable graph are weakly-chordal and, thus, perfect.

  $\implies$ Many NP-hard problems get easy on such graphs!

Pseudo-cographs and galled-tree explainable graphs

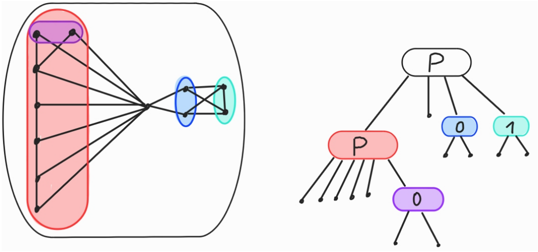
Basic Properties

- Every cograph is a pseudo-cograph and, in particular, galled-tree explainable.
- G is galled-tree explainable $\iff$ $\overline{G}$ is galled-tree explainable.
  [closed under complementation]
- G is galled-tree explainable $\iff$ every induced subgraph of G is galled-tree explainable.
  [heritable]
- Pseudo-cographs as well as galled-tree explainable graph are weakly-chordal and, thus, perfect.

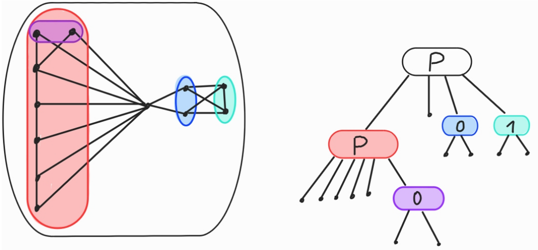
  $\Longrightarrow$ Many NP-hard problems get easy on such graphs!

This looks like a very interesting, novel graph class !!

Explicit Modular Decomposition



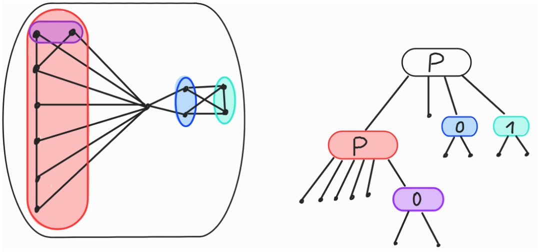
Explicit Modular Decomposition



General Aim:

- Preserve the main feature of the MD-tree,
  try to modify only the prime-vertices ”P”
  to obtain 0/1-labeled rooted networks to explain graphs
- “unbox” the structure of prime modules

Explicit Modular Decomposition



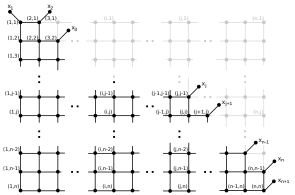
General Aim:

- Preserve the main feature of the MD-tree,
  try to modify only the prime-vertices ”P”
  to obtain 0/1-labeled rooted networks to explain graphs
- “unbox” the structure of prime modules

As a first attempt we may ask:

Is there a 0/1-labeled network that can explain EVERY given graph?

Explicit Modular Decomposition: Half-grids

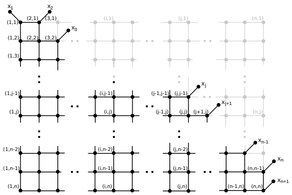


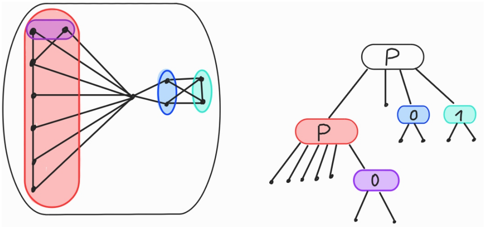
Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from the MD-tree by locally replacing "P"-vertices by half-grids = median graphs.

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids



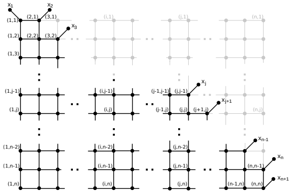


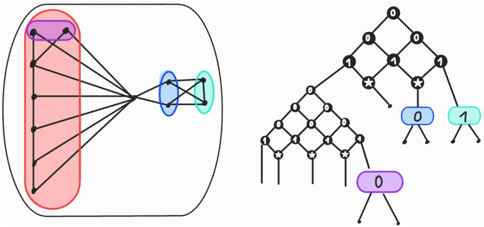
Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from the MD-tree by locally replacing "P"-vertices by half-grids = median graphs.

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids



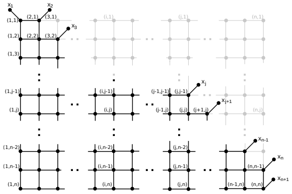


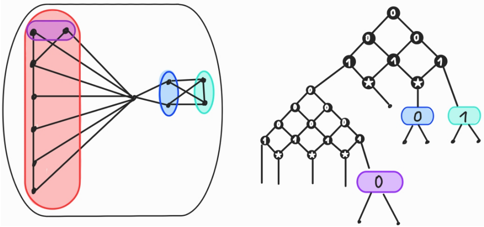
Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from the MD-tree by locally replacing "P"-vertices by half-grids = median graphs.

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

Explicit Modular Decomposition: Half-grids





Theorem (2021)

Every graph G can be explained by a 0/1-labeled half-grid and thus, by a network obtained from the MD-tree by locally replacing "P"-vertices by half-grids = median graphs.

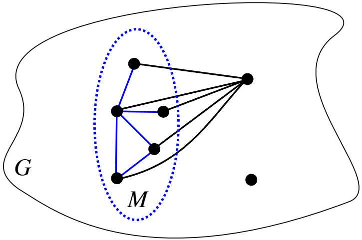
Half-grids are rather "heavy" ($O(|V(G)|^2)$ edges and vertices) and of less interest from the structural point and biological point of view.

$\Longrightarrow$ Can we go simpler?

Bruckmann, Stadler, Hellmuth, **From Modular Decomposition Trees to Rooted Median Graphs**, *Discr. Appl. Math*, 2021

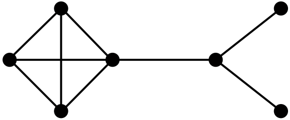
Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



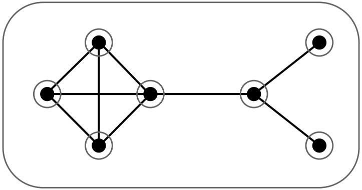
Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



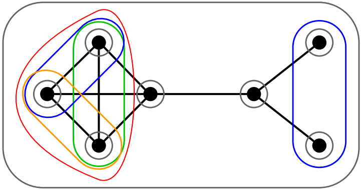
Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



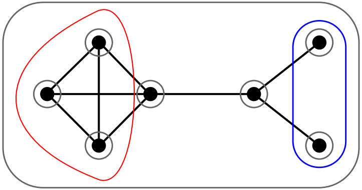
Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



Basics: Modular Decomposition (MD)

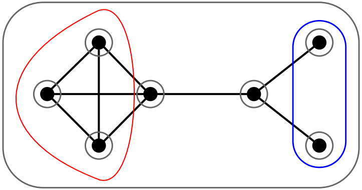
$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



MD = set of all *non-overlapping* modules

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$

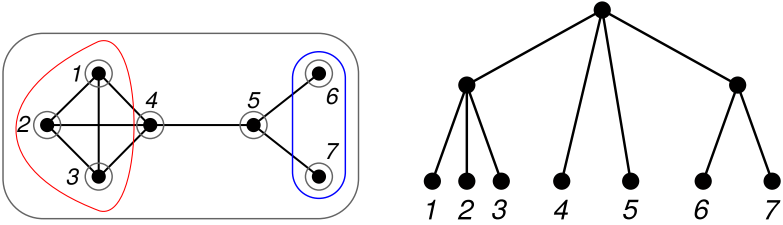


MD = set of all *non-overlapping* modules

MD is uniquely determined and can be computed in linear time

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



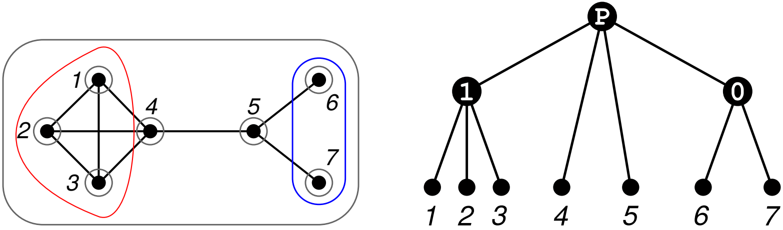
MD = set of all *non-overlapping* modules

MD is uniquely determined and can be computed in linear time

MD forms a hierarchy = rooted tree

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



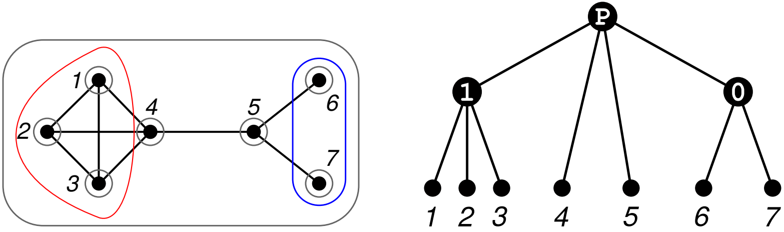
MD = set of all *non-overlapping* modules

MD is uniquely determined and can be computed in linear time

MD forms a hierarchy = rooted tree

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



MD = set of all *non-overlapping* modules

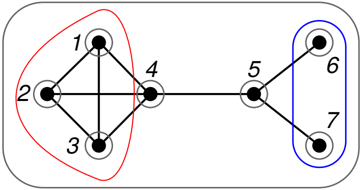
- MD is uniquely determined and can be computed in linear time
- MD forms a hierarchy = rooted tree

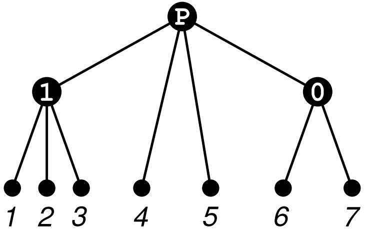
There are different type of modules:

- Parallel (0), Series (1) and Prime (P) modules
  defined in terms of connectedness conditions (omitted).

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



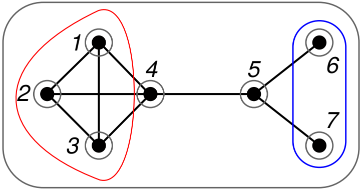


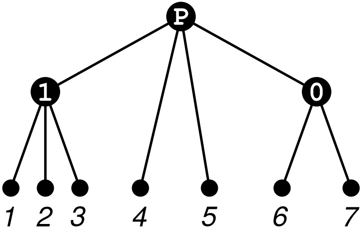
BUT, full information of G is provided only if G does not contain prime modules.

- $t(\text{lca}(3,4)) = \mathtt{P}$ and $\{3,4\} \in E(G)$
- $t(\text{lca}(3,5)) = \mathtt{P}$ and $\{3,5\} \notin E(G)$

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$





BUT, full information of G is provided only if G does not contain prime modules.

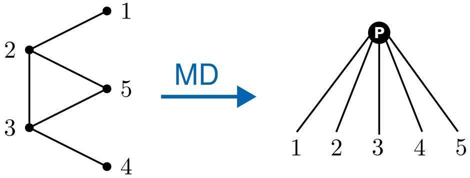
- $t(\text{lca}(3,4)) = \texttt{P}$ and $\{3,4\} \in E(G)$
- $t(\text{lca}(3,5)) = \texttt{P}$ and $\{3,5\} \notin E(G)$

For cographs:

- The MD does not contain prime modules
- $\{x,y\} \in E \iff t(\text{lca}(x,y)) = 1$
  Thus, full information about cographs is provided by MD

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



BUT, full information of G is provided only if G does not contain prime modules.

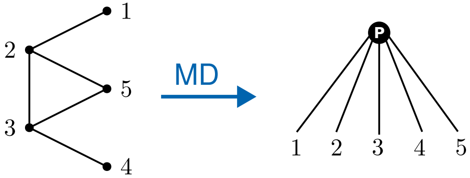
- $t(\text{lca}(3,4)) = \mathtt{P}$ and $\{3,4\} \in E(G)$
- $t(\text{lca}(3,5)) = \mathtt{P}$ and $\{3,5\} \notin E(G)$

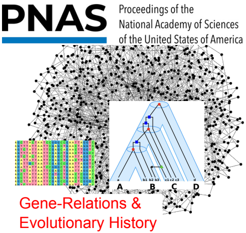
For cographs:

- The MD does not contain prime modules
- $\{x,y\} \in E \iff t(\text{lca}(x,y)) = 1$
  Thus, full information about cographs is provided by MD

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$





BUT, full information of G is provided only if G does not contain prime modules.

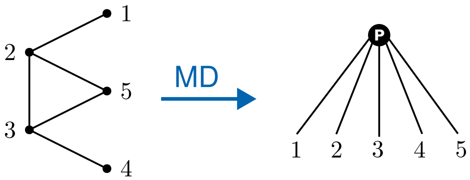
We used MD and cographs for studies in the context of evolutionary biology

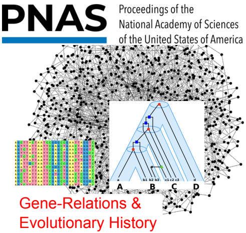
Milestone: How to use “noise” in the data as an additional source of information!

Hellmuth & Wiesecke et al., **Phylogenomics with Paralogs**, *PNAS*, 2015

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$





BUT, full information of G is provided only if G does not contain prime modules.

We used MD and cographs for studies in the context of evolutionary biology

Milestone: How to use "noise" in the data as an additional source of information!

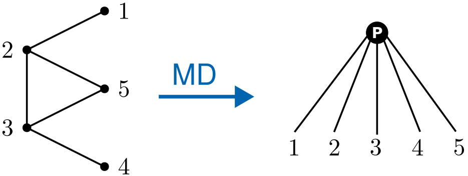
We observed:

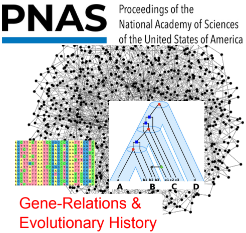
- It is crucial to understand the structure of subgraphs induced by prime modules!
- Evolution is not always tree-like and often better explained by networks!

Hellmuth & Wiesecke et al., **Phylogenomics with Paralogs**, *PNAS*, 2015

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$



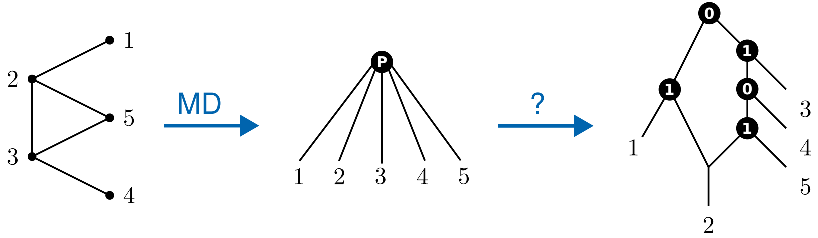


Central Questions:

Can we explain graphs by structures that go beyond trees?

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$

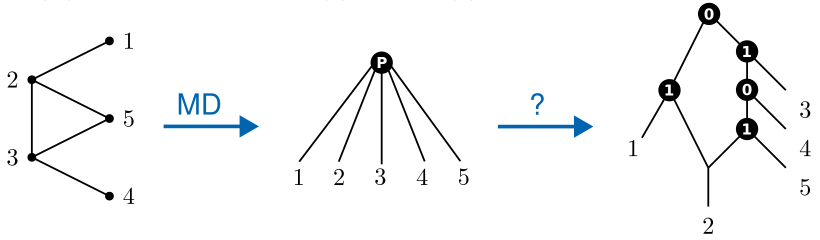


Central Questions:

Can we explain graphs by structures that go beyond trees?

Basics: Modular Decomposition (MD)

$M \subseteq V(G)$ is a **module** in G if $N(x) \setminus M = N(y) \setminus M$ for all $x, y \in M$

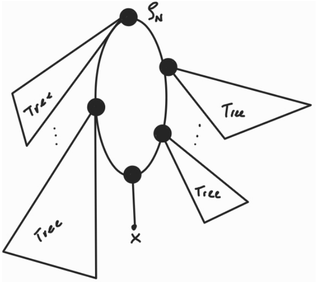


Central Questions:

Can we explain graphs by structures that go beyond trees?

Aim: Understand graph classes that can be explained by 0/1-labeled networks.

Pseudo-cographs: Characterization

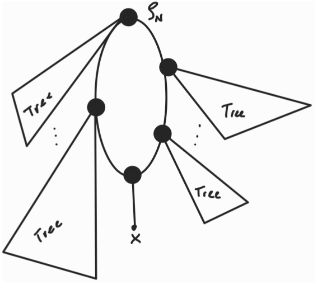


Theorem (2022)

G is a pseudo-cograph $\iff$ $|V(G)| \leq 2$ or G can be explained by a galled-tree (N,t) that contains precisely one cycle C such that $\rho_C = \rho_N$ and the (unique) hybrid has precisely one child.

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Pseudo-cographs: Characterization



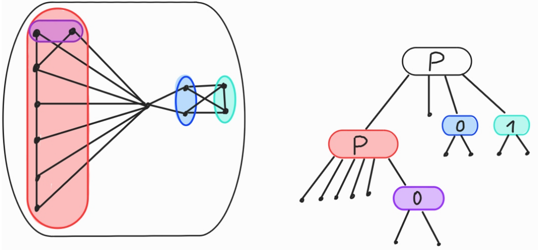
Theorem (2022)

G is a pseudo-cograph $\iff$ $|V(G)| \leq 2$ or G can be explained by a galled-tree (N,t) that contains precisely one cycle C such that $\rho_C = \rho_N$ and the (unique) hybrid has precisely one child.

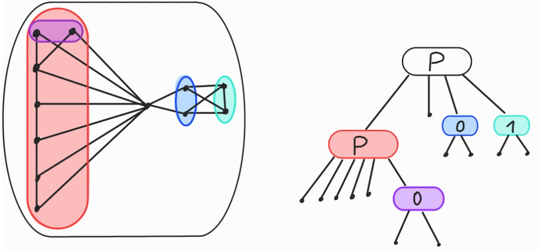
Pseudo-cographs can be recognized in linear time and a corresponding 0/1-labeled network can be constructed within the same time complexity.

Scholz & Hellmuth, **From Modular Decomposition Trees to Level-1 networks: Pseudo-Cographs, Polar-Cats and Prime Polar-Cats**, *Discr. Appl. Math*, 2022

Explicit Modular Decomposition



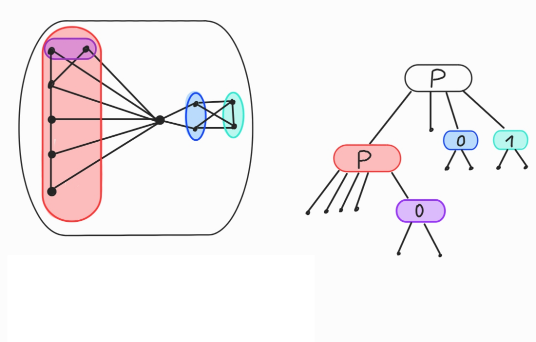
Explicit Modular Decomposition



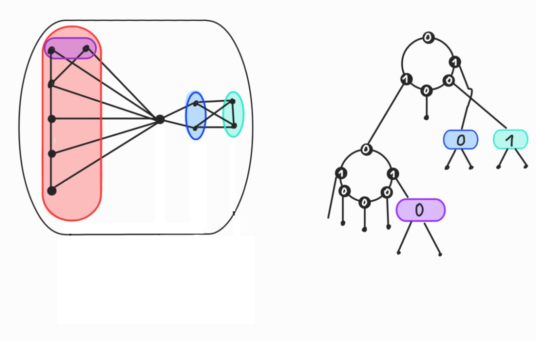
General Aim:

- Preserve the main features of the MD-tree,
  try to modify only the prime-vertices ”P”
  to obtain 0/1-labeled rooted networks to explain graphs
- “unbox” the structure of prime modules

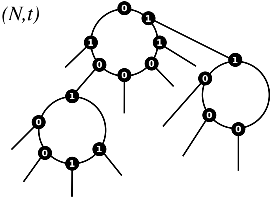
MD and simple cycles



MD and simple cycles

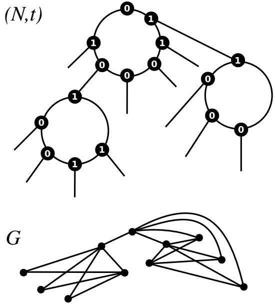


MD and simple cycles



Galled-tree: = 0/1-labeled network “consisting” of edges and simple cycles

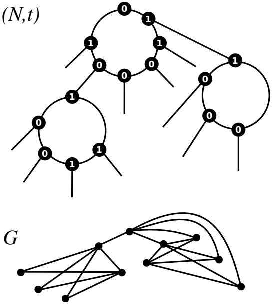
MD and simple cycles



G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathsf{lca}_N(x,y)) = 1$.

Galled-tree: = 0/1-labeled network “consisting” of edges and simple cycles

MD and simple cycles

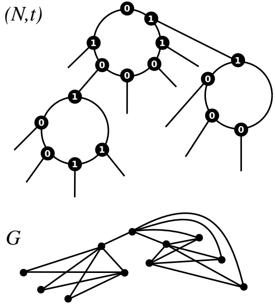


G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathrm{lca}_N(x,y)) = 1.$

Galled-tree: = 0/1-labeled network "consisting" of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GATEX)

MD and simple cycles



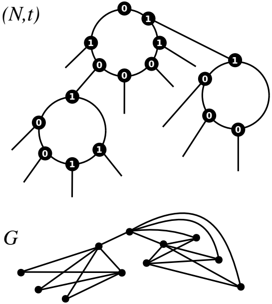
G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathrm{lca}_N(x,y)) = 1$.

Galled-tree: = 0/1-labeled network "consisting" of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GATEX)

Can we characterize the class of graphs of GATEX graphs?

MD and simple cycles



G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathsf{lca}_N(x,y)) = 1$.

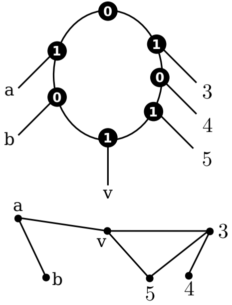
Galled-tree: = 0/1-labeled network "consisting" of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GATEX)

Can we characterize the class of graphs of GATEX graphs?

. . . let us start simple . . .

MD and simple cycles



Galled-tree: = 0/1-labeled network “consisting” of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GATEX)

Can we characterize the class of graphs of GATEX graphs?

. . . let us start simple . . .

Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

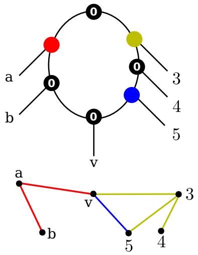
Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

Open Questions:

- 0/1/.../n-labeled networks to explain edge-colored graphs.

$$t(\mathsf{lca}_T(x,y)) = i \iff \{x,y\} \in E \text{ has color } i$$



Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

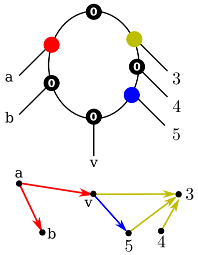
Open Questions:

- 0/1/.../n-labeled networks to explain edge-colored graphs.

  $$t(\mathrm{lca}_T(x,y)) = i \iff \{x,y\} \in E \text{ has color } i$$

- What if we have directed graphs that we want to explain by labeled networks?

  Can we use ordered networks to explain them?



Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

Open Questions:

- 0/1/.../n-labeled networks to explain edge-colored graphs.

$$t(\mathrm{lca}_T(x,y)) = i \iff \{x,y\} \in E \text{ has color } i$$

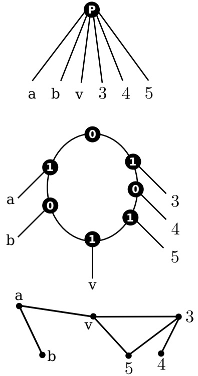
- What if we have directed graphs that we want to explain by labeled networks?

  Can we use ordered networks to explain them?

- In the MD-tree (T,t) clusters = modules

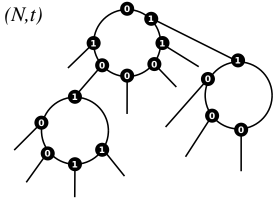
  In (N,t), the modules of G form a subset of the clusters in (N,t)

  $\Longrightarrow$ generalization of modules!



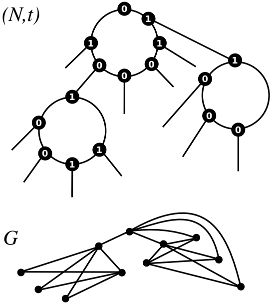
The general case: GATEX graphs

The general case: GATEX graphs



Galled-tree: = 0/1-labeled rooted network “consisting” of edges and simple cycles

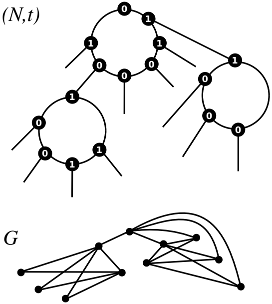
The general case: GaTEx graphs



G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathrm{lca}_N(x,y)) = 1$.

Galled-tree: = 0/1-labeled rooted network “consisting” of edges and simple cycles

The general case: GATEX graphs

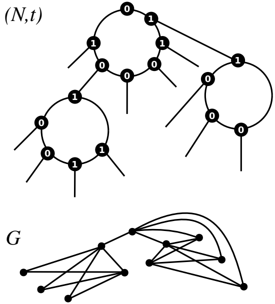


G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathrm{lca}_N(x,y)) = 1.$

Galled-tree: = 0/1-labeled rooted network “consisting” of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GATEX)

The general case: GaTEx graphs



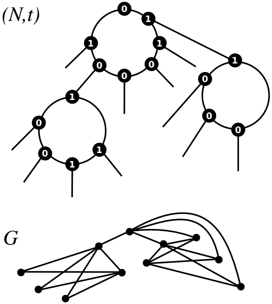
G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathrm{lca}_N(x,y)) = 1$.

Galled-tree: = 0/1-labeled rooted network “consisting” of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GaTEx)

Can we characterize the class of GaTEx graphs?

The general case: GaTEx graphs



G explained by (N,t) if $\{x,y\} \in E(G) \iff t(\mathrm{lca}_N(x,y)) = 1$.

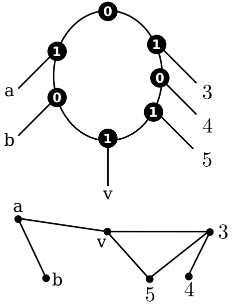
Galled-tree: = 0/1-labeled rooted network “consisting” of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GaTEx)

Can we characterize the class of GaTEx graphs?

. . . let us start simple . . .

The general case: GATEX graphs



Galled-tree: = 0/1-labeled rooted network “consisting” of edges and simple cycles

We call such graphs **Ga**lled-**T**ree **Ex**plainable (GATEX)

Can we characterize the class of GATEX graphs?

. . . let us start simple . . .

Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

Open Questions:

- **Edge-Colored Graphs**
  0/1/.../n-labeled networks to explain edge-colored graphs.
  $$t(\mathsf{lca}_T(x,y)) = i \iff \{x,y\} \in E \text{ has color } i$$

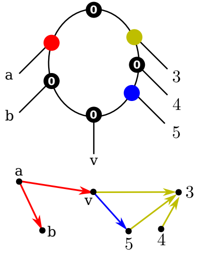
Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

Open Questions:

- **Edge-Colored Graphs**
  0/1/.../n-labeled networks to explain edge-colored graphs.
  $$t(\mathrm{lca}_T(x,y)) = i \iff \{x,y\} \in E \text{ has color } i$$
- **Directed Graphs**
  What if we have directed graphs that we want to explain by labeled networks?

  Can we use ordered networks to explain them?



Summary: GATEX graphs

- we characterized GATEX graphs (=graphs that can be explained by 0/1 galled trees)
- GATEX graphs are closely related to other famous graph classes
- Several NP-hard problems become easy on GATEX graphs

Open Questions:

- **Edge-Colored Graphs**
  0/1/.../n-labeled networks to explain edge-colored graphs.
  $$t(\mathrm{lca}_T(x,y)) = i \iff \{x,y\} \in E \text{ has color } i$$
- **Directed Graphs**
  What if we have directed graphs that we want to explain by labeled networks?

  Can we use ordered networks to explain them?
- **Hypergraphs and Set-Systems**