Canonical decompositions in bounded treedepth and bounded shrubdepth graphs

Wojciech Przybyszewski
Joint work with Pierre Ohlmann, Michał Pilipczuk, Szymon Toruńczyk

LoGAlg 2023

Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

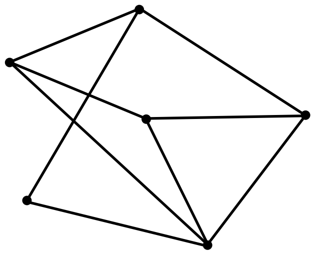
Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:



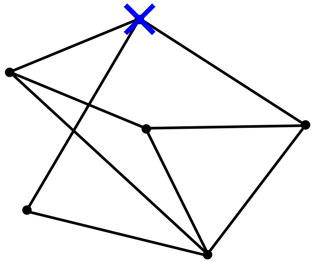
Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:



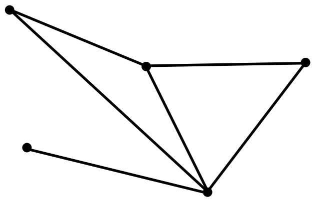
Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:

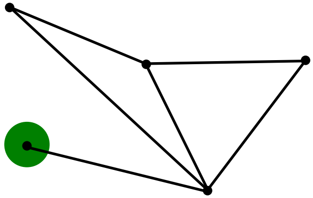


Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.
Example play of the treedepth game:



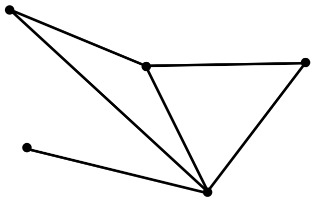
Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:



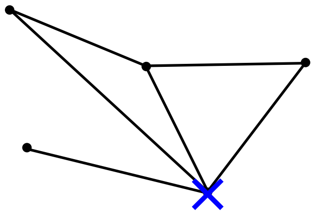
Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:



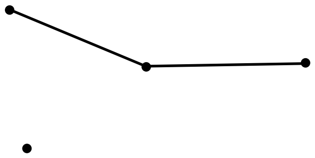
Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:



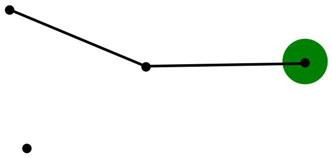
Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:

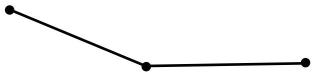


Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.
Example play of the treedepth game:

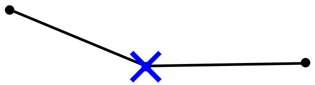


Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.
Example play of the treedepth game:



Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:

Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:

Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Example play of the treedepth game:

Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Definition

A treedepth of a graph G is the minimum number of rounds that are enough for Splitter to always win the treedepth game, no matter how Connector is playing.

Treedepth game

The treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v to delete
2. Connector chooses G_{i+1} as a connected component in $G_i - v$.

Splitter wins once G_i has size 1.

Definition

A treedepth of a graph G is the minimum number of rounds that are enough for Splitter to always win the treedepth game, no matter how Connector is playing.

Observation: We don't need to assume that G is a finite graph for this definition to make sense.

Progressing moves in the treedepth game

Theorem.

There exists a function $f : \mathbb{N} \to \mathbb{N}$ such that if a graph G has treedepth d then Splitter has at most $f(d)$ progressing moves[1].

[1] A vertex v is a progressing move for Splitter if every connected component C of $G - \{v\}$ has strictly smaller treedepth than G.

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Proof.

Assume statement doesn't hold. Denote $V(G) = \{v_i : i \in I\}$.

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Proof.

Assume statement doesn't hold. Denote $V(G) = \{v_i : i \in I\}$.

Consider the following theory over the signature that consists of constant symbols $\{v_i : i \in I\} \cup \{v_\infty\}$ and one binary relation E:

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Proof.

Assume statement doesn't hold. Denote $V(G) = \{v_i : i \in I\}$.

Consider the following theory over the signature that consists of constant symbols $\{v_i : i \in I\} \cup \{v_\infty\}$ and one binary relation E:

- $v_i \neq v_j$ for every $i, j \in I$;

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Proof.

Assume statement doesn't hold. Denote $V(G) = \{v_i : i \in I\}$.

Consider the following theory over the signature that consists of constant symbols $\{v_i : i \in I\} \cup \{v_\infty\}$ and one binary relation E:

- $v_i \neq v_j$ for every $i, j \in I$;
- $E(v_i, v_j)$ for every $(v_i, v_j) \in E(G)$ and $\neg E(v_i, v_j)$ for every $(v_i, v_j) \notin E(G)$;

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Proof.

Assume statement doesn't hold. Denote $V(G) = \{v_i : i \in I\}$.

Consider the following theory over the signature that consists of constant symbols $\{v_i : i \in I\} \cup \{v_\infty\}$ and one binary relation E:

- $v_i \neq v_j$ for every $i, j \in I$;
- $E(v_i, v_j)$ for every $(v_i, v_j) \in E(G)$ and $\neg E(v_i, v_j)$ for every $(v_i, v_j) \notin E(G)$;
- Splitter wins the treedepth game in d rounds if he plays optimally;

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Proof.

Assume statement doesn't hold. Denote $V(G) = \{v_i : i \in I\}$.

Consider the following theory over the signature that consists of constant symbols $\{v_i : i \in I\} \cup \{v_\infty\}$ and one binary relation E:

- $v_i \neq v_j$ for every $i, j \in I$;
- $E(v_i, v_j)$ for every $(v_i, v_j) \in E(G)$ and $\neg E(v_i, v_j)$ for every $(v_i, v_j) \notin E(G)$;
- Splitter wins the treedepth game in d rounds if he plays optimally;
- $v_\infty \neq v_i$ for every $i \in I$;

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.

Proof.

Assume statement doesn't hold. Denote $V(G) = \{v_i : i \in I\}$.

Consider the following theory over the signature that consists of constant symbols $\{v_i : i \in I\} \cup \{v_\infty\}$ and one binary relation E:

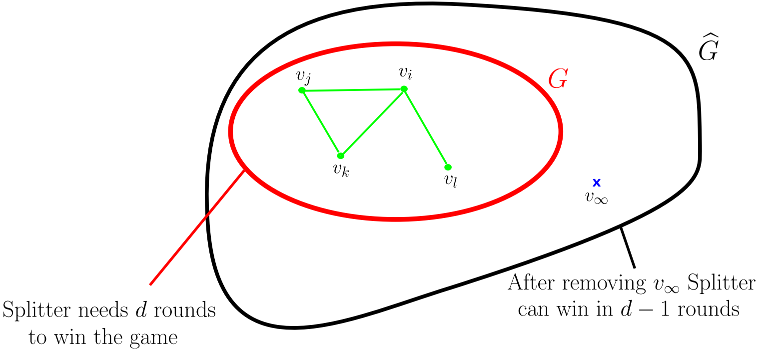
- $v_i \neq v_j$ for every $i, j \in I$;
- $E(v_i, v_j)$ for every $(v_i, v_j) \in E(G)$ and $\neg E(v_i, v_j)$ for every $(v_i, v_j) \notin E(G)$;
- Splitter wins the treedepth game in d rounds if he plays optimally;
- $v_\infty \neq v_i$ for every $i \in I$;
- v_∞ is a progressing move.

□

Progressing moves in infinite graphs

Lemma.

For every infinite graph G of treedepth d Splitter has finitely many progressing moves.



Progressing moves in infinite graphs

Theorem.

There exists a function $f : \mathbb{N} \to \mathbb{N}$ such that if a graph G has treedepth d then Splitter has at most $f(d)$ progressing moves.

Progressing moves in infinite graphs

Theorem.

There exists a function $f : \mathbb{N} \to \mathbb{N}$ such that if a graph G has treedepth d then Splitter has at most $f(d)$ progressing moves.

Proof.

Assume the statement is not true. Consider the following theory:

- Splitter can win the treedepth game in at most d rounds;

Progressing moves in infinite graphs

Theorem.

There exists a function $f : \mathbb{N} \to \mathbb{N}$ such that if a graph G has treedepth d then Splitter has at most $f(d)$ progressing moves.

Proof.

Assume the statement is not true. Consider the following theory:

- Splitter can win the treedepth game in at most d rounds;
- there are at least m progressing moves for every $m \in \mathbb{N}$.

Progressing moves in infinite graphs

Theorem.

There exists a function $f : \mathbb{N} \to \mathbb{N}$ such that if a graph G has treedepth d then Splitter has at most $f(d)$ progressing moves.

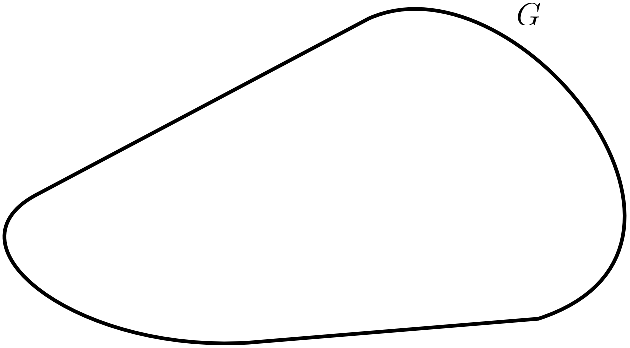
Proof.

Assume the statement is not true. Consider the following theory:

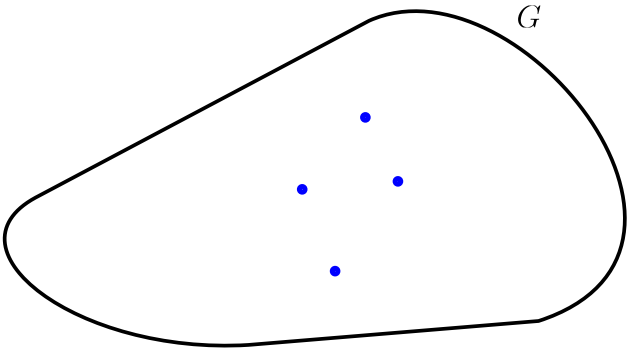
- Splitter can win the treedepth game in at most d rounds;
- there are at least m progressing moves for every $m \in \mathbb{N}$.

Compactness yields a model that contradicts the previous lemma. $\square$

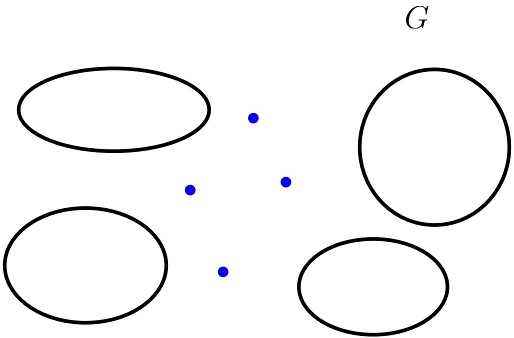
Canonical decomoposition of graphs of bounded treedepth



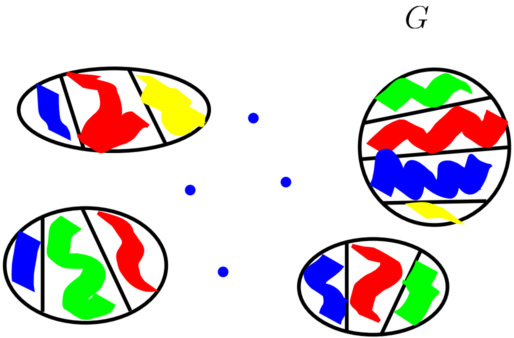
Canonical decomoposition of graphs of bounded treedepth



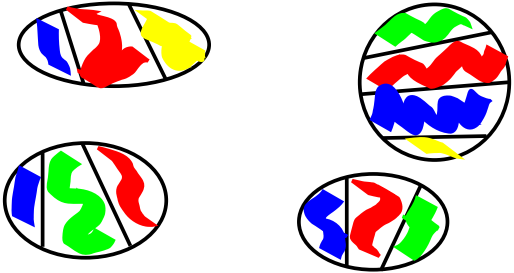
Canonical decomoposition of graphs of bounded treedepth



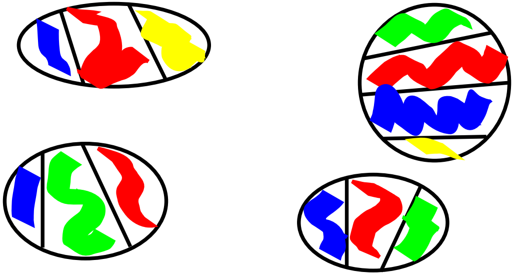
Canonical decomoposition of graphs of bounded treedepth



Canonical decomoposition of graphs of bounded treedepth



Canonical decomoposition of graphs of bounded treedepth



Observation: This yields a decomposition algorithm working in time $f(d) \cdot n^2$ on graphs of treedepth at most d.

Graph isomorphism for bounded treedepth

Theorem. [Bouland, Dawar, Kopczyński, 2012]

Graph isomorphism can be solved on graphs of treedepth at most d in time $f(d) \cdot n^3 \cdot \log n$.

Graph isomorphism for bounded treedepth

Theorem. [Bouland, Dawar, Kopczyński, 2012]

Graph isomorphism can be solved on graphs of treedepth at most d in time $f(d) \cdot n^3 \cdot \log n$.

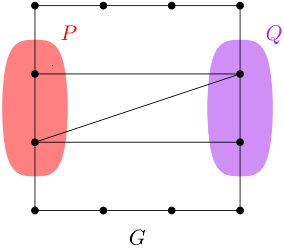
Remark: The running time can be further improved to $f(d) \cdot n \cdot \log^2 n$.

Flips

Denote by $G \oplus (P, Q)$ the graph obtained from G by complementing edges between pairs of vertices from $P \times Q$.

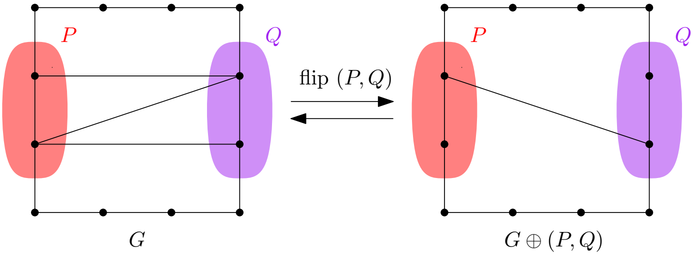
Flips

Denote by $G \oplus (P, Q)$ the graph obtained from G by complementing edges between pairs of vertices from $P \times Q$.



Flips

Denote by $G \oplus (P, Q)$ the graph obtained from G by complementing edges between pairs of vertices from $P \times Q$.



Shrubdepth Game

The Treedepth game is played on a graph G_1. In round i

1. Splitter chooses a vertex v
2. Connector chooses G_{i+1} as a connected component in $G - v$.

Splitter wins once G_i has size 1.

Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

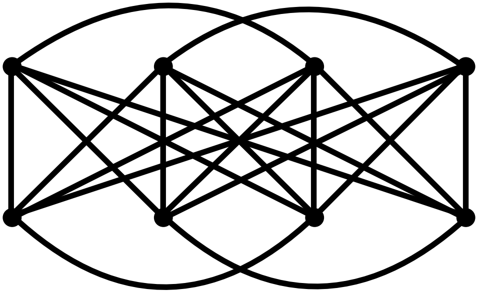
Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:



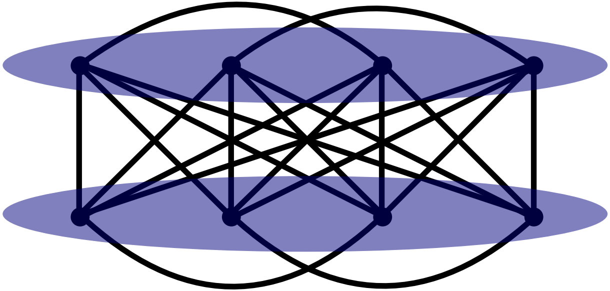
Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:



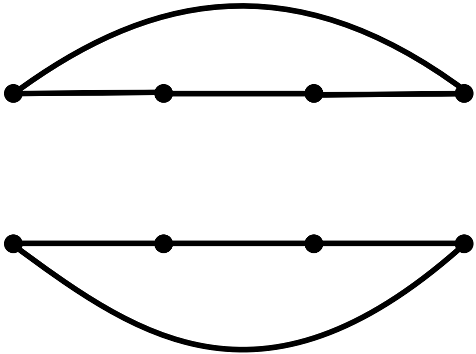
Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:



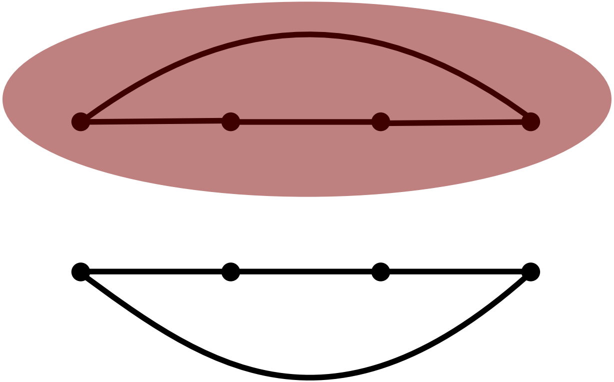
Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:



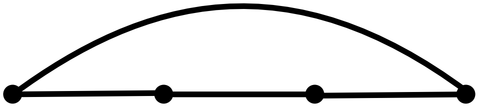
Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:



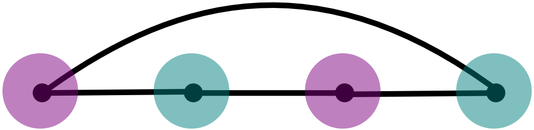
Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:



Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:

Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:

Shrubdepth Game

The Shrubdepth game is played on a graph G_1. In round i

1. Flipper chooses two sets P, Q
2. Connector chooses G_{i+1} as a connected component in $G_i \oplus (P, Q)$.

Flipper wins once G_i has size 1.

Example play of the shrubdepth game:

Beyond sparsity

1. We found canonical moves for Splitter in the treedepth game
2. to obtain canonical decompositions and a graph isomorphism algorithm for graphs of bounded treedepth.

Beyond sparsity

1. We want to find canonical moves for Flipper in the shrubdepth game
2. to obtain canonical decompositions and a graph isomorphism algorithm for graphs of bounded shrubdepth.

Beyond sparsity

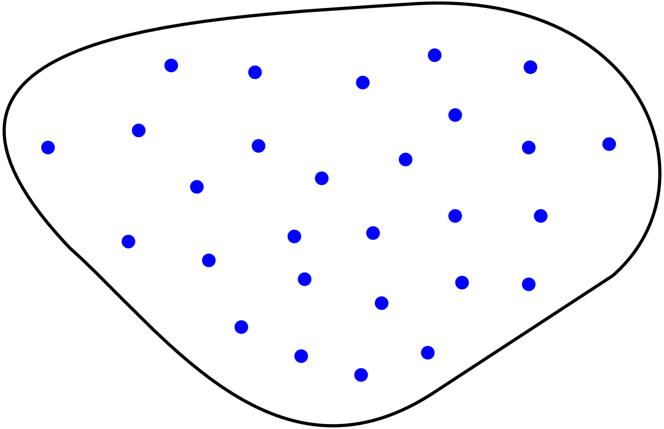
1. We want to find canonical moves for Flipper in the shrubdepth game
2. to obtain canonical decompositions and a graph isomorphism algorithm for graphs of bounded shrubdepth.

Defintion. [Ganian, Hliněný, Nešetřil, Obdržálek, Ossona de Mendez, 2017]

A graph G has *shrubdepth* at most d if Flipper can win the shrubdepth game on G in at most d rounds.

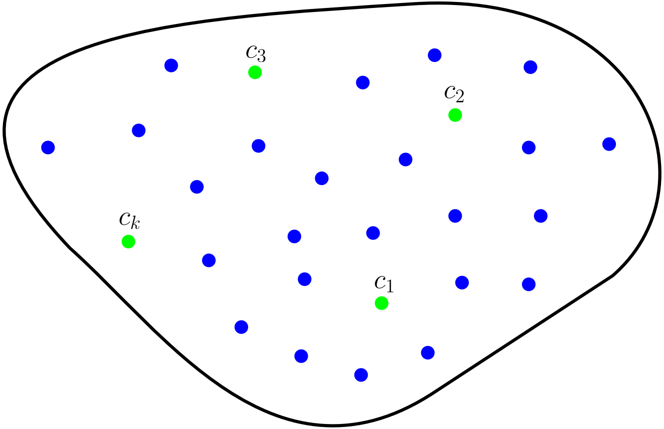
Finitary Substitute Lemma

$G \in \mathcal{C}$ $\qquad$ $\phi(x_1, x_2; y_1, \ldots, y_k)$ $\qquad$ ψ



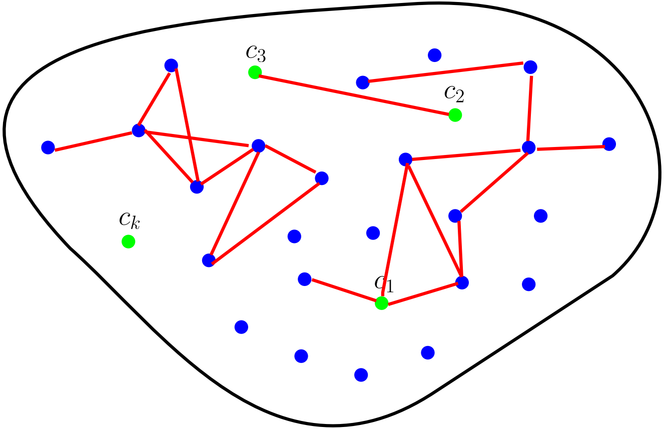
Finitary Substitute Lemma

$G \in \mathcal{C}$ $\phi(x_1, x_2; c_1, \ldots, c_k)$ ψ



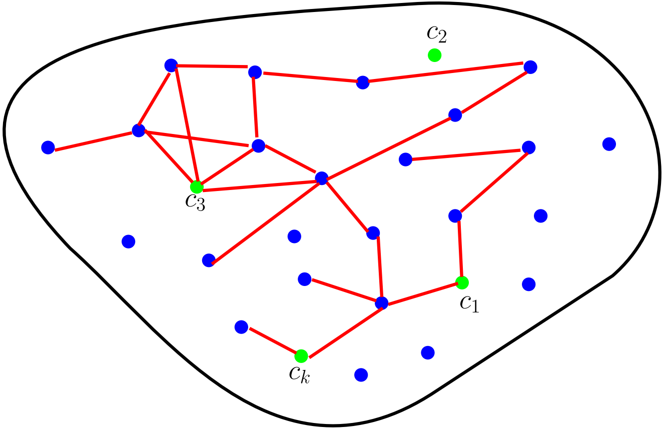
Finitary Substitute Lemma

$G \in \mathcal{C}$ $\phi(x_1, x_2; c_1, \ldots, c_k)$ ψ



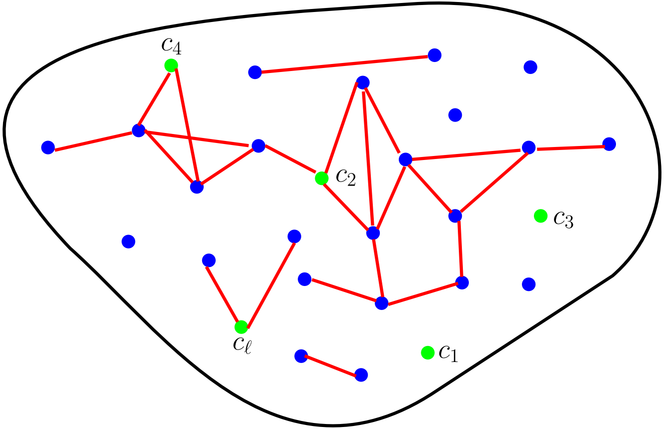
Finitary Substitute Lemma

$G \in \mathcal{C}$ $\qquad$ $\phi(x_1, x_2; c_1, \ldots, c_k)$ $\qquad$ ψ



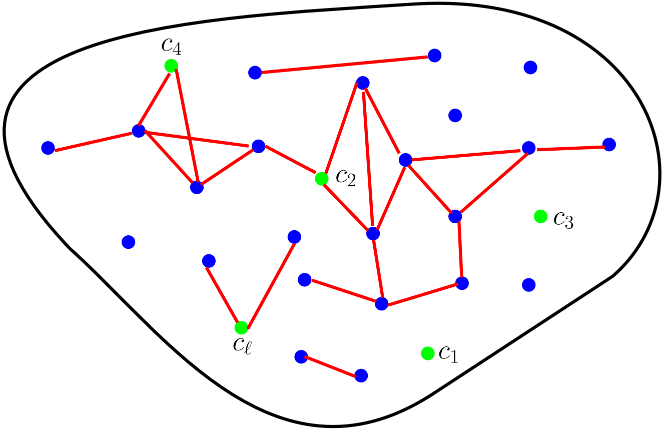
Finitary Substitute Lemma

$G \in \mathcal{C}$ $\quad$ $\phi'(x_1; x_2; y_1, \ldots, y_\ell)$ $\quad$ ψ



Finitary Substitute Lemma

$G \in \mathcal{C}$ $\qquad$ $\phi'(x_1; x_2; y_1, \ldots, y_\ell)$ $\qquad$ ψ



Proof uses a number of tools from stability theory [Shelah], most importantly properties of forking independence in stable theories.

Graph isomorphism on bounded shrubdepth

Theorem. [Ohlmann, Pilipczuk, Przybyszewski, Toruńczyk, 2023]

Graph isomorphism can be solved on graphs of shrubdepth at most d in time $f(d) \cdot n^2$.