Graph classes defined via vertex ordering avoiding patterns

Laurent Feuilloley

joint work with Yacine Boufkhad, Pierre Charbit, and Michel Habib

Université Paris Diderot

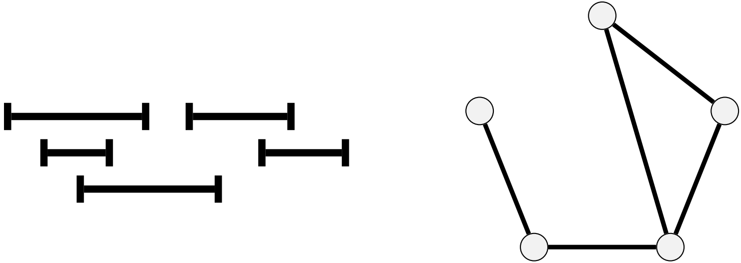
GROW · Toronto · October 2017

Disclaimer

This is mostly a survey

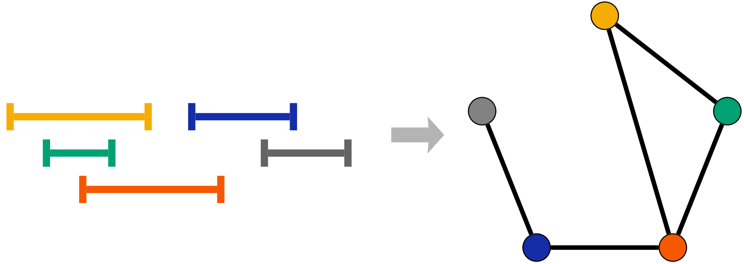
Warm-up : Interval graphs

Interval graphs



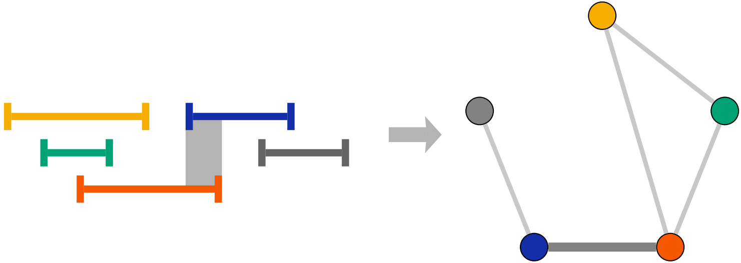
Interval graphs

From intervals to graphs



Interval graphs

From intervals to graphs

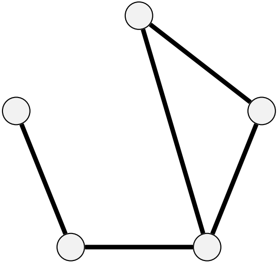


Interval graphs

From graphs to intervals

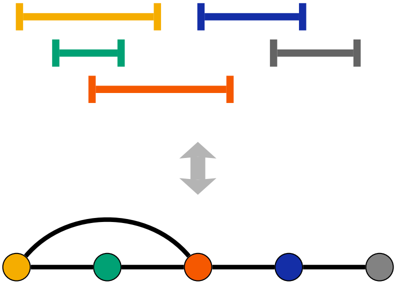
?

⬅



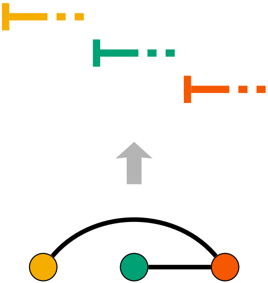
Interval graphs

with an ordering



Interval graphs

with an ordering

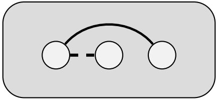


Interval graphs

Characterization

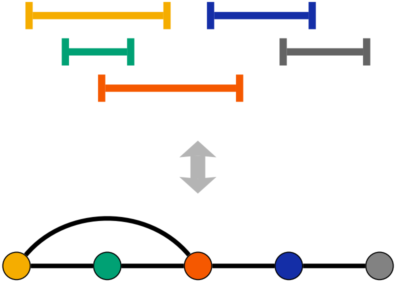
An ordered graph represents an interval graph

$\Leftrightarrow$

it avoids the pattern : 

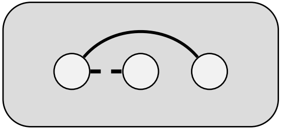
Interval graphs

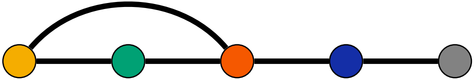
Characterization



Interval graphs

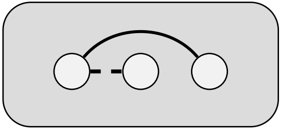
Characterization

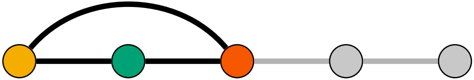




Interval graphs

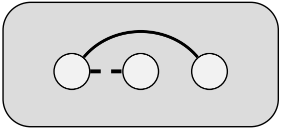
Characterization

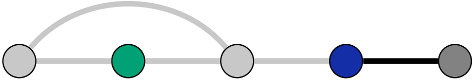




Interval graphs

Characterization





Interval graphs

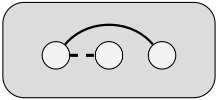
Characterization

A graph is an interval graph

$\Leftrightarrow$

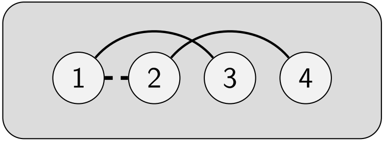
there exists a vertex ordering

that avoids :



Definitions

Pattern



For an ordered subgraph to match the pattern :

- plain edges must be present,
- dashed edges must be absent,
- non-edges have no constraint.

Vertex ordering characterizations

A graph is a

$\Leftrightarrow$

there exists a vertex ordering that avoids :

Pattern 1

Pattern 2

Pattern 3

...

[Damaschke 90]

Examples

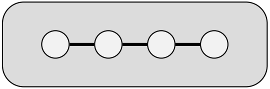
A zoo of classes

k-colourable

A graph is a 3-colourable

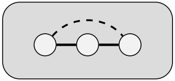
$\Leftrightarrow$

there exists a vertex ordering
that avoids :

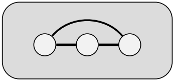


On three nodes

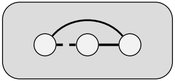
Comparability



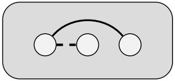
Triangle-free



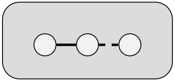
Chordal



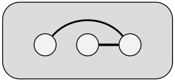
Interval



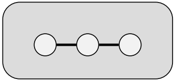
Split



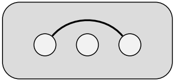
Tree



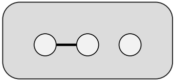
Bipartite



Path



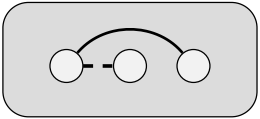
Star



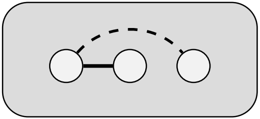
Structure

Complement

Interval $\rightleftharpoons$ Co-interval



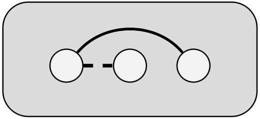
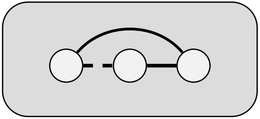
$\rightleftharpoons$



Inversion of dashed/plain edges $\Leftrightarrow$ Complement class

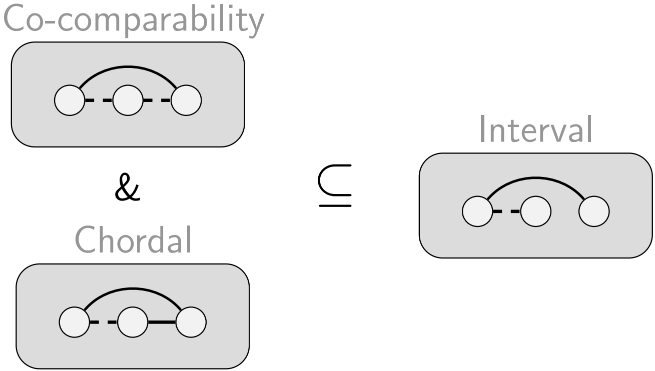
Inclusion

Interval $\subseteq$ Chordal

 $\subseteq$ 

Inclusion of patterns $\Rightarrow$ Inclusion of classes

Pattern splitting



Co-comparability

&

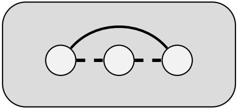
Chordal

⊆

Interval

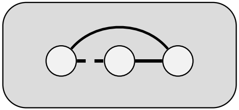
Pattern splitting

Co-comparability



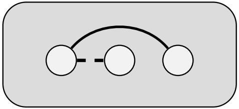
&

Chordal



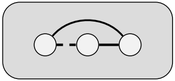
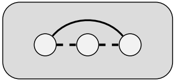
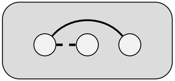
=

Interval

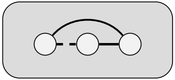
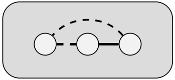
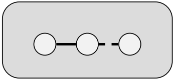


Pattern splitting

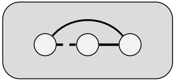
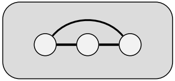
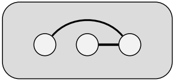
Chordal ∩ Co-comparability = Interval

 &  = 

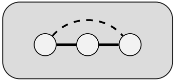
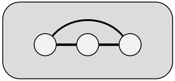
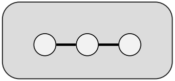
Chordal ∩ Co-chordal = Split

 &  = 

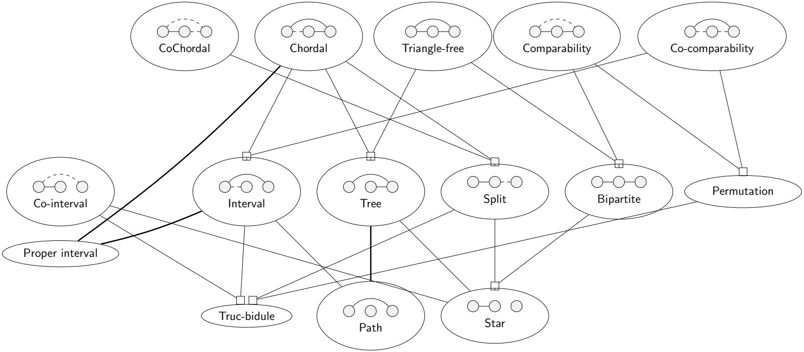
Chordal ∩ Triangle-free = Tree

 &  = 

Comparability ∩ Triangle-free = Bipartite

 &  = 

Diagram


CoChordal
Chordal
Triangle-free
Comparability
Co-comparability
Co-interval
Interval
Tree
Split
Bipartite
Permutation
Proper interval
Truc-bidule
Path
Star

Recognition

NP

Recognition of classes defined by forbidden patterns is in NP.

The ordering can be checked in polytime.

On three nodes

Theorem : Classes defined by patterns on three nodes can be recognized in polynomial-time.

Proof history :

- Class by class ;
- class by class with orderings ;
- a general algorithm [Hell, Mohar, Rafiey, 2014] ;
- a general algorithm with a simpler analysis ?

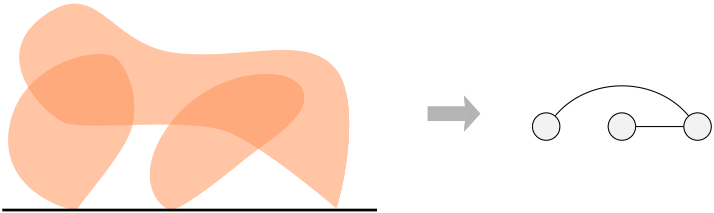
General case

- Some classes can be recognized in polytime, e.g. outerplanar graphs ;
- Some are NP-complete, e.g. k-colourability ;
- Almost all the classes defined by 2-connected patterns are NP-complete to recognize [Duffus, Ginn, Rödl, 95].
- It seems that there is no dichotomy [Nešetřil 17].

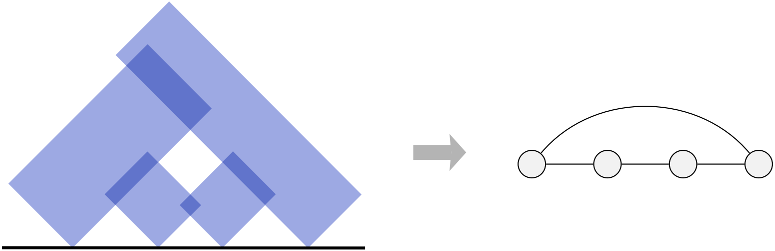
Geometry

Grounded intersection graphs

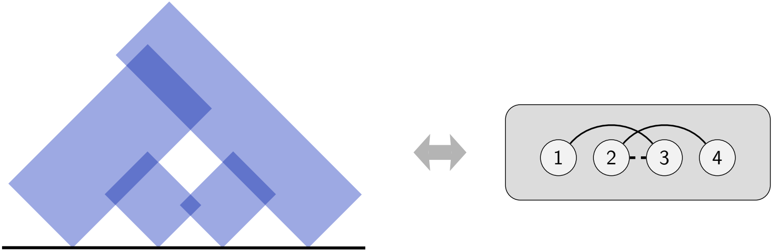
Grounded intersection graphs



Grounded rectangles graphs



Grounded rectangles graphs



Applications to algorithms

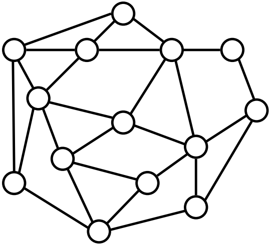
Applications to algorithms

Tomorrow !

Applications to distributed decision

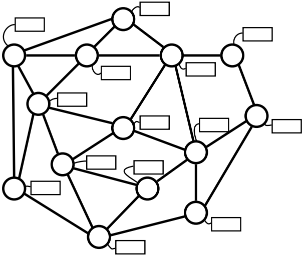
A distributed NP

1. A prover gives to each node a small certificate
2. Every node gathers some t-neighbourhood (structure and certificates) and chooses to accept or reject.
3. The graph is accept iff all nodes accept.



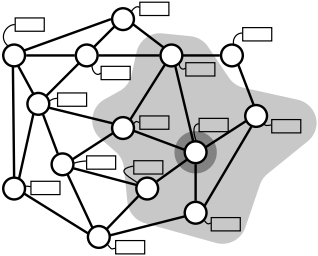
A distributed NP

1. A prover gives to each node a small certificate
2. Every node gathers some t-neighbourhood (structure and certificates) and chooses to accept or reject.
3. The graph is accept iff all nodes accept.



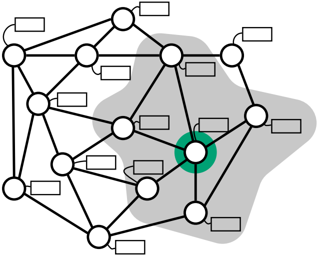
A distributed NP

1. A prover gives to each node a small certificate
2. Every node gathers some t-neighbourhood (structure and certificates) and chooses to accept or reject.
3. The graph is accept iff all nodes accept.



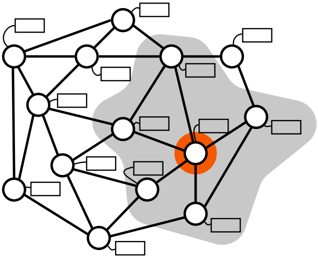
A distributed NP

1. A prover gives to each node a small certificate
2. Every node gathers some t-neighbourhood (structure and certificates) and chooses to accept or reject.
3. The graph is accept iff all nodes accept.



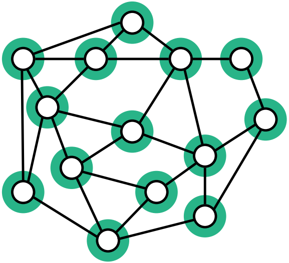
A distributed NP

1. A prover gives to each node a small certificate
2. Every node gathers some t-neighbourhood (structure and certificates) and chooses to accept or reject.
3. The graph is accept iff all nodes accept.



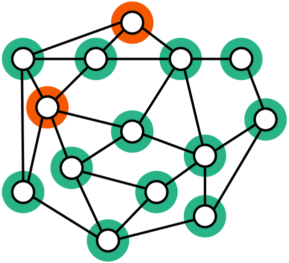
A distributed NP

1. A prover gives to each node a small certificate
2. Every node gathers some t-neighbourhood (structure and certificates) and chooses to accept or reject.
3. The graph is accept iff all nodes accept.



A distributed NP

1. A prover gives to each node a small certificate
2. Every node gathers some t-neighbourhood (structure and certificates) and chooses to accept or reject.
3. The graph is accept iff all nodes accept.



Distributed NP recognition

The ordering is a useful certificate
that can be checked locally for many classes.

Take-home message

- Vertex ordering characterizations are all around us.

- There are a lot of open questions worth investigating !