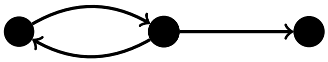
Enumeration of Preferred Extensions in Almost Oriented Digraphs

Serge Gaspers Ray Li

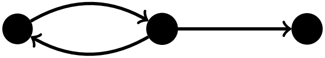
UNSW Sydney, Australia

MFCS 2019

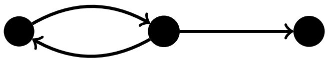
Argumentation Frameworks



Argumentation Frameworks



Admissibility:

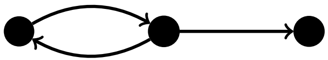


Admissibility:

Independence:

$$\forall a, b \in S.(a, b) \notin E(G)$$

Argumentation Frameworks



Admissibility:

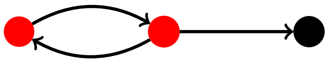
Independence:

$$\forall a, b \in S.(a, b) \notin E(G)$$

Defends itself:

$$N^{-}(S) \subseteq N^{+}(S).$$

Argumentation Frameworks



Admissibility:

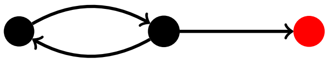
Independence:

$$\forall a, b \in S.(a, b) \notin E(G)$$

Defends itself:

$$N^{-}(S) \subseteq N^{+}(S).$$

Argumentation Frameworks



Admissibility:

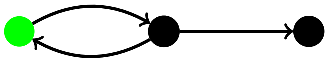
Independence:

$$\forall a, b \in S.(a, b) \notin E(G)$$

Defends itself:

$$N^-(S) \subseteq N^+(S).$$

Argumentation Frameworks



Admissibility:

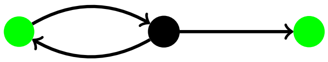
Independence:

$$\forall a, b \in S.(a, b) \notin E(G)$$

Defends itself:

$$N^{-}(S) \subseteq N^{+}(S).$$

Argumentation Frameworks



Admissibility:

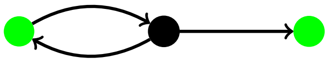
Independence:

$$\forall a, b \in S. (a, b) \notin E(G)$$

Defends itself:

$$N^-(S) \subseteq N^+(S).$$

Argumentation Frameworks



Admissibility:

Independence:

$$\forall a, b \in S.(a, b) \notin E(G)$$

Defends itself:

$$N^{-}(S) \subseteq N^{+}(S).$$

Preferred: Maximal among admissible subsets.

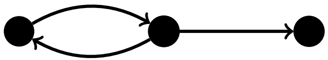
Oriented Argumentation Frameworks

Oriented: No 2-cycles.

Oriented Argumentation Frameworks

Oriented: No 2-cycles.

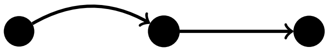
Not Oriented



Oriented Argumentation Frameworks

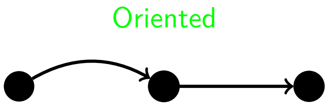
Oriented: No 2-cycles.

Oriented



Oriented Argumentation Frameworks

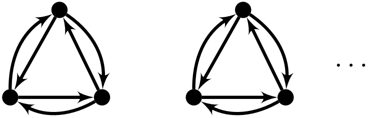
Oriented: No 2-cycles.



Resolution Order: $r(G) := \#$ vertices in a 2-cycle.

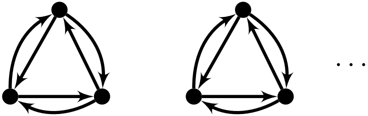
Overview of Results

Existing Results:
At most $3^{n/3}$ preferred extensions, $O^*(3^{n/3})$ enumeration algorithm.



Overview of Results

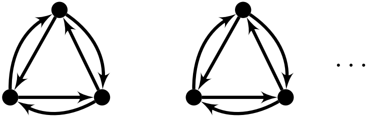
Existing Results:
At most $3^{n/3}$ preferred extensions, $O^*(3^{n/3})$ enumeration algorithm.



New Results:
Recall $r(G) :=$ #vertices in a 2-cycle
Define $r := r(G)/n$.

Overview of Results

Existing Results:
At most $3^{n/3}$ preferred extensions, $O^*(3^{n/3})$ enumeration algorithm.

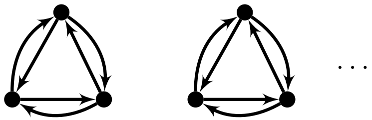


New Results:
Recall $r(G) :=$ #vertices in a 2-cycle
Define $r := r(G)/n$.

- An algorithm based on Monotone Local Search.
  Running time $\approx O^*((1.48^r \cdot 1.29^{1-r})^n)$.

Overview of Results

Existing Results:
At most $3^{n/3}$ preferred extensions, $O^*(3^{n/3})$ enumeration algorithm.

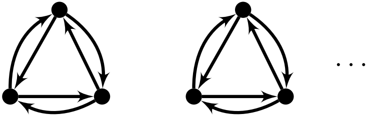


New Results:
Recall $r(G) :=$ #vertices in a 2-cycle
Define $r := r(G)/n$.

- An algorithm based on Monotone Local Search.
  Running time $\approx O^*((1.48^r \cdot 1.29^{1-r})^n)$.
- A direct branching algorithm for oriented graphs, with a translation result to lift it to a general graph.
  Running time $\approx O^*((1.52^r \cdot 1.23^{1-r})^n)$.

Overview of Results

Existing Results:
At most $3^{n/3}$ preferred extensions, $O^*(3^{n/3})$ enumeration algorithm.

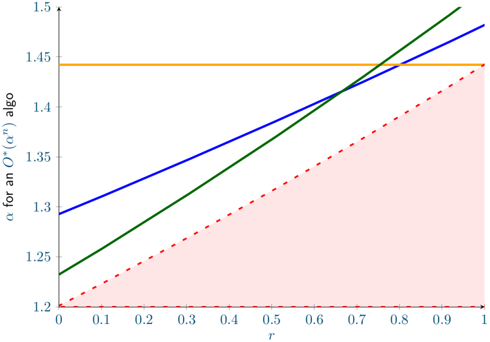


New Results:
Recall $r(G) :=$ #vertices in a 2-cycle
Define $r := r(G)/n$.

- An algorithm based on Monotone Local Search.
  Running time $\approx O^*((1.48^r \cdot 1.29^{1-r})^n)$.
- A direct branching algorithm for oriented graphs, with a translation result to lift it to a general graph.
  Running time $\approx O^*((1.52^r \cdot 1.23^{1-r})^n)$.
- A combinatorial lower bound of

$$\Omega((3^{\frac{r}{3}} 3^{\frac{1-r}{6}})^{|V|}) \approx ((1.44^r \cdot 1.20^{1-r})^n)$$

New Results



- Orange: Existing $O^*(3^{n/3})$ algorithm.
- Blue: MLS algorithm, $\approx O^*((1.48^r \cdot 1.29^{1-r})^n)$.
- Green: Branching Algorithm $\approx O^*((1.52^r \cdot 1.23^{1-r})^n)$.
- Red: Combinatorial lower bound, $\Omega((3^{\frac{r}{3}} 3^{\frac{1-r}{6}})^n)$.

Parameterized Problem Introduction

Maximal Admissible Subset Enumeration (MASE)	
Input:	Graph G, set $S \subseteq V(G)$, integer k
Parameter:	k
Output:	Enumerate all maximal admissible sets $T \subseteq S$ such that $\|S \setminus T\| \leq k$.

Parameterized Problem Introduction

Maximal Admissible Subset Enumeration (MASE)

Input:	Graph G, set $S \subseteq V(G)$, integer k
Parameter:	k
Output:	Enumerate all maximal admissible sets $T \subseteq S$ such that $\lvert S \setminus T \rvert \leq k$.

Theorem

There is an $O^*(2^k)$ algorithm for MASE.
Under SETH, this is optimal.

Parameterized Problem Introduction

Maximal Admissible Subset Enumeration (MASE)

Input:	Graph G, set $S \subseteq V(G)$, integer k
Parameter:	k
Output:	Enumerate all maximal admissible sets $T \subseteq S$ such that $\lvert S \setminus T \rvert \leq k$.

Theorem

There is an $O^*(2^k)$ algorithm for MASE.
Under SETH, this is optimal.

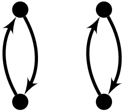
Theorem

There is an $O^*(2^{\frac{k}{2}})$ algorithm for MASE on oriented graphs.
Under SETH, this is optimal.

Why is Orientedness helpful?

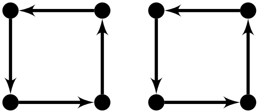
Base Case (max degree ≤ 2)

General Graph: $2^{\frac{n}{2}}$



$\dots$

Oriented Graph: $2^{\frac{n}{4}}$



$\dots$

Why is Orientedness helpful?

Let $v \in G[S]$ have the maximum number of neighbors.

- v has many neighbors: Easy

Why is Orientedness helpful?

Let $v \in G[S]$ have the maximum number of neighbors.

- v has many neighbors: Easy
- v has degree ≤ 2: Easy

Why is Orientedness helpful?

Let $v \in G[S]$ have the maximum number of neighbors.

- v has many neighbors: Easy
- v has degree ≤ 2: Easy
- v has in-degree 0:
  (Undefendable) Let $u \in S$ be a vertex such that there exists a vertex $a \in V(G)$, a attacks u and $a \notin N^+(S)$. Then there is no admissible subset of S that contains u so we can safely set $S \leftarrow S \setminus \{u\}$.

Why is Orientedness helpful?

Let $v \in G[S]$ have the maximum number of neighbors.

- v has many neighbors: Easy
- v has degree ≤ 2: Easy
- v has in-degree 0:
  (Undefendable) Let $u \in S$ be a vertex such that there exists a vertex $a \in V(G)$, a attacks u and $a \notin N^+(S)$. Then there is no admissible subset of S that contains u so we can safely set $S \leftarrow S \setminus \{u\}$.

Proof.

If u is in our admissible subset then a must be attacked by our admissible subset.
But no subset of S attacks a as S does not attack a itself. □

Why is Orientedness helpful?

Let $v \in G[S]$ have the maximum number of neighbors.

- v has many neighbors: Easy
- v has degree ≤ 2: Easy
- v has in-degree 0:
  (Undefendable) Let $u \in S$ be a vertex such that there exists a vertex $a \in V(G)$, a attacks u and $a \notin N^+(S)$. Then there is no admissible subset of S that contains u so we can safely set $S \leftarrow S \setminus \{u\}$.

Proof.

If u is in our admissible subset then a must be attacked by our admissible subset.
But no subset of S attacks a as S does not attack a itself. $\square$

- v has in-degree 1: Branch on in-neighbor and use **(Undefendable)**.

$O^*(2^{\frac{k}{2}})$ MASE algorithm

Maximal **Oriented** Admissible Subset Enumeration (MASE)

Input:	**Oriented** Graph G, set $S \subseteq V(G)$, integer k
Parameter:	k
Output:	Enumerate all maximal admissible sets $T \subseteq S$ such that $\|S \setminus T\| \leq k$.

Direct branching algorithm. First keep applying (Undefendable).
Cases:

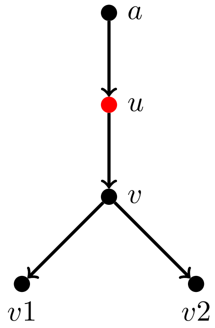
Case	Requirement to apply	Running Time
Base	S is conflict-free.	Solves in $O^*(1)$ time, returns ≤ 1 set.
1	$G[S]$ has maximum total degree ≤ 2.	Vector $(2,2), \alpha = \sqrt{2}$.
2	$G[S]$ has maximum total degree ≥ 4.	Vector $(4,1), \alpha \approx \sqrt{1.91}$.
3	$G[S]$ contains a vertex with total degree 3.	Vector $(2,3), \alpha \approx \sqrt{1.76}$.

Case 3

Case 3: There exists $v \in G[S]$ with total degree 3.

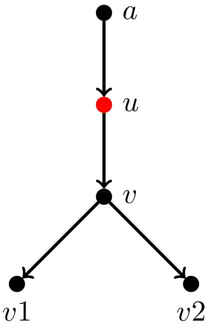
Case 3

Case 3: There exists $v \in G[S]$ with total degree 3.
Branch on u:



Case 3

Case 3: There exists $v \in G[S]$ with total degree 3.
Branch on u:



Gives a $(3, 2)$ branching.

Overall Branching Algorithm

Case	Requirement to apply
1	$\exists v \in G[S]$ with total degree ≥ 7.
2	$\exists v \in G[S]$ with degree $(1, -)$.
3	$\exists v \in G[S]$ with in-degree $\neq$ out-degree.
4	$G[S]$ has a weakly connected component where every vertex has degree $(2, 2)$.
5	$G[S]$ has a weakly connected component where every vertex has degree $(3, 3)$.
6	There is a weakly connected component in $G[S]$ where every vertex has in-degree $=$ out-degree.

Extending to Resolution Order

For FPT: Just set $\mu(I) = \frac{k}{2} + \frac{r(G[S])}{4}$.
Then there is a $O^*(2^{\mu(I)})$ algorithm for MASE on general graphs.

Extending to Resolution Order

Theorem

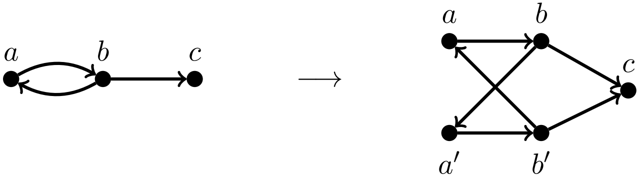
There is a linear time algorithm that transforms any AF G into an oriented AF G' with $|V(G')| = |V(G)| + r(G) + 3$ such that there is a bijection between the preferred extensions of G and the preferred extensions of G' that can be applied in linear time.

Extending to Resolution Order

Theorem

There is a linear time algorithm that transforms any AF G into an oriented AF G' with $|V(G')| = |V(G)| + r(G) + 3$ such that there is a bijection between the preferred extensions of G and the preferred extensions of G' that can be applied in linear time.

Basic idea:

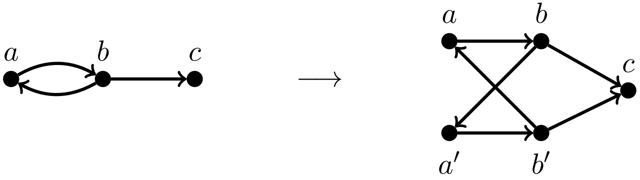


Extending to Resolution Order

Theorem

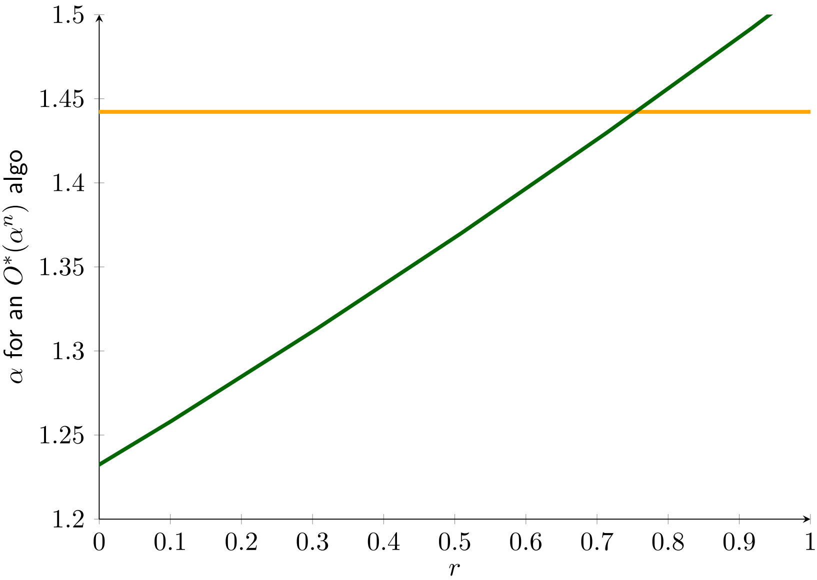
There is a linear time algorithm that transforms any AF G into an oriented AF G' with $|V(G')| = |V(G)| + r(G) + 3$ such that there is a bijection between the preferred extensions of G and the preferred extensions of G' that can be applied in linear time.

Basic idea:



So our $O^*(1.23^n)$ algorithm on oriented graphs becomes a $O^*((1.23^{2r} \cdot 1.23^{1-r})^n)$ algorithm on general graphs.

Results So Far



Monotone Local Search Idea

Idea: Suppose we are enumerating extensions of size exactly p. Smartly pick a value $q \geq p$ and randomly sample subsets of size q then apply our parameterized algorithm to these subsets.

Monotone Local Search Idea

Idea: Suppose we are enumerating extensions of size exactly p. Smartly pick a value $q \geq p$ and randomly sample subsets of size q then apply our parameterized algorithm to these subsets.

Problem: Our MASE running time is sensitive to number of vertices in a 2-cycle.

Monotone Local Search Idea

Idea: Suppose we are enumerating extensions of size exactly p. Smartly pick a value $q \geq p$ and randomly sample subsets of size q then apply our parameterized algorithm to these subsets.

Problem: Our MASE running time is sensitive to number of vertices in a 2-cycle.

Solution: Randomly sample separately among vertices in a 2-cycle and vertices in no 2-cycles.

Running Time Analysis

Turns out we can analyze the two parts separately.

Running Time Analysis

Turns out we can analyze the two parts separately.

- Let $B :=$ set of vertices from S that are in at least one 2-cycle.
- Let $D := V(G) \setminus B$.

Running Time Analysis

Turns out we can analyze the two parts separately.

- Let $B :=$ set of vertices from S that are in at least one 2-cycle.
- Let $D := V(G) \setminus B$.

Overall Running Time

$= \text{O(MLS on } B \text{ with an } O^*(2^{\frac{k}{2}+\frac{|S|}{4}}) \text{ subroutine)}$

$\times \text{ O(MLS on } D \text{ with an } O^*(2^{\frac{k}{2}}) \text{ subroutine)}$

(where S is the size of the candidate set)

MLS Result

O(MLS on B with an $O^*(2^{\frac{k}{2}+\frac{|S|}{4}})$ subroutine)
$\times$ O(MLS on D with an $O^*(2^{\frac{k}{2}})$ subroutine)

MLS Result

O(MLS on B with an $O^*(2^{\frac{k}{2}+\frac{|S|}{4}})$ subroutine)
$\times$ O(MLS on D with an $O^*(2^{\frac{k}{2}})$ subroutine)

Theorem (Gaspers, Lee 2017)

MLS with an $O^*(b^{|S|}c^k)$ algorithm has running time

$$O^*((1+b-\frac{1}{c})^{n+o(n)})$$

MLS Result

O(MLS on B with an $O^*(2^{\frac{k}{2}+\frac{|S|}{4}})$ subroutine)
$\times$ O(MLS on D with an $O^*(2^{\frac{k}{2}})$ subroutine)

Theorem (Gaspers, Lee 2017)

MLS with an $O^*(b^{|S|}c^k)$ algorithm has running time

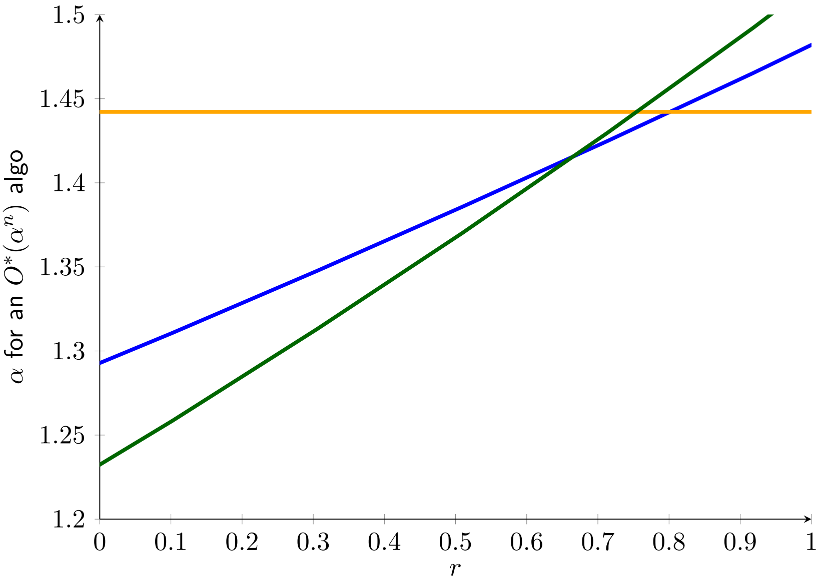
$$O^*((1+b-\frac{1}{c})^{n+o(n)})$$

Theorem

There is an algorithm for enumeration of preferred extensions with running time

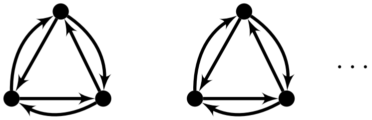
$$O^*\left(\left(2-\frac{1}{\sqrt{2}}\right)^{|D|+o(n)}\right)\cdot O^*\left(\left(1+2^{\frac{1}{4}}-\frac{1}{\sqrt{2}}\right)^{|B|+o(n)}\right)$$
$$\approx O^*\left((1.2929)^{|D|+o(n)}\right)\cdot O^*\left((1.4822)^{|B|+o(n)}\right)$$

Results So Far



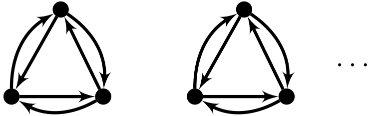
Combinatorial Lower Bounds

On general graphs

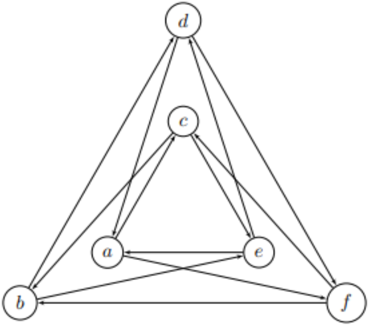


Combinatorial Lower Bounds

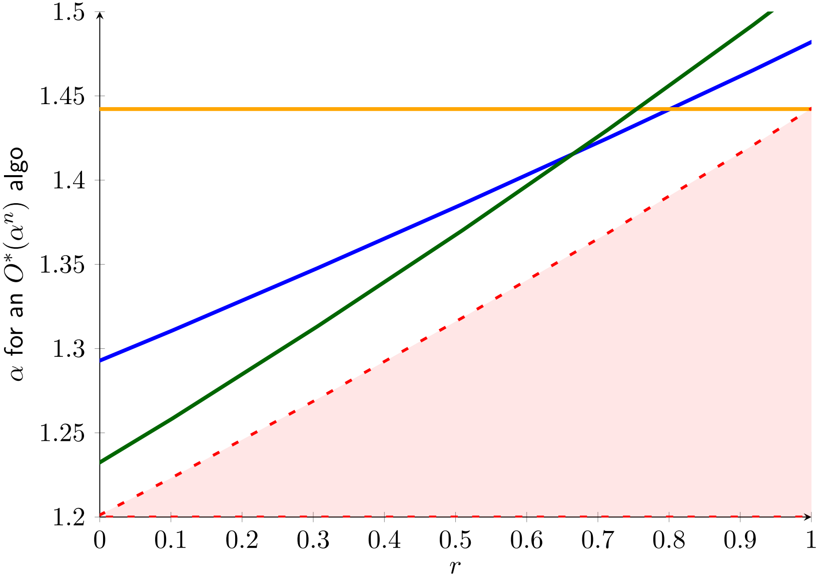
On general graphs



On oriented graphs instead repeat these



Full Results



Summary

- We introduce a parameter $r(G)$ that has an intuitive meaning within the context.

Summary

- We introduce a parameter $r(G)$ that has an intuitive meaning within the context.
- Furthermore, it leads to better results because vertices in oriented graphs with low degree are easily understood.

Summary

- We introduce a parameter $r(G)$ that has an intuitive meaning within the context.
- Furthermore, it leads to better results because vertices in oriented graphs with low degree are easily understood.
- Through some tricks, we can lift results from oriented graphs to results parameterized by $r(G)$.

Questions

- Can we close the gap? It suffices to close it for oriented graphs.

Questions

- Can we close the gap? It suffices to close it for oriented graphs.
  - Lower bound: $3^{n/6} > 1.2009^n$.

Questions

- Can we close the gap? It suffices to close it for oriented graphs.
  - Lower bound: $3^{n/6} > 1.2009^n$.
  - Upper bound: $O(1.2322^n)$.

Questions

- Can we close the gap? It suffices to close it for oriented graphs.
  - Lower bound: $3^{n/6} > 1.2009^n$.
  - Upper bound: $O(1.2322^n)$.
- Look at different semantics?

Questions

- Can we close the gap? It suffices to close it for oriented graphs.
  - Lower bound: $3^{n/6} > 1.2009^n$.
  - Upper bound: $O(1.2322^n)$.
- Look at different semantics?
- Look into other natural NP-hard problems where we can parameterize by orientedness?

Questions

- Can we close the gap? It suffices to close it for oriented graphs.
  - Lower bound: $3^{n/6} > 1.2009^n$.
  - Upper bound: $O(1.2322^n)$.
- Look at different semantics?
- Look into other natural NP-hard problems where we can parameterize by orientedness?

Questions

- Can we close the gap? It suffices to close it for oriented graphs.
  - Lower bound: $3^{n/6} > 1.2009^n$.
  - Upper bound: $O(1.2322^n)$.
- Look at different semantics?
- Look into other natural NP-hard problems where we can parameterize by orientedness?

Thank you!