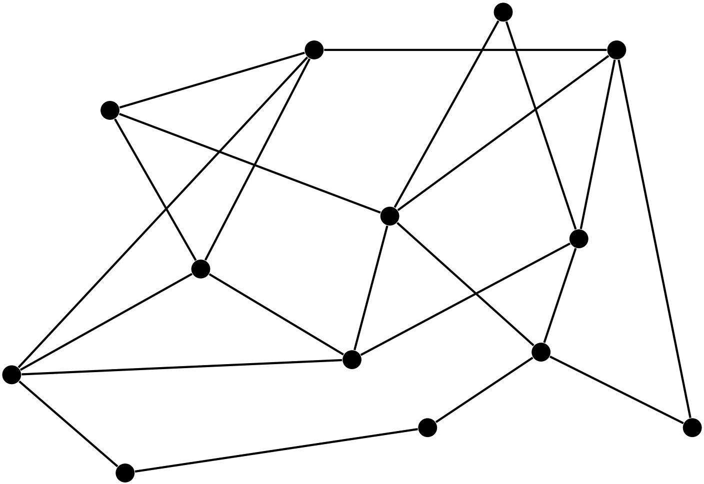
Finding Hamiltonian Cycle in Graphs of Bounded Treewidth: Experimental Evaluation[1]

Marcin Pilipczuk, Michał Ziobro

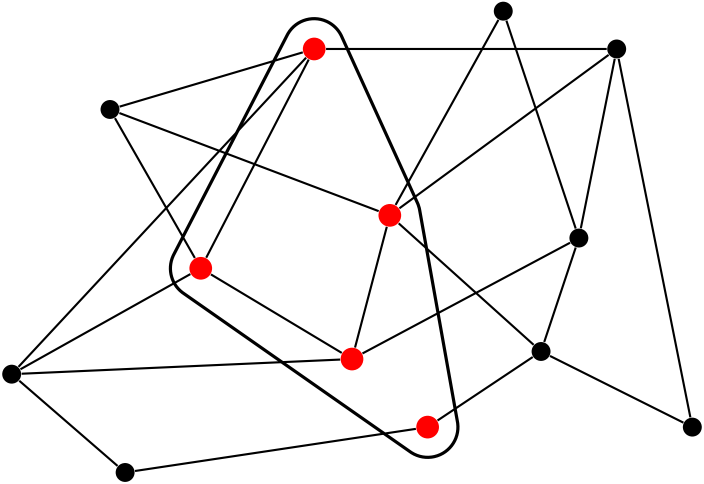
March 12, 2019

[1]Supported by the "Recent trends in kernelization: theory and experimental evaluation" project, carried out within the Homing programme of the Foundation for Polish Science co-financed by the European Union under the European Regional Development Fund.

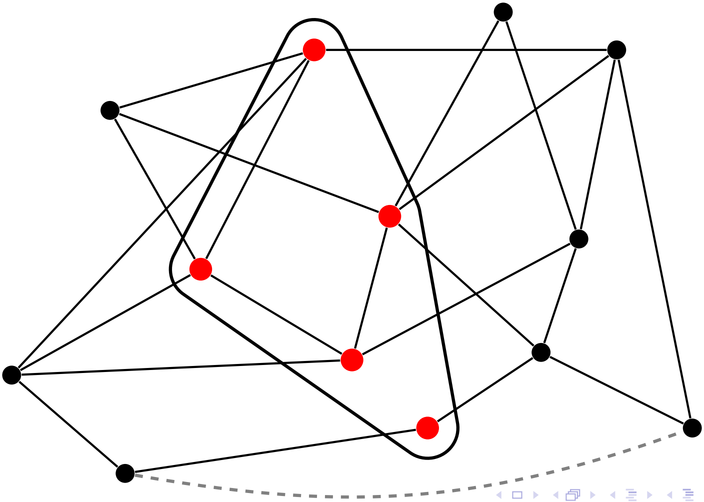
Separator



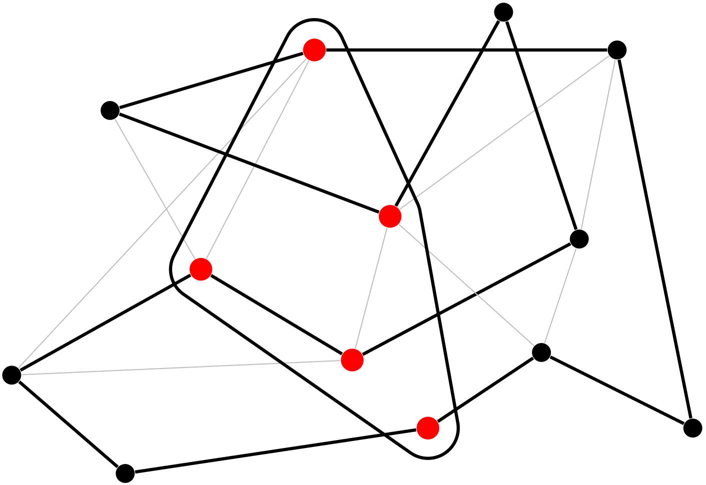
Separator



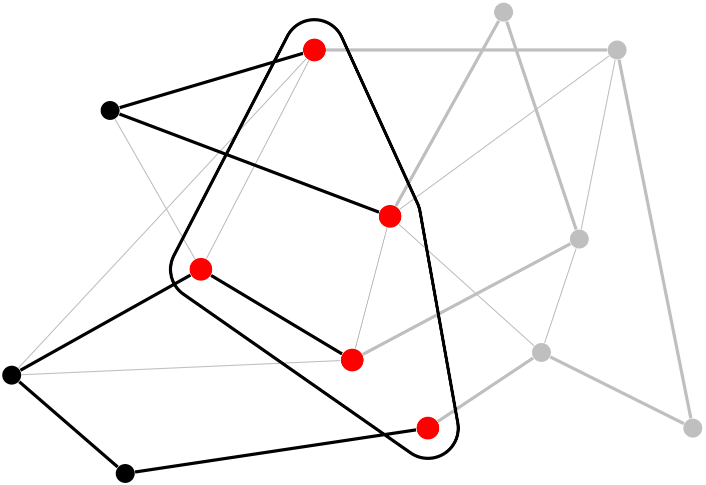
Separator



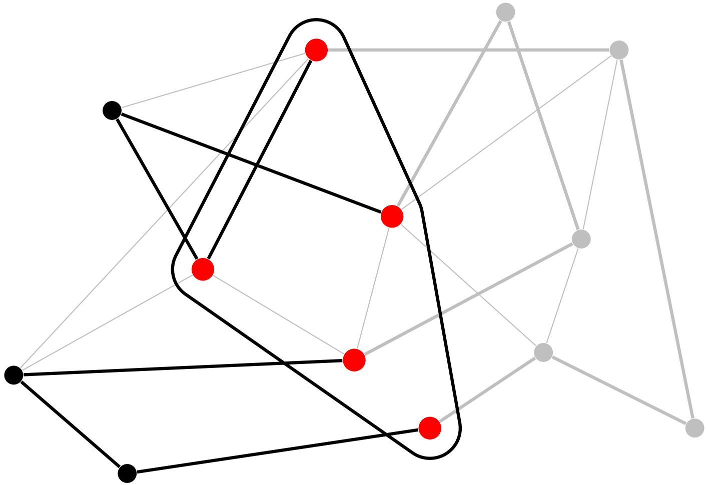
Separator



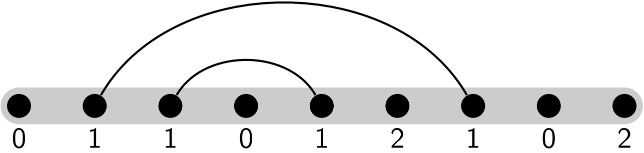
Separator



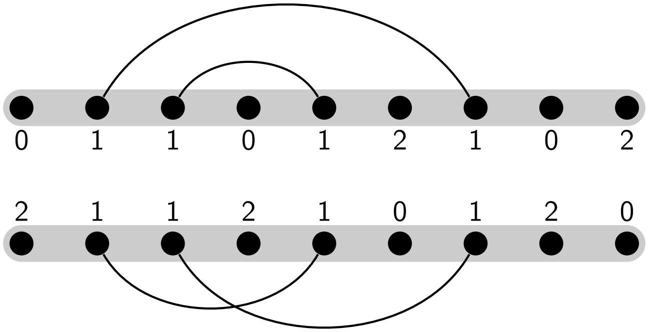
Separator



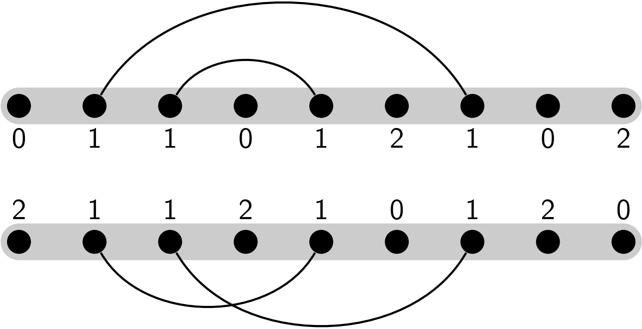
Fingerprints



Fingerprints

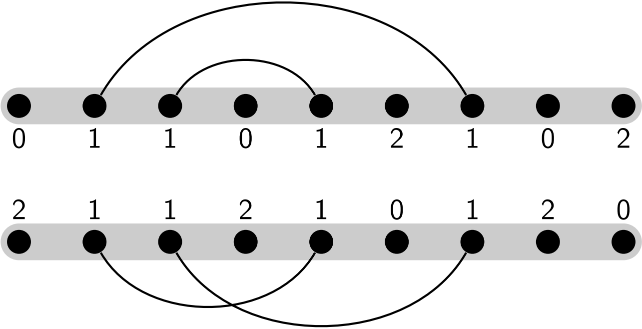


Fingerprints



same vertex degrees $\Rightarrow$ one class

Fingerprints



same vertex degrees $\Rightarrow$ one class

k vertices of degree 1 $\Rightarrow$ $k!!$ possible fingerprints

Naive algorithm

tree decomposition — set of separators covering whole graph
treewidth — size of largest separator in the tree decomposition (one with the smallest largest separator)

Naive algorithm

tree decomposition — set of separators covering whole graph
treewidth — size of largest separator in the tree decomposition (one with the smallest largest separator)

Basic idea:

- fingerprint set for trivial separator
- fingerprint set for $S' \sim S \Rightarrow$ fingerprint set for S

Naive algorithm

tree decomposition — set of separators covering whole graph
treewidth — size of largest separator in the tree decomposition (one with the smallest largest separator)

Basic idea:

- fingerprint set for trivial separator
- fingerprint set for $S' \sim S \Rightarrow$ fingerprint set for S

FPT dynamic algorithm with running time $2^{O(t \ln t)} O(n^c)$, where $t =$ treewidth

Representative sets

class — tuple of degrees of vertices, $\in \{0, 1, 2\}^S$
fingerprint — a class plus a matching on deg-1 vtcs

number of classes is small (3^t)

Representative sets

class — tuple of degrees of vertices, $\in \{0, 1, 2\}^S$
fingerprint — a class plus a matching on deg-1 vtcs

number of classes is small (3^t)

bottleneck - size of a class ($k!!$)

Representative sets

class — tuple of degrees of vertices, $\in \{0, 1, 2\}^S$
fingerprint — a class plus a matching on deg-1 vtcs

number of classes is small (3^t)

bottleneck - size of a class ($k!!$)

representative set F' of F : $f \in F$ fits $g \Rightarrow \exists_{f' \in F'} f'$ fits g

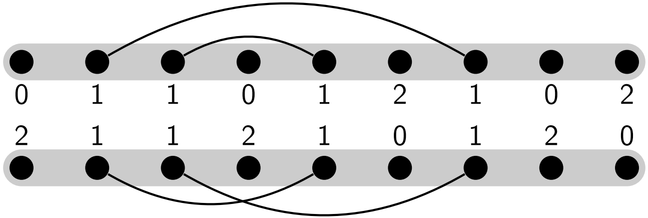
Representative sets

class — tuple of degrees of vertices, $\in \{0,1,2\}^S$
fingerprint — a class plus a matching on deg-1 vtcs

number of classes is small (3^t)

bottleneck - size of a class ($k!!$)

representative set F' of F : $f \in F$ fits $g \Rightarrow \exists_{f' \in F'} f'$ fits g



Representative sets

class — tuple of degrees of vertices, $\in \{0, 1, 2\}^S$
fingerprint — a class plus a matching on deg-1 vtcs

number of classes is small (3^t)

bottleneck - size of a class (k!!)

representative set F' of F : $f \in F$ fits $g \Rightarrow \exists_{f' \in F'} f'$ fits g

- 2^{k-1} (Bodleander et al., 2012)
- $2^{k/2-1}$ (Cygan et al., 2013)

Representative sets

class — tuple of degrees of vertices, $\in \{0,1,2\}^S$
fingerprint — a class plus a matching on deg-1 vtcs

number of classes is small (3^t)

bottleneck - size of a class ($k!!$)

representative set F' of F : $f \in F$ fits $g \Rightarrow \exists_{f' \in F'} f'$ fits g

- 2^{k-1} (Bodleander et al., 2012)
- $2^{k/2-1}$ (Cygan et al., 2013)

both are *rank-based* approaches $\Rightarrow$ size of representative set bounded by rank of certain matrix

Representative sets

class — tuple of degrees of vertices, $\in \{0,1,2\}^S$
fingerprint — a class plus a matching on deg-1 vtcs

number of classes is small (3^t)

bottleneck - size of a class ($k!!$)

representative set F' of F : $f \in F$ fits $g \Rightarrow \exists_{f' \in F'} f'$ fits g

- 2^{k-1} (Bodleander et al., 2012) **(rank-based 1)**
- $2^{k/2-1}$ (Cygan et al., 2013) **(rank-based 2)**

both are *rank-based* approaches $\Rightarrow$ size of representative set bounded by rank of certain matrix

Cut-and-count approach

- randomized
- algebraic
- theoretically fastest

Cut-and-count approach

- randomized
- algebraic
- theoretically fastest

Evaluation of a poly over large field of characteristic 2:

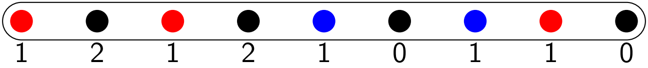
$$\sum_{(R,B)\in\mathcal{C}} \prod_{e\in R\cup B} x_e$$

Cut-and-count approach

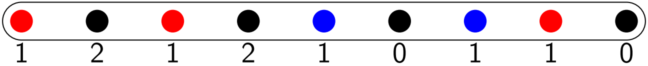
- randomized
- algebraic
- theoretically fastest

Evaluation of a poly over large field of characteristic 2:

$$\sum_{(R,B)\in\mathcal{C}} \prod_{e\in R\cup B} x_e$$

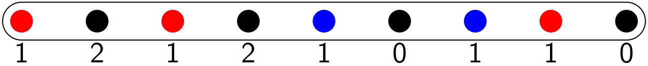


Cut-and-count approach



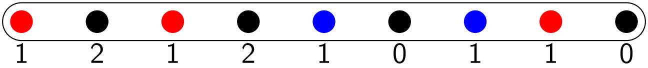
- 4^t states, deg-1 vertices red or blue,

Cut-and-count approach



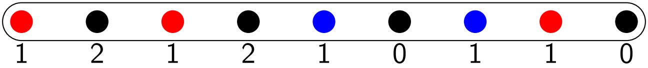
- 4^t states, deg-1 vertices red or blue,
- evaluate some polynomial over $GF(2^s)$,

Cut-and-count approach



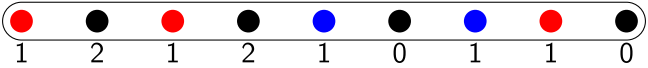
- 4^t states, deg-1 vertices red or blue,
- evaluate some polynomial over $GF(2^s)$,
- monomials from non-solutions cancel out, from solutions stay,

Cut-and-count approach



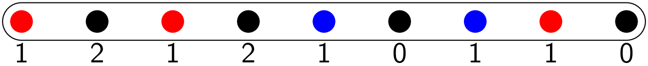
- 4^t states, deg-1 vertices red or blue,
- evaluate some polynomial over $GF(2^s)$,
- monomials from non-solutions cancel out, from solutions stay,
- Schwarz-Zippel: random values, from $GF(2^{64})$,

Cut-and-count approach



- 4^t states, deg-1 vertices red or blue,
- evaluate some polynomial over $GF(2^s)$,
- monomials from non-solutions cancel out, from solutions stay,
- Schwarz-Zippel: random values, from $GF(2^{64})$,
- naive join nodes: 9^t,

Cut-and-count approach



- 4^t states, deg-1 vertices red or blue,
- evaluate some polynomial over $GF(2^s)$,
- monomials from non-solutions cancel out, from solutions stay,
- Schwarz-Zippel: random values, from $GF(2^{64})$,
- naive join nodes: 9^t,
- transform at join nodes: 4^t, but problems with $GF(2^{64})$.

Data sets

FHCP Challenge - 1001 instances

Data sets

FHCP Challenge - 1001 instances
623 instances with treewidth below 10 (*fill-in* heuristic)

Data sets

FHCP Challenge - 1001 instances
623 instances with treewidth below 10 (*fill-in* heuristic)
19 instances with treewidth between 17 and 29 (heuristic by Ben Strasser, 2nd place on PACE 2017)

Data sets

FHCP Challenge - 1001 instances
623 instances with treewidth below 10 (*fill-in* heuristic)
19 instances with treewidth between 17 and 29 (heuristic by Ben Strasser, 2nd place on PACE 2017)

- A - instances with small treewidth from FHCP Challenge
- B - randomly sampled subset of A (for adjusting hyperparameters)

Data sets

FHCP Challenge - 1001 instances
623 instances with treewidth below 10 (*fill-in* heuristic)
19 instances with treewidth between 17 and 29 (heuristic by Ben Strasser, 2nd place on PACE 2017)

- A - instances with small treewidth from FHCP Challenge
- B - randomly sampled subset of A (for adjusting hyperparameters)
- C - instances with treewidth between 17 and 29 from FHCP Challenge
- D - subset of C which were solved by at least one of our algorithms (for adjusting hyperparameters)
- E - our few random instances

Small treewidth results

- **1st rank-based**: 18.59% times slower than naive,
- **2nd rank-based**: 10.97% faster,
- **cut-and-count**: solved 499 from 623 instances (TL: 600s)

test	$\|V\|$	tw	naive	rank-based 1	rank-based 2	c&c
0556	3274	9	20.655	27.794	28.024	128.231
0728	4170	9	30.861	38.578	38.823	279.871
0947	6598	9	128.733	143.144	142.427	467.181
0584	3411	9	105.371	114.240	73.291	-
0746	4286	9	631.261	619.601	381.351	-
0778	4561	8	17.468	16.974	13.069	-
0950	6620	9	196.572	206.482	124.641	-

Large treewidth results

test	$\lvert V \rvert$	tw	naive	rank-based 1	rank-based 2	c&c
0074	462	28	38.737	109.655	110.040	-
0253	1578	29	93.343	167.458	167.440	-
0268	1644	25	36.449	70.157	69.111	-
0272	1662	25	554.271	1260.329	1230.722	-
0298	1806	23	10.035	18.611	18.492	-
0172	1002	25	1.156	1.298	.554	-
0199	1200	25	13.513	15.419	3.369	-
E0002	600	18	204.197	-	28.882	-
E0003	700	20	-	-	711.778	-
E0007	360	15	1575.475	-	328.191	-

Conclusions

- Even on tests with small treewidth rank-based approach can help, but please use 2nd rank-based approach.

Conclusions

- Even on tests with small treewidth rank-based approach can help, but please use 2nd rank-based approach.
- For sparse graphs with a large treewidth a cost of dividing fingerprints into families is often greater than gain from reducing number of them.

Conclusions

- Even on tests with small treewidth rank-based approach can help, but please use 2nd rank-based approach.
- For sparse graphs with a large treewidth a cost of dividing fingerprints into families is often greater than gain from reducing number of them.
- Cut-and-count approach is impractical.

Conclusions

- Even on tests with small treewidth rank-based approach can help, but please use 2nd rank-based approach.
- For sparse graphs with a large treewidth a cost of dividing fingerprints into families is often greater than gain from reducing number of them.
- Cut-and-count approach is impractical.
- **Conjecture**: reducing only classes with 4 vertices of degree one may be the best.