Tree-decompositions with bags of small diameter

Nicolas Nisse

Université Côte d'Azur, Inria, CNRS, I3S, France

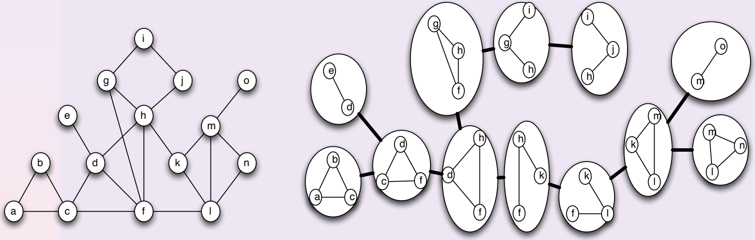
COATI Christmas seminar, December 2022

joint work with Thomas Dissaux, Guillaume Ducoffe and Simon Nivelle

Tree-Decompositions

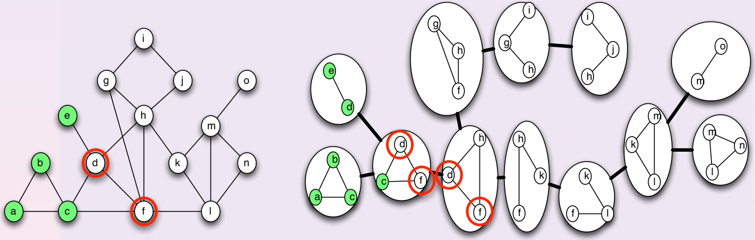
[Robertson and Seymour 83]

Tree-decomposition: Representation of a graph as a Tree with connectivity properties



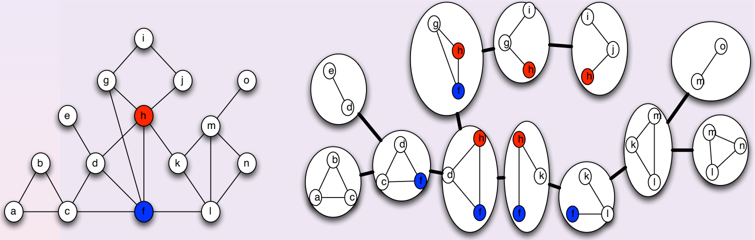
Tree T + family $\mathcal{X} = (X_t)_{t \in V(T)}$ of "bags" (sets of vertices of G)
Important: intersection of two adjacent bags = separator of G

Tree-decomposition: Representation of a graph as a Tree with connectivity properties



Tree T + family $\mathcal{X} = (X_t)_{t \in V(T)}$ of "bags" (sets of vertices of G)
Important: intersection of two adjacent bags = separator of G

Tree-decomposition: Representation of a graph as a Tree with connectivity properties

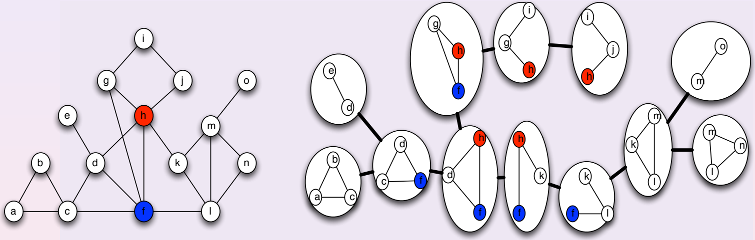


Tree T + family $\mathcal{X} = (X_t)_{t \in V(T)}$ of "bags" (sets of vertices of G)
Important: intersection of two adjacent bags = separator of G

- $\bigcup_{t \in V(T)} X_t = V(G)$;
- for any $uv \in E(G)$, there exists a bag X_t containing u and v;
- for any $v \in V(G)$, $\{t \in V(T) \mid v \in X_t\}$ induces a subtree.

Tree-Decompositions [Robertson and Seymour 83]

Tree-decomposition: Representation of a graph as a Tree with connectivity properties



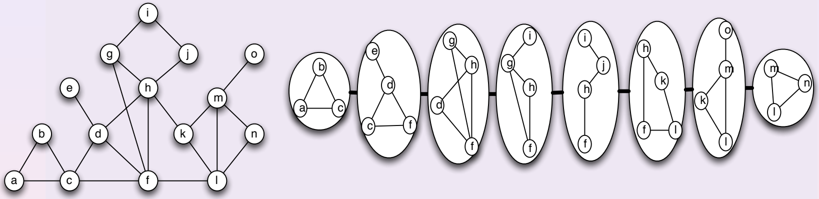
Tree T + family $\mathcal{X} = (X_t)_{t \in V(T)}$ of "bags" (sets of vertices of G)
Important: intersection of two adjacent bags = separator of G

Width of $(T, \mathcal{X})$: size of largest bag (minus 1)
Treewidth of a graph G, $tw(G)$: min width over all tree-decompositions.

Path-Decompositions

[Robertson and Seymour 83]

Path-decomposition: Representation of a graph as a Path with connectivity properties



Sequence $(X_1, \cdots, X_q)$ of "bags" (sets of vertices of G) s.t.

- $\bigcup_{1 \leq i \leq q} X_i = V(G)$;
- for any $uv \in E(G)$, there exists a bag X_i containing u and v;
- for any $1 \leq i \leq j \leq k \leq q$, $X_i \cap X_k \subseteq X_j$.

Width of $(T, \mathcal{X})$: size of largest bag (minus 1)
Pathwidth of a graph G, $pw(G)$: min width over all path-decompositions.

Many important Algorithmic Applications of *tw*

- cornerstone of Graph Minors Theorem [Robertson and Seymour 1983-2004]

  $\Rightarrow$ any graph property ($\Pi(G) \leq k$) that is closed under minor is FPT in k

- problems expressible in MSOL solvable in polynomial time in graphs of bounded treewidth (dynamic programming) [Courcelle, 90]

  any such problem is FPT in *tw*

- design of sub-exponential algorithms in some graph classes (e.g., planar, bounded genus, H-minor-free...) (bi-dimensionality) [Demaine *et al.* 04]
- design of FPT algorithms (meta-kernelization/protrusions) [Fomin *et al.* 09]

Main Problem: Computing tree-decomposition

Deciding if $tw(G) \leq k$? $\Rightarrow$ Very hard!

Exact algorithms

- NP-hard if k part of the input [Arnborg,Corneil,Prokurowski 87]
- FPT: algorithm in time $O(2^{k^3} n)$ [Bodlaender,Kloks 96]
- "practical" algorithms only for graph with treewidth ≤ 4 e.g., [Sanders 96]
- Branch & Bound algorithms (for small graphs) [Bodlaender *et al.* 12] [Coudert,Mazauric,N. 14]

Approximation algorithms

- 2-approximation in time $O(2^k n)$ [Korhonen 21]
- $\sqrt{\log OPT}$-approximation in polynomial-time (SDP) [Feige *et al.* 05]
- assuming Small Set Expansion Conjecture, no poly-time constant-ratio approximation [Wu,Austrin,Pitassi,Liu 14]
- $3/2$-approximation in planar graphs in time $O(n^3)$ [Seymour,Thomas 93]

Heuristics

- Mainly based on local complementations of edges (minimum fill-in: perfect elimination ordering of vertices) [Bodlaender *et al.*]

Approach: focus on other measure(s)

Two main problems:

What to do when the treewidth is large? how to compute "good" decompositions?

Instead of constraining the size of bags $\Rightarrow$ constraint bags' metric/structural properties

Some examples

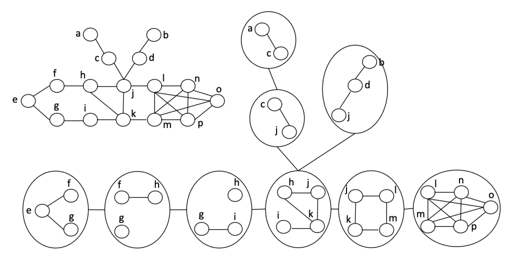
- bags' diameter (*treelength*) [Dourisboure,Gavoille 07, Lokstanov 10, Coudert,Ducoffe,N. 16]
  PTAS for TSP when bounded treelength, metric dimension FPT in treelength+max. degree...
- bags with short dominating path [Kosowski,Li,N.,Suchan 15]
  compact routing in distributed computing
- bags' chromatic number (*tree-chromatic number*) [Seymour 16]
- bags' radius (*treebreadth*) [Dragan,Köhler 14, Ducoffe,Legay,N. 20]
- bags' independence number (*tree-independence number*) [Dallard,Milanic,Storgel 21]
  Maximum Weight Independent Packing problem FPT in tree-indep. number...

In this talk, we focus on bags' diameter

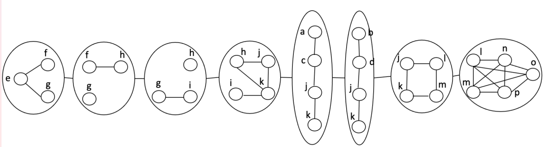
Treelength and pathlength

Length of $(T, \mathcal{X})$: $\ell(T, \mathcal{X}) = \max_{t \in V(T)} \max_{u,v \in X_t} dist_G(u, v)$. [Dourisboure,Gavoille 07]

Treelength of G, $t\ell(G)$: min. length among all tree-decompositions.



Pathlength of G, $p\ell(G)$: min. length among all path-decompositions. [Dragan,Köhler 14]



Tree/path-length *vs.* Tree/path-width

Incomparable in general:

- **Cliques:**
- **Cycles:**

Tree/path-length *vs.* Tree/path-width

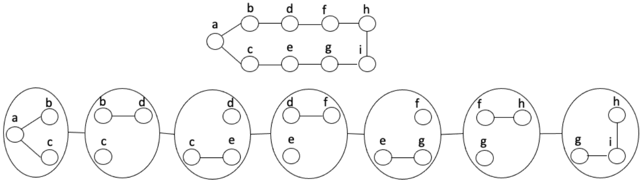
Incomparable in general:

- **Cliques:** width arbitrary larger than length
  $t\ell(K_n) = p\ell(K_n) = 1$ $tw(K_n) = pw(K_n) = n-1.$
- **Cycles:**

Tree/path-length *vs.* Tree/path-width

Incomparable in general:

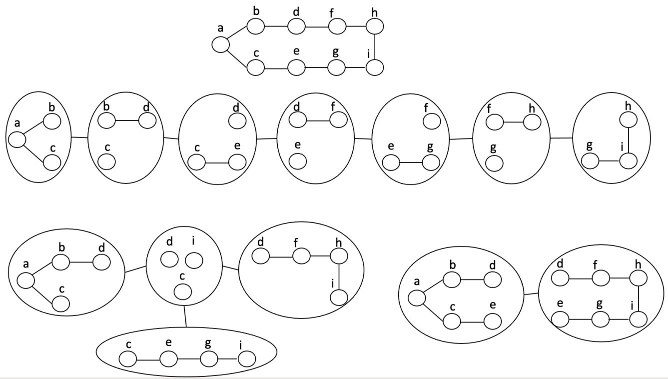
- **Cliques:** width arbitrary larger than length
  $t\ell(K_n) = p\ell(K_n) = 1$ $tw(K_n) = pw(K_n) = n-1$.
- **Cycles:**
  $tw(C_n) = pw(C_n) = 2$.



Tree/path-length *vs.* Tree/path-width

Incomparable in general:

- **Cliques:** width arbitrary larger than length
  $t\ell(K_n) = p\ell(K_n) = 1$ $\quad$ $tw(K_n) = pw(K_n) = n-1$.
- **Cycles:** length arbitrary larger than width
  $t\ell(C_n) = \lceil \frac{n}{3} \rceil$ [DG07] and $p\ell(C_n) = \lfloor \frac{n}{2} \rfloor$ [Dissaux,N. 22] $\quad$ $tw(C_n) = pw(C_n) = 2$.



Tree/path-length *vs.* Tree/path-width

Incomparable in general:

- **Cliques:** width arbitrary larger than length
  $t\ell(K_n) = p\ell(K_n) = 1$ $\quad tw(K_n) = pw(K_n) = n - 1.$
- **Cycles:** length arbitrary larger than width
  $t\ell(C_n) = \lceil \frac{n}{3} \rceil$ [DG07] and $p\ell(C_n) = \lfloor \frac{n}{2} \rfloor$ [Dissaux,N. 22] $\quad tw(C_n) = pw(C_n) = 2.$

A subgraph H of a graph G is **isometric** if the distances are "preserved".
That is, if $dist_H(u, v) = dist_G(u, v)$ for every $u, v \in V(H)$.

Tree/path-length closed under taking isometric subgraph [Dourisboure,Gavoile 07]

For every isometric subgraph H of G, $t\ell(H) \leq t\ell(G)$ and $p\ell(H) \leq p\ell(G)$.

Tree/path-length *vs.* Tree/path-width

Incomparable in general:

- **Cliques:** width arbitrary larger than length
  $t\ell(K_n) = p\ell(K_n) = 1$ $tw(K_n) = pw(K_n) = n - 1$.
- **Cycles:** length arbitrary larger than width
  $t\ell(C_n) = \lceil \frac{n}{3} \rceil$ [DG07] and $p\ell(C_n) = \lfloor \frac{n}{2} \rfloor$ [Dissaux,N. 22] $tw(C_n) = pw(C_n) = 2$.

A subgraph H of a graph G is **isometric** if the distances are "preserved".
That is, if $dist_H(u, v) = dist_G(u, v)$ for every $u, v \in V(H)$.

Tree/path-length closed under taking isometric subgraph [Dourisboure,Gavoile 07]

For every isometric subgraph H of G, $t\ell(H) \leq t\ell(G)$ and $p\ell(H) \leq p\ell(G)$.

Let $is(G)$ be the length of a largest isometric cycle in G.

Corollary: For any graph G, $\lceil \frac{is(G)}{3} \rceil \leq t\ell(G)$ and $\lfloor \frac{is(G)}{2} \rfloor \leq p\ell(G)$.

Tree/path-length *vs.* Tree/path-width

Incomparable in general:

- **Cliques:** width arbitrary larger than length
  $t\ell(K_n) = p\ell(K_n) = 1$ $\quad tw(K_n) = pw(K_n) = n-1$.
- **Cycles:** length arbitrary larger than width
  $t\ell(C_n) = \lceil \frac{n}{3} \rceil$ [DG07] and $p\ell(C_n) = \lfloor \frac{n}{2} \rfloor$ [Dissaux,N. 22] $\quad tw(C_n) = pw(C_n) = 2$.

A subgraph H of a graph G is **isometric** if the distances are "preserved".
That is, if $dist_H(u,v) = dist_G(u,v)$ for every $u,v \in V(H)$.

Tree/path-length closed under taking isometric subgraph [Dourisboure,Gavoile 07]

For every isometric subgraph H of G, $t\ell(H) \leq t\ell(G)$ and $p\ell(H) \leq p\ell(G)$.

Let $is(G)$ be the length of a largest isometric cycle in G.

Corollary: For any graph G, $\lceil \frac{is(G)}{3} \rceil \leq t\ell(G)$ and $\lfloor \frac{is(G)}{2} \rfloor \leq p\ell(G)$.

Cliques and large isometric cycles are the "single" extreme cases. [Coudert,Ducoffe,N. 16]

$tl(G) = \Theta(tw(G))$ in any apex-free graph G with bounded largest isometric cycles.

Computation of tree/path-decompositions

	Treewidth $tw(G) \leq k$?	**Pathwidth** $pw(G) \leq k$?	**Treelength** $t\ell(G) \leq k$?	**Pathlength** $p\ell(G) \leq k$?
k part of the input	NP-complete [Arnborg et al. 87]			
exact FPT (parameter k)	in time $2^{O(k^3)}n$ [Bodlaender,Kloks 96]		NP-c for $k=2$ [Lokshtanov 10]	NP-c for $k=2$ [Ducoffe,Legay,N. 20]
approximation algorithms (in general graphs)	$tw \log^{\frac{1}{2}}(tw)$ in time $n^{O(1)}$ [Feige et al. 08]; $2 \cdot k$ in time $2^{O(k)}n$ [Korhonen 21]	$pw \log^{\frac{3}{2}}(pw)$ in time $n^{O(1)}$ [Feige et al. 08]	$3 \cdot t\ell$ in time $O(n)$ [Dourisboure,Gavoille 07]; no $\frac{3}{2}$-approx unless $P=NP$ [Lokshtanov 10]	$2 \cdot p\ell$ in time $O(n)$ [Dragan et al. 17]
planar graphs	Open; $\frac{3}{2}$-approx in time $O(n^3)$ [Seymour,Thomas 93]	NP-complete [Monien, [Sudborough 88]	Open	

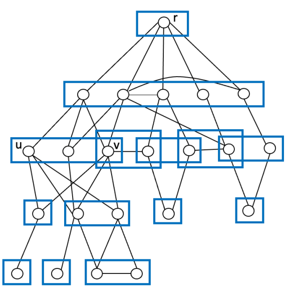
Computation of tree/path-decompositions

	Treewidth $tw(G) \leq k$?	Pathwidth $pw(G) \leq k$?	Treelength $t\ell(G) \leq k$?	Pathlength $p\ell(G) \leq k$?
k part of the input	NP-complete [Arnborg et al. 87]			
exact FPT (parameter k)	in time $2^{O(k^3)}n$ [Bodlaender,Kloks 96]		NP-c for $k=2$ [Lokshtanov 10]	NP-c for $k=2$ [Ducoffe,Legay,N. 20]
approximation algorithms (in general graphs)	$tw \log^{\frac{1}{2}}(tw)$ in time $n^{O(1)}$ [Feige et al. 08] $2 \cdot k$ in time $2^{O(k)}n$ [Korhonen 21]	$pw \log^{\frac{3}{2}}(pw)$	$3 \cdot t\ell$ in time $O(n)$ [Dourisboure,Gavoille 07] no $\frac{3}{2}$-approx unless $P = NP$ [Lokshtanov 10]	$2 \cdot p\ell$ [Dragan et al. 17]
planar graphs	Open $\frac{3}{2}$-approx in time $O(n^3)$ [Seymour,Thomas 93]	NP-complete [Monien, [Sudborough 88]	Open	

Algorithm based on a particular BFS (LexM)

Roughly: 2 vertices are in a same bag if

- they are in the same BFS-level
- there is a path between them with internal vertices further from the root



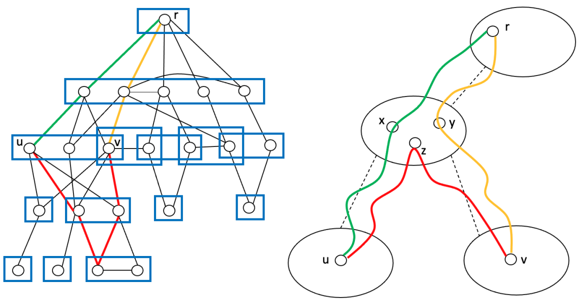
3 approximation for treelength

[Dourisboure,Gavoille 07]

Algorithm based on a particular BFS (LexM)

Roughly: 2 vertices are in a same bag if

- they are in the same BFS-level
- there is a path between them with internal vertices further from the root



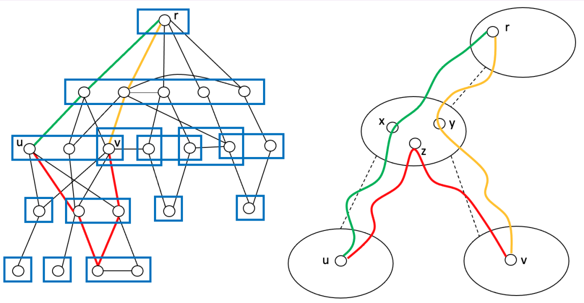
3 approximation for treelength

[Dourisboure,Gavoille 07]

Algorithm based on a particular BFS (LexM)

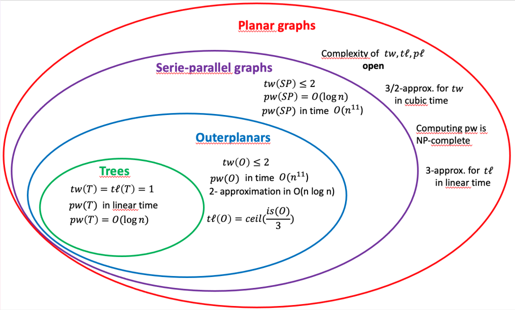
Roughly: 2 vertices are in a same bag if

- they are in the same BFS-level
- there is a path between them with internal vertices further from the root

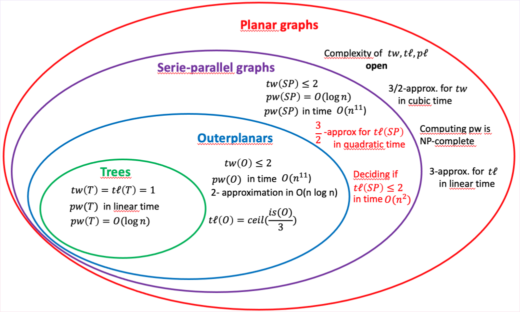


Better approximation in general graphs? in planar graphs? Use this for treewidth?

Planar graphs: known results



Planar graphs: known results and our contributions

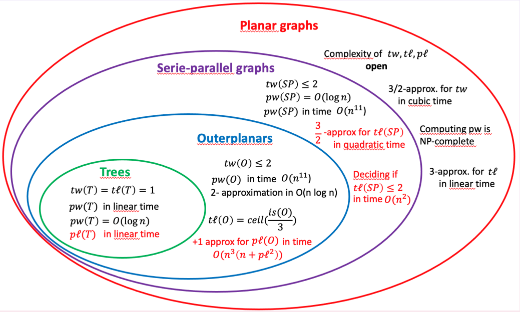


Treelength in Serie-Parallel

[Dissaux,Ducoffe,N.,Nivelle, LAGOS 21]

- $\frac{3}{2}$-approx. in $O(n^2)$-time;
- Exact for melon graphs;
- Characterization of SP graphs G s.t. $t\ell(G) \leq 2$.

Planar graphs: known results and our contributions



Treelength in Serie-Parallel

[Dissaux,Ducoffe,N.,Nivelle, LAGOS 21]

- $\frac{3}{2}$-approx. in $O(n^2)$-time;
- Exact for melon graphs;
- Characterization of SP graphs G s.t. $t\ell(G) \leq 2$.

Pathlength in Outerplanars

[Dissaux,N., LATIN 22]

- $p\ell(T)$ in linear time in trees;
- Cycles: $pl(C_n) = \lfloor \frac{n}{2} \rfloor$;
- $(+1)$-approximation in poly-time.

Treelength in Serie-Parallel graphs

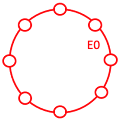
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



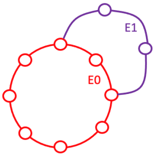
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



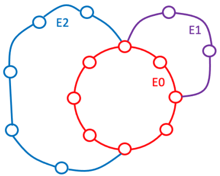
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



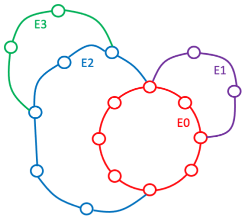
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



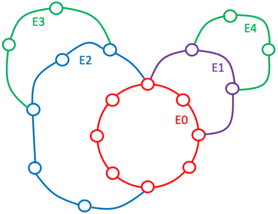
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



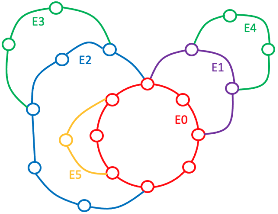
(2-connected) Serie-Parallel graphs

Serie parallel = K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



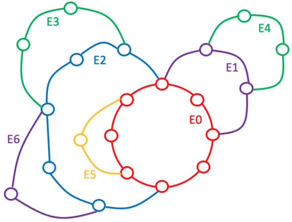
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



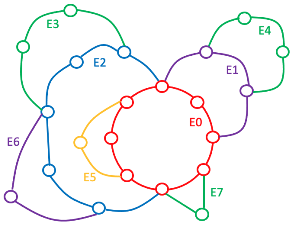
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



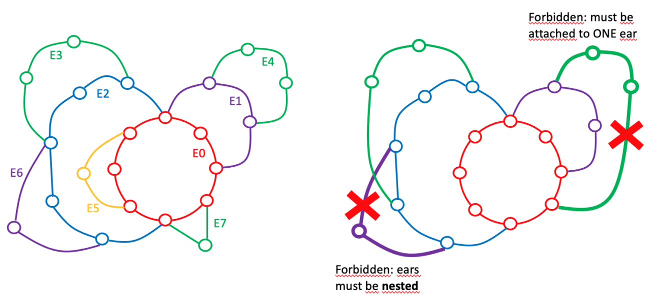
(2-connected) Serie-Parallel graphs

Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ Nested Ear decomposition [Eppstein 92]

Recursive construction:

- Start with graph G_0 that consists of a cycle E_0;
- At step $i > 0$, obtain G_i by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.

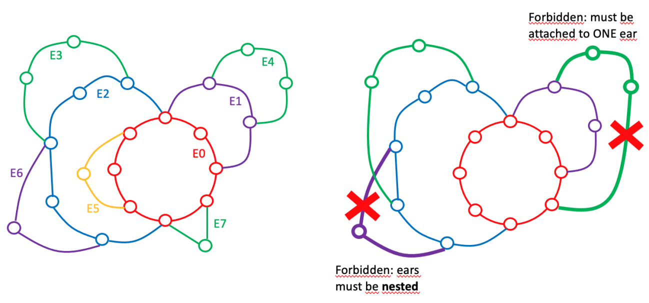


Serie parallel $=$ K_4-minor free graphs

G serie-parallel $\Leftrightarrow$ isometric Nested Ear decomposition **[Dissaux,Ducoffe,N.,Nivelle 21]**

Recursive construction:

- Start with graph G_0 that consists of a **largest isometric** cycle E_0;
- At step $i > 0$, obtain G_i, **isometric subgraph**, by adding an ear E_i (a path) attached, in a nested way, to a previous ear E_j, $j < i$.



Isometric nested Ear decomposition can be computed in time $O(n^2)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.

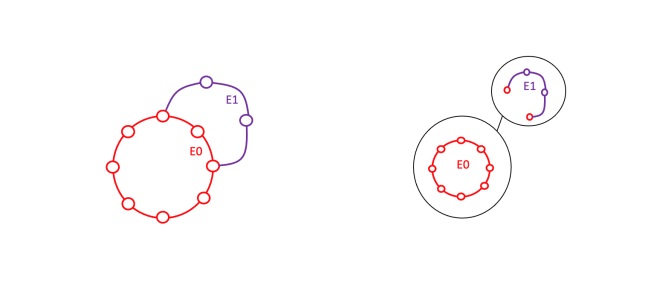
Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.



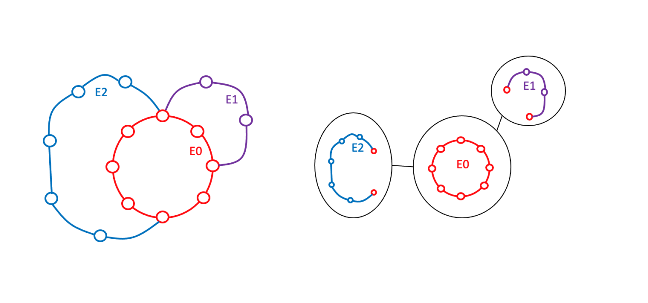
Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.



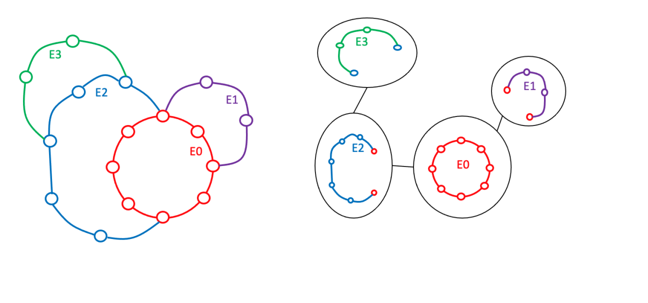
Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.



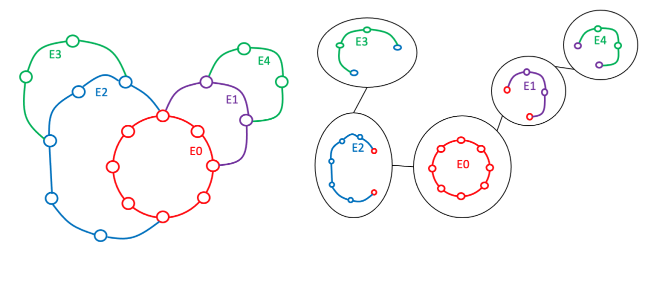
Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.



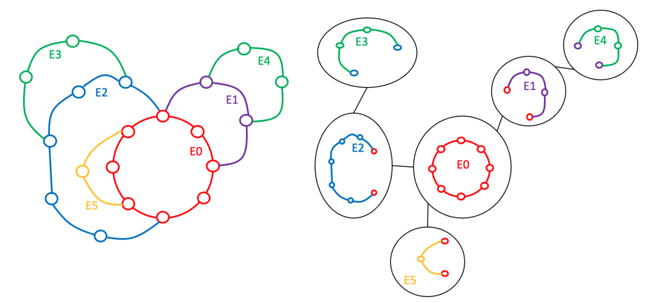
Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.



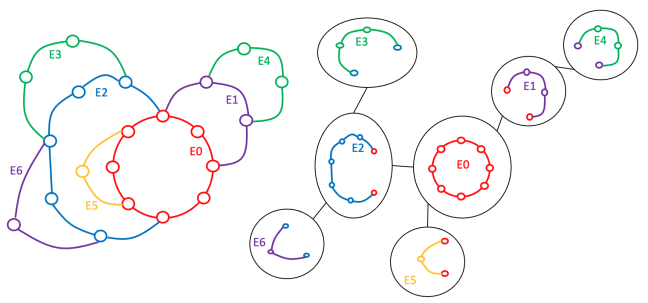
Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.



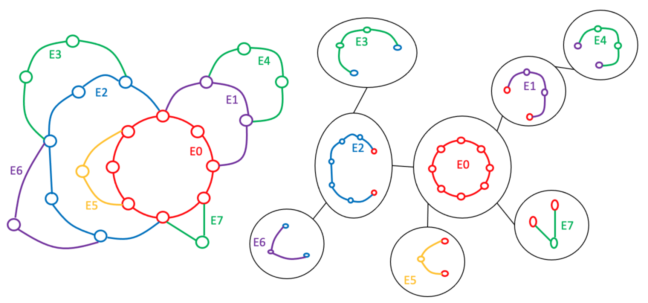
Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

$\frac{3}{2}$-approximation for $t\ell$ in Serie-Parallel graphs

Very simple algorithm [Dissaux,Ducoffe,N.,Nivelle 21]

Let $(E_0, \cdots, E_p)$ an isometric nested ear decomposition of a Serie-parallel graph G

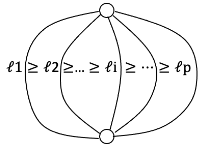
- Start with gone bag B_0 containing E_0;
- For $i = 1$ to p, add a bag B_i containing E_i and adjacent to a bag B_j that contains an ear E_j, $j < i$, to which E_i is attached.



Each bag $\approx$ subgraph of an isometric cycle $\Rightarrow$ length $\leq \frac{is(G)}{2}$. Recall, $\frac{is(G)}{3} \leq t\ell(G)$.

The simplest (?) subclass of Serie-Parallel graphs

Melon graph: paths linking two vertices



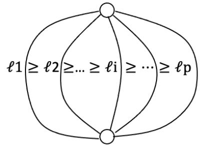
Theorem: [Dissaux,Ducoffe,N.,Nivelle LAGOS 21]

Let G be a melon graph with paths of lengths $\ell_1 \geq \cdots \geq \ell_p$

- $t\ell(G) = \lceil \frac{\ell_1+\ell_p}{3} \rceil = \lceil \frac{is(G)}{3} \rceil$ if $\ell_p \leq \lceil \frac{\ell_1+\ell_p}{3} \rceil$;
- $t\ell(G) = \ell_p$ if $\lceil \frac{\ell_1+\ell_p}{3} \rceil \leq \ell_p \leq \lceil \frac{\ell_1+\ell_2}{3} \rceil$;
- $t\ell(G) = \lceil \frac{\ell_1+\ell_2}{3} \rceil$ otherwise.

The simplest (?) subclass of Serie-Parallel graphs

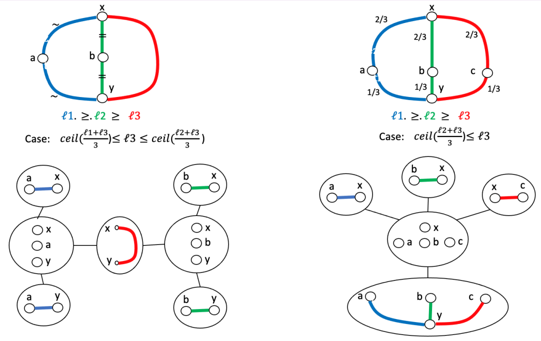
Melon graph: paths linking two vertices



Theorem: [Dissaux,Ducoffe,N.,Nivelle LAGOS 21]

Let G be a melon graph with paths of lengths $\ell_1 \geq \cdots \geq \ell_p$

- $t\ell(G) = \lceil \frac{\ell_1+\ell_p}{3} \rceil = \lceil \frac{is(G)}{3} \rceil$ if $\ell_p \leq \lceil \frac{\ell_1+\ell_p}{3} \rceil$;
- $t\ell(G) = \ell_p$ if $\lceil \frac{\ell_1+\ell_p}{3} \rceil \leq \ell_p \leq \lceil \frac{\ell_1+\ell_2}{3} \rceil$;
- $t\ell(G) = \lceil \frac{\ell_1+\ell_2}{3} \rceil$ otherwise.



Deciding if $t\ell(G) \leq 2$ in Serie-Parallel graphs

.

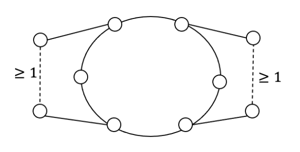
Characterization by forbidden isometric subgraphs [Dissaux,Ducoffe,N.,Nivelle LAGOS 21]

Let G be a Serie-Parallel graphs. Then, $t\ell(G) \leq 2$ if and only if $is(G) \leq 6$ and G has no Dumbo graph as isometric subgraph.

Polynomial-time algorithm that, given G Serie-parallel:

- either returns an isometric cycle larger than 6 or an isometric Dumbo subgraph;
- or compute a tree-decomposition of G of length at most 2.

Dumbo graph:



Deciding if $t\ell(G) \leq 2$ in Serie-Parallel graphs

.

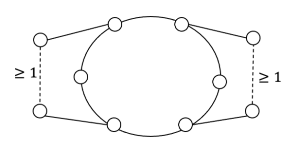
Characterization by forbidden isometric subgraphs [Dissaux,Ducoffe,N.,Nivelle LAGOS 21]

Let G be a Serie-Parallel graphs. Then, $t\ell(G) \leq 2$ if and only if $is(G) \leq 6$ and G has no Dumbo graph as isometric subgraph.

Polynomial-time algorithm that, given G Serie-parallel:

- either returns an isometric cycle larger than 6 or an isometric Dumbo subgraph;
- or compute a tree-decomposition of G of length at most 2.

Dumbo graph:



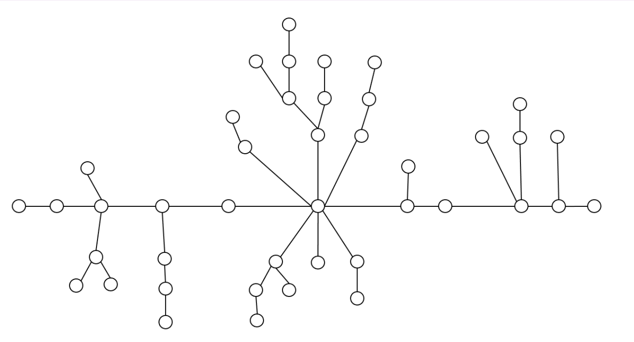
Proof:

- by induction on the number of Ears;
- must ensure that: if a forthcoming ear is attached to two vertices x and y, then there is a bag containing them;
- tedious case analysis depending on the length of the ears.

Pathlength in Outerplanar graphs

Pathlength of trees

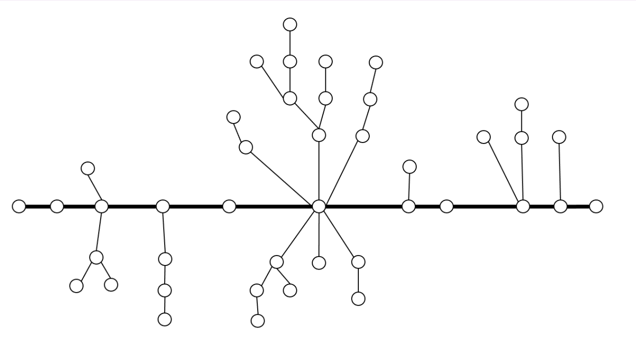
Linear time algorithm for any tree T [Dissaux,N. LATIN 22]



Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

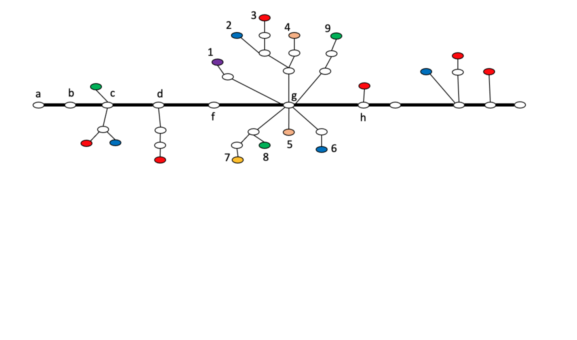
Let $D = (v_1, \cdots, v_d)$ be a diameter. .



Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

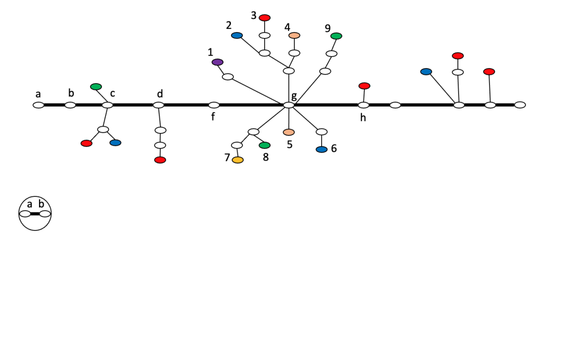
Let $D = (v_1, \cdots, v_d)$ be a diameter. .



Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.

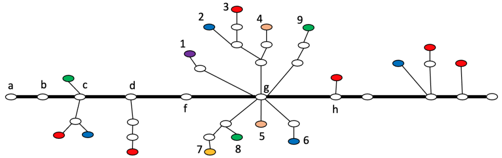


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

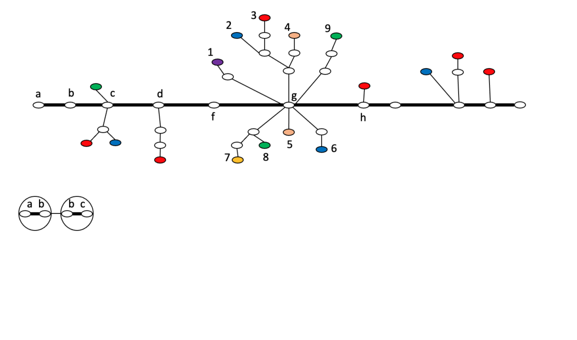


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

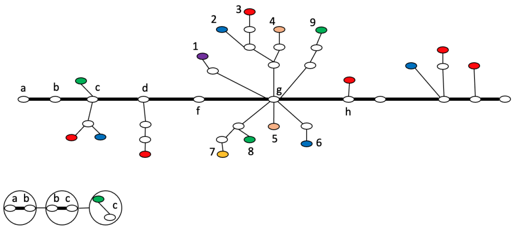


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

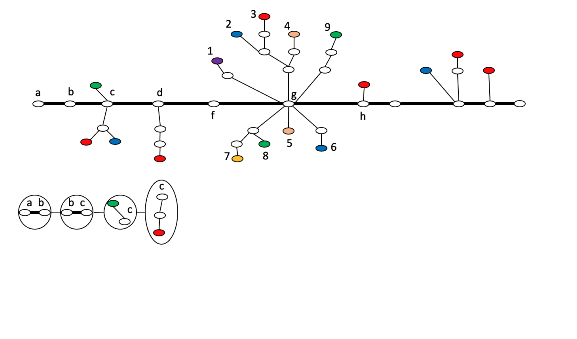


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

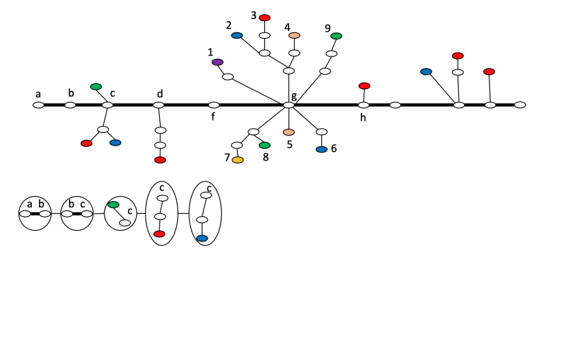


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

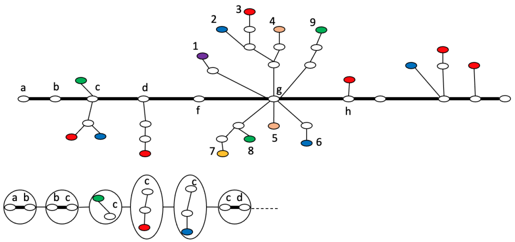


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

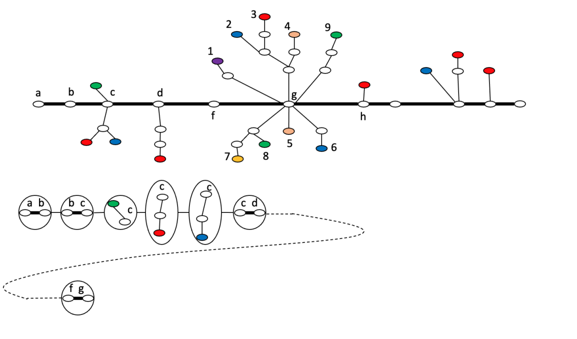


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

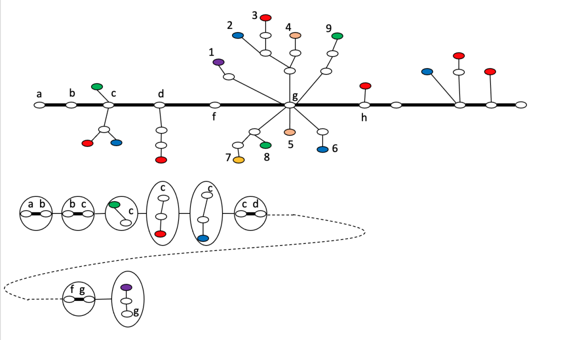


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

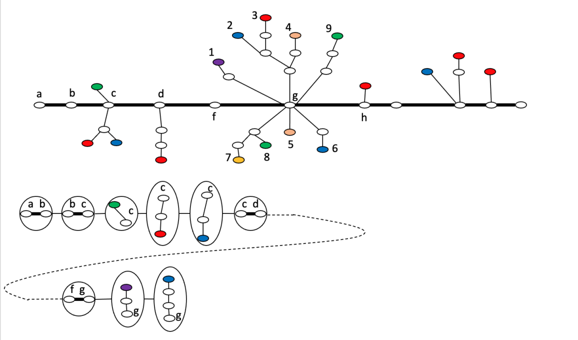


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

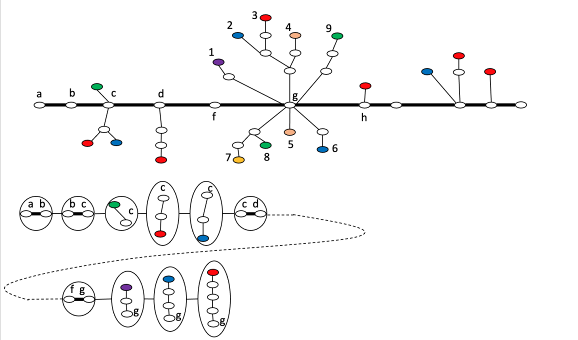


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

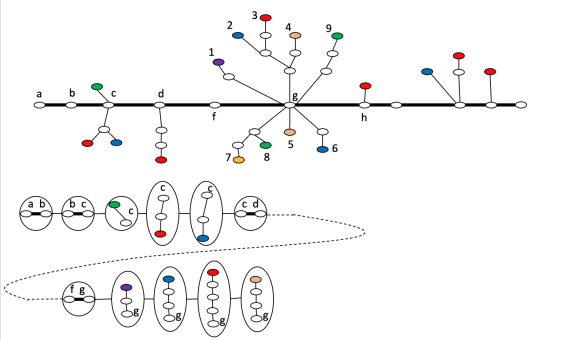


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

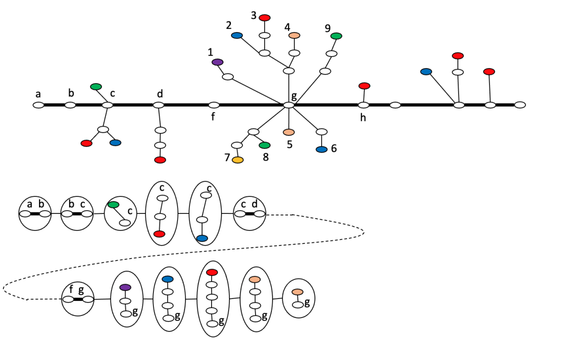


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

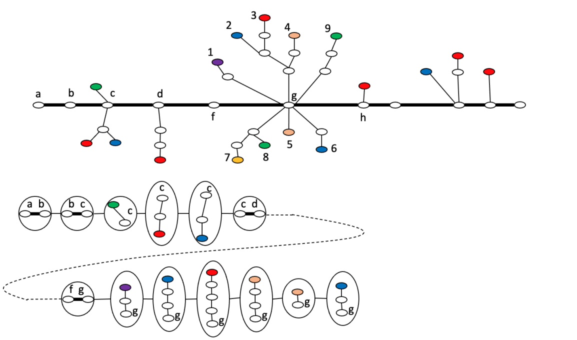


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

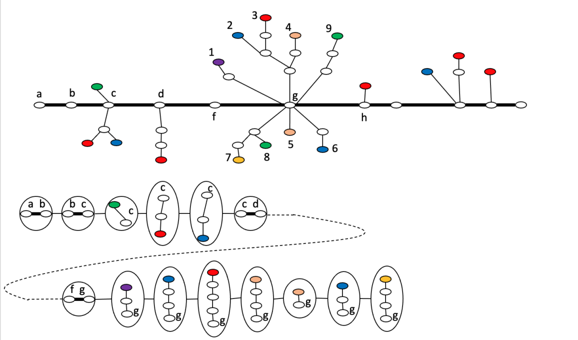


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

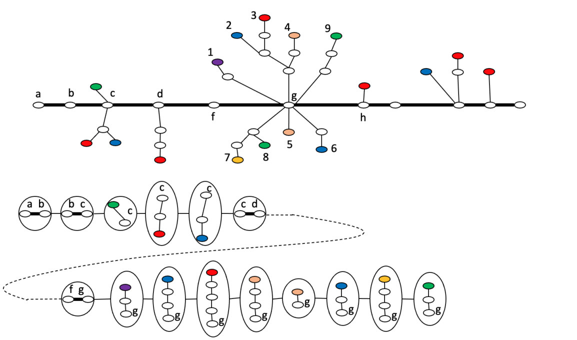


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

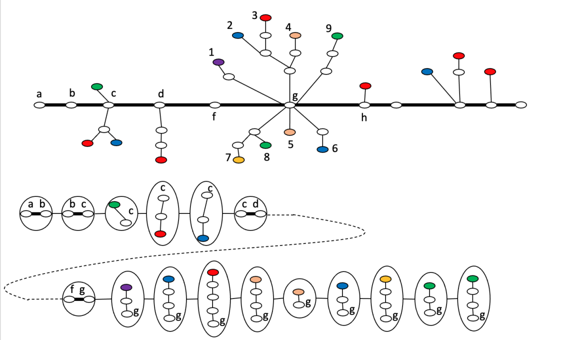


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

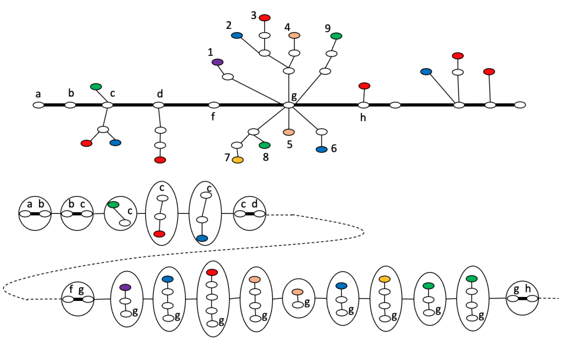


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

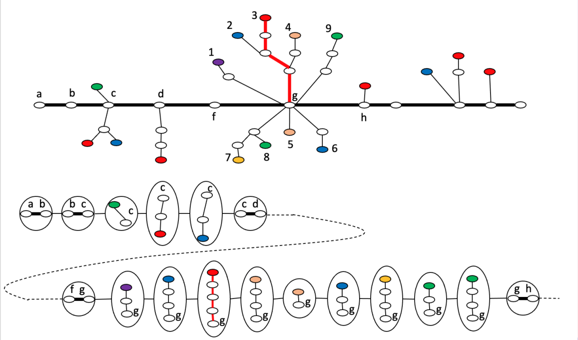


Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.

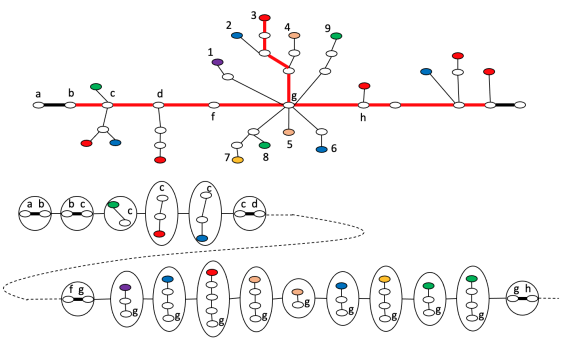


Length:
$k = \max_{v \in V(T)} dist(v, D)$
so $p\ell(T) \leq k$

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.



Length:
$k = \max_{v \in V(T)} dist(v, D)$
so $p\ell(T) \leq k$

T contains the star S_k with 3 branches of length k as isometric subgraph.

$p\ell(S_k) = k$ [DG 07]

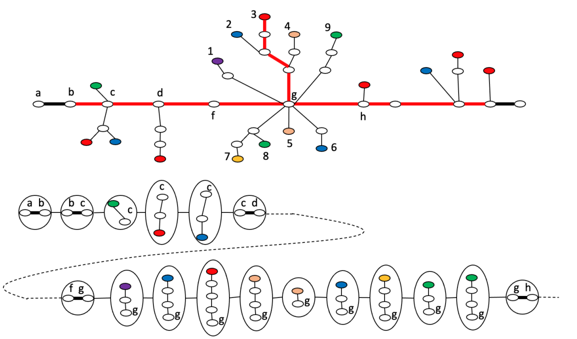
So, $p\ell(T) \geq k$.

Pathlength of trees

Linear time algorithm for any tree T [Dissaux,N. LATIN 22]

Let $D = (v_1, \cdots, v_d)$ be a diameter. Start with one bag $\{v_1, v_2\}$.
When arriving at a bag $\{v_{i-1}, v_i\}$:

- order the leaves "around" v_i in any DFS manner;
- $\forall$ path P_f from v_i to a leaf f "around" it, in the DFS order, add a bag $V(P_f)$
- then, add one bag $\{v_i, v_{i+1}\}$.



Length:
$k = \max_{v \in V(T)} dist(v, D)$
so $p\ell(T) \leq k$

T contains the star S_k with 3 branches of length k as isometric subgraph.

$p\ell(S_k) = k$ [DG 07]

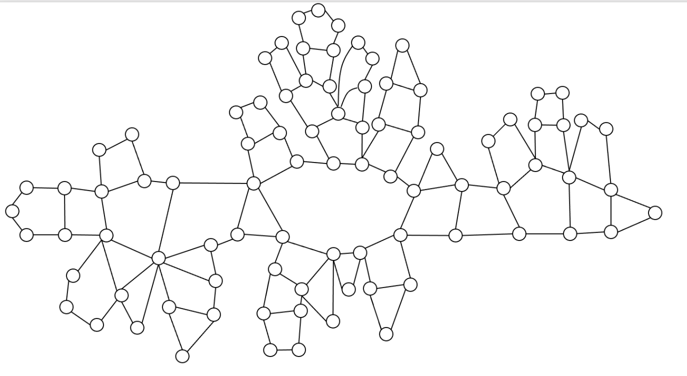
So, $p\ell(T) \geq k$.

$\Rightarrow p\ell(T) = k$.

Pathlength of Outerplanar graphs

Outerplanar: $K_4, K_{2,3}$ minor-free $\Leftrightarrow \exists$ planar embedding with all vertices on outer-face

Example of 2-connected outerplanar:



Pathlength of Outerplanar graphs

Outerplanar: $K_4, K_{2,3}$ minor-free $\Leftrightarrow \exists$ planar embedding with all vertices on outer-face

Let $k \geq 0$. There exists an algorithm that: [Dissaux,N. LATIN 22]

given an outerplanar graph G, in time $O(n^3(n+k^2))$,

- either returns a path-decomposition of length $\leq k+1$,
- or states that $p\ell(G) > k$.

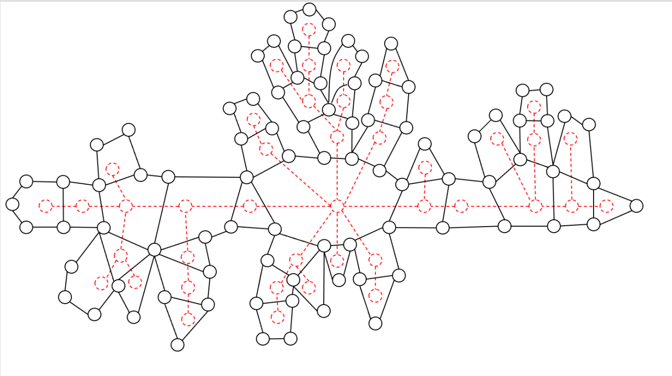
Two steps:

1. Show that, for every $k \geq p\ell(G)$, there exists a path-decomposition of length $\leq k+1$ with "good" properties;
2. Compute such a decomposition in polynomial-time.

Open: Does there exist an exact polynomial-time algorithm?

Pathlength of Outerplanar graphs: use the dual

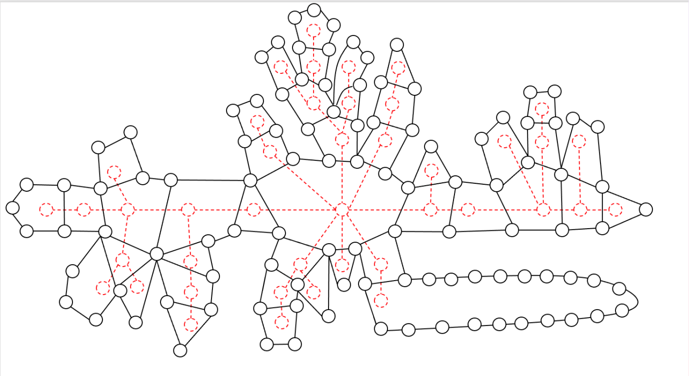
Weak dual of outerplanar graph G is a tree G^*
Idea: "Mimic" the strategy on trees: follow a diameter, add "branches" in this order.



Pathlength of Outerplanar graphs: use the dual

Weak dual of outerplanar graph G is a tree G^*
Idea: "Mimic" the strategy on trees: follow a diameter, add "branches" in this order.
Problem: no relation between diameters of G and G^*

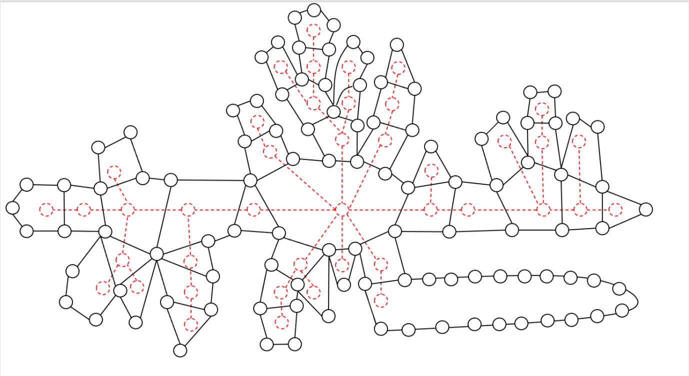
Which "main" path to follow?



Pathlength of Outerplanar graphs: use the dual

Weak dual of outerplanar graph G is a tree G^*
Idea: "Mimic" the strategy on trees: follow a diameter, add "branches" in this order.
Problem: no relation between diameters of G and G^*

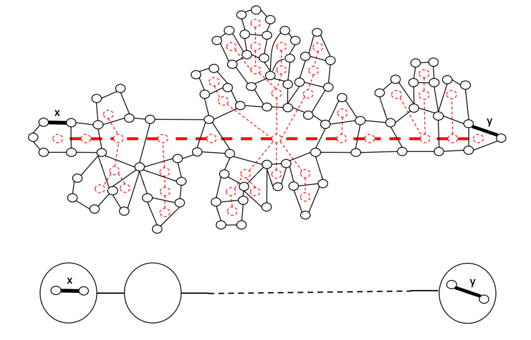
Which "main" path to follow? **We will try them all!**



(x, y)-Path-Decomposition

Let $x, y \in E(G)$. (x, y)-Path-Decomposition: x in the first bag, y in the last bag.
$p\ell(G, x, y)$: minimum length of a (x, y)-Path-Decomposition of G.

Lemma: $p\ell(G) = \min_{x,y \in E(G)} p\ell(G, x, y)$.



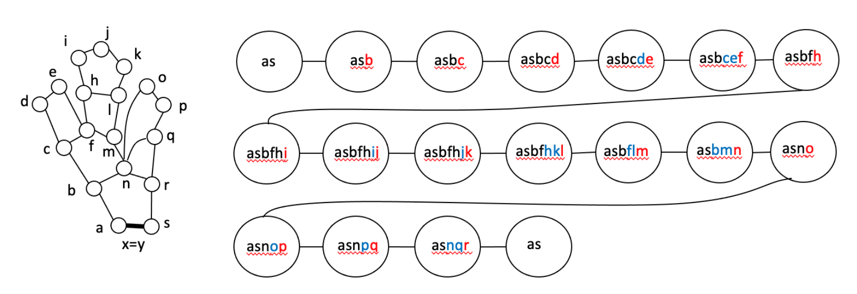
We will try to compute an optimal (x, y)-Path-Decomposition for every $x, y \in E(G)$.
("only" $O(n^2)$ possibilities).

(x, x)-Path-Decomposition: greedy algorithm

Computation of (x, y)-Path-Decomposition: Case $x = y = \{a, s\}$.

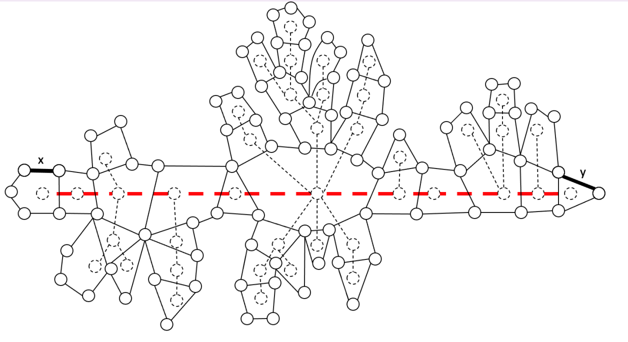
Greedy algorithm: add the vertices in the path-decomposition P in a DFS ordering (from x) guided by the outer-face.

Lemma: $length(P) = \max_{v \in V(G)} \max\{dist(a, v), dist(s, v)\} \leq p\ell(G, x, x)$.



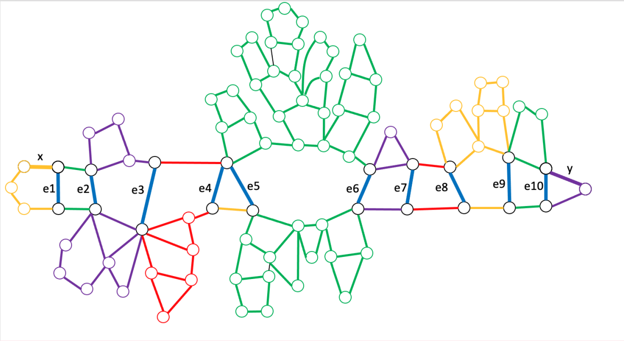
(x, y)-Path-Decomposition: x, y not in a same face

$x \neq y \in E(G) \Rightarrow$ define a path in the dual



(x, y)-Path-Decomposition: x, y not in a same face

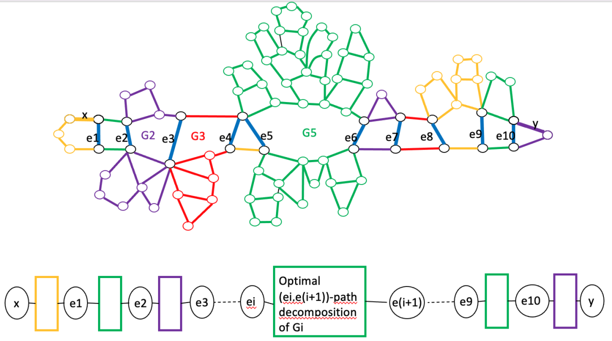
$x \neq y \in E(G) \Rightarrow$ define a path in the dual
Case $x \neq y \in E(G)$, and there are $e_1, \cdots, e_q$ edge separators of x and y.



(x, y)-Path-Decomposition: x, y not in a same face

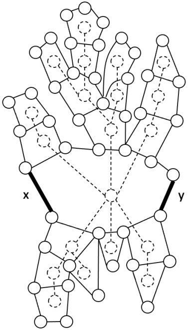
$x \neq y \in E(G) \Rightarrow$ define a path in the dual
Case $x \neq y \in E(G)$, and there are $e_1, \cdots, e_q$ edge separators of x and y.

Lemma: If $x, y \in E(G)$ not in the same face, there exists an (x, y)-Path-Decomposition with length $p\ell(G, x, y)$ which is "well separated".



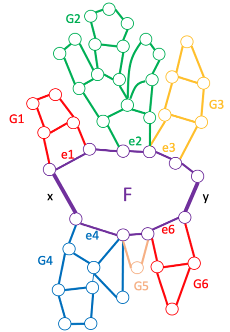
(x, y)-Path-Decomposition: x, y in a same face

$x \neq y \in E(F)$ for some face F.
Components of $G \setminus F$: "branches".



(x, y)-Path-Decomposition: x, y in a same face

$x \neq y \in E(F)$ for some face F.
Components of $G \setminus F$: "branches".

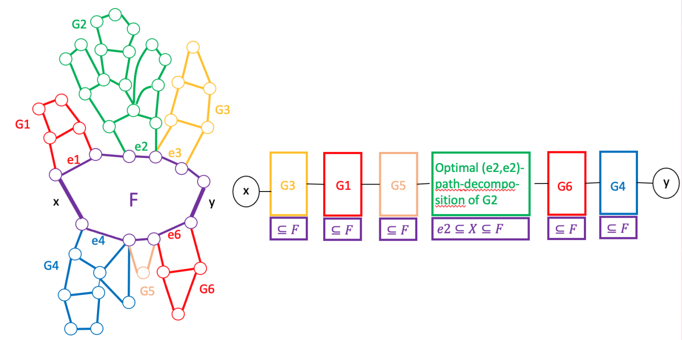


(x, y)-Path-Decomposition: x, y in a same face

$x \neq y \in E(F)$ for some face F.
Components of $G \setminus F$: "branches".

Lemma: If $x, y \in E(F)$, there exists an (x, y)-Path-Decomposition with length $p\ell(G, x, y) + 1$ which "proceeds branch by branch".
Moreover, for each branch G_i, it is a greedy (e_i, e_i)-path-decomposition.

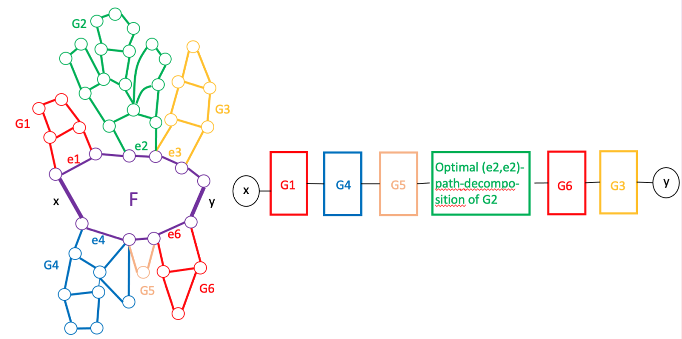
Can we guess the ordering of the "branches"?



(x, y)-Path-Decomposition: x, y in a same face

$x \neq y \in E(F)$ for some face F.
Components of $G \setminus F$: "branches".

Lemma: If $x, y \in E(F)$, there exists an (x, y)-Path-Decomposition with length $p\ell(G, x, y) + 1$ which "proceeds branch by branch", from left to right. Moreover, for each branch G_i, it is a greedy (e_i, e_i)-path-decomposition.

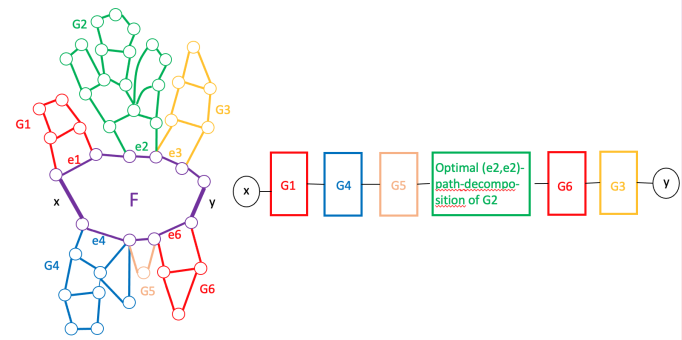


(x,y)-Path-Decomposition: x,y in a same face

$x \neq y \in E(F)$ for some face F.
Components of $G \setminus F$: "branches".

Lemma: If $x, y \in E(F)$, there exists an (x,y)-Path-Decomposition with length $p\ell(G,x,y) + 1$ which "proceeds branch by branch".
Moreover, for each branch G_i, it is a greedy (e_i, e_i)-path-decomposition.

By dynamic programming, in time $O(n + |F|^2) \leq n + is(G)^2 = O(n + p\ell(G)^2)$.



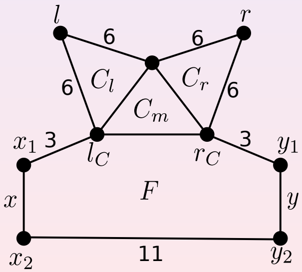
Pathlength of Outerplanar graphs

Let $k \geq 0$. There exists an algorithm that: [Dissaux,N. LATIN 22]

given an outerplanar graph G, in time $O(n^3(n+k^2))$,

- either returns a path-decomposition of length $\leq k+1$,
- or states that $p\ell(G) > k$.

Remark: the $+1$ cannot be avoided in our algorithm.



Further work

Treelength:

- Complexity in Serie-parallel graphs?
- Complexity in Planar graphs?
- Complexity in bouded treewidth graphs?
- New algorithmic applications?

Pathlength:

- Complexity in Outerplanar graphs?
- Complexity in Serie-parallel graphs?
- Complexity in Planar graphs?
- Complexity in bouded treewidth graphs?
- New algorithmic applications?

Treewidth:

- Complexity in Planar graphs?

In general, better practical approximation algorithms?

Further work

Treelength:

- Complexity in Serie-parallel graphs?
- Complexity in Planar graphs?
- Complexity in bouded treewidth graphs?
- New algorithmic applications?

Pathlength:

- Complexity in Outerplanar graphs?
- Complexity in Serie-parallel graphs?
- Complexity in Planar graphs?
- Complexity in bouded treewidth graphs?
- New algorithmic applications?

Treewidth:

- Complexity in Planar graphs?

In general, better practical approximation algorithms?

Thank you!