Finding subdigraphs in digraphs of bounded directed treewidth

Raul Lopes
LIRMM, Université de Montpellier, CNRS, Montpellier, France
Algorithmen und Komplexität, Technische Univ. Hamburg, Germany

Ignasi Sau
LIRMM, Université de Montpellier, CNRS, Montpellier, France

LAGOS 2025, Buenos Aires, Argentina

Undirected vs directed treewidth

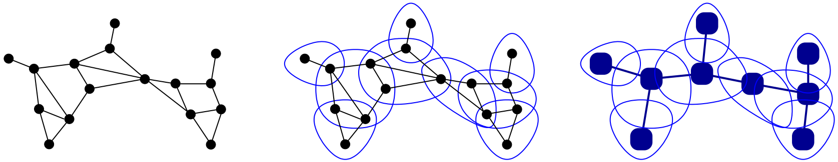
Treewidth of **undirected** graphs: invariant that measures the structural similarity of a graph to a forest.

[Bertelè, Brioschi. 1972]

[Halin. 1976]

[Arnborg, Proskurowski. 1984]

[Robertson, Seymour. 1984]



Undirected vs directed treewidth

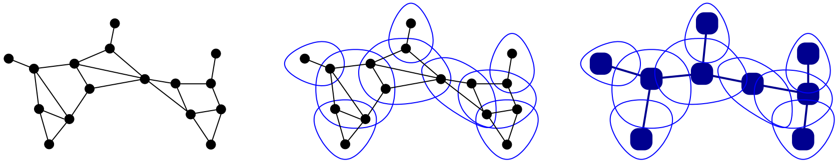
Treewidth of **undirected** graphs: invariant that measures the structural similarity of a graph to a forest.

[Bertelè, Brioschi. 1972]

[Halin. 1976]

[Arnborg, Proskurowski. 1984]

[Robertson, Seymour. 1984]



Treewidth of **directed** graphs: invariant that measures the structural similarity of a digraph to a DAG.

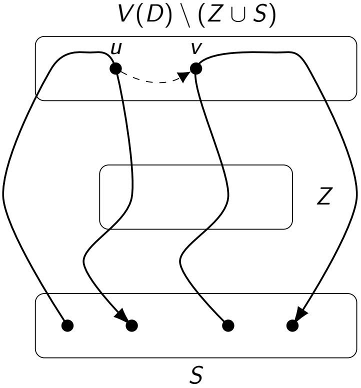
[Johnson, Robertson, Seymour, Thomas. 2001]

Preliminaries: guarded sets

Preliminaries: guarded sets

Let D be a digraph, $Z \subseteq V(D)$, and $S \subseteq V(D) \setminus Z$.

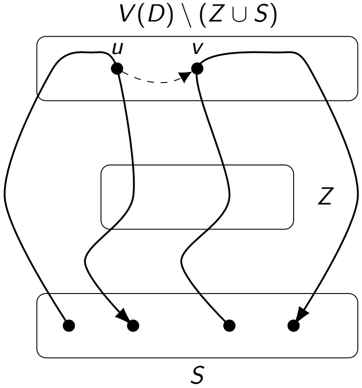
We say that S is Z-guarded if there is no directed walk in $D \setminus Z$ with first and last vertices in S that uses a vertex of $D \setminus (Z \cup S)$.



Preliminaries: guarded sets

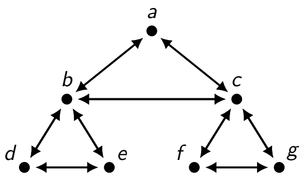
Let D be a digraph, $Z \subseteq V(D)$, and $S \subseteq V(D) \setminus Z$.

We say that S is Z-guarded if there is no directed walk in $D \setminus Z$ with first and last vertices in S that uses a vertex of $D \setminus (Z \cup S)$.

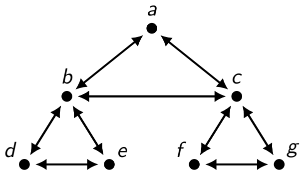


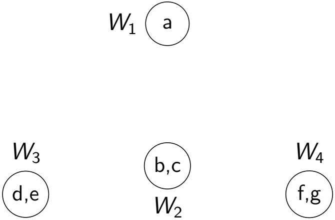
S is w-guarded if S is X-guarded by some $X \subseteq V(D)$ with $|X| \leq w$.

Directed treewidth ("distance to a DAG")



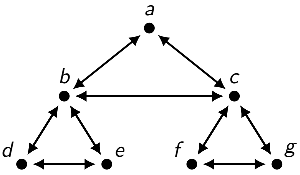
Directed treewidth (“distance to a DAG”)

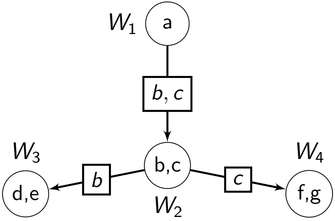




- *Partition* $V(D)$ into non-empty bags $\{W_1, W_2, W_3, W_4\}$, corresponding to the nodes of an out-arborescence T.

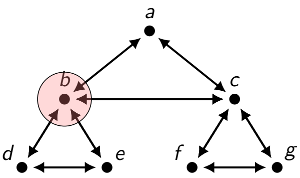
Directed treewidth ("distance to a DAG")

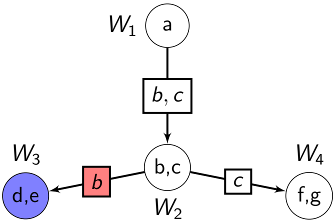




- *Partition* $V(D)$ into non-empty bags $\{W_1, W_2, W_3, W_4\}$, corresponding to the nodes of an out-arborescence T.
- For every arc $e \in E(T)$: set $X_e \subseteq V(D)$ that guards the subdigraph defined by its descendants (these sets may be empty or intersect)

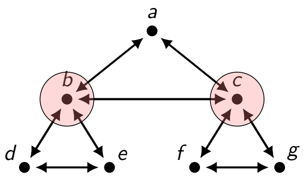
Directed treewidth ("distance to a DAG")

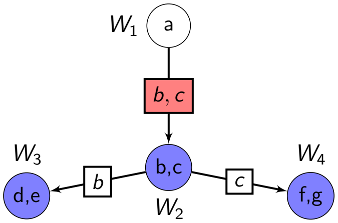




- *Partition* $V(D)$ into non-empty bags $\{W_1, W_2, W_3, W_4\}$, corresponding to the nodes of an out-arborescence T.
- For every arc $e \in E(T)$: set $X_e \subseteq V(D)$ that guards the subdigraph defined by its descendants (these sets may be empty or intersect)
- $\{b\}$ guards $\{d, e\}$.
  - No path starting and ending in $\{d, e\}$ using vertices from the other side of the guard $(= V(D) \setminus \{d, e\})$ in $D \setminus \{b\}$.

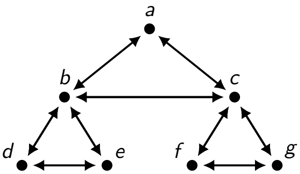
Directed treewidth ("distance to a DAG")

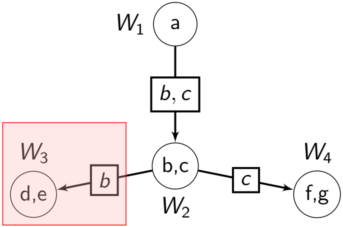




- *Partition* $V(D)$ into non-empty bags $\{W_1, W_2, W_3, W_4\}$, corresponding to the nodes of an out-arborescence T.
- For every arc $e \in E(T)$: set $X_e \subseteq V(D)$ that guards the subdigraph defined by its descendants (these sets may be empty or intersect)
- $\{b\}$ guards $\{d, e\}$, $\{b, c\}$ guards $V(D) \setminus \{a\}$.
  - No path starting and ending in $\{d, e\}$ using vertices from the other side of the guard $(=V(D) \setminus \{d, e\})$ in $D \setminus \{b\}$.

Directed treewidth ("distance to a DAG")





- *Partition* $V(D)$ into non-empty bags $\{W_1, W_2, W_3, W_4\}$, corresponding to the nodes of an out-arborescence T.
- For every arc $e \in E(T)$: set $X_e \subseteq V(D)$ that guards the subdigraph defined by its descendants (these sets may be empty or intersect)
- $\{b\}$ guards $\{d, e\}$, $\{b, c\}$ guards $V(D) \setminus \{a\}$.
  - No path starting and ending in $\{d, e\}$ using vertices from the other side of the guard $(=V(D) \setminus \{d, e\})$ in $D \setminus \{b\}$.
- *Width* = $\boxed{\text{size of largest set of (bag} \cup \text{adjacent guards)} -1}$.
  - $= 2$ in the example.

More preliminaries: parameterized complexity in one slide

More preliminaries: parameterized complexity in one slide

Instance of a parameterized problem: total size n, *parameter* k.

More preliminaries: parameterized complexity in one slide

Instance of a parameterized problem: total size n, *parameter* k.

- XP problem: solvable in time $f(k) \cdot n^{g(k)}$.

More preliminaries: parameterized complexity in one slide

Instance of a parameterized problem: total size n, *parameter* k.

- XP problem: solvable in time $f(k) \cdot n^{g(k)}$.
    - Example: $\mathcal{O}(n^k)$.

More preliminaries: parameterized complexity in one slide

Instance of a parameterized problem: total size n, *parameter* k.

- XP problem: solvable in time $f(k) \cdot n^{g(k)}$.
  - Example: $\mathcal{O}(n^k)$.

- FPT problem: solvable in time $f(k) \cdot n^c$ for an absolute constant c.

More preliminaries: parameterized complexity in one slide

Instance of a parameterized problem: total size n, *parameter* k.

- XP problem: solvable in time $f(k) \cdot n^{g(k)}$.
  - Example: $\mathcal{O}(n^k)$.
- FPT problem: solvable in time $f(k) \cdot n^c$ for an absolute constant c.
  - Example: $\mathcal{O}(2^k \cdot n^2)$.

More preliminaries: parameterized complexity in one slide

Instance of a parameterized problem: total size n, *parameter* k.

- XP problem: solvable in time $f(k) \cdot n^{g(k)}$.
  - Example: $\mathcal{O}(n^k)$.
- FPT problem: solvable in time $f(k) \cdot n^c$ for an absolute constant c.
  - Example: $\mathcal{O}(2^k \cdot n^2)$.
- W[i]-hard problem, for $i \geq 1$: strong evidence that it is *not* FPT.

More preliminaries: parameterized complexity in one slide

Instance of a parameterized problem: total size n, *parameter* k.

- XP problem: solvable in time $f(k) \cdot n^{g(k)}$.
  - Example: $\mathcal{O}(n^k)$.
- FPT problem: solvable in time $f(k) \cdot n^c$ for an absolute constant c.
  - Example: $\mathcal{O}(2^k \cdot n^2)$.
- W[i]-hard problem, for $i \geq 1$: strong evidence that it is *not* FPT.
- para-NP-hard problem: NP-hard for a fixed value of the parameter.

Back to undirected and directed treewidth

How can we exploit algorithmically bounded (un)directed treewidth?

Back to undirected and directed treewidth

How can we exploit algorithmically bounded (un)directed treewidth?

Undirected treewidth rules!

Theorem (Courcelle. 1990)

Every problem expressible in MSO_2 *can be solved in time* $f(\mathrm{tw}) \cdot n$ *on graphs on* n *vertices and treewidth at most* tw.

Back to undirected and directed treewidth

How can we exploit algorithmically bounded (un)directed treewidth?

Undirected treewidth rules!

Theorem (Courcelle. 1990)

Every problem expressible in MSO_2 *can be solved in time* $f(\mathrm{tw}) \cdot n$ *on graphs on* n *vertices and treewidth at most* tw. *(Also for optimizing)*

Back to undirected and directed treewidth

How can we exploit algorithmically bounded (un)directed treewidth?

Undirected treewidth rules!

Theorem (Courcelle. 1990)

Every problem expressible in MSO_2 can be solved in time $f(\mathsf{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw. (Also for optimizing)

Hamiltonian Path, Max Cut, Max Independent Set, Min Dominating Set, Feedback Vertex Set, 3-Coloring, Longest Path...

Back to undirected and directed treewidth

How can we exploit algorithmically bounded (un)directed treewidth?

Undirected treewidth rules!

Theorem (Courcelle. 1990)

Every problem expressible in MSO_2 can be solved in time $f(\mathrm{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw. (Also for optimizing)

HAMILTONIAN PATH, MAX CUT, MAX INDEPENDENT SET, MIN DOMINATING SET, FEEDBACK VERTEX SET, 3-COLORING, LONGEST PATH...

What about directed treewidth (dtw)?

Back to undirected and directed treewidth

How can we exploit algorithmically bounded (un)directed treewidth?

Undirected treewidth rules!

Theorem (Courcelle. 1990)

Every problem expressible in MSO_2 can be solved in time $f(\mathrm{tw}) \cdot n$ on graphs on n vertices and treewidth at most tw. (Also for optimizing)

Hamiltonian Path, Max Cut, Max Independent Set, Min Dominating Set, Feedback Vertex Set, 3-Coloring, Longest Path...

What about directed treewidth (dtw)?

- Directed Hamiltonian Path is $W[2]$-hard parameterized by dtw.
- Max Directed Cut is NP-hard restricted to DAGs ($\mathrm{dtw} = 0$).

[Lampis, Kaouri, Mitsou. 2011]

Only bad news for directed treewidth?

Only bad news for directed treewidth? No!

$\boxed{\mathbf{1}}$ Directed Hamiltonian Path and Directed k-Disjoint Paths can be solved, respectively, in time $n^{\mathcal{O}(\mathsf{dtw})}$ and $n^{\mathcal{O}(\mathsf{dtw}+k)}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

Only bad news for directed treewidth? No!

$\boxed{\mathbf{1}}$ Directed Hamiltonian Path and Directed k-Disjoint Paths can be solved, respectively, in time $n^{\mathcal{O}(\mathsf{dtw})}$ and $n^{\mathcal{O}(\mathsf{dtw}+k)}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

Fix an integer k and an MSO_2 formula φ, and let H be a digraph s.t.:

Only bad news for directed treewidth? No!

$\boxed{\mathbf{1}}$ Directed Hamiltonian Path and Directed k-Disjoint Paths can be solved, respectively, in time $n^{\mathcal{O}(\mathsf{dtw})}$ and $n^{\mathcal{O}(\mathsf{dtw}+k)}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

Fix an integer k and an MSO_2 formula φ, and let H be a digraph s.t.:

- H consists of the union of at most k directed paths.
- H satisfies the property described by φ.

Only bad news for directed treewidth? No!

$\boxed{\mathbf{1}}$ Directed Hamiltonian Path and Directed k-Disjoint Paths can be solved, respectively, in time $n^{\mathcal{O}(\mathrm{dtw})}$ and $n^{\mathcal{O}(\mathrm{dtw}+k)}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

Fix an integer k and an MSO_2 formula φ, and let H be a digraph s.t.:

- H consists of the union of at most k directed paths.
- H satisfies the property described by φ.

$\boxed{\mathbf{2}}$ Then, deciding whether a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\varphi,\mathrm{dtw})}$.

[de Oliveira Oliveira. 2016]

Only bad news for directed treewidth? No!

$\boxed{1}$ Directed Hamiltonian Path and Directed k-Disjoint Paths can be solved, respectively, in time $n^{\mathcal{O}(\mathrm{dtw})}$ and $n^{\mathcal{O}(\mathrm{dtw}+k)}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

Fix an integer k and an MSO_2 formula φ, and let H be a digraph s.t.:

- H consists of the union of at most k directed paths.
- H satisfies the property described by φ.

$\boxed{2}$ Then, deciding whether a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\varphi,\mathrm{dtw})}$.

[de Oliveira Oliveira. 2016]

Our question Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

Our results

Our question Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

Our results

$\boxed{\text{Our question}}$ Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"?

Our results

Our question Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"? XP.

Our results

$\boxed{\text{Our question}}$ Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"? XP. Ok, but for which parameter?

Our results

$\boxed{\text{Our question}}$ Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"? XP. Ok, but for which parameter?

If we take $|V(H)|$, the problem is XP for every H: time $|V(D)|^{\mathcal{O}(|V(H)|)}$.

Our results

$\boxed{\text{Our question}}$ Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"? XP. Ok, but for which parameter?

If we take $|V(H)|$, the problem is XP for every H: time $|V(D)|^{\mathcal{O}(|V(H)|)}$.

In the spirit of the existing XP algorithms, we want to know for which collections $\mathcal{A}$ of allowed digraphs the following holds:

Our results

$\boxed{\text{Our question}}$ Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"? XP. Ok, but for which parameter?

If we take $|V(H)|$, the problem is XP for every H: time $|V(D)|^{\mathcal{O}(|V(H)|)}$.

In the spirit of the existing XP algorithms, we want to know for which collections $\mathcal{A}$ of allowed digraphs the following holds:

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.

Our results

$\boxed{\text{Our question}}$ Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"? XP. Ok, but for which parameter?

If we take $|V(H)|$, the problem is XP for every H: time $|V(D)|^{\mathcal{O}(|V(H)|)}$.

In the spirit of the existing XP algorithms, we want to know for which collections $\mathcal{A}$ of allowed digraphs the following holds:

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

Our results (informally)

$\boxed{\text{Our question}}$ Are there subdigraphs H, other than the directed paths, that can be found efficiently in digraphs of bounded directed treewidth?

What do we mean by "efficiently"? XP. Ok, but for which parameter?

If we take $|V(H)|$, the problem is XP for every H: time $|V(D)|^{\mathcal{O}(|V(H)|)}$.

In the spirit of the existing XP algorithms, we want to know for which collections $\mathcal{A}$ of allowed digraphs the following holds:

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\text{dtw})}$.

$\boxed{\text{Our results}}$ Essentially, only for $\mathcal{A} = \{\text{paths}, \text{stars}\}$.

Our results (more formally)

Our results (more formally)

Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars.

Our results (more formally)

Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw *contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathrm{dtw})}$.*

Our results (more formally)

Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw *contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.*

What about digraphs H that cannot be covered by few paths or stars?

Our results (more formally)

Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathrm{dtw})}$.

What about digraphs H that cannot be covered by few paths or stars?

For all these subdigraphs H, the problem is NP-hard on DAGs ($\mathrm{dtw} = 0$):

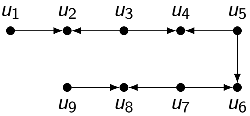
Our results (more formally)

Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathrm{dtw})}$.

What about digraphs H that cannot be covered by few paths or stars?

For all these subdigraphs H, the problem is NP-hard on DAGs ($\mathrm{dtw} = 0$):



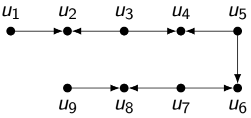
Our results (more formally)

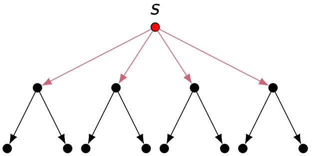
Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathrm{dtw})}$.

What about digraphs H that cannot be covered by few paths or stars?

For all these subdigraphs H, the problem is NP-hard on DAGs ($\mathrm{dtw} = 0$):





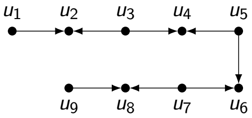
Our results (more formally)

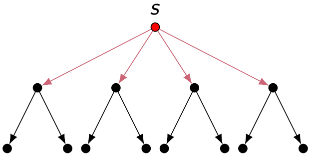
Theorem

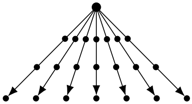
Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.

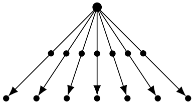
What about digraphs H that cannot be covered by few paths or stars?

For all these subdigraphs H, the problem is NP-hard on DAGs ($\mathsf{dtw} = 0$):









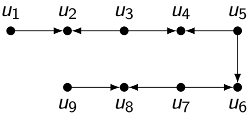
Our results (more formally)

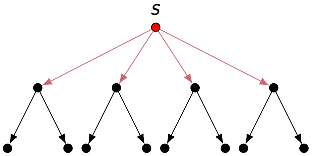
Theorem

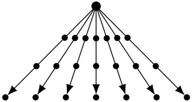
Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathrm{dtw})}$.

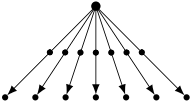
What about digraphs H that cannot be covered by few paths or stars?

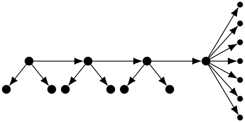
For all these subdigraphs H, the problem is NP-hard on DAGs ($\mathrm{dtw} = 0$):











Our techniques

Our techniques: XP algorithm

Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathrm{dtw})}$.

Our techniques: XP algorithm

Theorem

Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.

We generalize the dynamic programming approach to solve this problem:

Directed k-Disjoint Paths can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

Our techniques: XP algorithm

Theorem

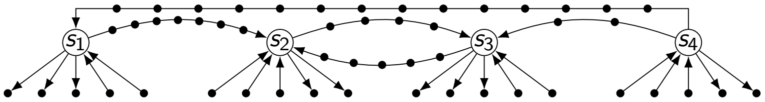
Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.

We generalize the dynamic programming approach to solve this problem:

Directed k-Disjoint Paths can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

But now we also have stars!



Our techniques: XP algorithm

Theorem

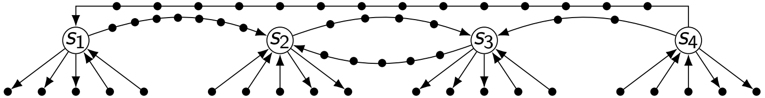
Let H be a digraph that consists of the union of at most k directed paths or stars. Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.

We generalize the dynamic programming approach to solve this problem:

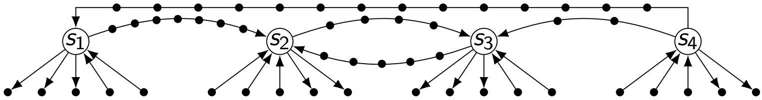
Directed k-Disjoint Paths can be solved in time $n^{\mathcal{O}(k+\mathsf{dtw})}$.

[Johnson, Robertson, Seymour, Thomas. 2001]

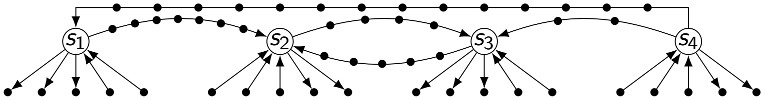
But now we also have stars!



Our techniques: XP algorithm

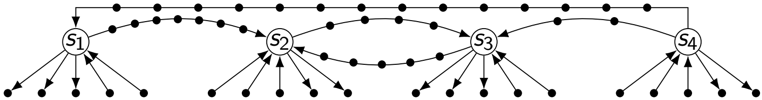


Our techniques: XP algorithm



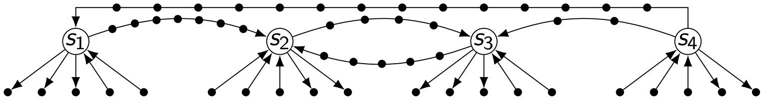
- If we had only stars: guess centers (XP time), and reduce the problem to an system of integer inequations (relating the sizes of the desired stars to the sizes of the neighborhoods of the centers),

Our techniques: XP algorithm



- If we had only stars: guess centers (XP time), and reduce the problem to an system of integer inequations (relating the sizes of the desired stars to the sizes of the neighborhoods of the centers), which can be solved in FPT time in the number of variables. [Lenstra. 1983]

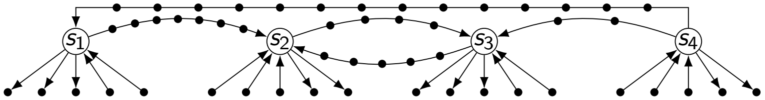
Our techniques: XP algorithm



- If we had only stars: guess centers (XP time), and reduce the problem to an system of integer inequations (relating the sizes of the desired stars to the sizes of the neighborhoods of the centers), which can be solved in FPT time in the number of variables. [Lenstra. 1983]

- Generalized version: for every candidate center $s_i \in V(D)$, we guess a number allowed-for-stars(s_i) (in XP time), representing how many vertices from $N(s_i)$ we want to use as leaves of all the stars.

Our techniques: XP algorithm



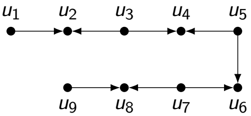
- If we had only stars: guess centers (XP time), and reduce the problem to an system of integer inequations (relating the sizes of the desired stars to the sizes of the neighborhoods of the centers), which can be solved in FPT time in the number of variables. [Lenstra. 1983]

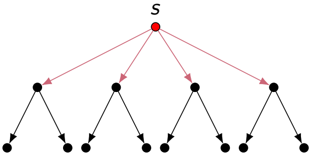
- Generalized version: for every candidate center $s_i \in V(D)$, we guess a number allowed-for-stars(s_i) (in XP time), representing how many vertices from $N(s_i)$ we want to use as leaves of all the stars.

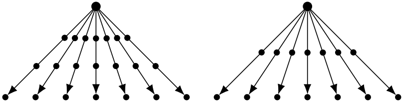
- Finally, we solve a generalized version of Directed k-Disjoint Paths where, in addition to the pairs of terminals, we are given, for each terminal s_i a set $X_{s_i} \subseteq N(s_i)$, with $|X_{s_i}| =$ allowed-for-stars(s_i), and we want to route the paths avoiding these sets.

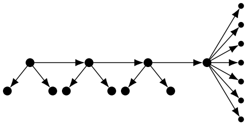
Our techniques: hardness results

We present several (more or less involved) NP-hardness reductions:









Further research

Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

$\boxed{\text{Our results}}$ Essentially, "only" for $\mathcal{A} = \{\text{paths}, \text{stars}\}$.

Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

$\boxed{\text{Our results}}$ Essentially, “only” for $\mathcal{A} = \{\text{paths}, \text{stars}\}$.

... but $\mathcal{A} = \{\text{matchings}\}$ is even polynomial for $k = 1$!

Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

$\boxed{\text{Our results}}$ Essentially, "only" for $\mathcal{A} = \{\text{paths}, \text{stars}\}$.

... but $\mathcal{A} = \{\text{matchings}\}$ is even polynomial for $k = 1$! (not for $k = 2$)

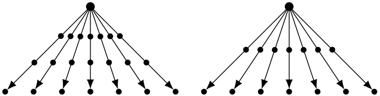
Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

Our results Essentially, "only" for $\mathcal{A} = \{$paths, stars$\}$.

... but $\mathcal{A} = \{$matchings$\}$ is even polynomial for $k = 1$! (not for $k = 2$)



hard

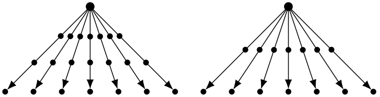
Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

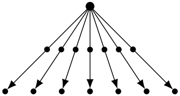
- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

Our results Essentially, “only” for $\mathcal{A} = \{\text{paths}, \text{stars}\}$.

... but $\mathcal{A} = \{\text{matchings}\}$ is even polynomial for $k = 1$! (not for $k = 2$)



hard



easy

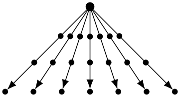
Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

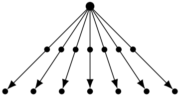
- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

Our results Essentially, "only" for $\mathcal{A} = \{\text{paths, stars}\}$.

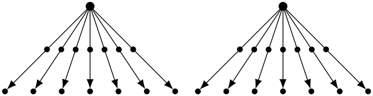
... but $\mathcal{A} = \{\text{matchings}\}$ is even polynomial for $k = 1$! (not for $k = 2$)



hard



easy



????

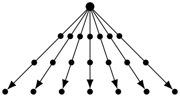
Further research

For which collections $\mathcal{A}$ of allowed digraphs the following holds?

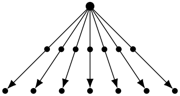
- If H is a digraph consisting of the union of $\leq k$ digraphs from $\mathcal{A}$.
- Then deciding if a digraph D on n vertices and directed treewidth dtw contains H as a subdigraph can be solved in time $n^{f(k,\mathrm{dtw})}$.

Our results Essentially, “only” for $\mathcal{A} = \{\text{paths}, \text{stars}\}$.

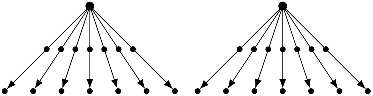
... but $\mathcal{A} = \{\text{matchings}\}$ is even polynomial for $k = 1$! (not for $k = 2$)



hard



easy



????

Strong connection with notoriously open problem: Exact Matching.

Gràcies!