Graph decompositions of small diameter

T. Dissaux

Université Côte d'Azur, Inria, CNRS, I3S, France
Joint work with Nicolas Nisse

GRASTA 2022

May 19th, 2022

Graph and tree-decomposition

Many problems of real life can be modeled and solved by **graphs**.
Often difficult **(NP-hard)**.

Prolific approach:

Dynamic programming using **Tree-decompositions** (widely studied during last 40 years).

Examples:

- **[Courcelle, 1990]** Every graph property definable in the monadic second-order logic of graphs can be decided in linear time on graphs of bounded treewidth.
- **[Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh, Thilikos, 2016]** (Meta) kernelization.

E.g., Max clique, Minimum Graph colorability, Gate Matrix Layout, Maximum Independent vertex Set, Minimum Fill-in, Minimum dominating set...

Tree-decompositions [Robertson, Seymour, 1984]

Idea: Decompose a graph into small pieces (**bags**), structured as a tree.

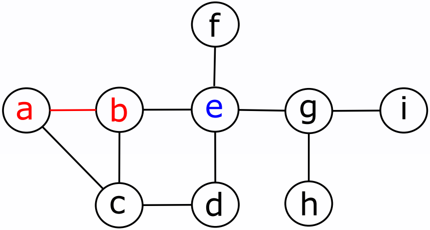
Tree-decomposition of $G = (V, E)$

A *tree-decomposition* T is a family $\mathcal{X} = \{X_t | t \in V(T)\}$ of bags s.t.:

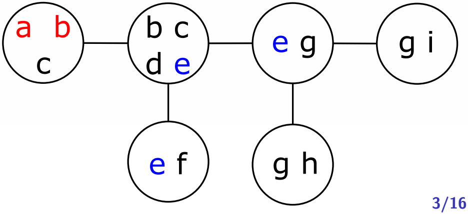
- $\bigcup_{t \in V(T)} X_t = V(G)$;
- $\forall \{u, v\} \in E(G)$, $\exists t \in V(T)$ s.t. that $u, v \in X_t$;
- $\forall v \in V(G)$, $\{t \in V(T) \mid v \in X_t\}$ induces a tree.

Width: max. size of the bags minus 1.
Treewidth, $tw(G)$: min. width among all tree-decompositions.

A graph G:



A tree-decomposition of G:



Tree-decompositions [Robertson, Seymour, 1984]

Idea: Decompose a graph into small pieces (**bags**), structured as a tree.

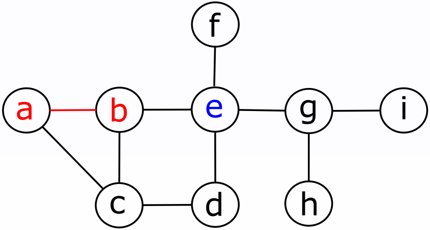
Tree-decomposition of $G = (V, E)$

A *tree-decomposition* T is a family $\mathcal{X} = \{X_t | t \in V(T)\}$ of bags s.t.:

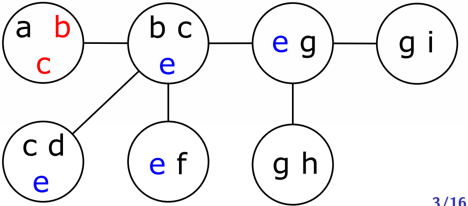
- $\bigcup_{t \in V(T)} X_t = V(G)$;
- $\forall \{u, v\} \in E(G)$, $\exists t \in V(T)$ s.t. that $u, v \in X_t$;
- $\forall v \in V(G)$, $\{t \in V(T) \mid v \in X_t\}$ induces a tree.

Width: max. size of the bags minus 1.
Treewidth, $tw(G)$: min. width among all tree-decompositions.

A graph G:



A tree-decomposition of G:



Treelength [Dourisboure, Gavoille, 2007]

Length: max. **diameter** of the bags.
Treelength, $tl(G)$: min. length among all tree-decompositions.

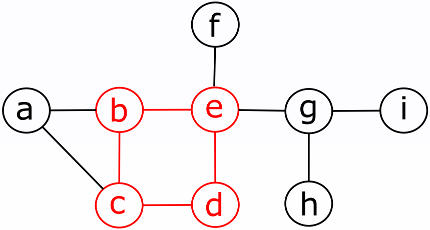
Applications:

TSP [Krauthgamer, Lee, 2006], Compact routing [Boguna, Papadopoulos, Krioukov, 2010]...

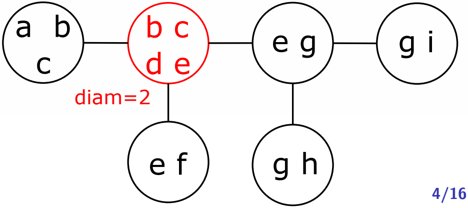
Remark:

Note that the red cycle $\{b, c, d, e\}$ can be used to prove that $tl(G) \geq 2$.

A graph G:



A tree-decomposition of G:



Path-decompositions

Path-decomposition of $G = (V, E)$

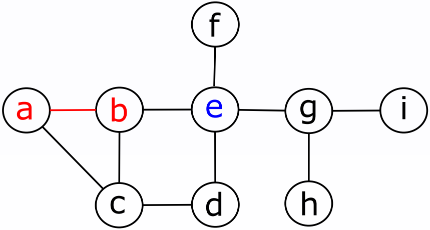
A *path-decomposition* P is a sequence $(X_1, \dots, X_p)$ of bags s.t.:

- $\bigcup_{1 \leq i \leq p} X_i = V(G)$;
- $\forall \{u, v\} \in E(G)$, $\exists 1 \leq i \leq p$ s.t. that $u, v \in X_i$;
- $\forall 1 \leq i \leq z \leq j \leq p$, $X_i \cap X_j \subseteq X_z$

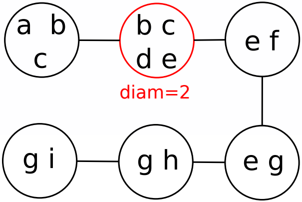
Width: max. size of the bags minus 1.
Pathwidth, $pw(G)$: min. width among all path-decompositions.

Length: max. diameter of the bags.
Pathlength, $pl(G)$: min. length among all path-decompositions.

A graph G:



A tree-decomposition of G:



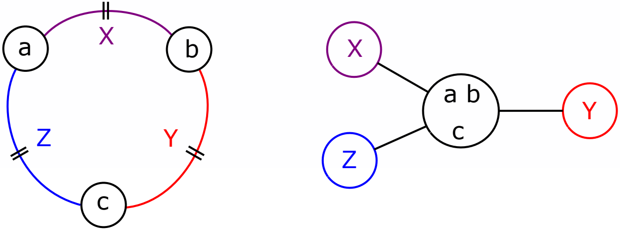
Difference between treewidth and treelength

Lemma :

$tl(K_n) = pl(K_n) = 1$ and $tw(K_n) = pw(K_n) = n-1$.

Lemma [Dourisboure,Gavoille, 2007]:

$tl(C_k) = \lceil \frac{k}{3} \rceil$ and $tw(C_k) = 2$. $pl(C_k) = \lfloor \frac{k}{2} \rfloor$ and $pw(C_k) = 2$.



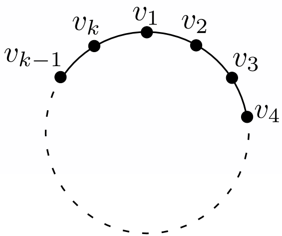
Difference between treewidth and treelength

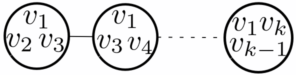
Lemma :

$tl(K_n) = pl(K_n) = 1$ and $tw(K_n) = pw(K_n) = n - 1$.

Lemma [Dourisboure,Gavoille, 2007]:

$tl(C_k) = \lceil \frac{k}{3} \rceil$ and $tw(C_k) = 2$. $pl(C_k) = \lfloor \frac{k}{2} \rfloor$ and $pw(C_k) = 2$.





Tree-decomposition of cycle, a good lowerbound

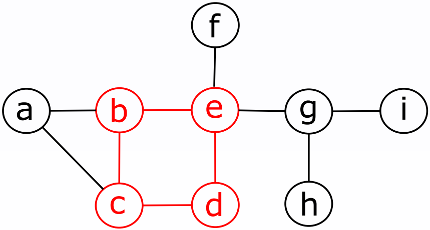
- $tl(C_k) = \lceil \frac{k}{3} \rceil$
- $pl(C_k) = \lfloor \frac{k}{2} \rfloor$

Lemma [Dourisboure,Gavoille, 2007]:

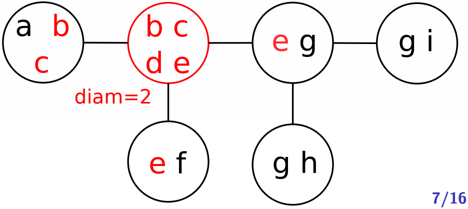
Let G be a graph and H an **isometric** (i.e. preserve distance) subgraph of G, then $tl(G) \geq tl(H)$.

Good lowerbound: $tl(G) \geq \lceil \frac{is(G)}{3} \rceil$
($is(G)$: largest size of an isometric cycle)

A graph G:



A tree-decomposition of G:



Tree-decomposition of cycle, a good lowerbound

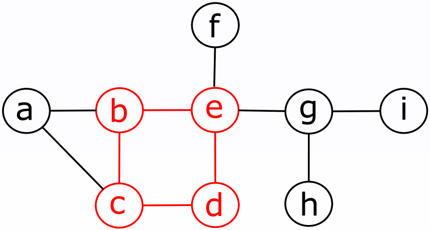
- $tl(C_k) = \lceil \frac{k}{3} \rceil$
- $pl(C_k) = \lfloor \frac{k}{2} \rfloor$

Lemma [Dourisboure,Gavoille, 2007]:

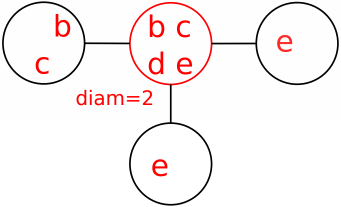
Let G be a graph and H an **isometric** (i.e. preserve distance) subgraph of G, then $tl(G) \geq tl(H)$.

Good lowerbound: $tl(G) \geq \lceil \frac{is(G)}{3} \rceil$ ($is(G)$: largest size of an isometric cycle)

A graph G:



A tree-decomposition of C_4:



Computation of tree/path-decompositions

	Treewidth: $tw(G) \leq k$	**Treelength:** $tl(G) \leq k$	**Pathwidth:** $pw(G) \leq k$	**Pathlength:** $pl(G) \leq k$
k part of the input	NP-complete [Arnborg and al, 1987]	NP-complete	NP-complete [Arnborg and al, 1987]	NP-complete
k fixed parameter	Polynomial [Bodlaender, Kloks, 1996]	NP-complete [Lokshtanov, 2010]	Polynomial [Bodlaender, Kloks, 1996]	NP-complete [Ducoffe and al, 2020]
Approximation algorithm	$\sqrt{log(Opt)}$ [Feige and al 2008]	3 [Dourisboure, Gavoille, 2007]	$O(log(n)^{\frac{3}{2}})$ [Feige and al, 2008]	2 [Dragan and al, 2017]
Planar graph	Open since 40 years	Open	NP-complete [Monien, Sudborough, 1988]	Open

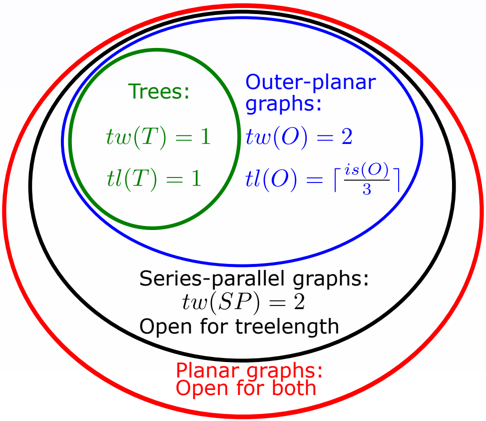
To approximate treewidth, use treelength ! [Coudert, Ducoffe, Nisse, 2016]

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

=> Study of treelength/pathlength in planar graphs

Our Contributions

Sub-classes of planar graphs:

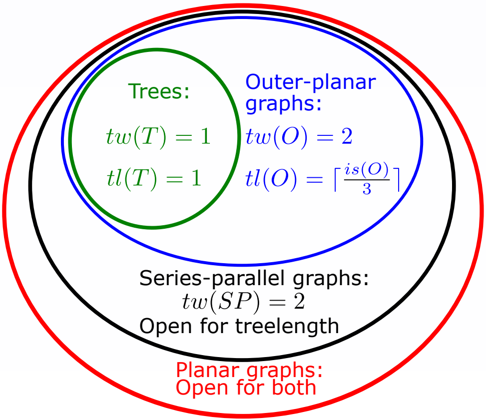


Treelength [T. Dissaux, G.Ducoffe, N. Nisse, S. Nivelle, 2021]

- $\frac{3}{2}$-Approximation Algorithm for SP graphs;
- Characterisation of SP graphs of treelength 2 (Forbiden isometric subgraphs).

Our Contributions

Sub-classes of planar graphs:



Treelength [T. Dissaux, G.Ducoffe, N. Nisse, S. Nivelle, 2021]

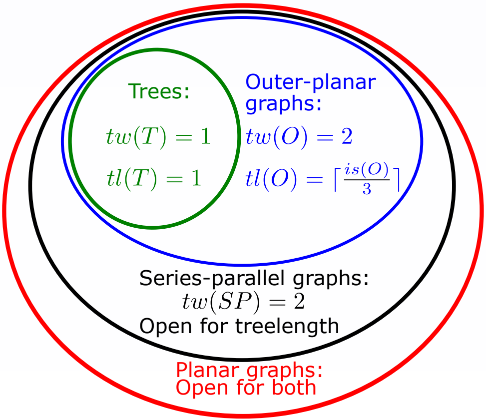
- $\frac{3}{2}$-Approximation Algorithm for SP graphs;
- Characterisation of SP graphs of treelength 2 (Forbiden isometric subgraphs).

Pathlength [T. Dissaux, N. Nisse, 2021]

- Linear time algorithm for trees;
- $pl(C_n) = \lfloor \frac{n}{2} \rfloor$;
- $(+1)$-approximation for Outerplanar graphs.

Our Contributions

Sub-classes of planar graphs:



Treelength [T. Dissaux, G.Ducoffe, N. Nisse, S. Nivelle, 2021]

- $\frac{3}{2}$-Approximation Algorithm for SP graphs;
- Characterisation of SP graphs of treelength 2 (Forbiden isometric subgraphs).

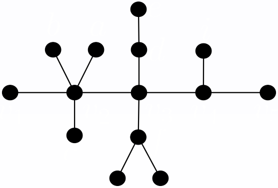
Pathlength [T. Dissaux, N. Nisse, 2021]

- Linear time algorithm for trees;
- $pl(C_n) = \lfloor \frac{n}{2} \rfloor$;
- (+1)-approximation for Outerplanar graphs.

Pathlength of trees

Algorithm:

A Tree T and an optimal path-decomposition of T:

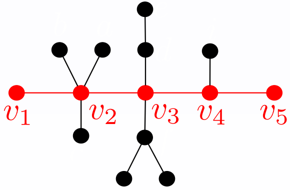


Pathlength of trees

Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.

A Tree T and an optimal path-decomposition of T:

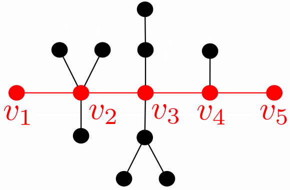


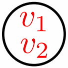
Pathlength of trees

Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.

A Tree T and an optimal path-decomposition of T:





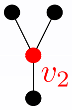
Pathlength of trees

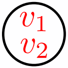
Algorithm:

Compute a diameter of T, i.e $P = (v_1, \dots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.

A Tree T and an optimal path-decomposition of T:





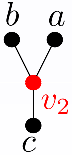
Pathlength of trees

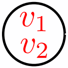
Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.

A Tree T and an optimal path-decomposition of T:





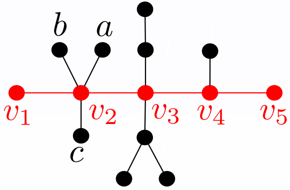
Pathlength of trees

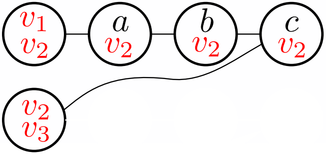
Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.
  Let us "add" the path between v_{i+1} and each leaf of T_{i+1} by following the ordering of these leafs.

A Tree T and an optimal path-decomposition of T:





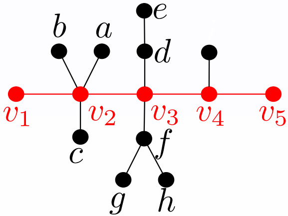
Pathlength of trees

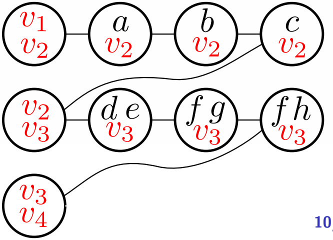
Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.
  Let us "add" the path between v_{i+1} and each leaf of T_{i+1} by following the ordering of these leafs.

A Tree T and an optimal path-decomposition of T:





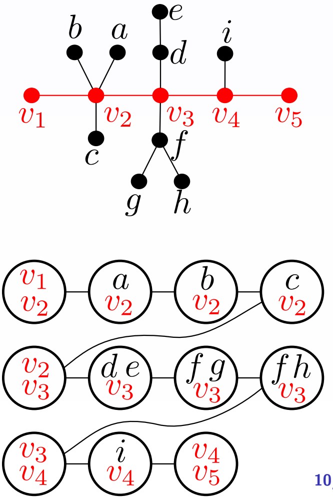
Pathlength of trees

Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.
  Let us "add" the path between v_{i+1} and each leaf of T_{i+1} by following the ordering of these leafs.

A Tree T and an optimal path-decomposition of T:



Pathlength of trees

Algorithm:

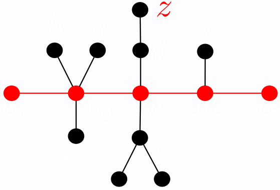
Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

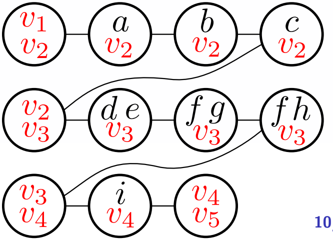
- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.
  Let us "add" the path between v_{i+1} and each leaf of T_{i+1} by following the ordering of these leafs.

Optimality:

Let z be any vertex maximizing its distance to P.

A Tree T and an optimal path-decomposition of T:





Pathlength of trees

Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

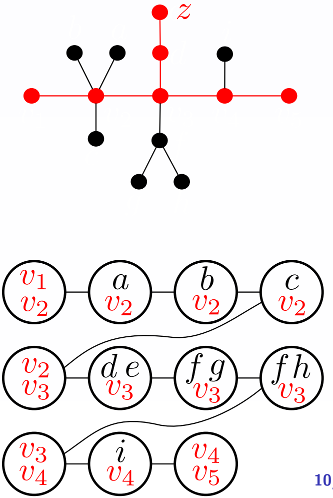
- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.
  Let us "add" the path between v_{i+1} and each leaf of T_{i+1} by following the ordering of these leafs.

Optimality:

Let z be any vertex maximizing its distance to P.

Note that T contains S_k **(as isometric subgraph)**, the tree composed by 3 paths of length $k = d(z, P)$ having one endpoint in common.

A Tree T and an optimal path-decomposition of T:



Pathlength of trees

Algorithm:

Compute a diameter of T, i.e $P = (v_1, \ldots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.
  Let us "add" the path between v_{i+1} and each leaf of T_{i+1} by following the ordering of these leafs.

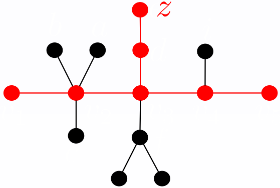
Optimality:

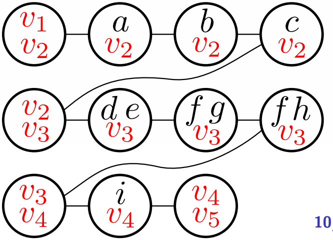
Let z be any vertex maximizing its distance to P.

Note that T contains S_k **(as isometric subgraph)**, the tree composed by 3 paths of length $k = d(z, P)$ having one endpoint in common.

Note that $pl(S_k) = k$ [Dourisboure, Gavoille, 2007]

A Tree T and an optimal path-decomposition of T:





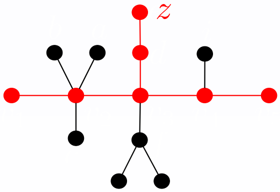
Pathlength of trees

Algorithm:

Compute a diameter of T, i.e $P = (v_1, \dots, v_t)$.
Let $X_1 = \{v_1, v_2\}$.
For any $1 \leq i < t$:

- Let us denote the vertices T_{i+1} by a DFS ordering starting from v_{i+1}.
  Let us "add" the path between v_{i+1} and each leaf of T_{i+1} by following the ordering of these leafs.

A Tree T and an optimal path-decomposition of T:

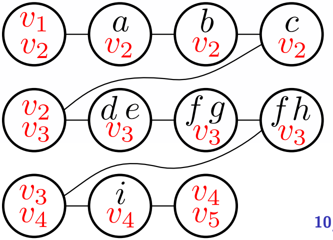


Optimality: $pl(T) \geq pl(S_k) \geq k$

Let z be any vertex maximizing its distance to P.

Note that T contains S_k **(as isometric subgraph)**, the tree composed by 3 paths of length $k = d(z, P)$ having one endpoint in common.

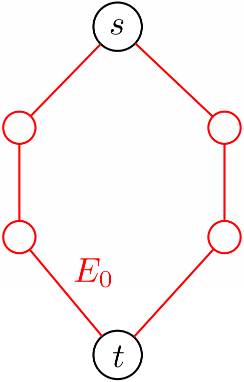
Note that $pl(S_k) = k$ [Dourisboure, Gavoille, 2007]



(+1)-Approximation of the pathlength of Outerplanar graphs

Outerplanar graphs

An outerplanar graph $G = (V, E)$ is a planar graph such that each vertex $v \in V$ is on the outer (unbounded) face.



(+1)-Approximation of the pathlength of Outerplanar graphs

Outerplanar graphs

An outerplanar graph $G = (V, E)$ is a planar graph such that each vertex $v \in V$ is on the outer (unbounded) face.

(+1)-Approximation

For any $k > 1$ and any Outerplanar graphs G, either the algorithm decides that $pl(G) > k$ or return a path-decomposition of length at most $pl(G) + 1$ in time $O(n^3(n + k^2))$.

(+1)-Approximation of the pathlength of Outerplanar graphs

Outerplanar graphs

An outerplanar graph $G = (V, E)$ is a planar graph such that each vertex $v \in V$ is on the outer (unbounded) face.

(+1)-Approximation

For any $k > 1$ and any Outerplanar graphs G, either the algorithm decides that $pl(G) > k$ or return a path-decomposition of length at most $pl(G) + 1$ in time $O(n^3(n + k^2))$.

This algorithm is based on:

- the fact that every $(\{x, y\})$ path-decomposition $D = (X_1, \ldots, X_p)$ of G contains an edge x in X_1 and an edge $y \in X_p$.
- a characterization of almost optimal $\{x, y\}$-path-decomposition of G, i.e. separated contiguous LtR $\{x, y\}$-path-decomposition.

(+1)-Approximation of the pathlength of Outerplanar graphs

Outerplanar graphs

An outerplanar graph $G = (V, E)$ is a planar graph such that each vertex $v \in V$ is on the outer (unbounded) face.

(+1)-Approximation

For any $k > 1$ and any Outerplanar graphs G, either the algorithm decides that $pl(G) > k$ or return a path-decomposition of length at most $pl(G) + 1$ in time $O(n^3(n + k^2))$.

This algorithm is based on:

- the fact that every ($\{x, y\}$) path-decomposition $D = (X_1, \dots, X_p)$ of G contains an edge x in X_1 and an edge $y \in X_p$.
- a characterization of almost optimal $\{x, y\}$-path-decomposition of G, i.e. separated contiguous LtR $\{x, y\}$-path-decomposition.

Restriction for the talk: The pathlength of 2-connected outerplanar graphs is much simpler.

(+1)-Approximation of the pathlength of Outerplanar graphs

Outerplanar graphs

An outerplanar graph $G = (V, E)$ is a planar graph such that each vertex $v \in V$ is on the outer (unbounded) face.

(+1)-Approximation

For any $k > 1$ and any Outerplanar graphs G, either the algorithm decides that $pl(G) > k$ or return a path-decomposition of length at most $pl(G) + 1$ in time $O(n^3(n + k^2))$.

This algorithm is based on:

- the fact that every $(\{x, y\})$ path-decomposition $D = (X_1, \ldots, X_p)$ of G contains an edge x in X_1 and an edge $y \in X_p$.
- a characterization of almost optimal $\{x, y\}$-path-decomposition of G, i.e. separated contiguous LtR $\{x, y\}$-path-decomposition.

Restriction for the talk: The pathlength of 2-connected outerplanar graphs is much simpler.

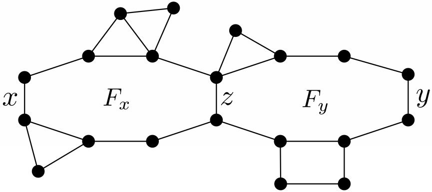
Characterization: "separated"

"Separated"

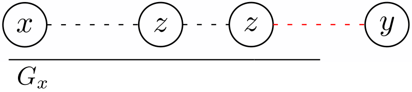
If there exists $z \in E_{int}$ such that z separates x and y, then a $\{x, y\}$-path-decomposition can be obtained from a $\{x, z\}$-path-decomposition and a $\{z, y\}$-path-decomposition.

Let G_x (resp. G_y) be the connected component of $G \setminus \{z\}$ containing x (resp. y) union z.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:



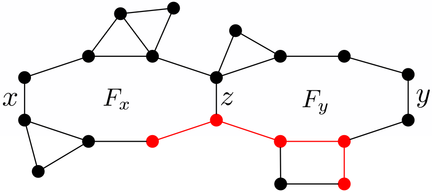
Characterization: "separated"

"Separated"

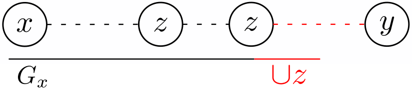
If there exists $z \in E_{int}$ such that z separates x and y, then a $\{x, y\}$-path-decomposition can be obtained from a $\{x, z\}$-path-decomposition and a $\{z, y\}$-path-decomposition.

Let G_x (resp. G_y) be the connected component of $G \setminus \{z\}$ containing x (resp. y) union z.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:



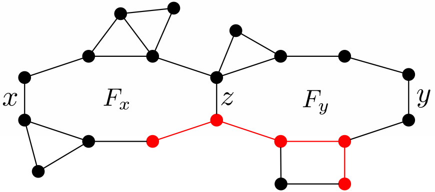
Characterization: "separated"

"Separated"

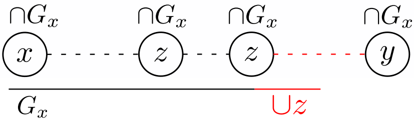
If there exists $z \in E_{int}$ such that z separates x and y, then a $\{x, y\}$-path-decomposition can be obtained from a $\{x, z\}$-path-decomposition and a $\{z, y\}$-path-decomposition.

Let G_x (resp. G_y) be the connected component of $G \setminus \{z\}$ containing x (resp. y) union z.

A 2-connected outerplanar graph G:



A $\{x, z\}$-path-decomposition of G_x of length at most $pl(G)$:



Characterization: "contiguous"

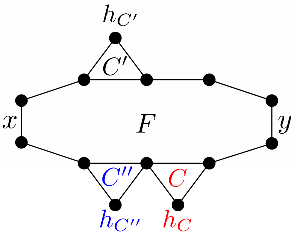
"Contiguous"

For any component C of $G \setminus F$, there exists $1 \leq a_C < b_C \leq p$ such

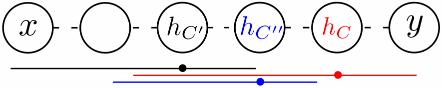
1. $C \cap X_i = \emptyset$ if only if $i \notin [a_C, b_C]$;
2. there exists $R_C \subseteq F$ such that for any $a_C \leq i \leq b_C$, $X_i \setminus C = R_F$.

For any component $C \in \mathcal{C}$, let h_C be any vertex that maximize its distance to F.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:



Characterization: "contiguous"

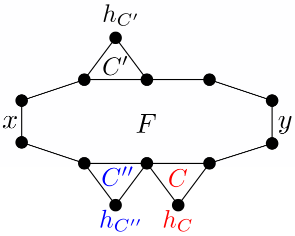
"Contiguous"

For any component C of $G \setminus F$, there exists $1 \leq a_C < b_C \leq p$ such

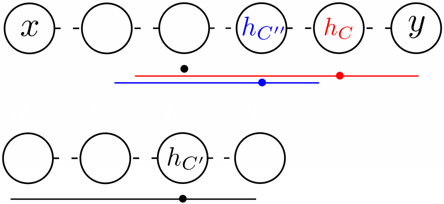
1. $C \cap X_i = \emptyset$ if only if $i \notin [a_C, b_C]$;
2. there exists $R_C \subseteq F$ such that for any $a_C \leq i \leq b_C$, $X_i \setminus C = R_F$.

For any component $C \in \mathcal{C}$, let h_C be any vertex that maximize its distance to F.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:



Characterization: "contiguous"

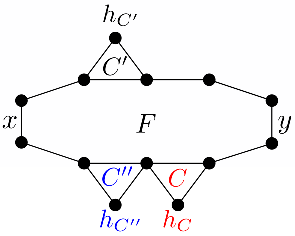
"Contiguous"

For any component C of $G \setminus F$, there exists $1 \leq a_C < b_C \leq p$ such

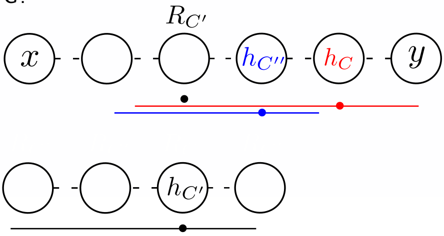
1. $C \cap X_i = \emptyset$ if only if $i \notin [a_C, b_C]$;
2. there exists $R_C \subseteq F$ such that for any $a_C \leq i \leq b_C$, $X_i \setminus C = R_F$.

For any component $C \in \mathcal{C}$, let h_C be any vertex that maximize its distance to F.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:



Characterization: "contiguous"

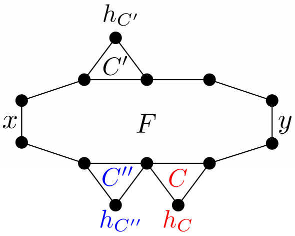
"Contiguous"

For any component C of $G \setminus F$, there exists $1 \leq a_C < b_C \leq p$ such

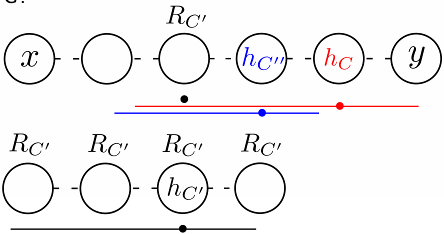
1. $C \cap X_i = \emptyset$ if only if $i \notin [a_C, b_C]$;
2. there exists $R_C \subseteq F$ such that for any $a_C \leq i \leq b_C$, $X_i \setminus C = R_F$.

For any component $C \in \mathcal{C}$, let h_C be any vertex that maximize its distance to F.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:



Characterization: "contiguous"

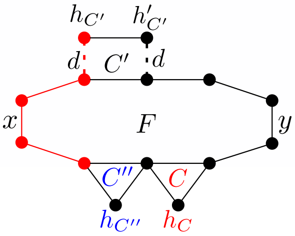
"Contiguous" Warning: The length increase by 1

For any component C of $G \setminus F$, there exists $1 \leq a_C < b_C \leq p$ such

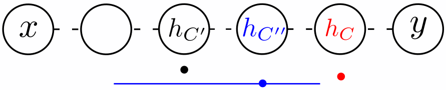
1. $C \cap X_i = \emptyset$ if only if $i \notin [a_C, b_C]$;
2. there exists $R_C \subseteq F$ such that for any $a_C \leq i \leq b_C$, $X_i \setminus C = R_F$.

For any component $C \in \mathcal{C}$, let h_C be any vertex that maximize its distance to F.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:



Characterization: "contiguous"

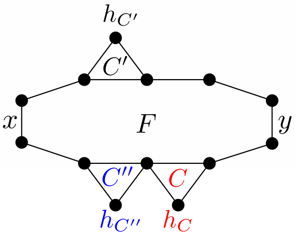
"Contiguous" Warning: The length increase by 1

For any component C of $G \setminus F$, there exists $1 \leq a_C < b_C \leq p$ such

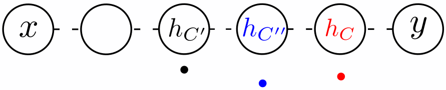
1. $C \cap X_i = \emptyset$ if only if $i \notin [a_C, b_C]$;
2. there exists $R_C \subseteq F$ such that for any $a_C \leq i \leq b_C$, $X_i \setminus C = R_F$.

For any component $C \in \mathcal{C}$, let h_C be any vertex that maximize its distance to F.

A 2-connected outerplanar graph G:



An optimal $\{x, y\}$-path-decomposition of G:

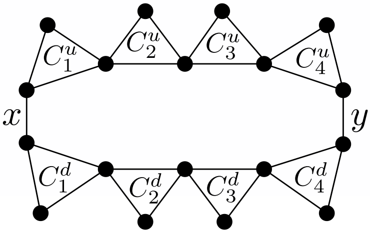


Characterization: "LtR" (Left to Right)

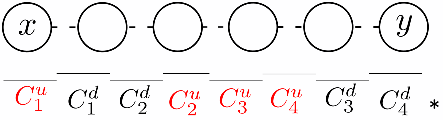
"LtR"

For any two component C and C' in C_{up} (resp. C_{down}), if $N(C)$ is closer to x than $N(C')$, then $b_C < a'_C$, i.e. C appears before C' in the $\{x, y\}$-path-decomposition.

A 2-connected outerplanar graph G:



An $\{x, y\}$-path-decomposition of G:



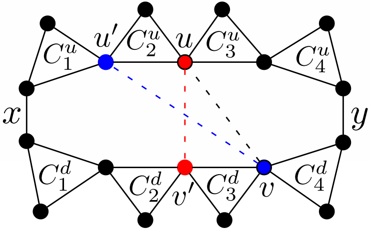
Characterization: "LtR" (Left to Right)

"LtR"

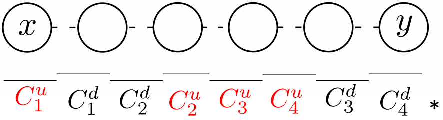
For any two component C and C' in C_{up} (resp. C_{down}), if $N(C)$ is closer to x than $N(C')$, then $b_C < a'_C$, i.e. C appears before C' in the $\{x, y\}$-path-decomposition.

Let $D[u, v]$ be a "partial" contiguous LtR path-decomposition of G such that the last bag contains u and v.

A 2-connected outerplanar graph G:

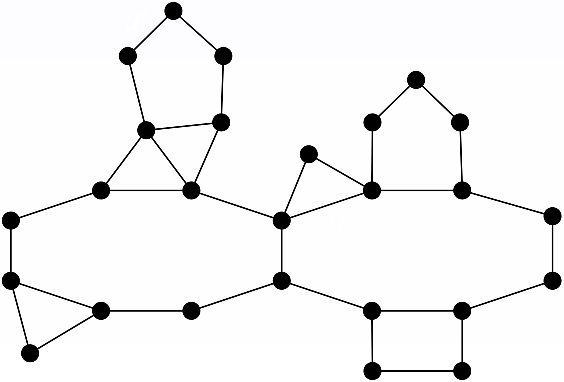


An $\{x, y\}$-path-decomposition of G:



(+1)-Approximation

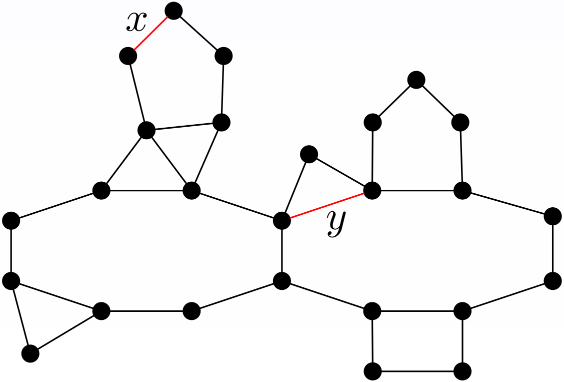
Algorithm:



(+1)-Approximation

Algorithm: $O(n^2)$

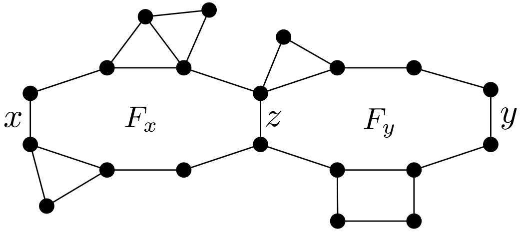
- First, fix $x, y \in E$ ($|E| = O(n)$ in planar graphs): $O(n^2)$;



(+1)-Approximation

Algorithm: $O(n^3)$

- First, fix $x, y \in E$ ($|E| = O(n)$ in planar graphs): $O(n^2)$;
- Second, restrict the computation when x, y belongs to the same face ("separated"): $O(n)$;



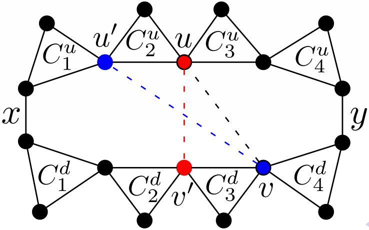
(+1)-Approximation

Algorithm: $O(n^3(n))$

- First, fix $x, y \in E$ ($|E| = O(n)$ in planar graphs): $O(n^2)$;
- Second, restrict the computation when x, y belongs to the same face ("separated"): $O(n)$;
- Third, compute a greedy path-decomposition for every component of $G \setminus F$: $O(n)$;

Lemma "greedy":

For any 2-connected outerplanar graph G and any $x \in E(G)$, a $\{x, x\}$-path-decomposition of G can be computed in time $O(n)$.



(+1)-Approximation

Algorithm: $O(n^3(n+k^2))$

- First, fix $x, y \in E$ ($|E| = O(n)$ in planar graphs): $O(n^2)$;
- Second, restrict the computation when x, y belongs to the same face ("separated"): $O(n)$;
- Third, compute a greedy path-decomposition for every component of $G \setminus F$: $O(n)$;
- Fourth, compute the "LtR" ordering for the component of $G \setminus F$: $O(k^2)$;
- Finally, it remains to compare the computed path-decompositions and keep the one that has minimum length.

(+1)-Approximation

Algorithm: $O(n^3(n+k^2))$

- First, fix $x, y \in E$ ($|E| = O(n)$ in planar graphs): $O(n^2)$;
- Second, restrict the computation when x, y belongs to the same face ("separated"): $O(n)$;
- Third, compute a greedy path-decomposition for every component of $G \setminus F$: $O(n)$;
- Fourth, compute the "LtR" ordering for the component of $G \setminus F$: $O(k^2)$;
- Finally, it remains to compare the computed path-decompositions and keep the one that has minimum length.

(+1)-Approximation

Algorithm: $O(n^3(n+k^2))$

- First, fix $x, y \in E$ ($|E| = O(n)$ in planar graphs): $O(n^2)$;
- Second, restrict the computation when x, y belongs to the same face ("separated"): $O(n)$;
- Third, compute a greedy path-decomposition for every component of $G \setminus F$: $O(n)$;
- Fourth, compute the "LtR" ordering for the component of $G \setminus F$: $O(k^2)$;
- Finally, it remains to compare the computed path-decompositions and keep the one that has minimum length.

Further Works

- SP graph of treelength k;
- Exact algorithm for the pathlength of Outerplanar graphs;
- Approximation algorithm (planar graphs, general graphs);

Further Works

- SP graph of treelength k;
- Exact algorithm for the pathlength of Outerplanar graphs;
- Approximation algorithm (planar graphs, general graphs);

Thank you