Algorithmic Aspects of Modular Decomposition

Christophe Paul

CNRS - LIRMM - Univsersité Montpellier II, France

October 24, 2006

Habilitation (in French) avalaible at `www.lirmm.fr/~paul`

Joint work with

- B.M. Bui Xuan, C. Crespelle (LIRMM),
- A. Bretscher, D. Corneil (U. of Toronto),
- M. Habib, F. de Montgolfier (LIAFA),
- L. Viennot (INRIA Rocquencourt)

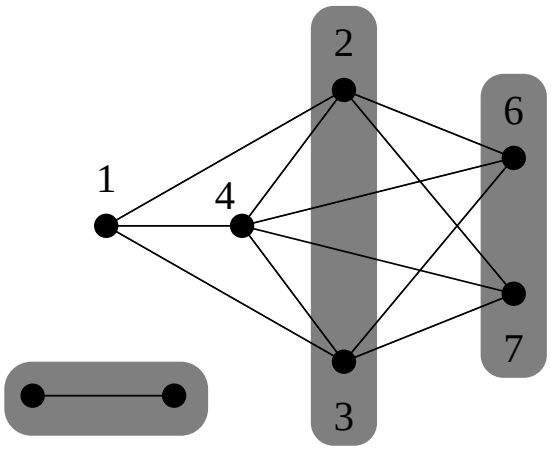
Some motivations

- A very natural operation on discrete structures (used is the proof of the perfect graph theorem)
- Help to capture the structure of many graphs families (basis of a theory for comparability graphs)
- Divide and conquer paradigm for optimization problems
- Provide compact representation of graphs (even for dynamic graphs)
- ...

1. Definitions and preliminaries
2. Ehrenfeucht et al.'s algorithm
   - Principle
   - Computation of $\mathcal{M}(G, v)$
   - Computation of $MD(G_{/\mathcal{M}(G,v)})$
3. New algorithmic trends
   - Factoring permutation
   - LexBFS

Modules

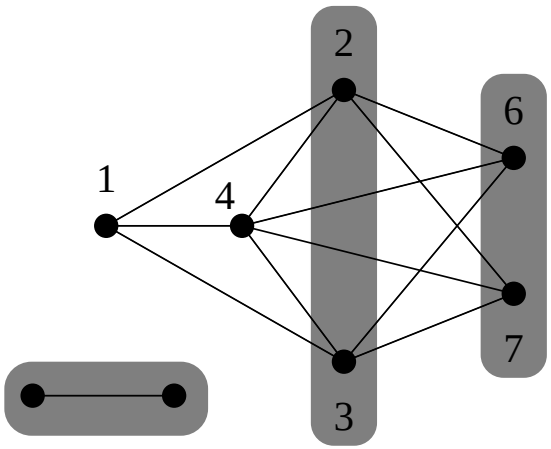
A subset of vertices M of a graph $G = (V, E)$ is a **module** iff
$\forall x \in V \setminus M$, either $M \subseteq N(x)$ or $M \cap N(x) = \emptyset$



Examples of modules

Modules

A subset of vertices M of a graph $G = (V, E)$ is a **module** iff
$\forall x \in V \setminus M$, either $M \subseteq N(x)$ or $M \cap N(x) = \emptyset$



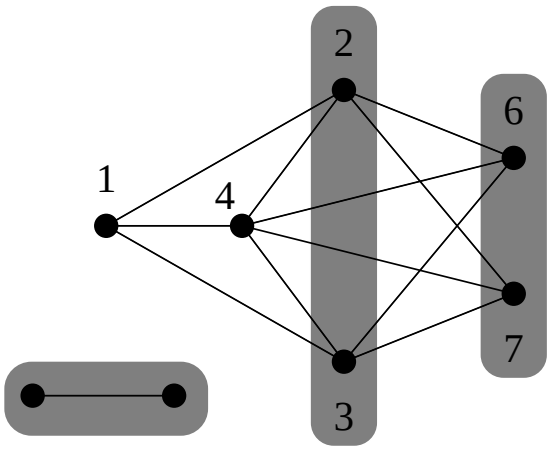
Examples of modules

- connected components
- connected components of $\overline{G}$

Modules

A subset of vertices M of a graph $G = (V, E)$ is a **module** iff

$$\forall x \in V \setminus M, \text{ either } M \subseteq N(x) \text{ or } M \cap N(x) = \emptyset$$

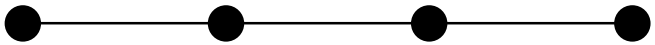


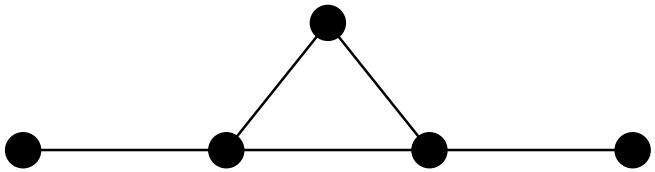
Examples of modules

- connected components
- connected components of $\overline{G}$
- any vertex subset of the complete graph (or the stable)

Prime graphs and degenerate graphs

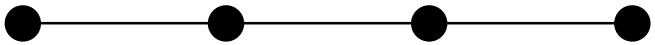
- A graph is **prime** is all its modules are trivial: e.g. the P_4.
- A graph is **degenerate** if any subset of vertices is a module: cliques and stables.

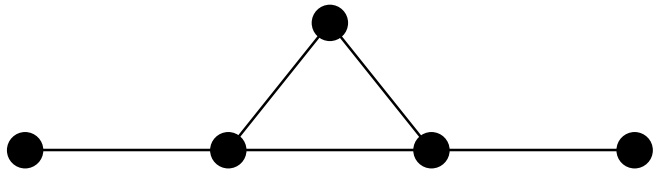




Prime graphs and degenerate graphs

- A graph is **prime** is all its modules are trivial: e.g. the P_4.
- A graph is **degenerate** if any subset of vertices is a module: cliques and stables.



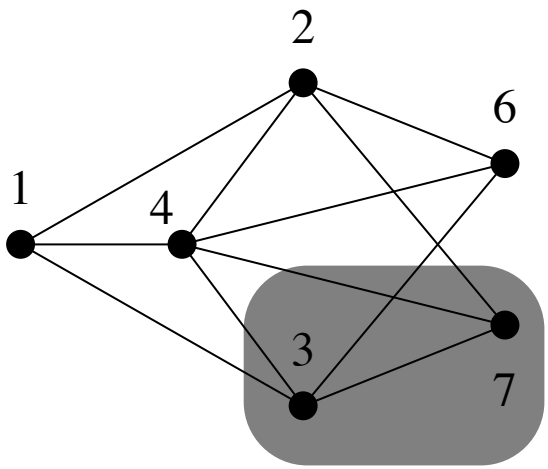


Lemma (Cournier, Ille 91)

If G is a prime graph, then any vertex but at most one belongs to a P_4.

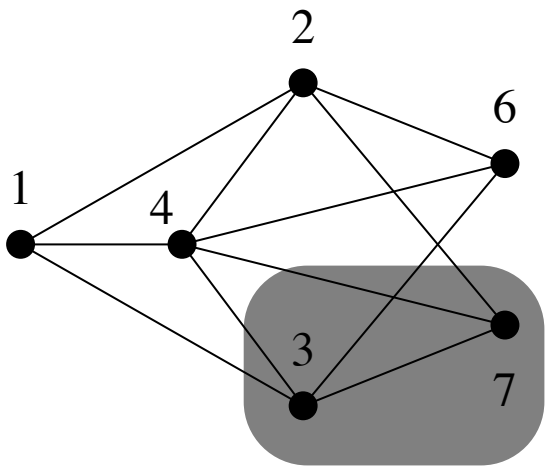
Splitter

For a graph $G = (V, E)$, the vertex x is a **splitter** for a set S of vertices if $\exists y, z \in S$ with $xy \in S$ and $xz \notin E$. We say that x **separate** y and z.



Splitter

For a graph $G = (V, E)$, the vertex x is a **splitter** for a set S of vertices if $\exists y, z \in S$ with $xy \in S$ and $xz \notin E$. We say that x **separate** y and z.

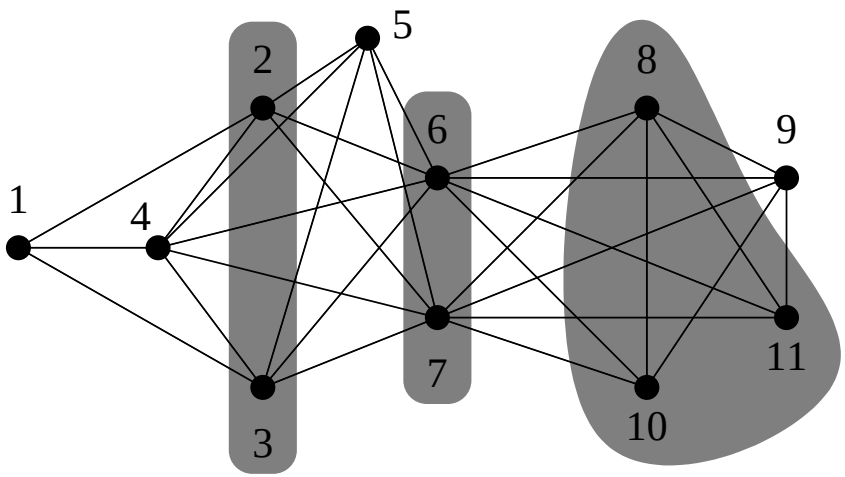


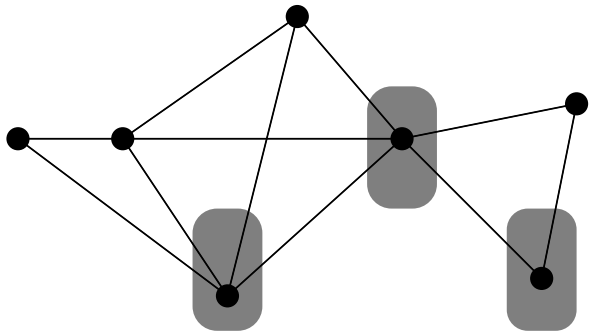
Lemma

If x is a splitter for the set S, then any module M containing S must also contain x.

Modular partition

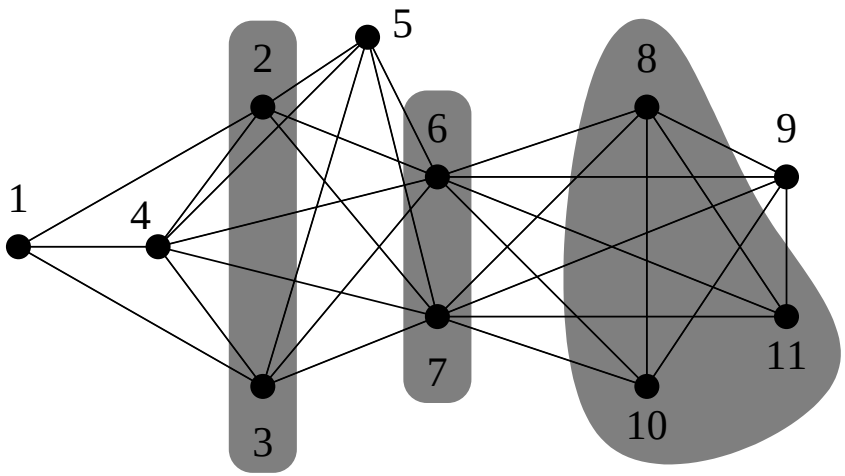
A partition $\mathcal{P}$ of the vertex set of a graph $G = (V, E)$ is a **modular partition** of G if any part is a module of G.

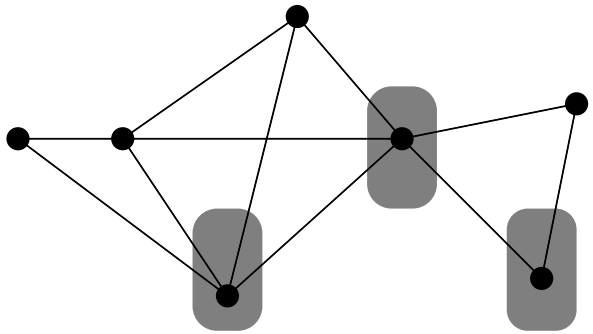




Modular partition

A partition $\mathcal{P}$ of the vertex set of a graph $G = (V, E)$ is a **modular partition** of G if any part is a module of G.

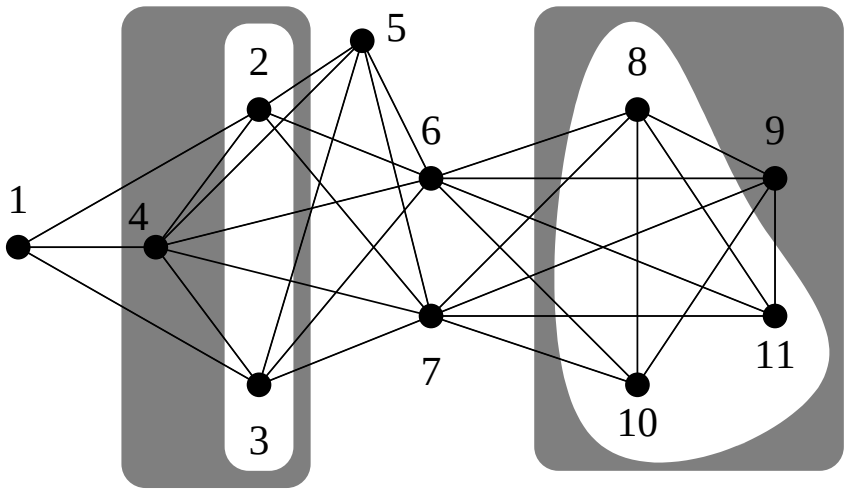


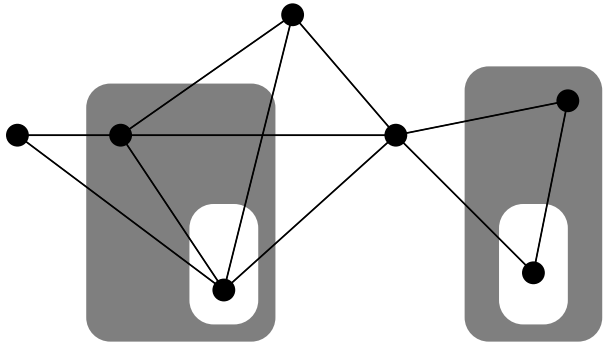


Let $\mathcal{P}$ be a modular partition of a graph $G = (V, E)$. The **quotient graph** $G_{/\mathcal{P}}$ is the induced subgraph obtained by choosing one vertex per part of $\mathcal{P}$.

Lemma (MR84)

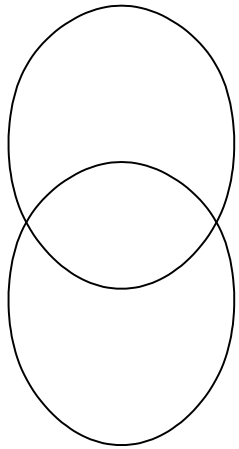
Let $\mathcal{P}$ be a modular partition of $G = (V, E)$.
$\mathcal{X} \subseteq \mathcal{P}$ is a module of $G_{/\mathcal{P}}$ iff $\cup_{M \in \mathcal{X}} M$ is a module of G.





Lemma

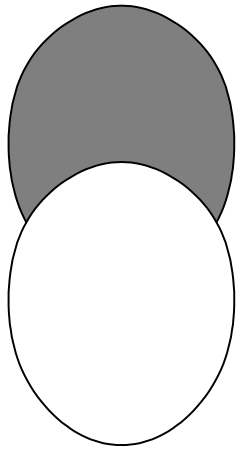
If M and M' are two overlapping modules then



Lemma

If M and M' are two overlapping modules then

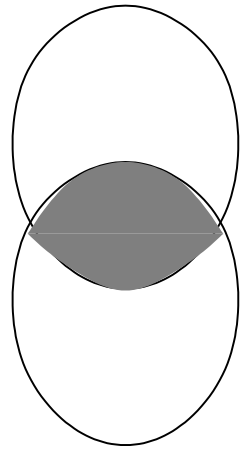
- *$M \setminus M'$ is a module*



Lemma

If M and M' are two overlapping modules then

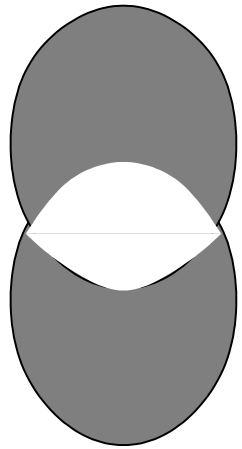
- *$M \setminus M'$ is a module*
- *$M \cap M'$ is a module*



Lemma

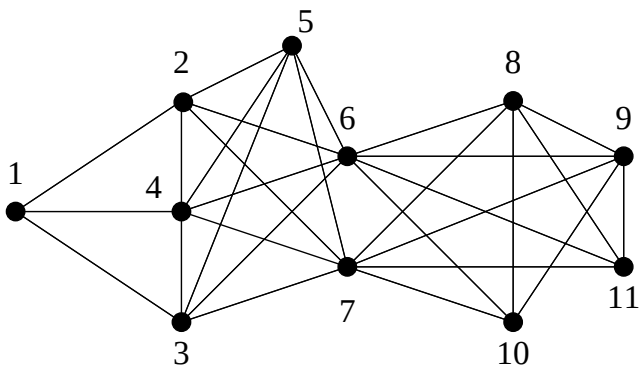
If M and M' are two overlapping modules then

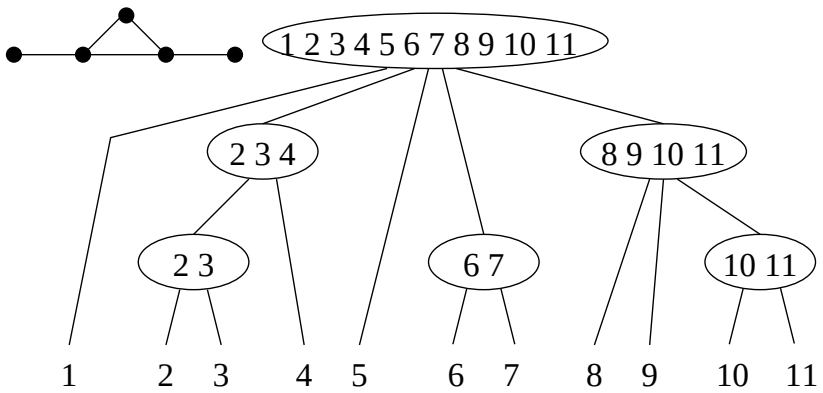
- *$M \setminus M'$ is a module*
- *$M \cap M'$ is a module*
- *$M \Delta M'$ is a module*



- The set of modules of a graph defines a **partitive family**
- A module is **strong** if it does not overlap any other module

The set of strong modules is nested into an inclusion tree (called the **modular decomposition tree** $MD(G)$ of G).





Theorem (Gal'67,CHM81)

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

1. *G is not connected, or*
2. *$\overline{G}$ is not connected, or*
3. *$G_{/\mathcal{M}(G)}$ is a prime graph, with $\mathcal{M}(G)$ the modular partition containing the maximal strong modules of G.*

$\Rightarrow$ "Brut force" algorithm in $O(n^3)$ time

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$
- Blass (1978) $O(n^3)$, Habib, Maurer (1979) $O(n^3)$

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$
- Blass (1978) $O(n^3)$, Habib, Maurer (1979) $O(n^3)$
- Corneil, Perl, Stewart (1981) $O(n+m)$ cograph recognition.

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$
- Blass (1978) $O(n^3)$, Habib, Maurer (1979) $O(n^3)$
- Corneil, Perl, Stewart (1981) $O(n+m)$ cograph recognition.
- McConnell, Spinrad (1989) $O(n^2)$ incremental

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$
- Blass (1978) $O(n^3)$, Habib, Maurer (1979) $O(n^3)$
- Corneil, Perl, Stewart (1981) $O(n+m)$ cograph recognition.
- McConnell, Spinrad (1989) $O(n^2)$ incremental
- Spinrad 1992] $O(n+m\alpha(m,n))$, Cournier, Habib (1993) $O(n+m\alpha(m,n))$

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$
- Blass (1978) $O(n^3)$, Habib, Maurer (1979) $O(n^3)$
- Corneil, Perl, Stewart (1981) $O(n+m)$ cograph recognition.
- McConnell, Spinrad (1989) $O(n^2)$ incremental
- Spinrad 1992] $O(n+m\alpha(m,n))$, Cournier, Habib (1993) $O(n+m\alpha(m,n))$
- McConnell, Spinrad (1994) $O(n+m)$, Cournier, Habib (1994) $O(n+m)$

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$
- Blass (1978) $O(n^3)$, Habib, Maurer (1979) $O(n^3)$
- Corneil, Perl, Stewart (1981) $O(n+m)$ cograph recognition.
- McConnell, Spinrad (1989) $O(n^2)$ incremental
- Spinrad 1992] $O(n+m\alpha(m,n))$, Cournier, Habib (1993) $O(n+m\alpha(m,n))$
- McConnell, Spinrad (1994) $O(n+m)$, Cournier, Habib (1994) $O(n+m)$
- Capelle, Habib (1997) $O(n+m)$ if a factoring permutation is given

Some algorithms

- Cowan, James, Stanton (1972) $O(n^4)$
- Blass (1978) $O(n^3)$, Habib, Maurer (1979) $O(n^3)$
- Corneil, Perl, Stewart (1981) $O(n+m)$ cograph recognition.
- McConnell, Spinrad (1989) $O(n^2)$ incremental
- Spinrad 1992] $O(n+m\alpha(m,n))$, Cournier, Habib (1993) $O(n+m\alpha(m,n))$
- McConnell, Spinrad (1994) $O(n+m)$, Cournier, Habib (1994) $O(n+m)$
- Capelle, Habib (1997) $O(n+m)$ if a factoring permutation is given
- Dahlhaus, Gustedt, McConnell (1997) $O(n+m)$

Some algorithms

- Habib, Paul, Viennot (1999) $O(n + mlogn)$ computes a factoring permutation, McConnell, Spinrad (2000) $O(n + mlogn)$

Some algorithms

- Habib, Paul, Viennot (1999) $O(n + mlogn)$ computes a factoring permutation, McConnell, Spinrad (2000) $O(n + mlogn)$
- Habib, Paul (2001) $O(n + m)$ cographs via a factoring permutation, Capelle, Habib, Montgolfier (2002) $O(n + m)$ if a factoring permutation is provided.

Some algorithms

- Habib, Paul, Viennot (1999) $O(n + mlogn)$ computes a factoring permutation, McConnell, Spinrad (2000) $O(n + mlogn)$
- Habib, Paul (2001) $O(n + m)$ cographs via a factoring permutation, Capelle, Habib, Montgolfier (2002) $O(n + m)$ if a factoring permutation is provided.
- Brestcher, Corneil, Habib, Paul (2006) $O(n + m)$ Cographs via LexBFS.

Some algorithms

- Habib, Paul, Viennot (1999) $O(n + mlogn)$ computes a factoring permutation, McConnell, Spinrad (2000) $O(n + mlogn)$
- Habib, Paul (2001) $O(n + m)$ cographs via a factoring permutation, Capelle, Habib, Montgolfier (2002) $O(n + m)$ if a factoring permutation is provided.
- Brestcher, Corneil, Habib, Paul (2006) $O(n + m)$ Cographs via LexBFS.
- ...

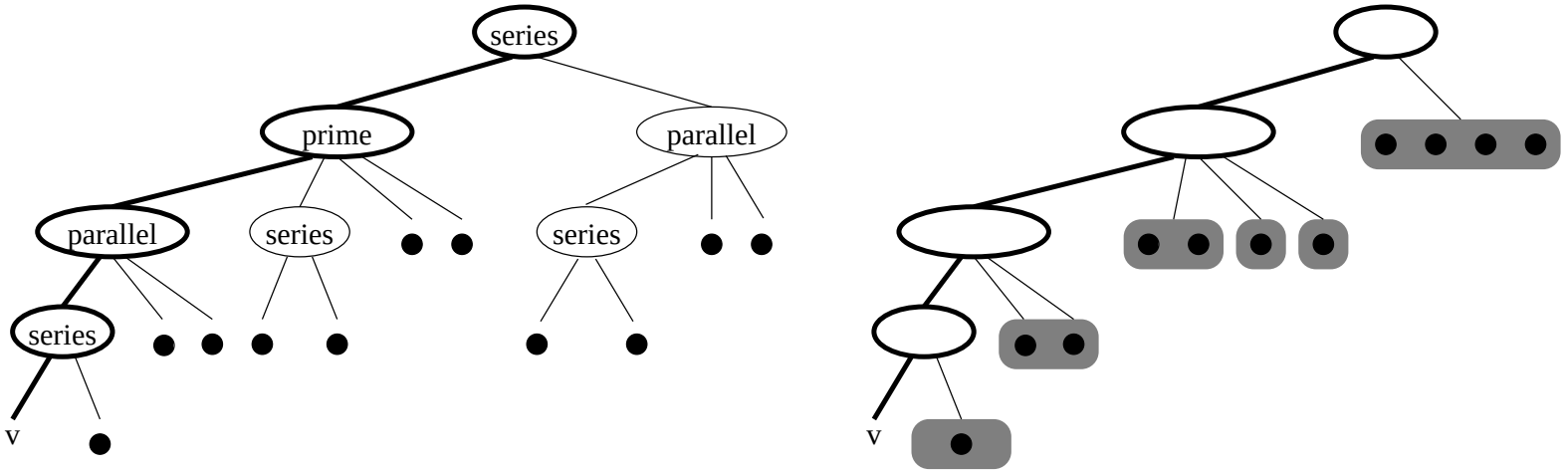
That was only for undirected graphs!!!

1 Definitions and preliminaries

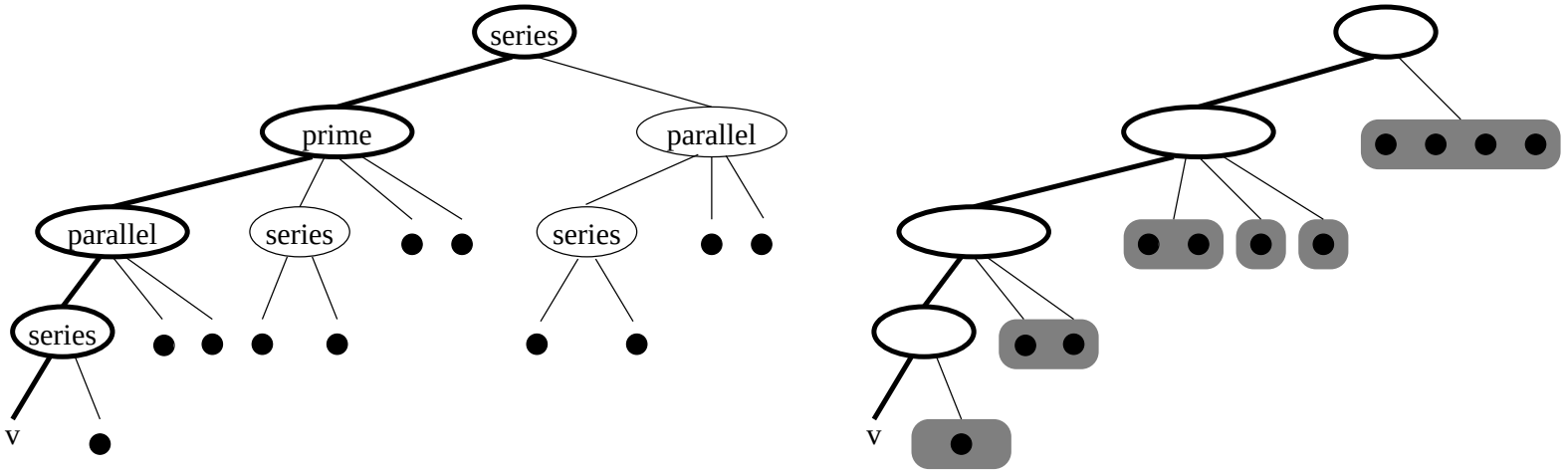
2 Ehrenfeucht et al.'s algorithm
- Principle
- Computation of $\mathcal{M}(G, v)$
- Computation of $MD(G_{/\mathcal{M}(G,v)})$

3 New algorithmic trends
- Factoring permutation
- LexBFS

$\mathcal{M}(G, v)$ is composed by $\{v\}$ and the maximal modules of G not containing v.



$\mathcal{M}(G, v)$ is composed by $\{v\}$ and the maximal modules of G not containing v.



Principle of the Ehrenfeucht et al.'s algorithm

1. Compute $\mathcal{M}(G, v)$
2. Compute $MD(G_{/\mathcal{M}(G,v)})$
3. For each $\mathcal{X} \in \mathcal{M}(G, v)$ compute $MD(G[\mathcal{X}])$

Lemma

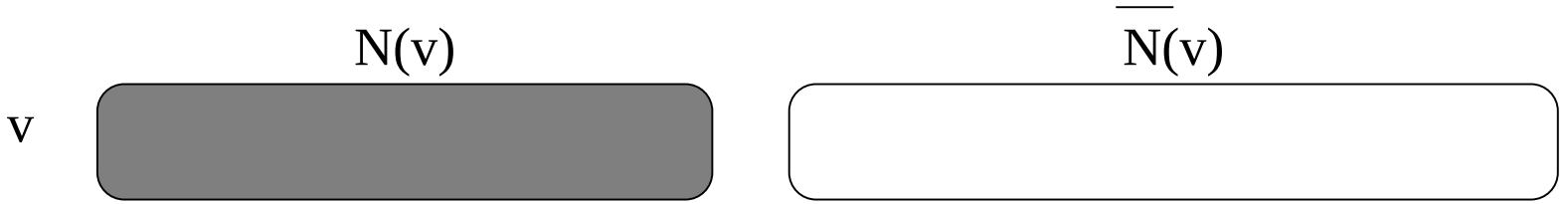
If x is a splitter of a set S, then for any module M contained in S either $M \subseteq S \cap N(x)$ or $M \subseteq M \cap \overline{N}(x)$

Lemma

If x is a splitter of a set S, then for any module M contained in S either $M \subseteq S \cap N(x)$ or $M \subseteq M \cap \overline{N}(x)$

Computation of $\mathcal{M}(G, v)$

$\Rightarrow O(n + m \log n)$ time using vertex partitioning algorihtm.

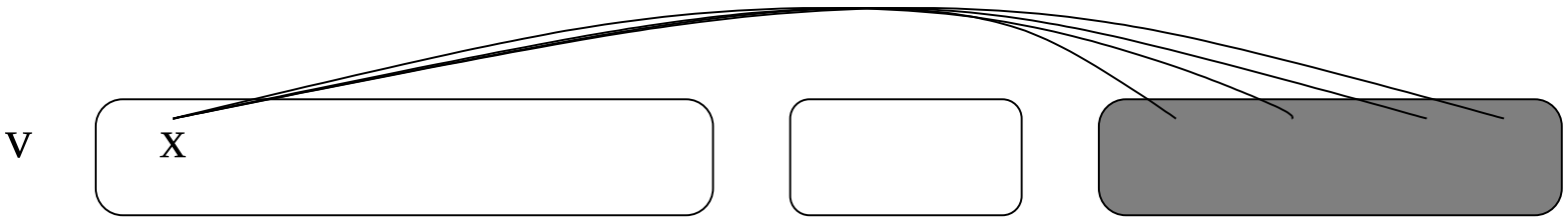


Lemma

If x is a splitter of a set S, then for any module M contained in S either $M \subseteq S \cap N(x)$ or $M \subseteq M \cap \overline{N}(x)$

Computation of $\mathcal{M}(G, v)$

$\Rightarrow O(n + m \log n)$ time using vertex partitioning algorihtm.

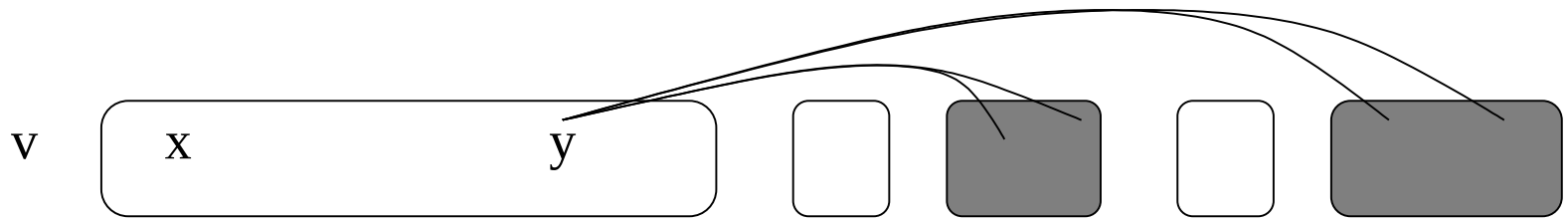


Lemma

If x is a splitter of a set S, then for any module M contained in S either $M \subseteq S \cap N(x)$ or $M \subseteq M \cap \overline{N}(x)$

Computation of $\mathcal{M}(G, v)$

$\Rightarrow O(n + m \log n)$ time using vertex partitioning algorihtm.

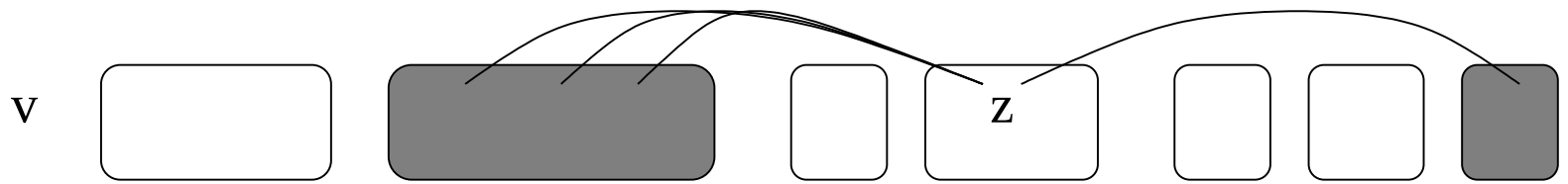


Lemma

If x is a splitter of a set S, then for any module M contained in S either $M \subseteq S \cap N(x)$ or $M \subseteq M \cap \overline{N}(x)$

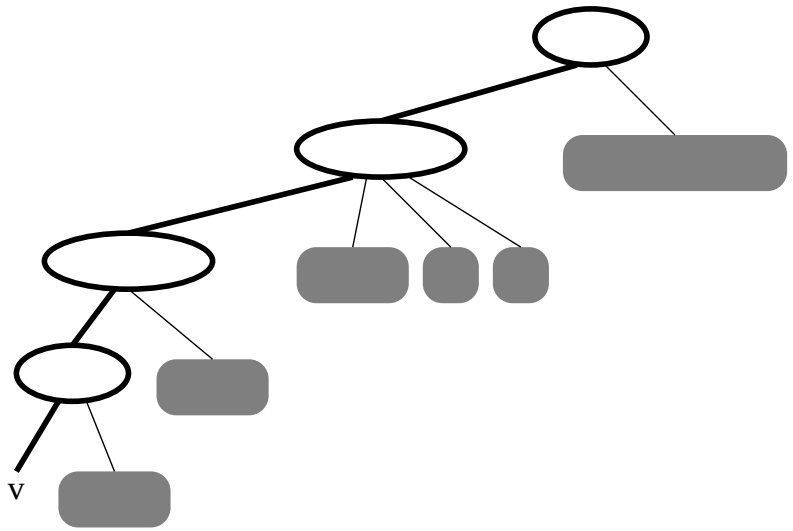
Computation of $\mathcal{M}(G, v)$

$\Rightarrow O(n + m \log n)$ time using vertex partitioning algorihtm.



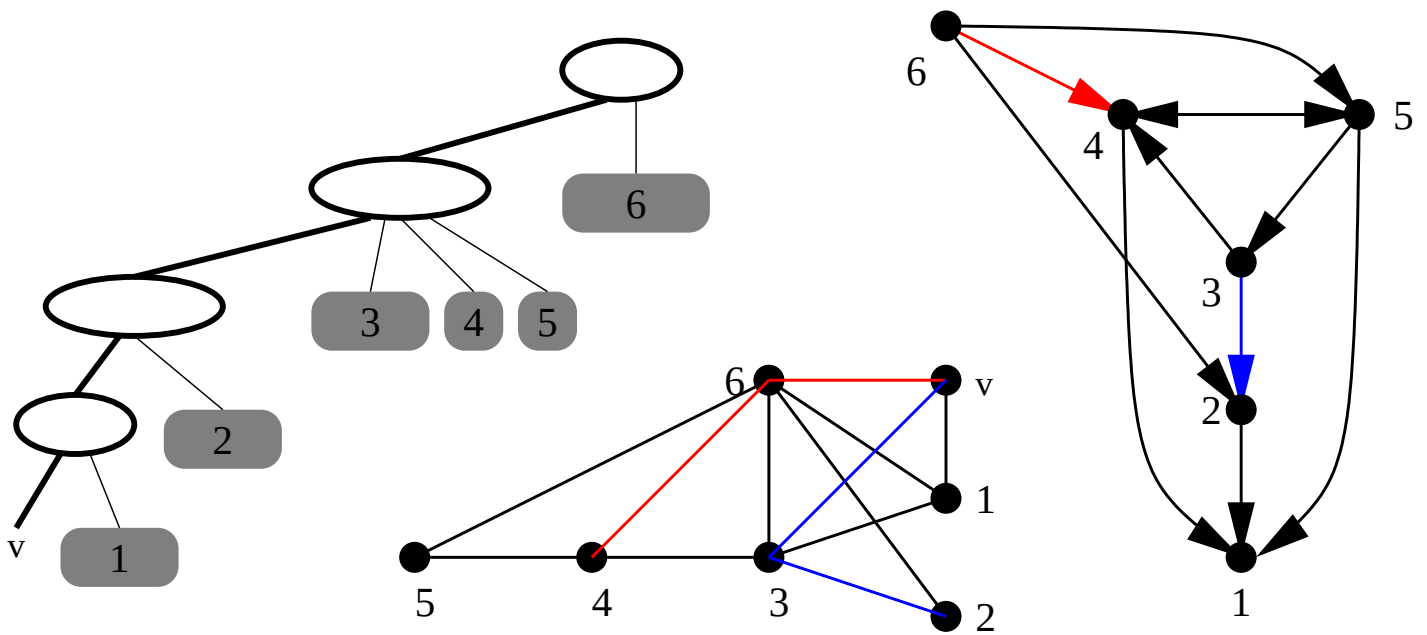
Computation of $MD(G_{/\mathcal{M}(G,v)})$

- The modules of $G_{/\mathcal{M}(G,v)}$ are linearly nested:
  any non-trivial module contains v



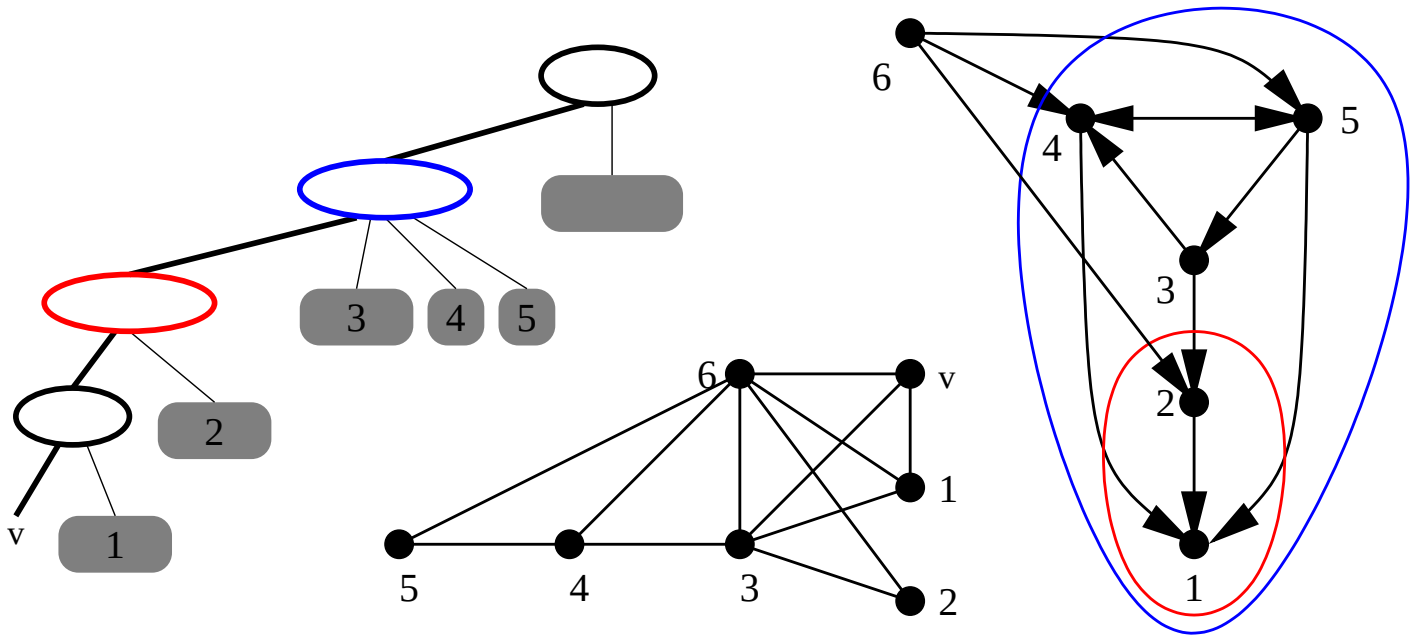
Computation of $MD(G_{/\mathcal{M}(G,v)})$

- The modules of $G_{/\mathcal{M}(G,v)}$ are linearly nested:
  any non-trivial module contains v
- The *forcing graph* $\mathcal{F}(G, v)$ has edge $\overrightarrow{xy}$ iff y separates x and v



Computation of $MD(G_{/\mathcal{M}(G,v)})$

- The modules of $G_{/\mathcal{M}(G,v)}$ are linearly nested:
  any non-trivial module contains v
- The *forcing graph* $\mathcal{F}(G, v)$ has edge $\overrightarrow{xy}$ iff y separates x and v



Complexity

- [Ehrenfeucht et al.'94] gives a $O(n^2)$ complexity.
- [MS00] gives a very simple $O(n + m \log n)$ algorithm based on vertex partitioning.
- [DGM'01] proposes a $O(n + m.\alpha(n, m))$ and a more complicated $O(n + m)$ implementation.

Other algorithms

- [CH94] and [MS94] present the first linear algorithms.
- [MS99] present a new linear time algorithm which extend to transitive orientation.

Still open

Find a simple linear time modular decomposition algorithm.

[Spinrad'03]

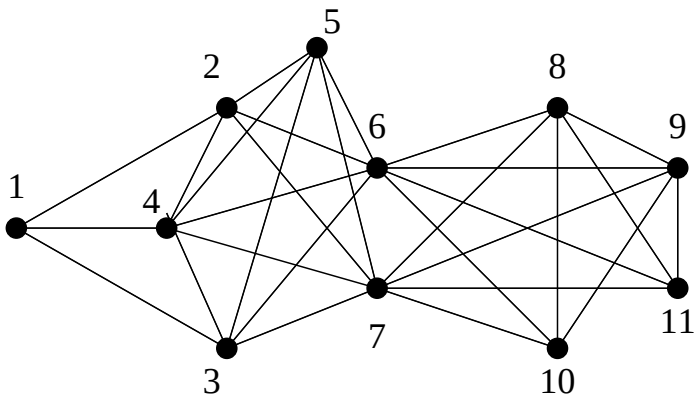
The new [linear time] algorithm [MS99] is currently too complex to describe easily [...] The first $O(n^2)$ partitioning algorithms were similarly complex; I hope and believe that in a number of years the linear algorithm can be simplified as well.

1 Definitions and preliminaries

2 Ehrenfeucht et al.'s algorithm
- Principle
- Computation of $\mathcal{M}(G, v)$
- Computation of $MD(G_{/\mathcal{M}(G,v)})$

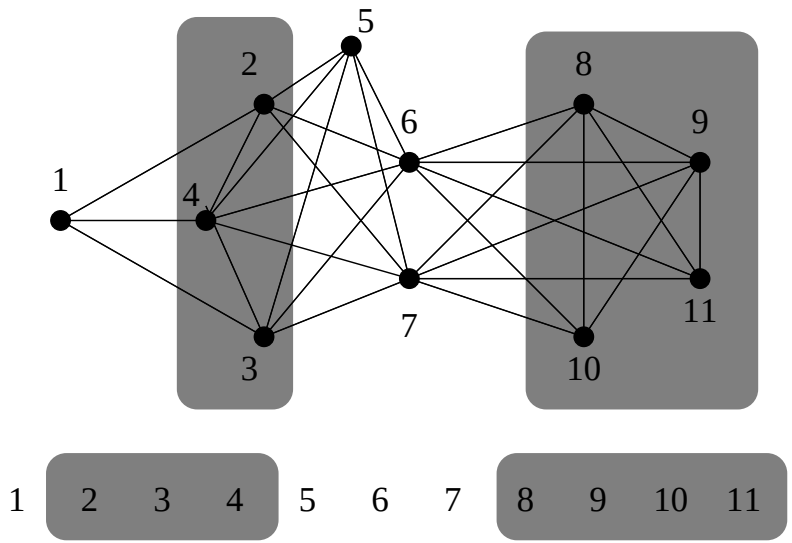
3 New algorithmic trends
- Factoring permutation
- LexBFS

A **factoring pemutation** of a graph $G = (V, E)$ is a permutation of V in which any strong module of G is a factor. [Capelle 97]

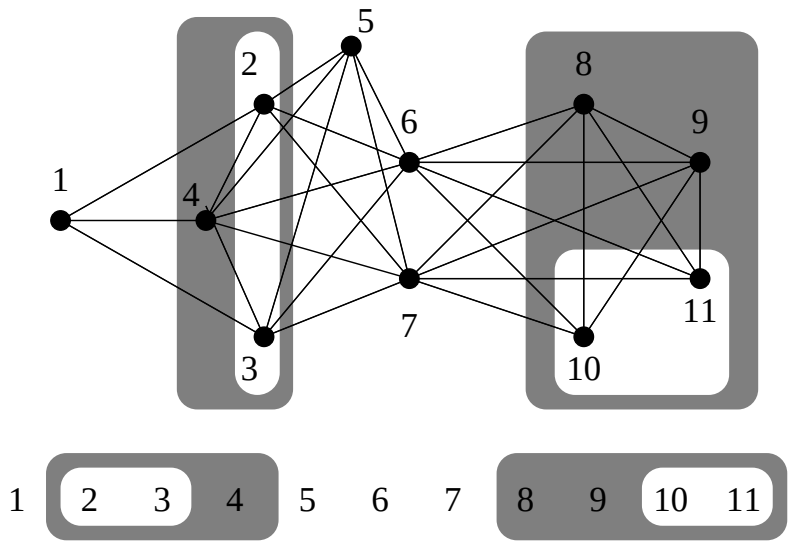


1 2 3 4 5 6 7 8 9 10 11

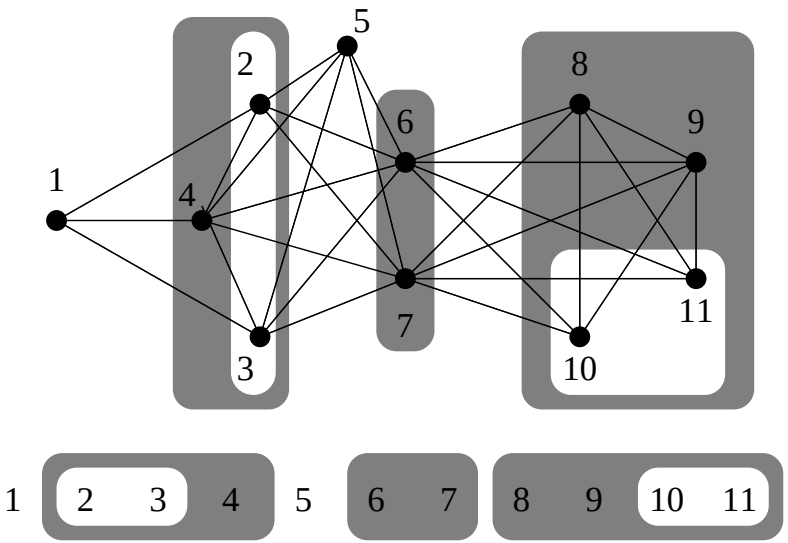
A **factoring pemutation** of a graph $G = (V, E)$ is a permutation of V in which any strong module of G is a factor. [Capelle 97]



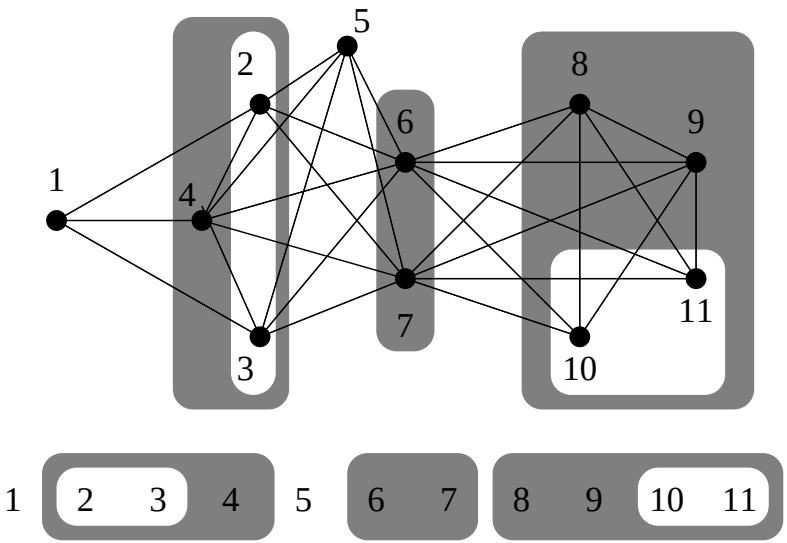
A **factoring pemutation** of a graph $G = (V, E)$ is a permutation of V in which any strong module of G is a factor. [Capelle 97]



A **factoring pemutation** of a graph $G = (V, E)$ is a permutation of V in which any strong module of G is a factor. [Capelle 97]

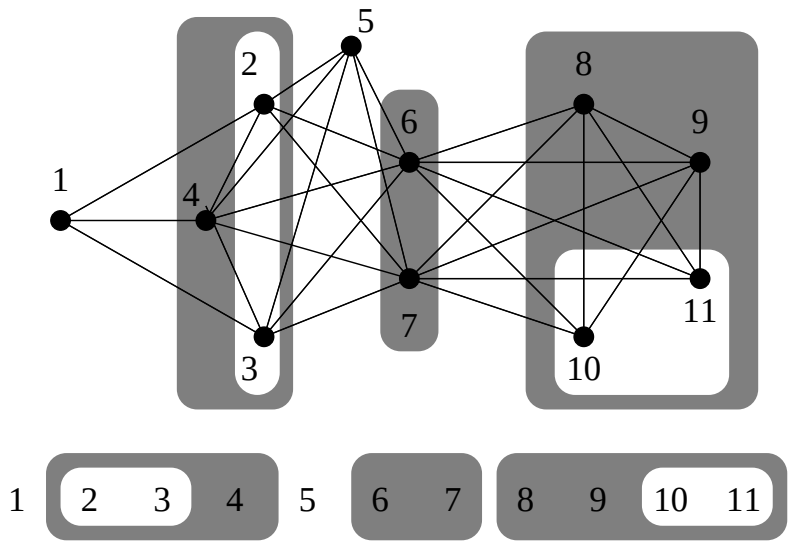


A **factoring pemutation** of a graph $G = (V, E)$ is a permutation of V in which any strong module of G is a factor. [Capelle 97]



- From G to factoring permutation: $O(n + m \log n)$ [HPV99]

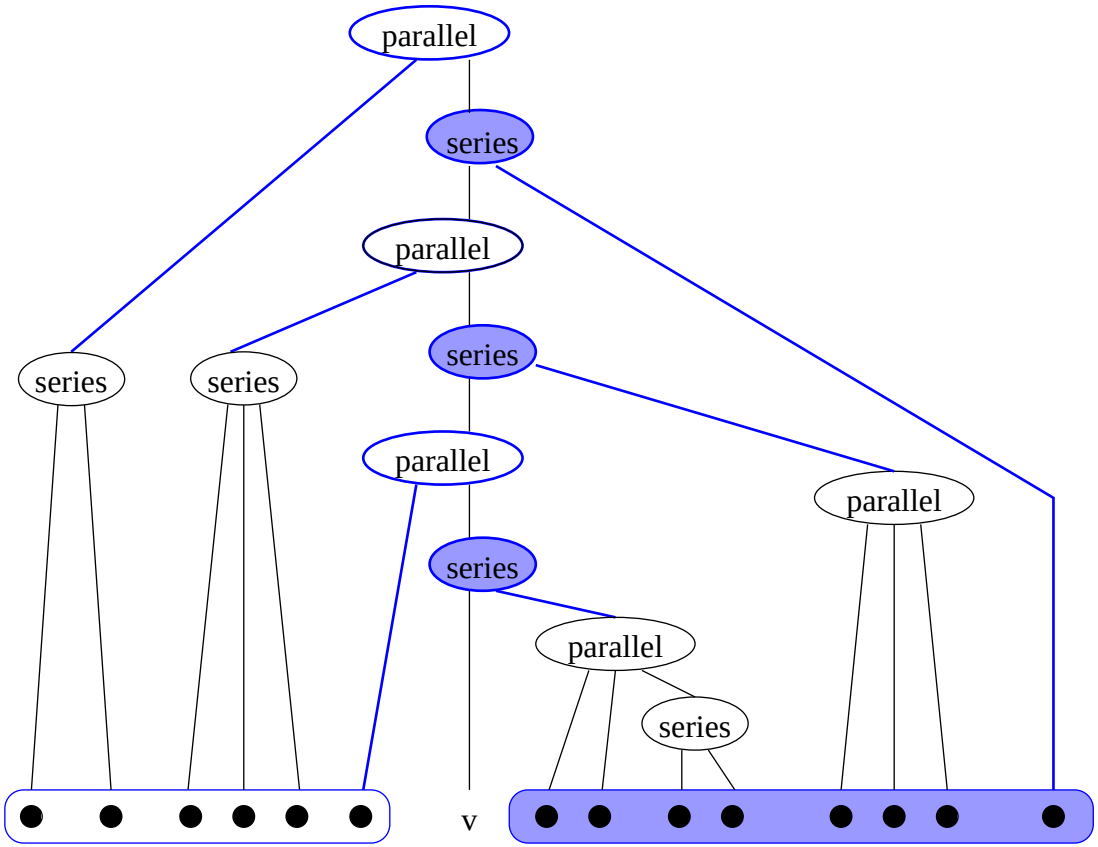
A **factoring pemutation** of a graph $G = (V, E)$ is a permutation of V in which any strong module of G is a factor. [Capelle 97]



- From G to factoring permutation: $O(n + m \log n)$ [HPV99]
- From factoring permutation to $MD(G)$: $O(n + m)$ [CdMH01] [UY00] [BXHP05]

Factoring permutation (of cographs) via vertex partitionning

At each step, the partition can be refined into a factoring permutation.

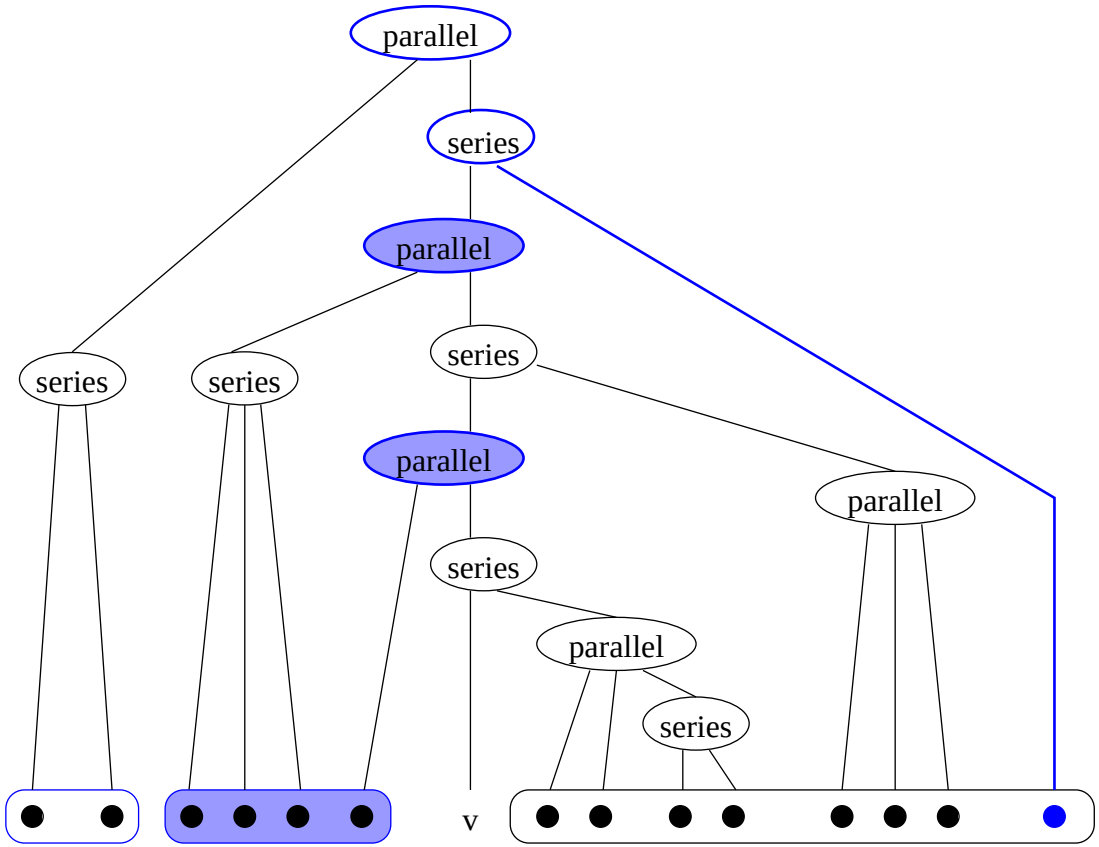


Invariant

Any strong module is a factor of the partition.

Factoring permutation (of cographs) via vertex partitionning

At each step, the partition can be refined into a factoring permutation.

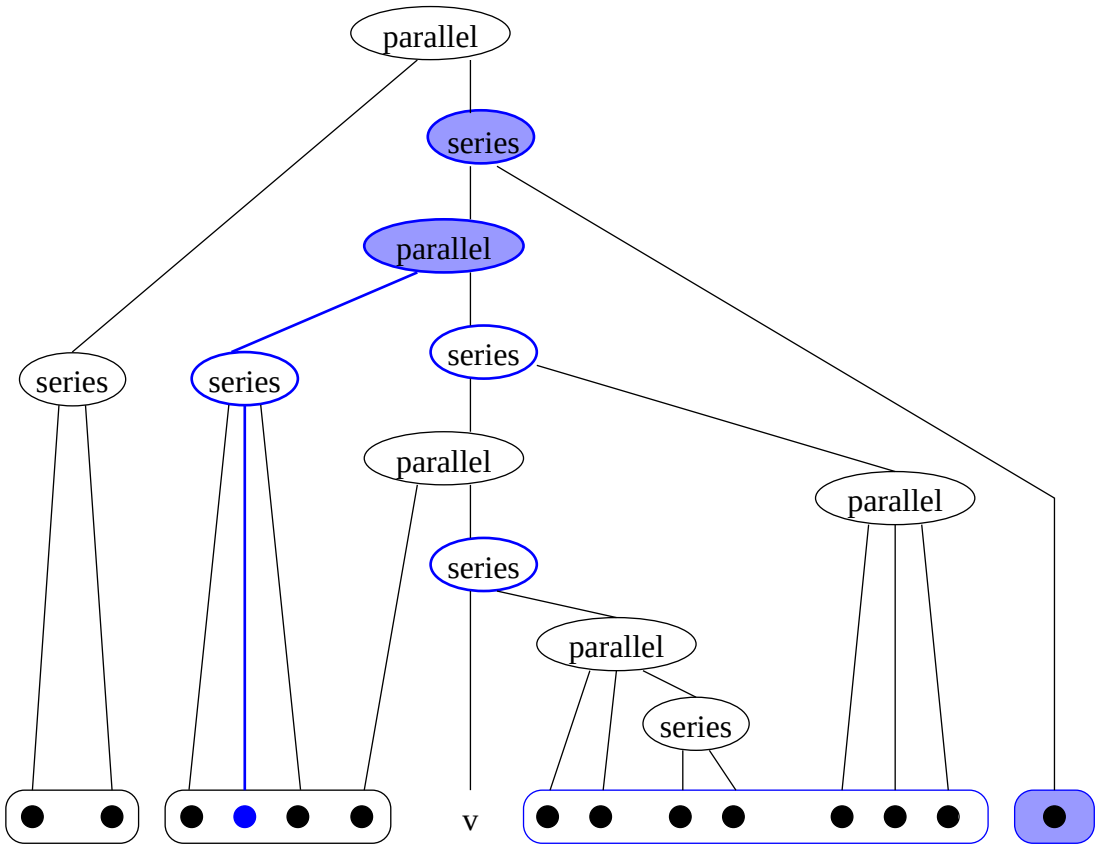


Invariant

Any strong module is a factor of the partition.

Factoring permutation (of cographs) via vertex partitionning

At each step, the partition can be refined into a factoring permutation.

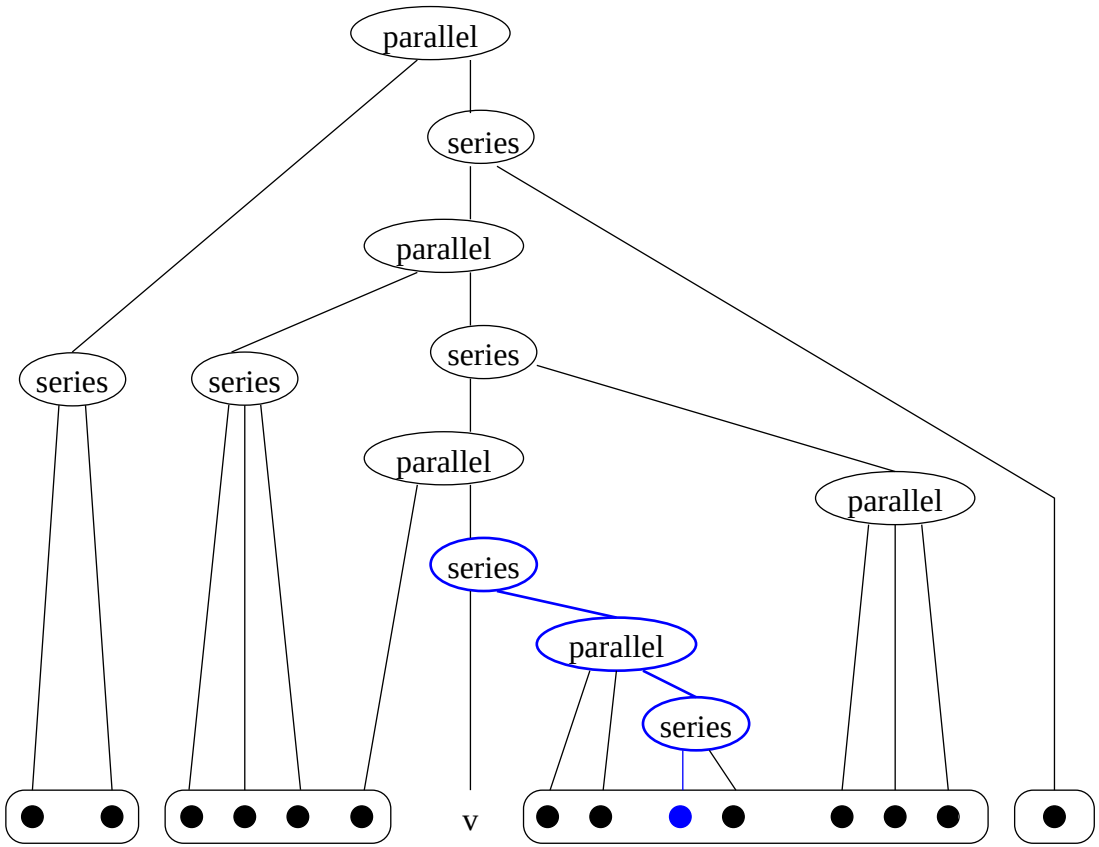


Invariant

Any strong module is a factor of the partition.

Factoring permutation (of cographs) via vertex partitionning

At each step, the partition can be refined into a factoring permutation.

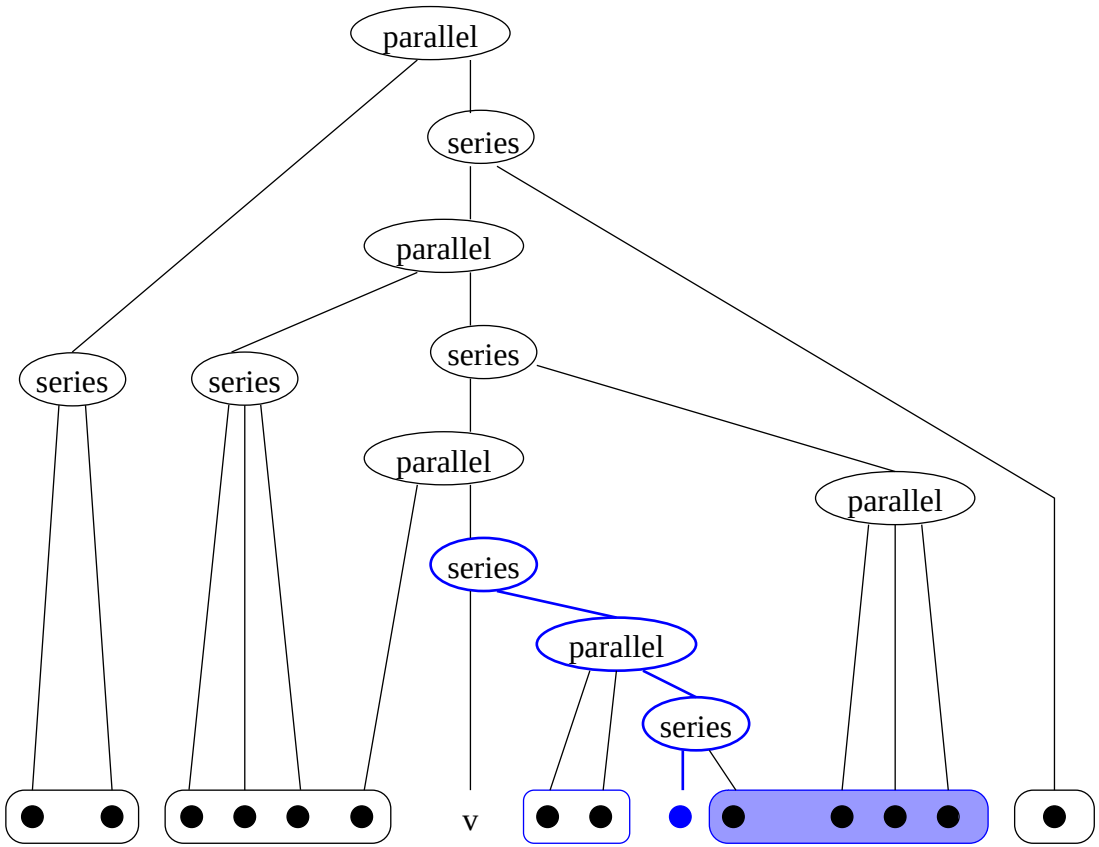


Invariant

Any strong module is a factor of the partition.

Factoring permutation (of cographs) via vertex partitionning

At each step, the partition can be refined into a factoring permutation.



Invariant

Any strong module is a factor of the partition.

1. Best known complexity: $O(n + m \log n)$ and $O(n + m)$ for cographs

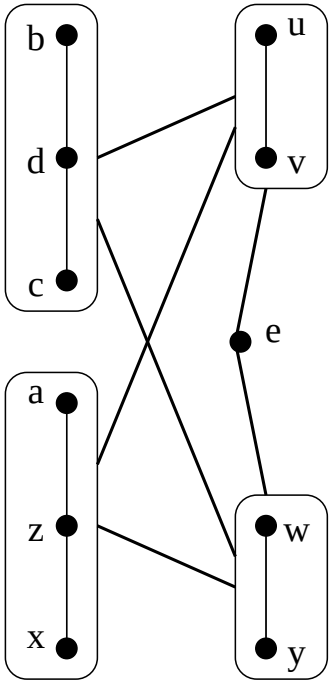
1. Best known complexity: $O(n + m \log n)$ and $O(n + m)$ for cographs
2. Recent algorithmic results around this notion [UY00] [BXHP05] [BCdMR05]

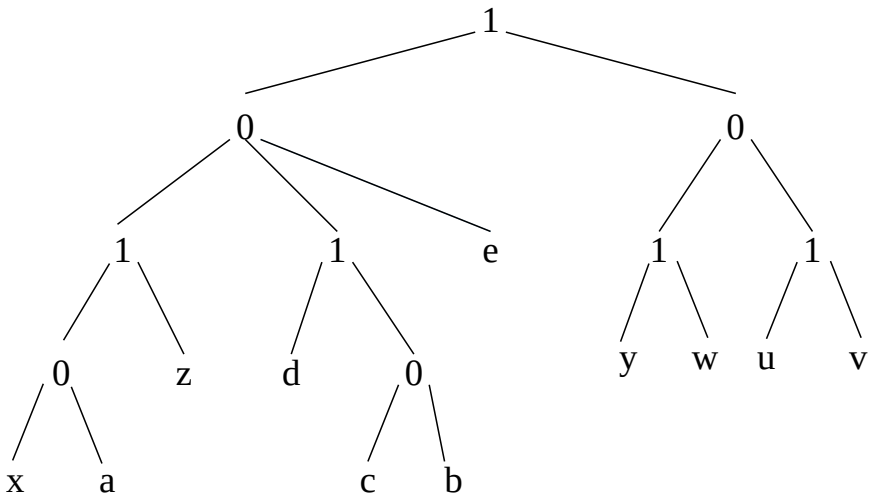
1. Best known complexity: $O(n + m \log n)$ and $O(n + m)$ for cographs
2. Recent algorithmic results around this notion [UY00] [BXHP05] [BCdMR05]
3. For some graph families, computing $MD(G)$ from a factoring permutation costs $O(n)$ time

1. Best known complexity: $O(n + m \log n)$ and $O(n + m)$ for cographs
2. Recent algorithmic results around this notion [UY00] [BXHP05] [BCdMR05]
3. For some graph families, computing $MD(G)$ from a factoring permutation costs $O(n)$ time
4. Generalization for other decomposition like for example the split decomposition ?...

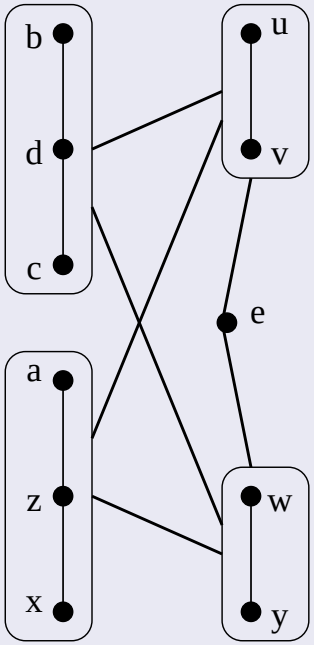
A cograph recognition algorithm [BCHP03]

1. $\sigma \leftarrow \text{LexBFS}(G)$
2. $\overline{\sigma} \leftarrow \text{LexBFS}^{-}(\overline{G}, \sigma)$
3. **If** σ and $\overline{\sigma}$ both have the NS-property **then**
   1. Answer "G is a cograph"
   2. Build $MD(G)$
4. **Else** Output a P_4


b
d
c
u
v
e
a
z
x
w
y

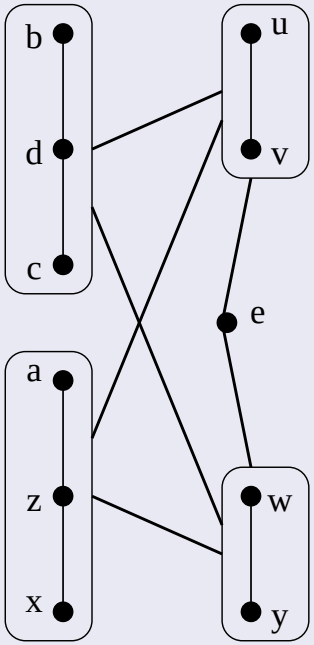

1
0
0
1
1
e
1
1
0
z
d
0
y
w
u
v
x
a
c
b

Computing a LexBFS ordering σ



x | y w z u v | a d c b e

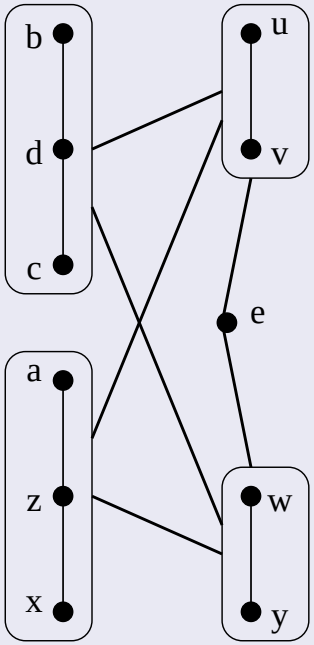
Computing a LexBFS ordering σ



x | y w z u v | a d c b e

x y | w z | u v | a d c b e

Computing a LexBFS ordering σ

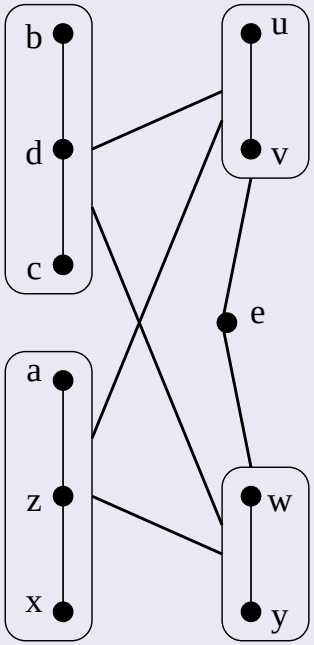


x | y w z u v | a d c b e

x | y | w z | u v | a d c b e

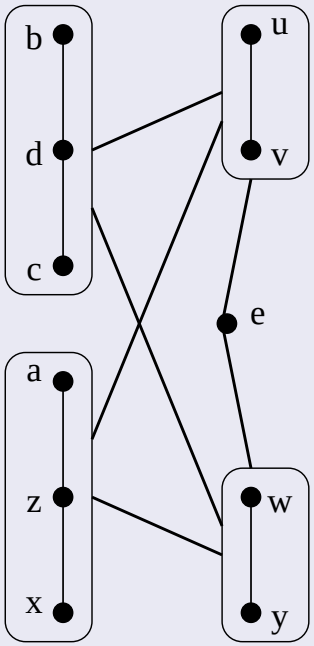
x | y | w | z | u v | a d c b e

Computing a LexBFS ordering σ



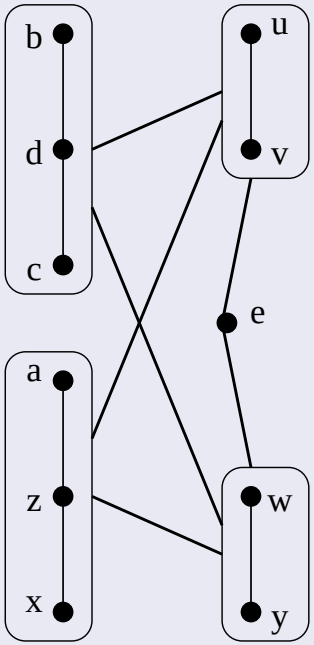
x	y	w	z	u	v	a	d	c	b	e
x	y	w	z	u	v	a	d	c	b	e
x	y	w	z	u	v	a	d	c	b	e
x	y	w	z	u	v	a	d	c	b	e

Computing a LexBFS ordering σ



x	y	w	z	u	v	a	d	c	b	e
x	y	w	z	u	v	a	d	c	b	e
x	y	w	z	u	v	a	d	c	b	e
x	y	w	z	u	v	a	d	c	b	e
x	y	w	z	u	v	a	d	c	b	e

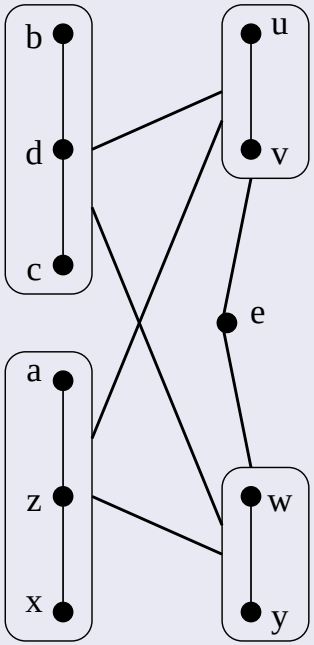
Computing $\overline{\sigma}$=LexBFS$^{-}(\overline{G}, \sigma)$



x | y w z u v | a | d c b e

x | a d c b e | y w z u v

Computing $\overline{\sigma}$=LexBFS$^{-}(\overline{G}, \sigma)$

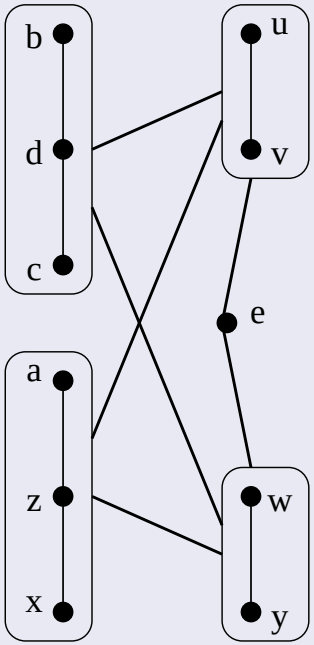


x | y w z u v | a d c b e

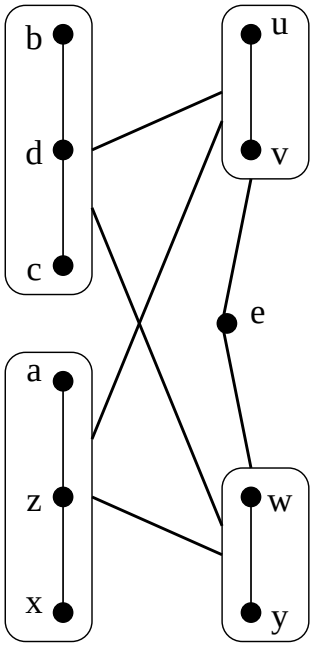
x | a d c b e | y w z u v

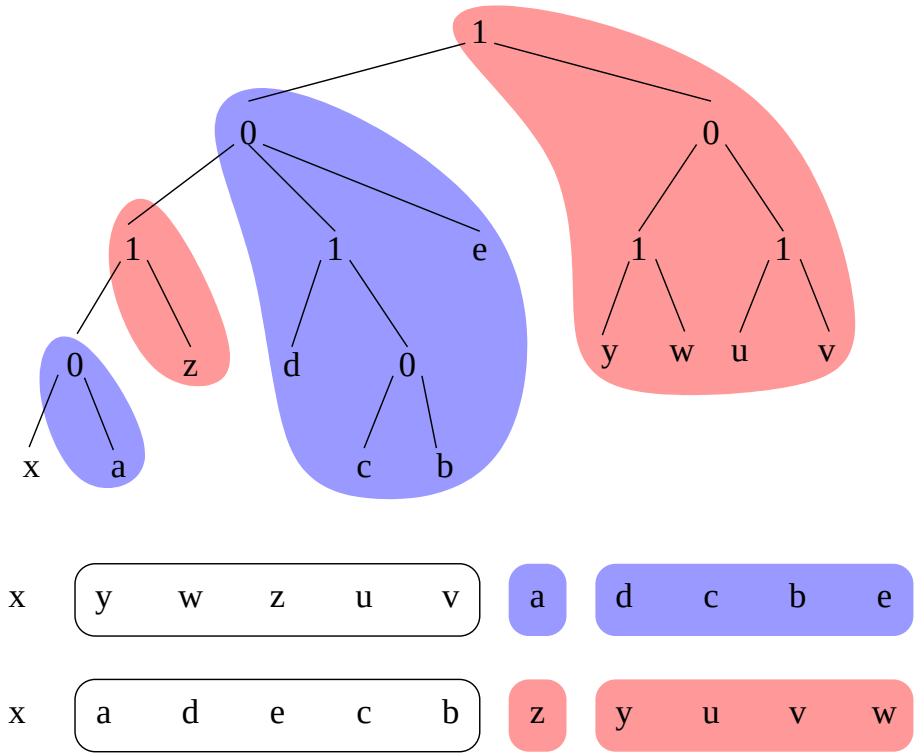
x a | d c b e | y w z u v

Computing $\overline{\sigma}$=LexBFS$^{-}(\overline{G}, \sigma)$



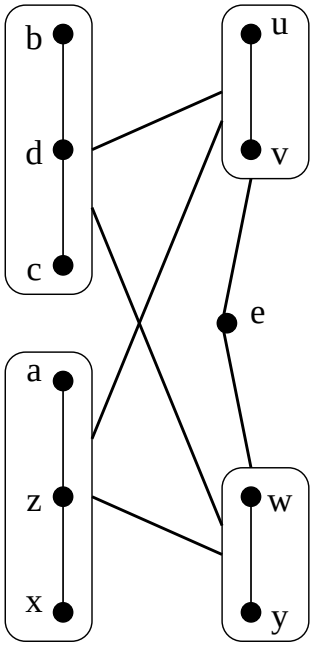
x	y	w	z	u	v	a	d	c	b	e
x	a	d	c	b	e	y	w	z	u	v
x	a	d	c	b	e	y	w	z	u	v
x	a	d	e	c	b	z	y	w	u	v

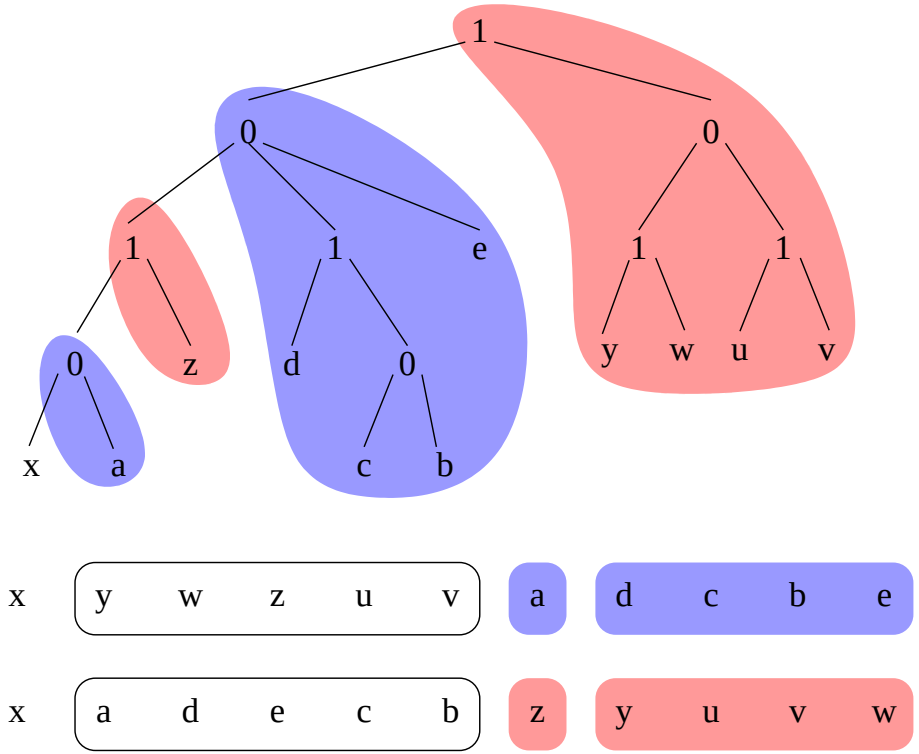




Lemma

$\mathcal{M}(G, x)$ is composed by the slices $S_i(x)$ of σ and $\overline{S_j}(x)$ of $\overline{\sigma}$.





Theorem

If G is a cograph, then $MD(G)$ can be retrieved from the slice structure of σ and $\overline{\sigma}$.

Generalisation to arbitrary graphs ?

1. There are many similarities between two-LexBFS-sweep algorithm and the linear implementation of Ehrenfeucht et al.'s algorithm [DGM01]
2. LexBFS is useful for the transitive orientation problem. Could it lead to a simple linear time algorithm for this problem ?

Miscellaneous : other works related to modular decomposition

1. Representation of dynamic graphs (cf. C. Crespelle's PhD-thesis)
   - cographs, permutation graphs ...

Miscellaneous : other works related to modular decomposition

1. Representation of dynamic graphs (cf. C. Crespelle's PhD-thesis)
   - cographs, permutation graphs ...
2. Perfect sorting by reversals
   - efficient parametrized algorithm based on the modular decomposition tree
   - characterization of the permutations having perfect parsimonious scenarios

Thank you...