Advanced Graph Algorithms

Conf. dr. ing. Guillaume Ducoffe

guillaume.ducoffe@fmi.unibuc.ro

Motivations

- Cographs have nice characterizations and algorithmic properties.
- We would like to extend these results to larger graph classes.
- How do we measure closeness to a cograph?
- How to recognize graphs that are “close-to-cographs”?

Reminder: twins

Definition

Two vertices u, v are twins if $N(u) \setminus \{v\} = N(v) \setminus \{u\}$.

Proposition: Being twins is an equivalence relation!

Proof: Assume $N(u) \setminus \{v\} = N(v) \setminus \{u\}$ and $N(v) \setminus \{w\} = N(w) \setminus \{v\}$.

- Assume $u \in N(v)$. Then, $u \in N(v) \setminus \{w\} \subseteq N(w)$. Similarly, $w \in N(u) \setminus \{v\} \subseteq N(v)$. Therefore,

$$\begin{aligned} N(w) \setminus \{u\} &= \{v\} \cup (N(w) \setminus \{u, v\}) = \{v\} \cup (N(v) \setminus \{u, w\}) \\ &= \{v\} \cup (N(u) \setminus \{v, w\}) = N(u) \setminus \{w\} \end{aligned}$$

- Otherwise, u and v are adjacent in the complement $\overline{G}$. Since being twins in G is equivalent to being twins in $\overline{G}$, we are back to the previous case.

Computing twins

Observation: True twins in G are False twins in $\overline{G}$, and vice-versa.

Computing false twins

Observation: True twins in G are False twins in $\overline{G}$, and vice-versa.

1) Initialize in $\mathcal{O}(n)$ time a **partition refinement** data structure with one group equal to $V(G)$.

2) For every $v \in V(G)$, refine existing groups according to $N(v)$. it takes $\mathcal{O}(d(v))$ time.

3) Two vertices are false twins if and only if they belong to the same final group.

Complexity: $\mathcal{O}(n + m)$.

Computing false twins

Observation: True twins in G are False twins in $\overline{G}$, and vice-versa.

1) Initialize in $\mathcal{O}(n)$ time a **partition refinement** data structure with one group equal to $V(G)$.

2) For every $v \in V(G)$, refine existing groups according to $N(v)$. it takes $\mathcal{O}(d(v))$ time.

3) Two vertices are false twins if and only if they belong to the same final group.

Complexity: $\mathcal{O}(n+m)$.

$\rightarrow$ For true twins, it suffices to refine according to $N[v]$.

Neighbourhood diversity

Definition (Neighbourhood diversity)

Number of twin equivalence classes.

- Complete graphs have neighbourhood diversity equal to 1.

- Stars, and more generally complete bipartite graphs, have neighbourhood diversity equal to 2.

- However, cographs have unbounded neighbourhood diversity!

$\Longrightarrow$ Need for a stronger property.

Modules

Definition (Module)

A vertex subset M such that, for every $x, y \in M$, we have $N(x) \setminus M = N(y) \setminus M$.

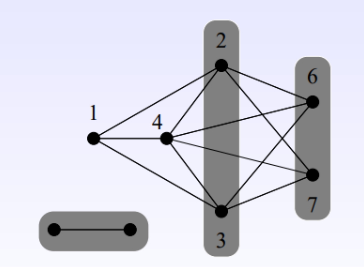
Remark: Twin classes are modules. The converse is not true in general.

- In every graph $G = (V, E)$, the sets $\emptyset$, V and $\{v\}$ for every vertex $v \in V$ are always modules.
- A graph is **prime** if it only has trivial modules.

$\rightarrow$ A cograph with > 1 vertices is never prime because it contains a pair of twins. It implies that every prime graph contains an induced P_4.

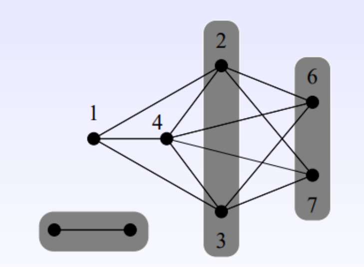
Examples

- Every connected component of a graph G is a module.
- Every co-connected component of a graph G is also a module.
- The same holds for any disjoint union of (co)-connected components.



Examples

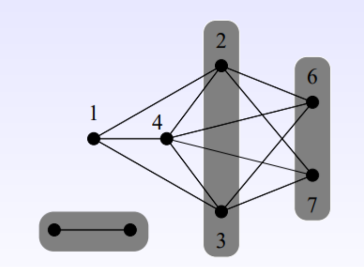
- Every connected component of a graph G is a module.
- Every co-connected component of a graph G is also a module.
- The same holds for any disjoint union of (co)-connected components.



$\rightarrow$ a prime graph must be connected and co-connected.

Examples

- Every connected component of a graph G is a module.
- Every co-connected component of a graph G is also a module.
- The same holds for any disjoint union of (co)-connected components.

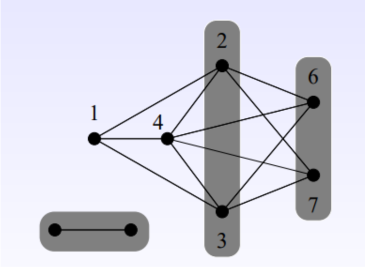


$\rightarrow$ a prime graph must be connected and co-connected.

$\rightarrow$ the number of modules can be exponential (e.g., in a clique).

Examples

- Every connected component of a graph G is a module.
- Every co-connected component of a graph G is also a module.
- The same holds for any disjoint union of (co)-connected components.



$\rightarrow$ a prime graph must be connected and co-connected.

$\rightarrow$ the number of modules can be exponential (e.g., in a clique).

$\rightarrow$ the nodes in a cotree represent modules of a cograph. We aim at obtaining a similar tree representation for the modules in an arbitrary graph.

Basic properties

(1) M is a module of G if and only if M is a module of $\overline{G}$.

Basic properties

(1) M is a module of G if and only if M is a module of $\overline{G}$.

(2) If M is a module of G, and M' is a module of $G[M]$, then M' is also a module of G.

Basic properties

(1) M is a module of G if and only if M is a module of $\overline{G}$.

(2) If M is a module of G, and M' is a module of $G[M]$, then M' is also a module of G.

(3) If M, M' are intersecting modules of G then $M \cap M'$, $M \cup M'$, $M \setminus M'$, $M' \setminus M$ and $M \Delta M'$ (symmetric difference) are also modules of G.

Strong modules

Definition

A module M is strong if it does not overlap any other module, *i.e.*, for any module $M' \neq M$, either $M \cap M' = \emptyset$, $M \subseteq M'$, or $M' \subseteq M$.

A **maximal strong module** is a strong module $M \neq V$ that is inclusion-wise maximal.

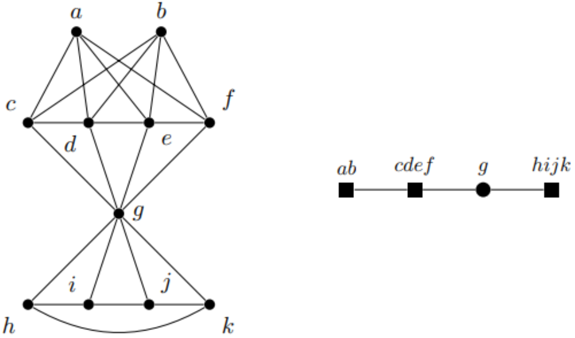
Proposition: the family $\mathcal{M}(G)$ of maximal strong modules of G is a partition of $V(G)$.

Proof: every vertex v is in a strong module, namely, $\{v\}$. Furthermore, since modules of $\mathcal{M}(G)$ are strong, they cannot overlap.

Quotient graph

Definition

The quotient subgraph of G, denoted by $G_{/\mathcal{M}(G)}$, is the induced subgraph obtained by keeping one vertex in every maximal strong module of G.



Modular decomposition theorem

Theorem (Gallai, 1967)

For every graph $G = (V, E)$ with at least four vertices, exactly one of the following conditions must be true:

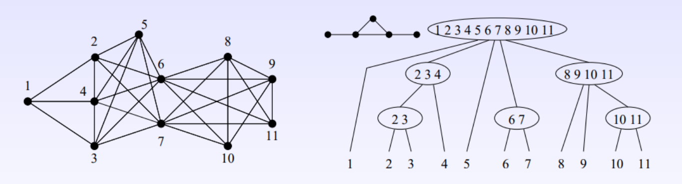
- *G is disconnected;*
- *$\overline{G}$ is disconnected;*
- *$G_{/\mathcal{M}(G)}$ is prime.*

Remark: for cographs, we always fall in one of the two first cases.

Modular decomposition tree

Generalization of the cotree, where we implicitly represent all modules of a graph G. Every node represents a different strong module of G.

- The root represents V
- The leaves represent the vertices of G
- For a strong module M with > 1 vertices, if $G[M]$ is (co-)disconnected then the children of M must represent the (co-)connected components of $G[M]$. Otherwise, the children of M represent $\mathcal{M}(G[M])$.



Computation of the modular decomposition tree

1) For every two vertices x, y, we compute the smallest module $m(x, y)$ that contains both x, y.

m(x,y) := {x,y}

while there exists a vertex v with both a neighbour and a non-neighbour in m(x,y):

    add all such vertices v to m(x,y)

2) If $m(x, y) = V$ for every x, y then G is prime.

3) Otherwise, let $A = m(x, y) \neq V$ be arbitrary. We replace A in G by a new vertex a, that results in a new graph G_a. We compute the modular decompostion of G_a and $G[A]$ separately.

State of the art

Our algorithm from the previous slide is polynomial, but far from linear.

Theorem (Tedder et al., 2008)

The modular decomposition tree of any graph can be computed in $\mathcal{O}(n+m)$ time.

This result will be admitted in the subsequent classes and seminars.

Questions