Treewidth computations I. Upper bounds

Hans L. Bodlaender, Arie M.C.A Koster

Armin Friedl

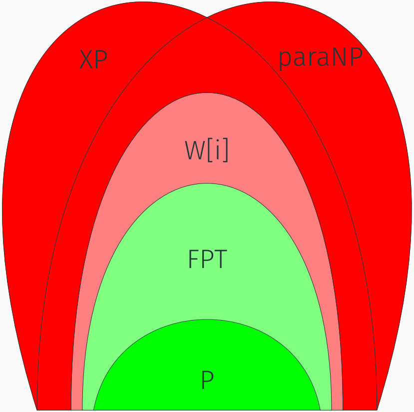
June 30, 2016

Table of contents

Motivation

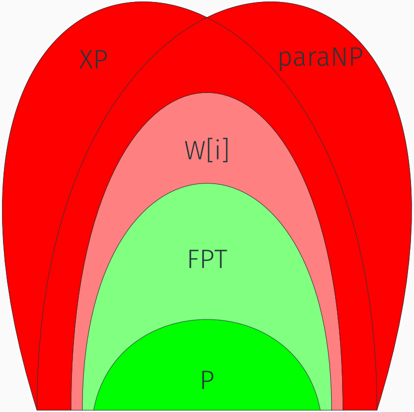
General Motivation

1. Choose infeasible problem
    - Combinatorial Problems
    - Computational Biology
    - Constraint Satisfaction
    - ...
2. Find FPT$_{tw}$ algorithm
3. Model problem as graph
4. Compute *tree composition* with small *tree width*



General Motivation

1. Choose infeasible problem
    - Combinatorial Problems
    - Computational Biology
    - Constraint Satisfaction
    - ...
2. Find FPT_{tw} algorithm
3. Model problem as graph
4. Compute *tree composition* with small *tree width*
5. Tract the intractable



Why Upper Bounds?

We want (efficiently)

- *High* Lower Bound: Tree dec. not the right tool
- *Low* Upper Bound: Tree dec. works
- Other combinations? -- Not so useful

What this paper is about

Exact algorithm: Huge constant factor [4]

$\rightarrow$ Find a non-optimal tree decomposition

$\rightarrow$ This is also an Upper Bound

Elimination Ordering Methods

Table of contents

Idea

Theorem [1, 2]

Equivalent:

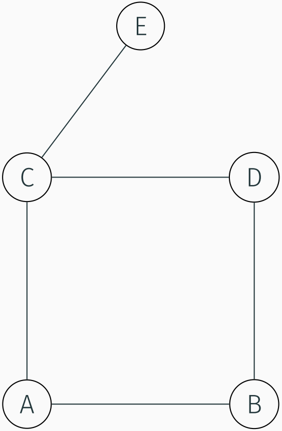
(i) G has a treewidth at most k.

(ii) There is an elimination ordering π, such that no vertex $v \in V$ has more than k neighbours with a higher number in π in G_π^+

Application

1. Take *some* elimination ordering π of G
2. Construct G_π^+, calculate k
3. $\xrightarrow{(i) \equiv (ii)}$ Upper Bound for treewidth

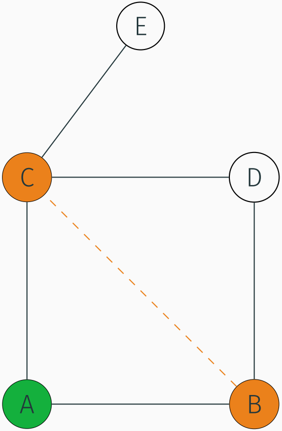
What is G_π^+ ?



$\pi = [A, B, C, D, E]$

Input: G, π
Output: G_π^+
$H = G$
foreach $v \in V_G$ **do**
  foreach w, x *of* $\mathrm{N}_H(v)$ **do**
    if $\pi(w), \pi(x) > \pi(v)$ **then**
      add {w,x} to E_H
return *H*

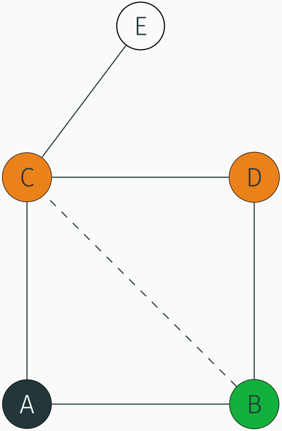
What is G_π^+ ?



$\pi = [A, B, C, D, E]$

Input: G, π
Output: G_π^+
$H = G$
foreach $v \in V_G$ **do**
  foreach w, x of $N_H(v)$ **do**
    if $\pi(w), \pi(x) > \pi(v)$ **then**
      add {w,x} to E_H
return H

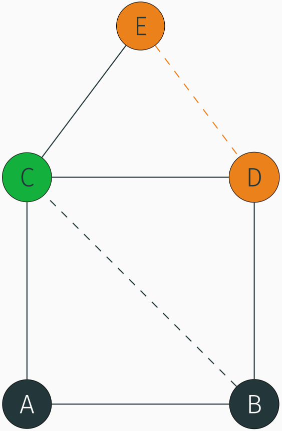
What is G_π^+ ?



$\pi = [A, B, C, D, E]$

Input: G, π
Output: G_π^+
$H = G$
foreach $v \in V_G$ **do**
  foreach w, x *of* $N_H(v)$ **do**
    if $\pi(w), \pi(x) > \pi(v)$ **then**
      add {w,x} to E_H
return *H*

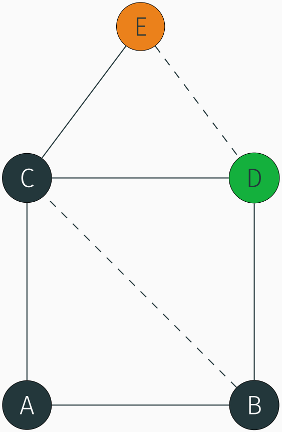
What is G_π^+ ?



$\pi = [A, B, C, D, E]$

Input: G, π
Output: G_π^+
$H = G$
foreach $v \in V_G$ **do**
  foreach w, x of $N_H(v)$ **do**
    if $\pi(w), \pi(x) > \pi(v)$ **then**
      add {w,x} to E_H
return H

What is G_π^+ ?



$\pi = [A, B, C, D, E]$

Input: G, π

Output: G_π^+

$H = G$

foreach $v \in V_G$ **do**

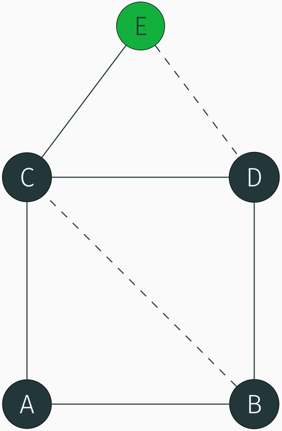
  foreach w, x of $N_H(v)$ **do**

    if $\pi(w), \pi(x) > \pi(v)$ **then**

      add {w,x} to E_H

return H

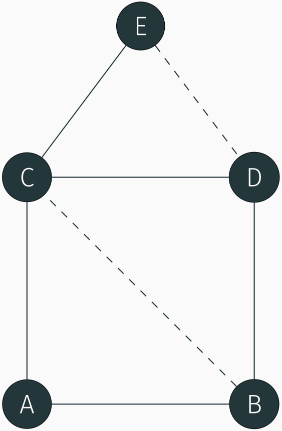
What is G_π^+ ?



$\pi = [A, B, C, D, E]$

Input: G, π
Output: G_π^+
$H = G$
foreach $v \in V_G$ **do**
  foreach w, x *of* $N_H(v)$ **do**
    if $\pi(w), \pi(x) > \pi(v)$ **then**
      add {w,x} to E_H
return *H*

What is G_π^+ ?



$\pi = [A, B, C, D, E]$

- G_π^+ is chordal
- G is a subgraph of G_π^+
- π is a perfect elimination ordering of G_π^+
- $width$ of *subtree graph* (also a tree decomposition) of G_π^+ is MAXCLIQUE(G_π^+) $- 1$ [2]
- There is a tree decomposition algorithm for G with $width =$ MAXCLIQUE(G_π^+) $- 1$, polynomial in n [1]

How to find the best elimination ordering?

How to find the best elimination ordering?

$$\begin{aligned}\text{Best} &= G_\pi^+ \text{ with } \text{Min}(\text{MAXCLIQUE}(G_\pi^+)) \\ &= \text{Computational Infeasible} \\ &= \text{see [3]}\end{aligned}$$

How to find ~~the best~~ a good elimination ordering?

No best. But the smaller the triangulation the better.
For minimal (not minimum): $\mathcal{O}(n^{2.376})$ [3]

Greedy Triangulation - Algorithm

Input: $G(V, E)$

Output: π

$H = G$

for $i = 1$ **to** n **do**

  Choose $v \in H$ by criteria X

  Set $\pi^{-1}(i) = v$

  Eliminate v from H (make $N_H(v)$ a clique and remove v)

return H

Greedy Triangulation - Algorithm

Input: $G(V, E)$

Output: π

$H = G$

for $i = 1$ **to** n **do**

  Choose $v \in H$ by criteria X

  Set $\pi^{-1}(i) = v$

  Eliminate v from H (make $N_H(v)$ a clique and remove v)

return H

How to choose X?

Greedy Triangulation - Criteria X

Minimum Degree/Greedy Degree

X = v with smallest degree in H

Performs well in practice

Greedy Fill In

X = v which causes smallest number of fill edges in G_π^+
= v with smallest number of pairs of non-adjacent neighbours

Slightly slower, slightly better bounds than MD/GD on average

Greedy Triangulation - Advanced Criteria

Lower Bound Based

Eliminate v from H, compute lower bound (LB) of treewidth
Choose v with Min$(2 * LB + \deg_H(v))$

Enhanced Minimum Fill In

Compute LB of G
Choose simplical or almost simplical v with $\deg(v)$ at most LB
otherwise: Greedy Fill In

...

Tabu Search

General Approach

(i) Keep list of α last solutions to avoid cycling
(ii) Find inital solution [= some elimination ordering]
(iii) Make small change to get *Neighbourhood*
(iv) Select neighbouring solution $\notin \alpha$ with smallest cost
(v) Repeat (iii), (iv) some time $\rightarrow$ return best solution

Neighbourhood Generation

Swap two vertices in elminiation ordering

Step Cost

(i) Width of generated neighbour
(ii) But many neighbours with equal width, better:
$\rightarrow w_\pi * n^2 + \sum v \in V|N_\pi^+(v)|^2$

Chordal Graph Recognition Heuristics

If it's chordal already, find perfect elminiation ordering (i.e. recognize it):

- Maximum Cardinality Search
- Lexicographical Breadth First Search

... tree decomposition depends on (perfect) elimination ordering found. Mostly determined by algorithms, except for first chosen v_n (from right to left).

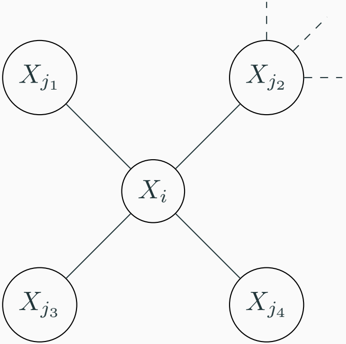
$\rightarrow$ try for all v

$\rightarrow$ adds factor $\mathcal{O}(n)$

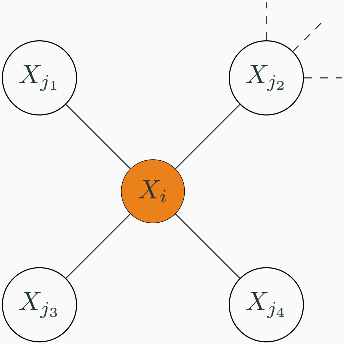
Separator Methods

Table of contents

Minimum Separating Vertex Set Heuristic

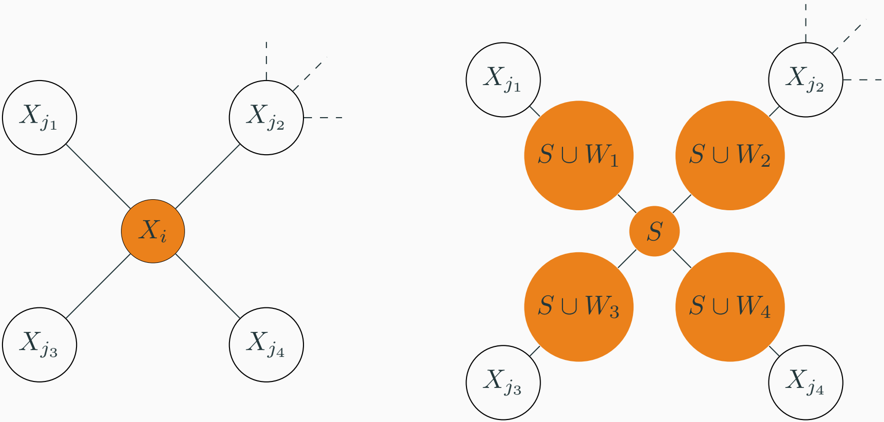


Minimum Separating Vertex Set Heuristic



Choose $i \in I$ such that $|X_i|$ maximal and $G[X_i]$ does not include a clique.

Minimum Separating Vertex Set Heuristic



Construct Graph H_i:

$H_i(X_i, E_{H_i}), E_{H_i} = \{\{v,w\} \in X_i \times X_i | \{v,w\} \in E \vee \exists j \neq i : v,w \in X_j\}$

Compute minimum separator S; $W_1, \ldots, W_r$ are components

Construct new tree decomposition

Others

- MinimalTriangulation (same principle as Minimum Separating Vertex Set Heuristic)
- Component Splitting

Results

Greedy Results

- Average of 50 randomly generated graphs
- Combinations of GreedyFillIn, GreedyDegree, Triangulation Minimisation
- Best Results for combinations with Triangulation Minimisation
- Worst Results for GreedyFillIn alone
- GreedyDegree is fast and perfoming well

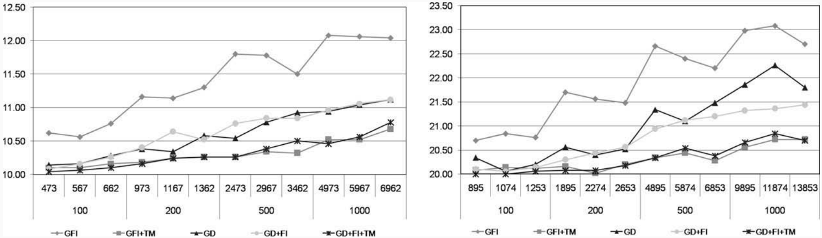


Figure 1: Results for Greedy Heuristics [1]

References I

H. L. Bodlaender and A. M. Koster.
Treewidth computations i. upper bounds.
Information and Computation, 208(3):259 -- 275, 2010.

F. Gavril.
The intersection graphs of subtrees in trees are exactly the chordal graphs.
Journal of Combinatorial Theory, Series B, 16(1):47 -- 56, 1974.

P. Heggernes.
Minimal triangulations of graphs: A survey.
Discrete Mathematics, 306(3):297 -- 317, 2006.
Minimal Separation and Minimal Triangulation.

References II

H. Röhrig.
Tree Decomposition: A Feasibility Study.
Master's thesis, 1998.

Thanks!