Trivially perfect graphs

Based on a joint work with
Pål Grønås Drange, Fedor V. Fomin, and Yngve Villanger
Trivially perfect graphs

- Based on a joint work with Pål Grønås Drange, Fedor V. Fomin, and Yngve Villanger
- A graph $H$ is \textit{trivially perfect} if it is a transitive closure of some rooted forest.
Based on a joint work with Pål Grønås Drange, Fedor V. Fomin, and Yngve Villanger

A graph $H$ is \textit{trivially perfect} if it is a transitive closure of some rooted forest.
Trivially perfect graphs

- Based on a joint work with Pål Grønås Drange, Fedor V. Fomin, and Yngve Villanger
- A graph $H$ is \textit{trivially perfect} if it is a transitive closure of some rooted forest.
Recursive definition: A graph $H$ is trivially perfect iff it is $K_1$ or it can be constructed using the following two operations:
Recursive definition: A graph $H$ is trivially perfect iff it is $K_1$ or it can be constructed using the following two operations:

- taking a disjoint union of two trivially perfect graphs, or
**Recursive definition**: A graph $H$ is trivially perfect iff it is $K_1$ or it can be constructed using the following two operations:

- taking a disjoint union of two trivially perfect graphs, or
- adding a universal vertex to a trivially perfect graph.

Forbidden obstacles: A graph $H$ is trivially perfect iff it does not contain $C_4$ or $P_4$ as an induced subgraph.

$\text{Threshold} \subseteq \text{TriviallyPerfect} \subseteq \text{Interval}$.

Tight connections with graph parameter treedepth.

From now on, all trivially perfect graphs will be connected.
Trivially perfect graphs: alternative definitions

- **Recursive definition:** A graph $H$ is trivially perfect iff it is $K_1$ or it can be constructed using the following two operations:
  - taking a disjoint union of two trivially perfect graphs, or
  - adding a universal vertex to a trivially perfect graph.

- **Forbidden obstacles:** A graph $H$ is trivially perfect iff it does not contain $C_4$ or $P_4$ as an induced subgraph.
**Recursive definition:** A graph $H$ is trivially perfect iff it is $K_1$ or it can be constructed using the following two operations:
- taking a disjoint union of two trivially perfect graphs, or
- adding a universal vertex to a trivially perfect graph.

**Forbidden obstacles:** A graph $H$ is trivially perfect iff it does not contain $C_4$ or $P_4$ as an induced subgraph.

**Threshold $\subseteq$ TriviallyPerfect $\subseteq$ Interval.**
Recursive definition: A graph $H$ is trivially perfect iff it is $K_1$ or it can be constructed using the following two operations:
- taking a disjoint union of two trivially perfect graphs, or
- adding a universal vertex to a trivially perfect graph.

Forbidden obstacles: A graph $H$ is trivially perfect iff it does not contain $C_4$ or $P_4$ as an induced subgraph.

Threshold $\subseteq$ TriviallyPerfect $\subseteq$ Interval.

Tight connections with graph parameter treedepth.
**Recursive definition:** A graph $H$ is trivially perfect iff it is $K_1$ or it can be constructed using the following two operations:

- taking a disjoint union of two trivially perfect graphs, or
- adding a universal vertex to a trivially perfect graph.

**Forbidden obstacles:** A graph $H$ is trivially perfect iff it does not contain $C_4$ or $P_4$ as an induced subgraph.

- Threshold $\subseteq$ TriviallyPerfect $\subseteq$ Interval.
- Tight connections with graph parameter *treedepth*.
- From now on, all trivially perfect graphs will be connected.
A universal clique decomposition of a graph $H$ is a rooted tree $T$, where every node $t$ is assigned a bag $B_t$ such that:

- $\{B_t : t \in V(T)\}$ form a partition of $V(H)$;
- non-leaf nodes of $T$ have at least two sons;
- $uv \in E(H)$ if and only if $u$ and $v$ share a bag, or their bags are in ancestor-descendant relation.
UC-decomposition on a picture
Universal Clique Decomposition

A *universal clique decomposition* of a graph $H$ is a rooted tree $T$, where every node $t$ is assigned a bag $B_t$ such that:

- $\{B_t : t \in V(T)\}$ form a partition of $V(H)$;
- non-leaf nodes of $T$ have at least two sons;
- $uv \in E(H)$ if and only if $u$ and $v$ share a bag, or their bags are in ancestor-descendant relation.

Lemma

A connected graph is trivially perfect if and only if it admits a universal clique decomposition. Moreover, the universal clique decomposition of a trivially perfect graph is unique.
**Trivially perfect completion**

- \( F \) is a *completion set* for \( G \) if \( G + F = (V(G), E(G) \cup F) \) is a TP-graph.
Trivially perfect completion

- \( F \) is a completion set for \( G \) if \( G + F = (V(G), E(G) \cup F) \) is a TP-graph.

- **Trivially Perfect Completion:** given \( G \) and \( k \), does there exist a completion \( F \) for \( G \) such that \( |F| \leq k \)?
Trivially perfect completion

- $F$ is a completion set for $G$ if $G + F = (V(G), E(G) \cup F)$ is a TP-graph.

**Trivially Perfect Completion:** given $G$ and $k$, does there exist a completion $F$ for $G$ such that $|F| \leq k$?

**WRONG:** Is there a completion set of size $\leq k$ that kills all induced $C_4$-s and $P_4$-s?

**RIGHT:** Can one find a rooted tree of bags that needs at most $k$ fill edges to be turned into a universal clique decomposition?
- $F$ is a completion set for $G$ if $G + F = (V(G), E(G) \cup F)$ is a TP-graph.

- **Trivially Perfect Completion**: given $G$ and $k$, does there exists a completion $F$ for $G$ such that $|F| \leq k$?

- **WRONG**: Is there a completion set of size $\leq k$ that kills all induced $C_4$-s and $P_4$-s?

- **RIGHT**: Can one find a rooted tree of bags that needs at most $k$ fill edges to be turned into a universal clique decomposition?
Lemma 1

Suppose $F$ is a minimal completion for $G$, and let $C$ be a subtree of the UC-decomposition of $G + F$. Then $G[V(C)] := G[\bigcup_{t \in V(C)} B_t]$ is connected.
**Lemma 1**

Suppose $F$ is a minimal completion for $G$, and let $C$ be a subtree of the UC-decomposition of $G + F$. Then $G[V(C)] := G[\bigcup_{t \in V(C)} B_t]$ is connected.
Lemma 1

Suppose $F$ is a minimal completion for $G$, and let $C$ be a subtree of the UC-decomposition of $G + F$. Then $G[V(C)] := G[\bigcup_{t \in V(C)} B_t]$ is connected.
Lemma 2

Assume $F$ is a minimal completion for $G$ and let $v$ be any vertex. Let $C_1, C_2, \ldots, C_p$ be the subtrees of the UC-decomposition of $G + F$ below the bag of $v$. Then $v$ has at least one neighbour in each of $V(C_1), V(C_2), \ldots, V(C_p)$ in $G$. 
Lemma 2

Assume $F$ is a minimal completion for $G$ and let $v$ be any vertex. Let $C_1, C_2, \ldots, C_p$ be the subtrees of the UC-decomposition of $G + F$ below the bag of $v$. Then $v$ has at least one neighbour in each of $V(C_1), V(C_2), \ldots, V(C_p)$ in $G$. 
Lemma 2

Assume $F$ is a minimal completion for $G$ and let $v$ be any vertex. Let $C_1, C_2, \ldots, C_p$ be the subtrees of the UC-decomposition of $G + F$ below the bag of $v$. Then $v$ has at least one neighbour in each of $V(C_1), V(C_2), \ldots, V(C_p)$ in $G$. 
Lemma 2

Assume $F$ is a minimal completion for $G$ and let $v$ be any vertex. Let $C_1, C_2, \ldots, C_p$ be the subtrees of the UC-decomposition of $G + F$ below the bag of $v$. Then $v$ has at least one neighbour in each of $V(C_1), V(C_2), \ldots, V(C_p)$ in $G$. 
For the UC-decomposition $\mathcal{T}$ of a TP graph $H$ and its node $t$, a block at $t$ is the pair $(Q, D)$, where $Q = B_t$ and $D$ is the set of vertices in the subtree of $\mathcal{T}$ rooted in $t$. 
For the UC-decomposition $\mathcal{T}$ of a TP graph $H$ and its node $t$, a block at $t$ is the pair $(Q, D)$, where $Q = B_t$ and $D$ is the set of vertices in the subtree of $\mathcal{T}$ rooted in $t$. 
For the UC-decomposition $\mathcal{T}$ of a TP graph $H$ and its node $t$, a block at $t$ is the pair $(Q, D)$, where $Q = B_t$ and $D$ is the set of vertices in the subtree of $\mathcal{T}$ rooted in $t$. 
For the UC-decomposition $T$ of a TP graph $H$ and its node $t$, a block at $t$ is the pair $(Q, D)$, where $Q = B_t$ and $D$ is the set of vertices in the subtree of $T$ rooted in $t$. 
Let $S_0$ be the set of all pairs $(Q, D)$ such that $Q \subseteq D \subseteq V(G)$ and $G[D]$ is connected. Of course $|S_0| \leq 3^n$. 

**Lemma 1** gives the following recurrence:

$$T[Q, D] = \#\text{edges needed to make } Q \text{ a UC} + \sum_{C \in \text{cc}(G[D] \setminus Q)} (\min_{Q_C \subseteq C} T[Q_C, C])$$
Let $S_0$ be the set of all pairs $(Q, D)$ such that $Q \subseteq D \subseteq V(G)$ and $G[D]$ is connected. Of course $|S_0| \leq 3^n$.

Define the following value:

$$T[Q, D] := \text{Minimum \#edges needed to turn } G[D] \text{ into a TP-graph with } Q \text{ being the universal clique}$$
Let $S_0$ be the set of all pairs $(Q, D)$ such that $Q \subseteq D \subseteq V(G)$ and $G[D]$ is connected. Of course $|S_0| \leq 3^n$.

Define the following value:

$$T[Q, D] := \text{Minimum } \#\text{edges needed to turn } G[D] \text{ into a TP-graph with } Q \text{ being the universal clique}$$

**Lemma 1** gives the following recurrence:

$$T[Q, D] = \#\text{edges needed to make } Q \text{ a UC} + \sum_{C \in cc(G[D \setminus Q])} \left( \min_{Q_C \subseteq C} T[Q_C, C] \right)$$
Trimmed dynamic programming

- Exact $O^*(4^n)$ algorithm: compute $T[\cdot, \cdot]$ for the whole family $S_0$, and output $\min_{Q \subseteq V(G)} T[Q, V(G)]$.

Idea: Consider some $S \subseteq S_0$ and trim the DP to $S$:

We define $T_S[Q, D]$ only for $(Q, D) \in S$.

In the recurrence for $T_S[Q, D]$ we take the minimum of $T_{S_C}[Q_C, C]$ over $Q_C \subseteq C$ such that $(Q_C, C) \in S$.

Easy: $T_S[Q, D] \geq T[Q, D]$ for every $(Q, D) \in S$.

Crucial: Suppose there exists a minimum completion $F$ such that every block of the UC-decomposition of $G + F$ belongs to $S$. Then $\min_{Q \subseteq V(G)} T[Q, V(G)] = \min_{Q \subseteq V(G)} T[Q, V(G)]$. 
Trimmed dynamic programming

- Exact $O^*(4^n)$ algorithm: compute $T[\cdot, \cdot]$ for the whole family $S_0$, and output $\min_{Q \subseteq V(G)} T[Q, V(G)]$.

- **Idea**: Consider some $S \subseteq S_0$ and trim the DP to $S$:
Trimmed dynamic programming

- Exact $O^*(4^n)$ algorithm: compute $T[\cdot, \cdot]$ for the whole family $S_0$, and output $\min_{Q \subseteq V(G)} T[Q, V(G)]$.

- **Idea**: Consider some $S \subseteq S_0$ and trim the DP to $S$:
  - We define $T_S[Q, D]$ only for $(Q, D) \in S$.
Trimmed dynamic programming

- Exact $O^*(4^n)$ algorithm: compute $T[\cdot, \cdot]$ for the whole family $S_0$, and output $\min_{Q \subseteq V(G)} T[Q, V(G)]$.

**Idea**: Consider some $S \subseteq S_0$ and trim the DP to $S$:

- We define $T_S[Q, D]$ only for $(Q, D) \in S$.
- In the recurrence for $T_S[\cdot, \cdot]$ we take the minimum of $T_S[Q_C, C]$ over $Q_C \subseteq C$ such that $(Q_C, C) \in S$. 

Trimmed dynamic programming

- Exact $O^*(4^n)$ algorithm: compute $T[\cdot, \cdot]$ for the whole family $S_0$, and output $\min_{Q \subseteq V(G)} T[Q, V(G)]$.

- **Idea:** Consider some $S \subseteq S_0$ and trim the DP to $S$:
  - We define $T_S[Q, D]$ only for $(Q, D) \in S$.
  - In the recurrence for $T_S[\cdot, \cdot]$ we take the minimum of $T_S[Q_C, C]$ over $Q_C \subseteq C$ such that $(Q_C, C) \in S$.

- **Easy:** $T_S[Q, D] \geq T[Q, D]$ for every $(Q, D) \in S$. 

---

Part II Subexponential FPT algorithms for completion problems
Trimmed dynamic programming

- **Exact** \(O^*(4^n)\) algorithm: compute \(T[\cdot, \cdot]\) for the whole family \(S_0\), and output \(\min_{Q \subseteq V(G)} T[Q, V(G)]\).

- **Idea**: Consider some \(S \subseteq S_0\) and trim the DP to \(S\):
  - We define \(T_S[Q, D]\) **only** for \((Q, D) \in S\).
  - In the recurrence for \(T_S[\cdot, \cdot]\) we take the minimum of \(T_S[Q_C, C]\) over \(Q_C \subseteq C\) such that \((Q_C, C) \in S\).

- **Easy**: \(T_S[Q, D] \geq T[Q, D]\) for every \((Q, D) \in S\).

- **Crucial**: Suppose there exists a minimum completion \(F\) such that every block of the UC-decomposition of \(G + F\) belongs to \(S\). Then

\[
\min_{(Q, V(G)) \in S} T_S[Q, V(G)] = \min_{Q \subseteq V(G)} T[Q, V(G)].
\]
Running time of the trimmed DP: $O^*(|S|^2)$. 

Goal: Enumerate a family $S$ of potential blocks that is:

- rich enough so that every block of $G + F$ is captured for some optimum $F$, providing that $|F| \leq k$,
- small enough so that the DP will be efficient.

Our family $S$ will be of size $k O(\sqrt{k})$ (starting from a polykernel).

Note: In family $S$ we just need to hit all the blocks of OPT. We may put into $S$ a lot of unnecessary states as well, and they do not make any harm.
Trimmed dynamic programming

- Running time of the trimmed DP: $O^*(|\mathcal{S}|^2)$.
- **Goal**: Enumerate a family $\mathcal{S}$ of potential blocks that is:

  - **Goal**: Enumerate a family $\mathcal{S}$ of potential blocks that is:
Running time of the trimmed DP: $O^*(|S|^2)$.

**Goal**: Enumerate a family $S$ of potential blocks that is:
- **rich** enough so that every block of $G + F$ is captured for some optimum $F$, providing that $|F| \leq k$, and
Running time of the trimmed DP: \( O^*(|S|^2) \).

**Goal:** Enumerate a family \( S \) of potential blocks that is:

- **rich** enough so that every block of \( G + F \) is captured for some optimum \( F \), providing that \( |F| \leq k \), and
- **small** enough so that the DP will be efficient.
Trimmed dynamic programming

- Running time of the trimmed DP: $O^*(|S|^2)$.
- **Goal**: Enumerate a family $S$ of potential blocks that is:
  - **rich** enough so that every block of $G + F$ is captured for some optimum $F$, providing that $|F| \leq k$, and
  - **small** enough so that the DP will be efficient.
- Our family $S$ will be of size $k^{O(\sqrt{k})}$ (starting from a polykernel).
Trimmed dynamic programming

- Running time of the trimmed DP: $O^*(|S|^2)$.
- **Goal**: Enumerate a family $S$ of potential blocks that is:
  - **rich** enough so that every block of $G + F$ is captured for some optimum $F$, providing that $|F| \leq k$, and
  - **small** enough so that the DP will be efficient.
  - Our family $S$ will be of size $k^{O(\sqrt{k})}$
    (starting from a polykernel).

- **Note**: In family $S$ we just need to hit all the blocks of OPT. We may put into $S$ a lot of unnecessary states as well, and they do not make any harm.
Let \((Q, D)\) be a block and \(\Omega_1, \Omega_2, \Omega_3\) be as on the figure.
Let \((Q, D)\) be a block and \(\Omega_1, \Omega_2, \Omega_3\) be as on the figure. Then \(Q = (\Omega_1 \cap \Omega_2) \setminus \Omega_3\),
Let \((Q, D)\) be a block and \(\Omega_1, \Omega_2, \Omega_3\) be as on the figure. Then \(Q = (\Omega_1 \cap \Omega_2) \setminus \Omega_3\), and \(D\) is the vertex set of the connected component of \(G \setminus \Omega_3\) that contains \(Q\).
Every block in $G + F$ can be described by a triple of maximal cliques in $G + F$. 
Every block in $G + F$ can be described by a triple of maximal cliques in $G + F$.

If we enumerate a family $\mathcal{P}$ of potential maximal cliques, then we have a family of $|\mathcal{P}|^3$ potential blocks.
Every block in $G + F$ can be described by a triple of maximal cliques in $G + F$.

If we enumerate a family $\mathcal{P}$ of potential maximal cliques, then we have a family of $|\mathcal{P}|^3$ potential blocks.  

- Family $\mathcal{P}$ has to capture every maximal clique in $G + F$ for some optimum completion $F$. 

Part II

Subexponential FPT algorithms for completion problems
Every block in \( G + F \) can be described by a triple of maximal cliques in \( G + F \).

If we enumerate a family \( \mathcal{P} \) of potential maximal cliques, then we have a family of \( |\mathcal{P}|^3 \) potential blocks.

- Family \( \mathcal{P} \) has to capture every maximal clique in \( G + F \) for some optimum completion \( F \).
- We are going to find such a family of size \( k^{O(\sqrt{k})} \). Let us fix some optimum completion \( F \) with \( |F| \leq k \).
**Polynomial kernel**: There exists a cubic kernel of Guo for \textsc{TP-\textsc{Completion}}. Hence we can assume that \( n = \mathcal{O}(k^3) \).

**Goal**: A family \( \mathbb{P} \) of \( k^{\mathcal{O}(\sqrt{k})} \) subsets of \( V(G) \) such that every maximal clique in \( G + F \) belongs to \( \mathbb{P} \).