Algorithmics of Modular decomposition

Christophe Paul

CNRS - LIRMM, Montpellier France

Algorithms & Permutations Workshop
February 20, 2012

Joint work with: A. Bergeron, S. Bérard, S. Bessy, B.M. Bui Xuan, C. Chauve, D. Corneil, F. Fomin, E. Gioan, M. Habib, A. Perez, S. Saurabh, S. Thomassé, M. Tedder, L. Viennot...

Modular decomposition of undirected graphs

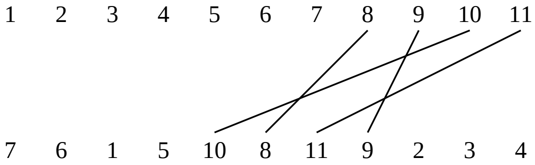
Ehrenfeucht et al's modular decomposition algorithm

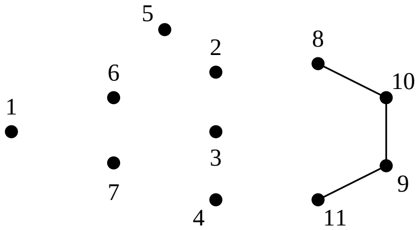
Common Intervals of permutations

Modular decomposition of tournaments

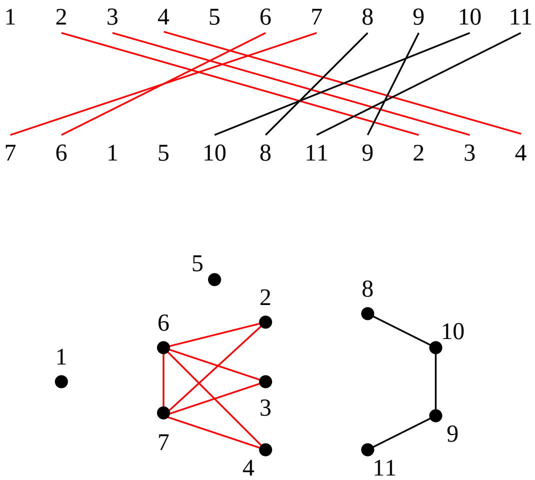
Kernelization algorithm for FAST

Permutations and permutation graphs

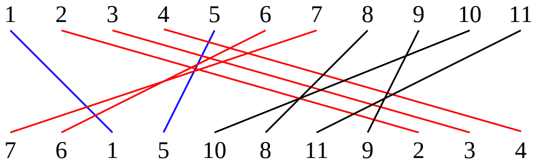


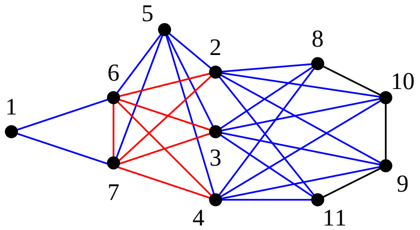


Permutations and permutation graphs



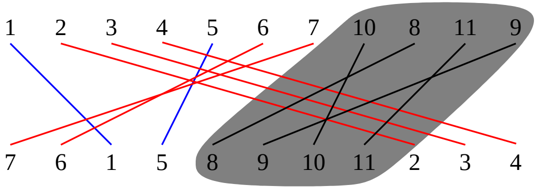
Permutations and permutation graphs

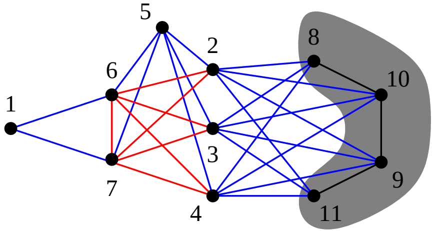




- Does a permutation graph have a unique representation ?

Permutations and permutation graphs

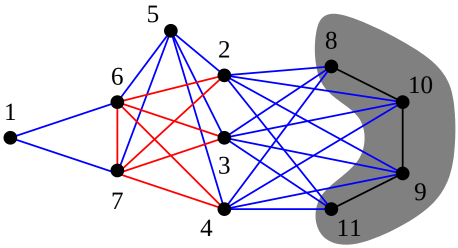




- Does a permutation graph have a unique representation ?

Modules

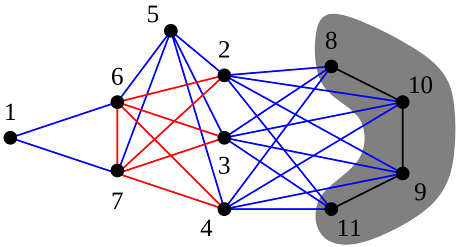
A subset of vertices M of a graph $G = (V, E)$ is a module iff
$$\forall x \in V \setminus M,\ \text{either } M \subseteq N(x) \text{ or } M \cap N(x) = \emptyset$$



Modules

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

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

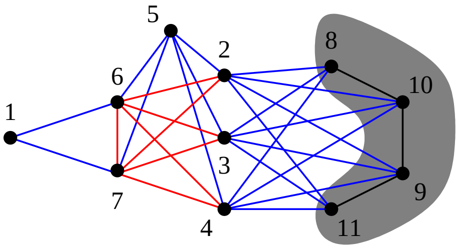


Examples of modules:

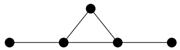
- connected components
- connected components of $\overline{G}$

Modules

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



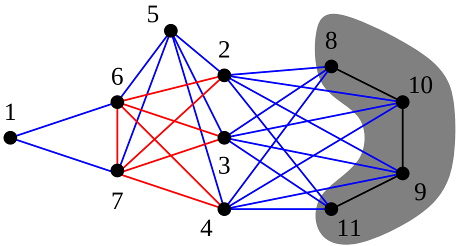
- A graph (a module) is prime is all its modules are trivial: e.g. the P_4.



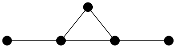
Modules

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

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



- A graph (a module) is prime is all its modules are trivial: e.g. the P_4.



- A graph (a module) is degenerate if every subset of vertices is a module: cliques and stables.

Permutation graph recognition

Theorem [Gallai'67]
A permutation graph has a unique representation (up to reversal) iff it is prime.

Permutation graph recognition

Theorem [Gallai'67]
A permutation graph has a unique representation (up to reversal) iff it is prime.

Recognition algorithm

- Recursively solve the problem on modules
- Solve the prime case (with linear time transitive orientation algorithm)

Permutation graph recognition

Theorem [Gallai'67]
A permutation graph has a unique representation (up to reversal) iff it is prime.

Recognition algorithm

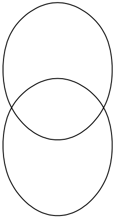
- Recursively solve the problem on modules
- Solve the prime case (with linear time transitive orientation algorithm)

Theorem [McConnell and Spinrad'99]
The permutation graph recognition problem can be solved in $O(n + m)$ time

- we need a linear time modular decomposition algorithm

Partitive families

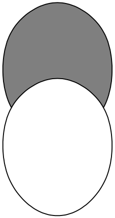
If M and M' are two overlapping modules then



Partitive families

If M and M' are two overlapping modules then

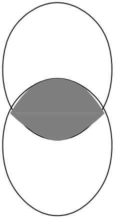
- $M \setminus M'$ is a module



Partitive families

If M and M' are two overlapping modules then

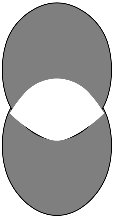
- $M \setminus M'$ is a module
- $M \cap M'$ is a module



Partitive families

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



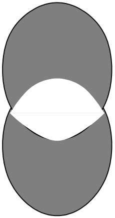
The set of modules of a graph forms a partitive family

A module is strong if it does not overlap any other module

Partitive families

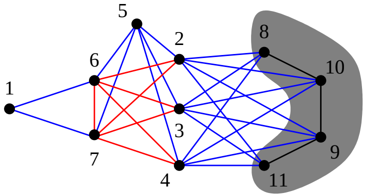
If M and M' are two overlapping modules then

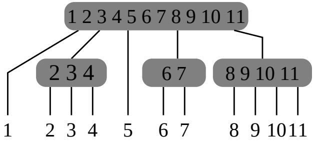
- $M \setminus M'$is a module
- $M \cap M'$ is a module
- $M \Delta M'$ is a module



The set of modules of a graph forms a partitive family

A module is strong if it does not overlap any other module

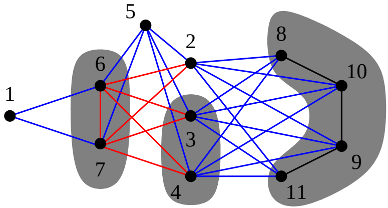




Strong modules are nested into an inclusion tree: the modular decomposition tree $MD(G)$

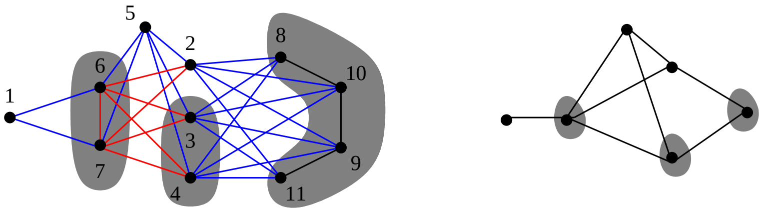
Modular partition and quotient graph

A partition $\mathcal{P}$ of the vertex set of a graph G is a modular partition if every part is a module of G.



Modular partition and quotient graph

A partition $\mathcal{P}$ of the vertex set of a graph G is a modular partition if every part is a module of G.



If $\mathcal{P}$ is a modular partition of G, the quotient graph $G_{/\mathcal{P}}$ is the induced subgraph obtained by choosing one vertex per part of $\mathcal{P}$.

Theorem [Gal'67,CHM81]

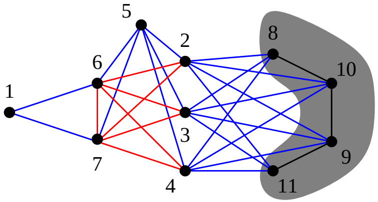
Let $G = (V, E)$ be a graph. Then either

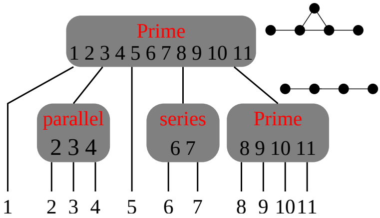
1. (parallel) G is not connected, or
2. (series) $\overline{G}$ is not connected, or

Theorem [Gal'67,CHM81]

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

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

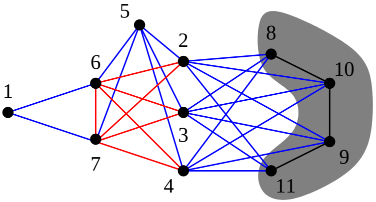


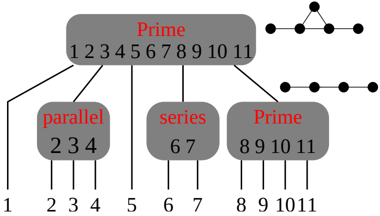


Theorem [Gal'67,CHM81]

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

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





Observation: If a P_4 on $\{a, b, c, d\}$ overlap a module M, then

$$|M \cap \{a, b, c, d\}| = 1$$

Modular decomposition algorithms

- $O(n^4)$ [Cowan, James, Stanton'72]
- $O(n^3)$ [Blass, 1978], [Habib, Maurer'79]
- $O(n^2)$ [McConnell, Spinrad'89]

Modular decomposition algorithms

- $O(n^4)$ [Cowan, James, Stanton'72]
- $O(n^3)$ [Blass, 1978], [Habib, Maurer'79]
- $O(n^2)$ [McConnell, Spinrad'89]

- $O(n + m\alpha(m, n))$ [Spinrad'92], [Cournier, Habib'93]
- $O(n + m)$ [McConnell, Spinrad'94], [Cournier, Habib'94]

Modular decomposition algorithms

- $O(n^4)$ [Cowan, James, Stanton'72]
- $O(n^3)$ [Blass, 1978], [Habib, Maurer'79]
- $O(n^2)$ [McConnell, Spinrad'89]
- $O(n + m\alpha(m, n))$ [Spinrad'92], [Cournier, Habib'93]
- $O(n + m)$ [McConnell, Spinrad'94], [Cournier, Habib'94]
- $O(n + m \log n)$ [Habib, Paul, Viennot'99] (factoring permutation), [McConnell, Spinrad'00]

Modular decomposition algorithms

- $O(n^4)$ [Cowan, James, Stanton'72]
- $O(n^3)$ [Blass, 1978], [Habib, Maurer'79]
- $O(n^2)$ [McConnell, Spinrad'89]

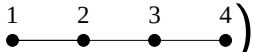
- $O(n + m\alpha(m, n))$ [Spinrad'92], [Cournier, Habib'93]
- $O(n + m)$ [McConnell, Spinrad'94], [Cournier, Habib'94]

- $O(n + m \log n)$ [Habib, Paul, Viennot'99] (factoring permutation), [McConnell, Spinrad'00]

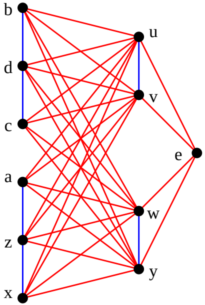
- $O(n + m)$ [Capelle, Habib'97] (factoring permutation) [Dahlhaus, Gustedt, McConnell'97], [Tedder, Corneil, Habib, Paul'08]

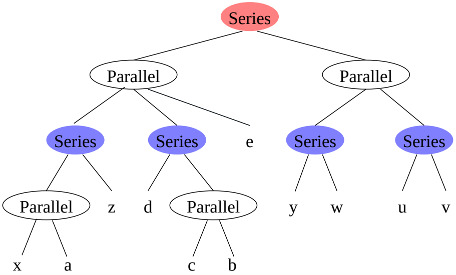
- other many others for variants of modular decomposition

Cographs - Totally decomposable graphs

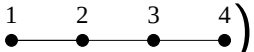
Theorem: A graph is a cograph (a P_4-free graph )

iff its modular decomposition tree does not contain any prime node

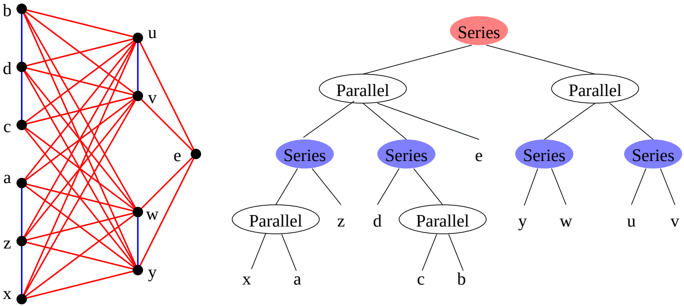




Cographs - Totally decomposable graphs

Theorem: A graph is a cograph (a P_4-free graph )

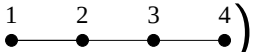
iff its modular decomposition tree does not contain any prime node

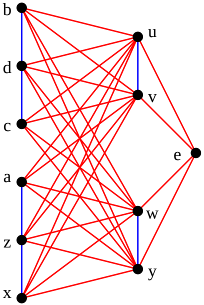


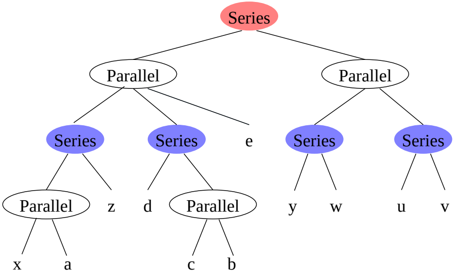
Cographs can be built from the single vertex with the disjoint union and series composition

Exercice: prove that cographs are permutation graphs

Cographs - Totally decomposable graphs

Theorem: A graph is a cograph (a P_4-free graph )
iff its modular decomposition tree does not contain any prime node





Linear time recognition algorithms

- incremental [Corneil, Pearl, Stewart'85]
- partition refinement [Habib, P.'05]
- LexBFS [Bretscher, Corneil, Habib, P.'08]

Modular decomposition of undirected graphs

Ehrenfeucht et al's modular decomposition algorithm

Common Intervals of permutations

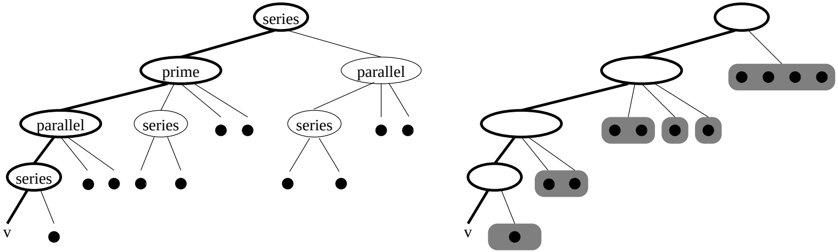
Modular decomposition of tournaments

Kernelization algorithm for FAST

Ehrenfeucht et al's modular decomposition algorithm

$\mathcal{M}(G, v)$ is the modular partition composed by

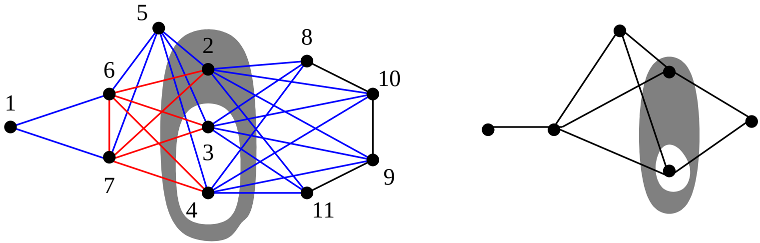
- $\{v\}$ and the maximal modules of G not containing v.



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 $MD(G[\mathcal{X}])$

Computation of $\mathcal{M}(G, v)$ (1)

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.



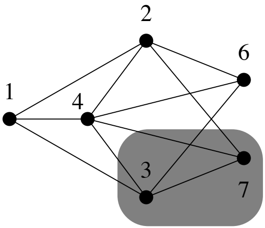
Computation of $\mathcal{M}(G, v)$ (1)

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.

A vertex x is a splitter for a set S of vertices if

$$\exists y, z \in S \text{ with } xy \in S \text{ and } xz \notin E$$

We say that x separate y and z.



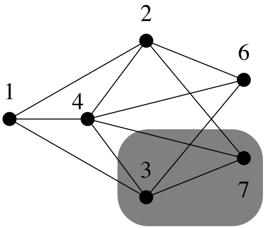
Computation of $\mathcal{M}(G, v)$ (1)

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.

A vertex x is a splitter for a set S of vertices if

$$\exists y, z \in S \text{ with } xy \in S \text{ and } xz \notin E$$

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.

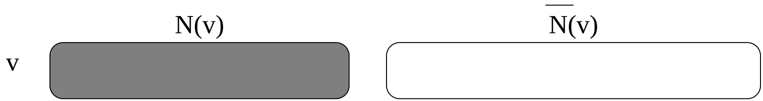
Computation of $\mathcal{M}(G, v)$ (2)

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

Computation of $\mathcal{M}(G, v)$ (2)

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

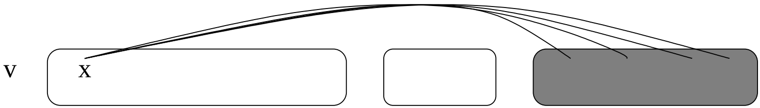
- $O(n + m \log n)$ time using vertex partitioning algorihtm



Computation of $\mathcal{M}(G, v)$ (2)

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

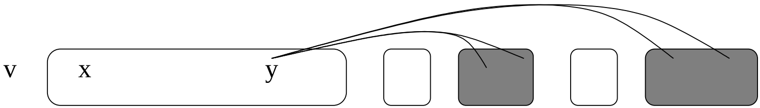
- $O(n + m \log n)$ time using vertex partitioning algorihtm



Computation of $\mathcal{M}(G, v)$ (2)

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

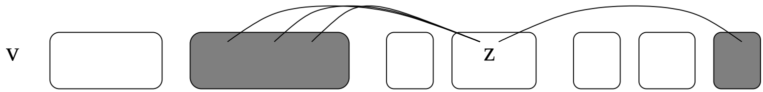
- $O(n + m \log n)$ time using vertex partitioning algorihtm



Computation of $\mathcal{M}(G, v)$ (2)

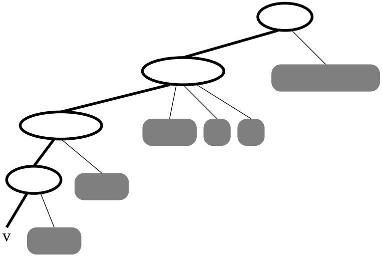
Lemma If v is a splitter of a set S, then for any module $M \subseteq S$ either $M \subseteq S \cap N(v)$ or $M \subseteq M \cap \overline{N}(v)$

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



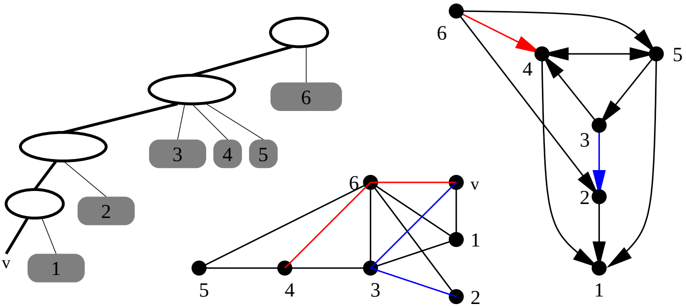
Computation of $MD(G_{/\mathcal{M}(G,v)})$ (3)

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



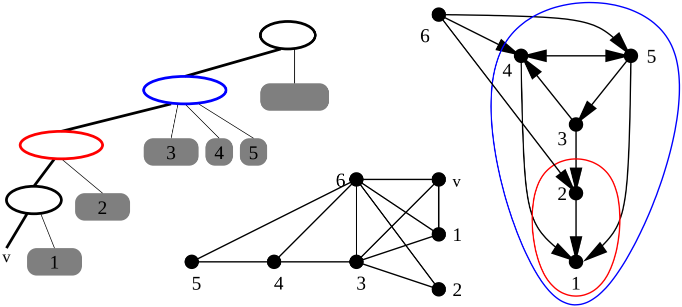
Computation of $MD(G_{/\mathcal{M}(G,v)})$ (3)

- 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 $MD(G_{/\mathcal{M}(G,v)})$ (3)

- 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 $MD(G_{/\mathcal{M}(G,v)})$ (4)

Complexity

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

Other algorithms

- [CH94] and [MS94]: the first linear algorithms.
- [MS99]: $O(n + m)$ algorithm which extends to transitive orientation.

Computation of $MD(G_{/\mathcal{M}(G,v)})$ (4)

Complexity

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

Other algorithms

- [CH94] and [MS94]: the first linear algorithms.
- [MS99]: $O(n + m)$ algorithm which extends to transitive orientation.

[Spinrad'03] The new [linear time] algorithm [MS99] is currently too complex to describe easily [...] I hope and believe that in a number of years the linear algorithm can be simplified as well.

- [Tedder, Corneil, Habib, P.'08] simple linear time algorithm

Modular decomposition of undirected graphs

Ehrenfeucht et al's modular decomposition algorithm

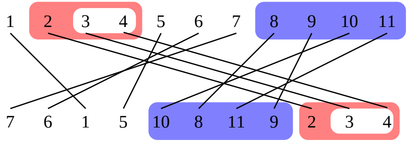
Common Intervals of permutations

Modular decomposition of tournaments

Kernelization algorithm for FAST

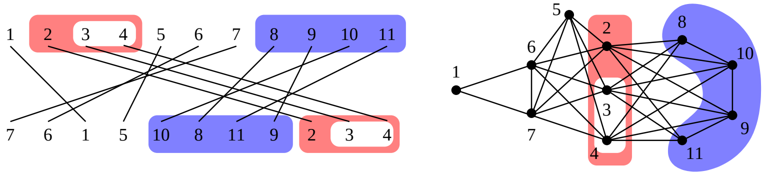
Back to permutations: common intervals

In a permutation σ, a set S of consecutive elements is called an interval. Likewise a set S is a common interval of several permutations $\sigma_1, \sigma_2 \ldots$ if it is an interval for every σ_i.



Back to permutations: common intervals

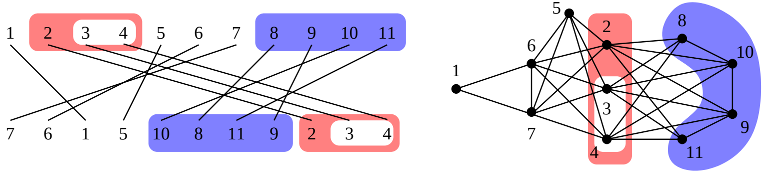
In a permutation σ, a set S of consecutive elements is called an interval. Likewise a set S is a common interval of several permutations $\sigma_1, \sigma_2 \dots$ if it is an interval for every σ_i.



Obs.: common intervals of σ_1, σ_2 are not modules of $G(\sigma_1, \sigma_2)$

Back to permutations: common intervals

In a permutation σ, a set S of consecutive elements is called an interval. Likewise a set S is a common interval of several permutations $\sigma_1, \sigma_2 \dots$ if it is an interval for every σ_i.

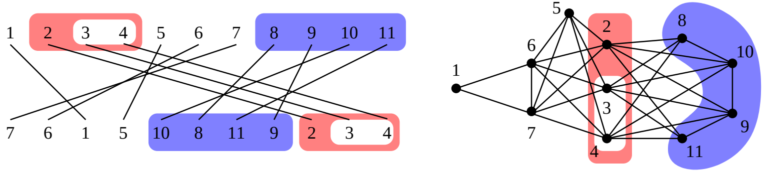


Obs.: common intervals of σ_1, σ_2 are not modules of $G(\sigma_1, \sigma_2)$

- Computation of all common intervals in linear time - $O(n^2)$ - [Uno, Yagura'00]

Back to permutations: common intervals

In a permutation σ, a set S of consecutive elements is called an interval. Likewise a set S is a common interval of several permutations $\sigma_1, \sigma_2 \ldots$ if it is an interval for every σ_i.



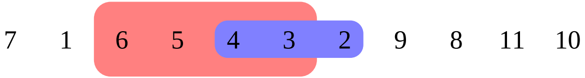
Obs.: common intervals of σ_1, σ_2 are not modules of $G(\sigma_1, \sigma_2)$

Strong (common) interval doesn't overlap other common intervals

Lemma [de Montgolfier] A set S is a strong interval of σ_1 and σ_2 iff it is a strong module of the permutation graph $G(\sigma_1, \sigma_2)$

Common intervals (2)

The family of common intervals is weakly partitive:



if I_1 and I_2 are two common intervals then

- $I_1 \cup I_2$ is a common interval
- $I_1 \cap I_2$ is a common interval
- $I_1 \setminus I_2$ and $I_2 \setminus I_1$ are common intervals

Common intervals (2)

The family of common intervals is weakly partitive:

7 | 1 | 6 5 | 4 3 2 | 9 8 11 10

7 1 5 3 8

if I_1 and I_2 are two common intervals then

- $I_1 \cup I_2$ is a common interval
- $I_1 \cap I_2$ is a common interval
- $I_1 \setminus I_2$ and $I_2 \setminus I_1$ are common intervals

Let $\mathcal{I}$ be a partition of $[1 \dots n]$ into common intervals of the permutation σ, then we denote by $\sigma_{/\mathcal{I}}$ the quotient permutation defined on $[1 \dots |\mathcal{I}|]$

Common intervals (2)

The family of common intervals is weakly partitive:

[7] [1] [6 5] [4 3 2] [9 8 11 10]

4 1 3 2 5

if I_1 and I_2 are two common intervals then

- $I_1 \cup I_2$ is a common interval
- $I_1 \cap I_2$ is a common interval
- $I_1 \setminus I_2$ and $I_2 \setminus I_1$ are common intervals

Let $\mathcal{I}$ be a partition of $[1 \dots n]$ into common intervals of the permutation σ, then we denote by $\sigma_{/\mathcal{I}}$ the quotient permutation defined on $[1 \dots |\mathcal{I}|]$

Common intervals (3)

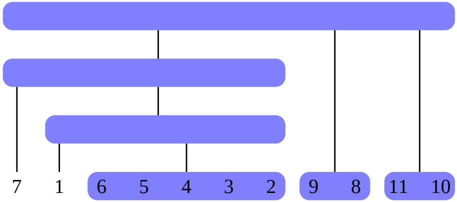
Theorem Let σ be a permutation on $[1, \ldots n]$ and $\mathcal{I}$ be the partition into maximal common intervals of σ, then either

1. $\sigma_{/\mathcal{I}} = \mathbb{1}_{|\mathcal{I}|}$ - the identity on $[1 \ldots |\mathcal{I}|]$
2. $\sigma_{/\mathcal{I}} = \overline{\mathbb{1}_{|\mathcal{I}|}}$ - the reverse identity on $[1 \ldots |\mathcal{I}|]$
3. $\sigma_{/\mathcal{I}}$ is prime - or simple (does not have non-trivial common interval)

Common intervals (3)

Theorem Let σ be a permutation on $[1, \ldots n]$ and $\mathcal{I}$ be the partition into maximal common intervals of σ, then either

1. $\sigma_{/\mathcal{I}} = \mathbb{1}_{|\mathcal{I}|}$ - the identity on $[1 \ldots |\mathcal{I}|]$
2. $\sigma_{/\mathcal{I}} = \overline{\mathbb{1}_{|\mathcal{I}|}}$ - the reverse identity on $[1 \ldots |\mathcal{I}|]$
3. $\sigma_{/\mathcal{I}}$ is prime - or simple (does not have non-trivial common interval)

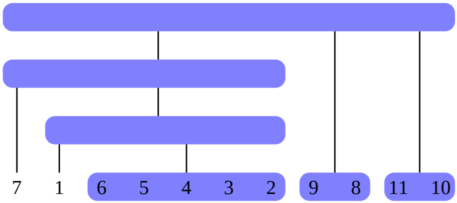


Theorem (see e.g. [Bergeron et al.'08])
The common interval tree can be computed in $O(n)$ time.

Common intervals (4)

A permutation σ is separable if it does not contains the pattern

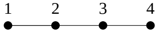
3 1 4 2

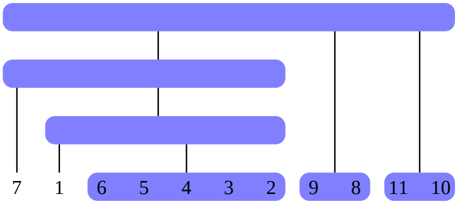


Common intervals (4)

A permutation σ is separable if it does not contains the pattern

$$3\ 1\ 4\ 2$$

- 3 1 4 2 corresponds to the P_4 

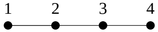


Common intervals (4)

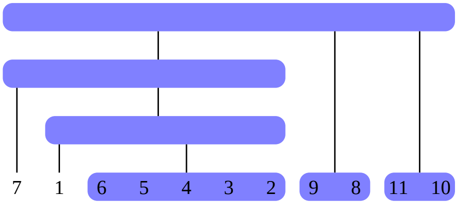
A permutation σ is separable if it does not contains the pattern

3 1 4 2

- 3 1 4 2 corresponds to the P_4



- a permutation is separable iff
  its common interval tree does not contains prime nodes
- a permutation is separable iff
  the permutation graph $G(\sigma, \mathbb{1})$ is a cograph (P_4-free graph)



Modular decomposition of undirected graphs

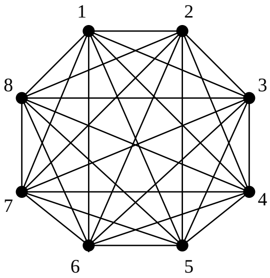
Ehrenfeucht et al's modular decomposition algorithm

Common Intervals of permutations

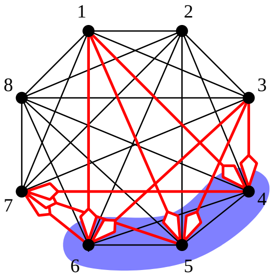
Modular decomposition of tournaments

Kernelization algorithm for FAST

Modular decomposition of tournaments



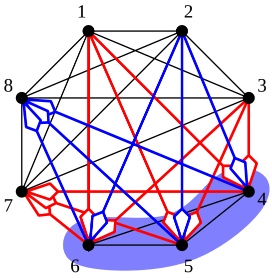
Modular decomposition of tournaments



A module in a tournament is a set S such that for every $x \notin S$

- either $\forall y \in S,\ x \rightarrow y$ or $\forall y \in S,\ y \rightarrow x$

Modular decomposition of tournaments

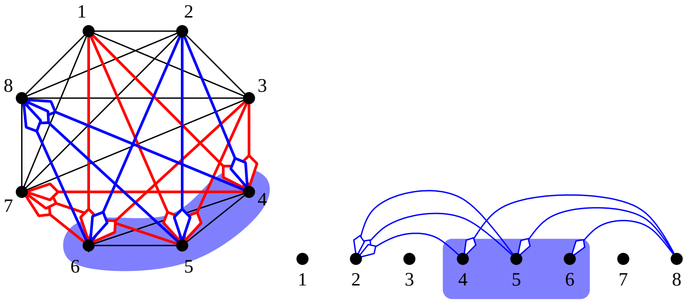


A tournament is transitive if there exists a permutation σ of $V(T)$ with no backward arcs

Theorem: Let T be a tournament and $\mathcal{M}(T)$ be the modular partition into maximal strong modules, then

1. either $T_{/\mathcal{M}(T)}$ is transitive - contains no backward arc
2. or $T_{/\mathcal{M}(T)}$ is prime

Modular decomposition of tournaments



A tournament is transitive if there exists a permutation σ of $V(T)$ with no backward arcs

Theorem: Let T be a tournament and $\mathcal{M}(T)$ be the modular partition into maximal strong modules, then

1. either $T_{/\mathcal{M}(T)}$ is transitive - contains no backward arc
2. or $T_{/\mathcal{M}(T)}$ is prime

Modular decomposition algorithm for tournaments

A factoring permutation of a tournament T (or a graph) is a permutation σ of its vertices such that

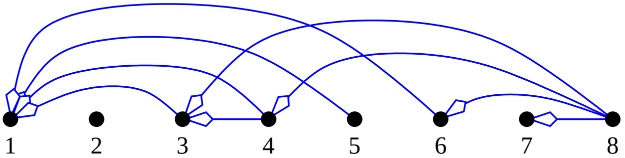
every (strong) module of T is an interval of σ

Modular decomposition algorithm for tournaments

A factoring permutation of a tournament T (or a graph) is a permutation σ of its vertices such that

every (strong) module of T is an interval of σ

- Factoring permutation via a partition refinement algorithm in linear time [de Mongolfier'03]

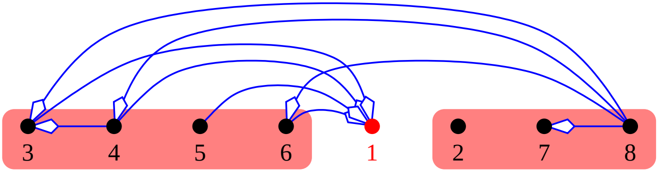


Modular decomposition algorithm for tournaments

A factoring permutation of a tournament T (or a graph) is a permutation σ of its vertices such that

every (strong) module of T is an interval of σ

- Factoring permutation via a partition refinement algorithm in linear time [de Mongolfier'03]

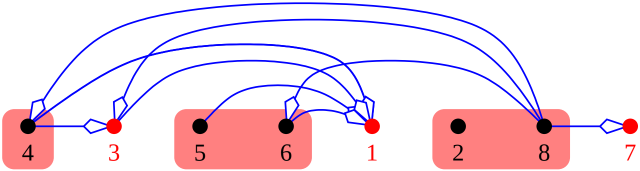


Modular decomposition algorithm for tournaments

A factoring permutation of a tournament T (or a graph) is a permutation σ of its vertices such that

every (strong) module of T is an interval of σ

- Factoring permutation via a partition refinement algorithm in linear time [de Mongolfier'03]

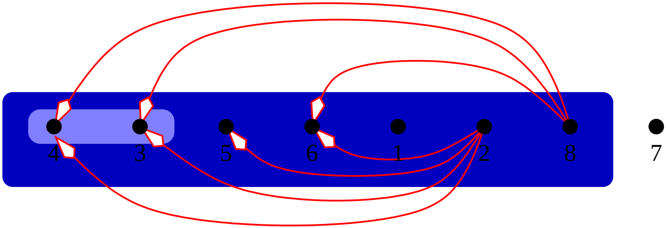


Modular decomposition algorithm for tournaments

A factoring permutation of a tournament T (or a graph) is a permutation σ of its vertices such that

every (strong) module of T is an interval of σ

- Factoring permutation via a partition refinement algorithm in linear time [de Mongolfier'03]

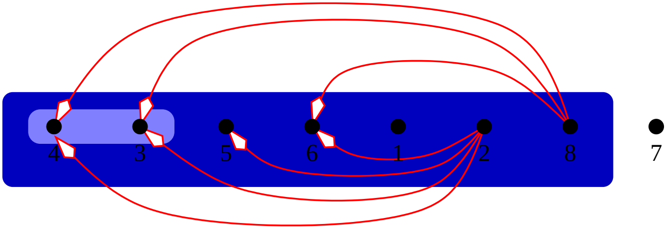


Modular decomposition algorithm for tournaments

A factoring permutation of a tournament T (or a graph) is a permutation σ of its vertices such that

every (strong) module of T is an interval of σ

- Factoring permutation via a partition refinement algorithm in linear time [de Mongolfier'03]



- Modular decomposition tree from a factoring permutation in linear time [Capelle'97]

Modular decomposition of undirected graphs

Ehrenfeucht et al's modular decomposition algorithm

Common Intervals of permutations

Modular decomposition of tournaments

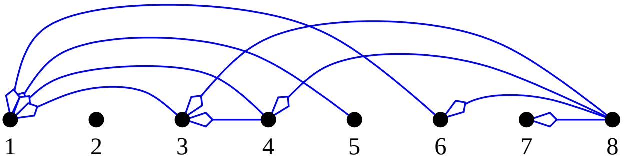
Kernelization algorithm for FAST

FAST: Feedback Arc Set in Tournament

- A tournament T and an integer k
- Find a set of at most k arcs whose reversal transform T into a transitive tournament

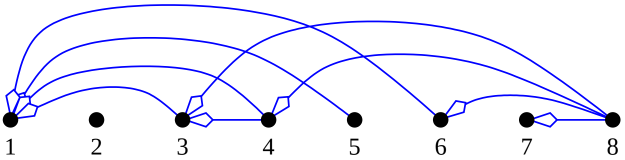
FAST: Feedback Arc Set in Tournament

- A tournament T and an integer k
- Find a permutation σ of the vertices with at most k backward edges



FAST: Feedback Arc Set in Tournament

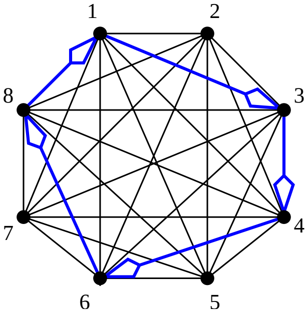
- A tournament T and an integer k
- Find a permutation σ of the vertices with at most k backward edges



- NP-Complete [Alon'06] [Charbit et al.'07]
- FTP [Raman, Saurabh'06] [Alon et al.'09]
- $(1+\epsilon)$-approximation scheme [Kenyon-Mathieu, Schudy'07]

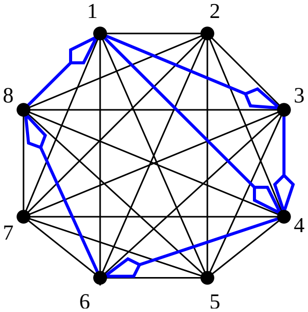
FAST (2)

Obs.: A tournament is transitive iff there is no (directed) triangle



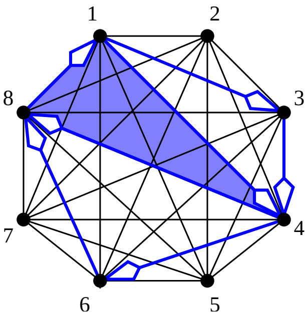
FAST (2)

Obs.: A tournament is transitive iff there is no (directed) triangle



FAST (2)

Obs.: A tournament is transitive iff there is no (directed) triangle

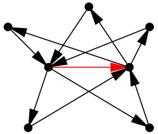


FAST (2)

Obs.: A tournament is transitive iff there is no (directed) triangle

Rule 1 [irrelevant vertex] If a vertex v is not contained in any triangle, then delete v

Rule 2 [sunflower] If there is an arc belonging to more that k distinct triangles, then reverse it and decrease k by 1



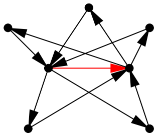
FAST (2)

Obs.: A tournament is transitive iff there is no (directed) triangle

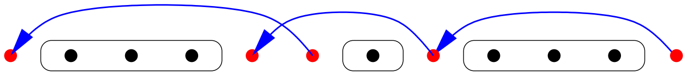
Rule 1 [irrelevant vertex] If a vertex v is not contained in any triangle, then delete v

A reduced tournament contains no source nor sink

Rule 2 [sunflower] If there is an arc belonging to more that k distinct triangles, then reverse it and decrease k by 1

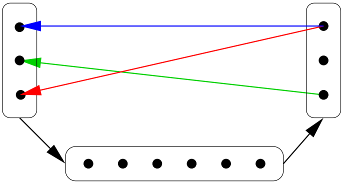


The span $s(\overrightarrow{uv})$ of a backward arc of a reduced tournament is $\leqslant 2k+2$



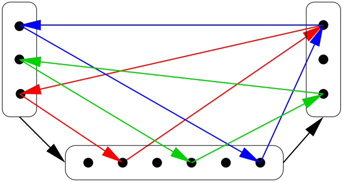
FAST (3)

Rule 3 [acyclic module] Let M be a maximal acyclic module. If there are at most $p = |M|$ arcs from $N^+(M)$ to $N^-(M)$, then reverse all these arcs and decrease k by p.



FAST (3)

Rule 3 [acyclic module] Let M be a maximal acyclic module.
If there are at most $p = |M|$ arcs from $N^+(M)$ to $N^-(M)$,
then reverse all these arcs and decrease k by p.

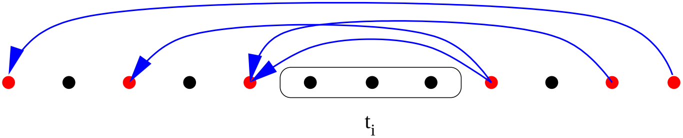


FAST (3)

Rule 3 [acyclic module] Let M be a maximal acyclic module.
If there are at most $p = |M|$ arcs from $N^+(M)$ to $N^-(M)$,
then reverse all these arcs and decrease k by p.

acyclic module Rule $\Rightarrow$

$$\sum t_i^2 \leqslant \sum s(\overrightarrow{uv})$$

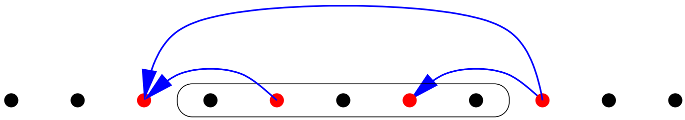


FAST (3)

Rule 3 [acyclic module] Let M be a maximal acyclic module.
If there are at most $p = |M|$ arcs from $N^+(M)$ to $N^-(M)$,
then reverse all these arcs and decrease k by p.

sunflower Rule $\Rightarrow$

$$\sum t_i^2 \leqslant \sum s(\overrightarrow{uv}) \leqslant k(2k+2)$$

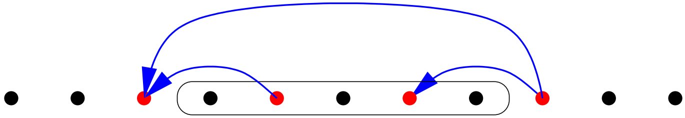


FAST (3)

Rule 3 [acyclic module] Let M be a maximal acyclic module. If there are at most $p = |M|$ arcs from $N^+(M)$ to $N^-(M)$, then reverse all these arcs and decrease k by p.

sunflower Rule $\Rightarrow$

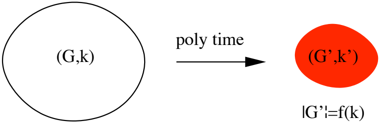
$$\sum t_i^2 \leqslant \sum s(\overrightarrow{uv}) \leqslant k(2k+2)$$



Theorem [Bessy et al.'09]: Every instance (T, k) of k-FAST can be reduced in polynomial time to an equivalent instance (T, k') such that

$$|T| \leqslant 2k + \sum t_i = O(k\sqrt{k}) \text{ and } k' \leqslant k$$

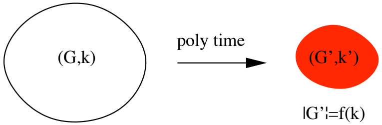
Kernelization algorithm



Given a parameterized instance $(\mathcal{I}, k)$ of a problem, a kernelization algorithm computes in polytime an equivalent instance $(\mathcal{I}', k')$ st.

$$k' = f(k) \qquad \text{and} \qquad |\mathcal{I}'| \leqslant g(k)$$

Kernelization algorithm

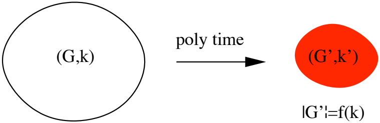


Given a parameterized instance $(\mathcal{I}, k)$ of a problem, a kernelization algorithm computes in polytime an equivalent instance $(\mathcal{I}', k')$ st.

$$k' = f(k) \qquad \text{and} \qquad |\mathcal{I}'| \leqslant g(k)$$

- we described a $O(k\sqrt{k})$-vertex kernel for FAST based on modular decomposition
- best known result: $O(k)$-vertex kernel ([Bessy et al.'09], [P., Perez, Thomassé'11])

Kernelization algorithm



Given a parameterized instance $(\mathcal{I}, k)$ of a problem, a kernelization algorithm computes in polytime an equivalent instance $(\mathcal{I}', k')$ st.

$$k' = f(k) \qquad \text{and} \qquad |\mathcal{I}'| \leqslant g(k)$$

Other modular decomposition kernelizations:

- $O(k^2)$-vertex kernel for CLUSTER EDITING
- $O(k^3)$-vertex kernel for COGRAPH EDITING
- also for MIN FLIP CONSENSUS TREE, CLOSEST 3-LEAF POWER...

Some conclusions

- Modular decomposition plays an important role in the context of many graph classes
  - permutation graphs, interval graphs, comparability graphs . . .
  - even perfect graph)

Some conclusions

- Modular decomposition plays an important role in the context of many graph classes
  - permutation graphs, interval graphs, comparability graphs ...
  - even perfect graph)
- Many examples of partitive (weakly partitive) families are known in various contexts
  - modules in undirected graphs, digraphs, hypergraphs
  - common interval of permutations

Some conclusions

- Modular decomposition plays an important role in the context of many graph classes
  - permutation graphs, interval graphs, comparability graphs ...
  - even perfect graph)
- Many examples of partitive (weakly partitive) families are known in various contexts
  - modules in undirected graphs, digraphs, hypergraphs
  - common interval of permutations
- Various generalizations
  - bimodular decomposition (module adapted to bipartite graphs)
  - bipartitive families : eg. split decomposition of graphs - $O(n + \alpha(n, m).m)$ circle graph recognition
  - crossing families, union-difference families of sets...
  - clique-width (cographs are clique-widht 2 graphs), rankwidth

To learn / read more

- Habib and P. "A survey of the algorithmic aspects of modular decomposition" in Computer Science Review 4:41-59, 2010

To learn / read more

- Habib and P. "A survey of the algorithmic aspects of modular decomposition" in Computer Science Review 4:41-59, 2010
- F. de Montgolfier, "Décomposition modulaire de graphes, théorie, extensions et algorithmes", Phd Thesis (in French), 2003
- B.M. Bui Xuan, "Tree-representation of set families in graph decompositions and efficient algorithms", Phd Thesis, 2008

To learn / read more

- Habib and P. "A survey of the algorithmic aspects of modular decomposition" in Computer Science Review 4:41-59, 2010
- F. de Montgolfier, "Décomposition modulaire de graphes, théorie, extensions et algorithmes", Phd Thesis (in French), 2003
- B.M. Bui Xuan, "Tree-representation of set families in graph decompositions and efficient algorithms", Phd Thesis, 2008
- C. Crespelle, "Représentations dynamiques de graphes", PhD Thesis (in French), 2007 attend I. Todinca's talk !!!

To learn / read more

- Habib and P. "A survey of the algorithmic aspects of modular decomposition" in Computer Science Review 4:41-59, 2010
- F. de Montgolfier, "Décomposition modulaire de graphes, théorie, extensions et algorithmes", Phd Thesis (in French), 2003
- B.M. Bui Xuan, "Tree-representation of set families in graph decompositions and efficient algorithms", Phd Thesis, 2008
- C. Crespelle, "Représentations dynamiques de graphes", PhD Thesis (in French), 2007 attend I. Todinca's talk !!!
- Common intervals and sorting by reversal:
  attend M. Bouvel's talk!!!