# Constructive enumeration and uniform random sampling of DAGs

Antoine Genitrini[1] > **Martin Pépin**[1] Alfredo Viola[2]

Work submitted to the LAGOS conference

December 10, 2020

[1]LIP6 — Sorbonne Université — Paris

[2]Universidad de la República — Montevideo

# Outline

Background

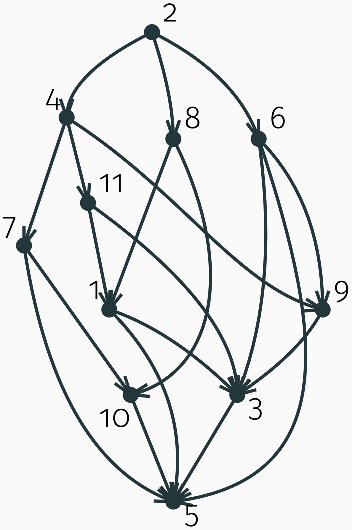
Directed Ordered Acyclic Graphs

Extensions

Conclusion and future work

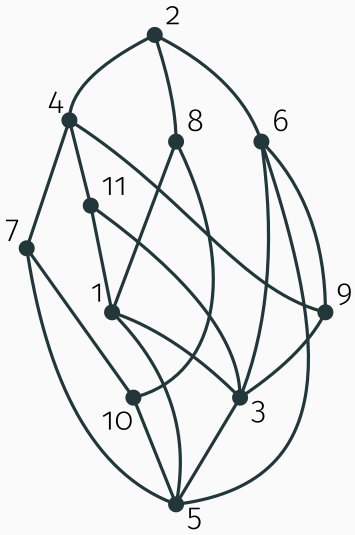
# Directed Acyclic Graphs

- A finite set of vertices $V$ e.g. $\{1, 2, \ldots, n\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles: $v_1 \to v_2 \to \cdots \to v_n = v_1$.



# Directed Acyclic Graphs

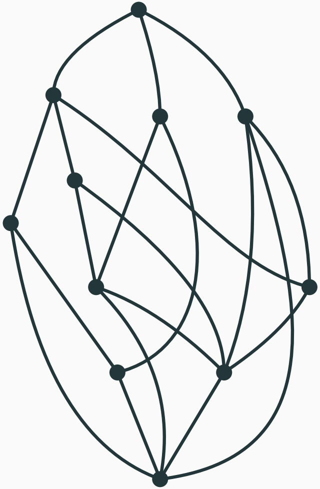
- A finite set of vertices $V$ e.g. $\{1, 2, \ldots, n\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles: $v_1 \to v_2 \to \cdots \to v_n = v_1$.



# Directed Acyclic Graphs

- A finite set of vertices $V$ e.g. $\{1, 2, \ldots, n\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles: $v_1 \to v_2 \to \cdots \to v_n = v_1$.

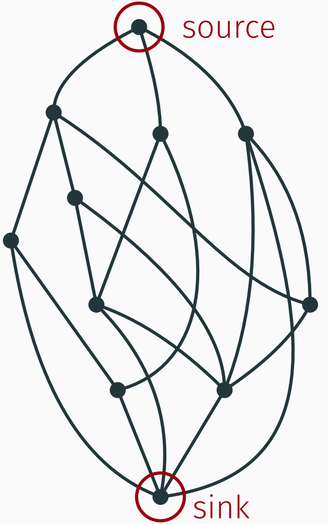
- If considered up to relabelling: **unlabelled DAGs**



# Directed Acyclic Graphs

- A finite set of vertices $V$ e.g. $\{1, 2, \ldots, n\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles: $v_1 \to v_2 \to \cdots \to v_n = v_1$.

- If considered up to relabelling:
  **unlabelled DAGs**



# State of the art

**Labelled** DAGs:

- Counting by number of vertices: [Rob73]

# State of the art

**Labelled** DAGs:

- Counting by number of vertices: [Rob73]
- Counting by number of edges: [Ges96]

# State of the art

**Labelled** DAGs:

- Counting by number of vertices: [Rob73]
- Counting by number of edges: [Ges96]
- Uniform sampling: [MDB01], [KM15]

# State of the art

**Labelled** DAGs:

- Counting by number of vertices: [Rob73]
- Counting by number of edges: [Ges96]
- Uniform sampling: [MDB01], [KM15]

**Unlabelled** DAGs:

- Counting by vertices and sources: [Rob77]

# State of the art

**Labelled** DAGs:

- Counting by number of vertices: [Rob73]
- Counting by number of edges: [Ges96] •
- Uniform sampling: [MDB01], [KM15]

**Unlabelled** DAGs:

- Counting by vertices and sources: [Rob77] •

**Problems:**

- Inclusion-exclusion

# State of the art

**Labelled** DAGs:

- Counting by number of vertices: [Rob73]
- Counting by number of edges: [Ges96] ●
- Uniform sampling: [MDB01], [KM15] ●

**Unlabelled** DAGs:

- Counting by vertices and sources: [Rob77] ●

**Problems:**

- Inclusion-exclusion
- No or little control over the number of edges

# Still missing

- Finer control over the number of edges?
- Sampling of unlabelled structures?

# Outline

Background

Directed Ordered Acyclic Graphs

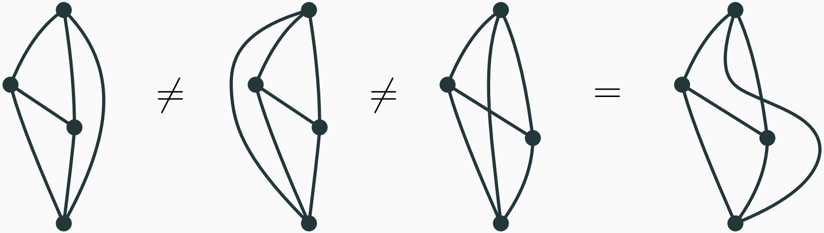
Extensions

Conclusion and future work

# A new kind of DAG

**Directed Ordered Acyclic Graphs (DOAGs)**

DOAG = Unlabelled DAG
+ a total order on the **outgoing** edges of each vertex
+ only one sink and one source



# Motivation

- Real-life implementations of DAGs have an **ordering**; →

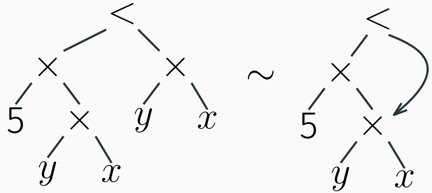
```c
struct vertex {
  int           out_degree;
  struct vertex *out_edges;
};
```

# Motivation

- Real-life implementations of DAGs have an **ordering**; →

```c
struct vertex {
  int           out_degree;
  struct vertex *out_edges;
};
```

- The memory layout of trees with hash-consing have an **ordering**; →

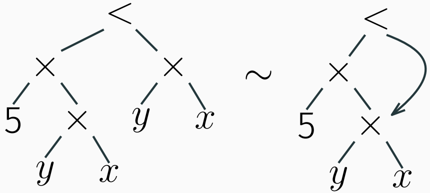


# Motivation

> Real-life implementations of DAGs have an **ordering**; →

```c
struct vertex {
  int            out_degree;
  struct vertex *out_edges;
};
```

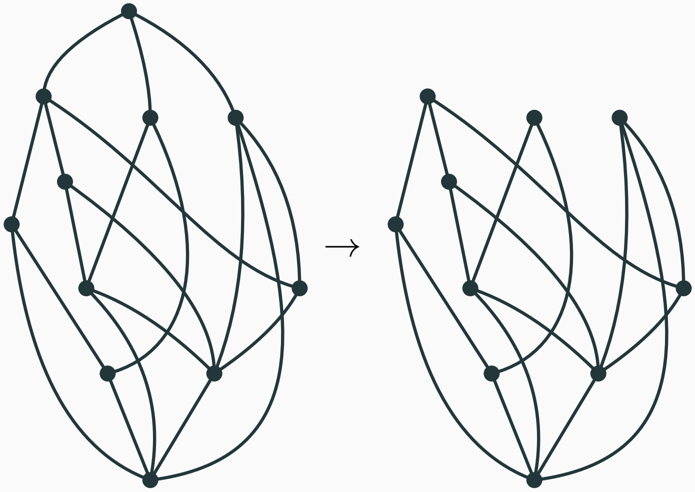
> The memory layout of trees with hash-consing have an **ordering**; →



> Models **unlabelled** objects.

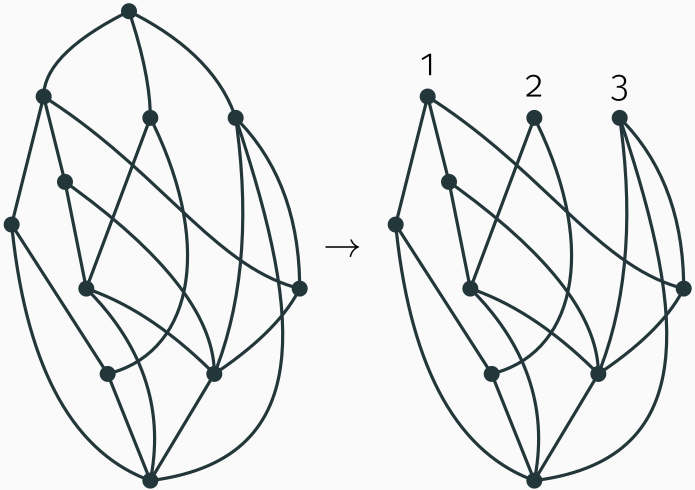
# Recursive decomposition: multi-source DOAGs

Idea: remove the source and see what is left.



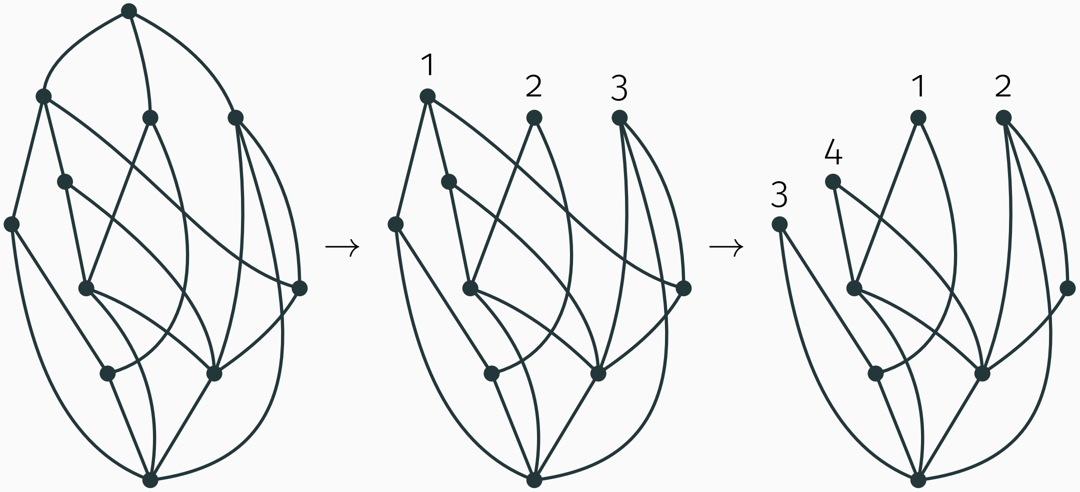
# Recursive decomposition: multi-source DOAGs

Idea: remove the source and see what is left.

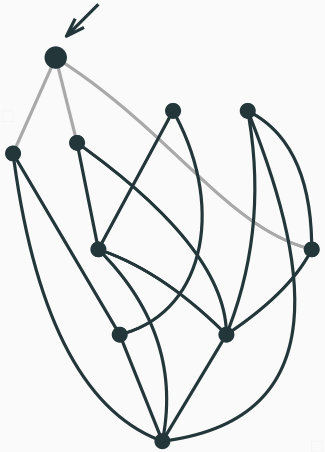


# Recursive decomposition: multi-source DOAGs

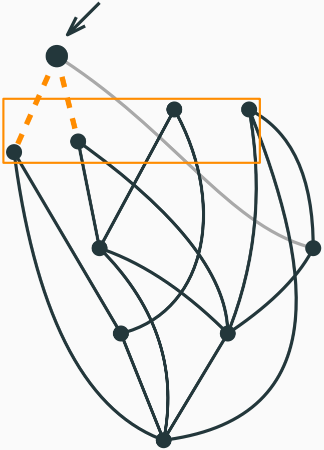
Idea: remove the source and see what is left.



# Recursive decomposition: multi-source DOAGs

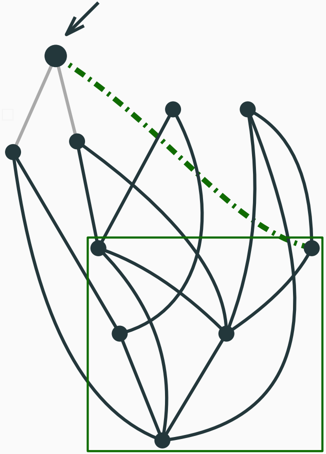


# Recursive decomposition: multi-source DOAGs



$q$ edges to sources;

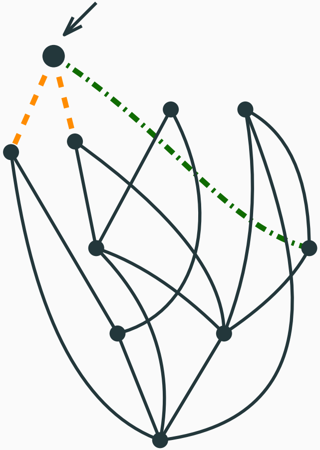
# Recursive decomposition: multi-source DOAGs



$q$ edges to sources;

$s$ edges to internal vertices;

# Recursive decomposition: multi-source DOAGs

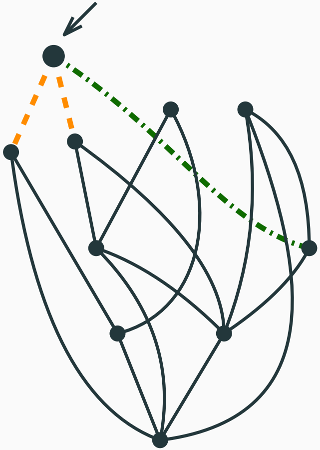


$q$ edges to sources;

$s$ edges to internal vertices;

$\binom{s+q}{q} s!$ ways to arrange the two sets of edges;

# Recursive decomposition: multi-source DOAGs



$q$ edges to sources;

$s$ edges to internal vertices;

$\binom{s+q}{q} s!$ ways to arrange the two sets of edges;

$$D_{n,m,k} = \#\text{DOAGs with } n \text{ vertices, } m \text{ edges, } k \text{ sources}$$

$$= \sum_{s+q>0} D_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s!$$

# Complexity of the counting

$$\begin{aligned} D_{1,m,k} &= \mathbb{1}_{\{m=0 \wedge k=1\}} && \\ D_{n,m,k} &= 0 && \text{when } k \leq 0 \\ D_{n,m,k} &= \sum_{s+q>0} D_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s! && \text{when } n > 1 \end{aligned}$$

# Complexity of the counting

$$D_{1,m,k} = \mathbb{1}_{\{m=0 \wedge k=1\}}$$

$$D_{n,m,k} = 0 \qquad \text{when } k \leq 0$$

$$D_{n,m,k} = \sum_{s+q>0} D_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s! \qquad \text{when } n > 1$$

**Complexity**

Computing $D_{n,m,k}$ for all $n, k \leq N$ and $m \leq M$ takes $O(N^4 M)$ arithmetic operations.

# Complexity of the counting

$$D_{1,m,k} = \mathbb{1}_{\{m=0 \wedge k=1\}}$$

$$D_{n,m,k} = 0 \qquad \text{when } k \leq 0$$

$$D_{n,m,k} = \sum_{s+q>0} D_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s! \qquad \text{when } n > 1$$

**Complexity**

Computing $D_{n,m,k}$ for all $n, k \leq N$ and $m \leq M$ takes $O(N^4 M)$ arithmetic operations.

In practice we reach $M = 400$, $N = M + 1$.

# Random sampling

Do the same, but backwards.

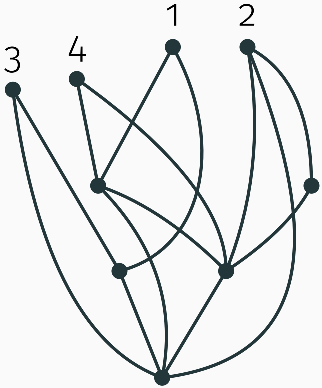
# Random sampling

Do the same, but backwards.

1. Select $(s, q)$ with probability $\frac{D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!}{D_{n,m,k}}$;

# Random sampling

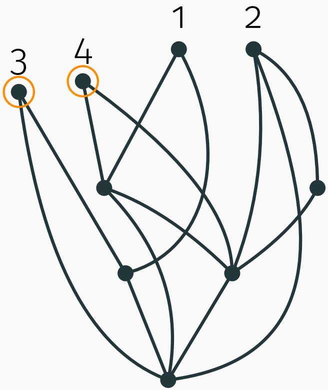
Do the same, but backwards.




1. Select $(s, q)$ with probability $\frac{D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!}{D_{n,m,k}}$;
2. Sample a $\mathrm{DOAG}_{n-1,m-s-q,k-1+q}$ recursively;

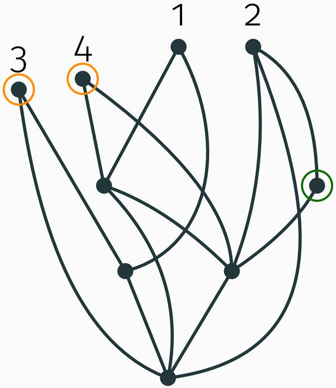
# Random sampling

Do the same, but backwards.




1. Select $(s, q)$ with probability $\frac{D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!}{D_{n,m,k}}$;
2. Sample a $\text{DOAG}_{n-1,m-s-q,k-1+q}$ recursively;
3. We already know the $q$ largest sources;

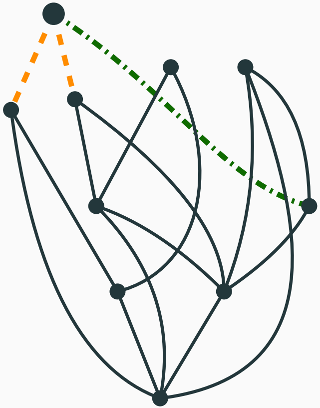
Do the same, but backwards.




1. Select $(s, q)$ with probability $\frac{D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!}{D_{n,m,k}}$;
2. Sample a $\text{DOAG}_{n-1,m-s-q,k-1+q}$ recursively;
3. We already know the $q$ largest sources;
4. Choose $s$ internal vertices;

# Random sampling

Do the same, but backwards.



1. Select $(s, q)$ with probability $\frac{D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!}{D_{n,m,k}}$;
2. Sample a $\text{DOAG}_{n-1,m-s-q,k-1+q}$ recursively;
3. We already know the $q$ largest sources;
4. Choose $s$ internal vertices;
5. Connect them to the new sources.

# Random sampling

How to select $s$ and $q$?

# Random sampling

How to select $s$ and $q$?

1. Pick an $x \sim \textsc{Unif}(\llbracket 0; D_{n,m,k} - 1 \rrbracket)$;

# Random sampling

How to select $s$ and $q$?

1. Pick an $x \sim \textsc{Unif}(\llbracket 0; D_{n,m,k} - 1 \rrbracket)$;
2. compute the partial sum of the terms $D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!$;

# Random sampling

How to select $s$ and $q$?

1. Pick an $x \sim \textsc{Unif}(\llbracket 0; D_{n,m,k} - 1 \rrbracket)$;
2. compute the partial sum of the terms $D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!$;
3. stop as soon as the sum becomes $> x$;

# Random sampling

How to select $s$ and $q$?

1. Pick an $x \sim \textsc{Unif}(\llbracket 0; D_{n,m,k} - 1 \rrbracket)$;
2. compute the partial sum of the terms $D_{n-1,m-s-q,k-1+q}\binom{s+q}{s}\binom{n-k-q}{s}s!$;
3. stop as soon as the sum becomes $> x$;
4. (bonus) sum in the lexicographic order for $(s+q, s)$.

# Complexity of the sampling algorithm

- Selecting s and $q$: $O((s+q)^2)$ arithmetic operations;
- the rest is cheap.

# Complexity of the sampling algorithm

- Selecting $s$ and $q$: $O((s+q)^2)$ arithmetic operations;
- the rest is cheap.

**Complexity**

Sampling a DOAG uniformly at random costs $O(\sum_v d_v^2)$ arithmetic operations where $v$ ranges over the vertices of the output and $d_v$ is the out-degree of $v$.

# Complexity of the sampling algorithm

- Selecting $s$ and $q$: $O((s+q)^2)$ arithmetic operations;
- the rest is cheap.

**Complexity**

Sampling a DOAG uniformly at random costs $O(\sum_v d_v^2)$ arithmetic operations where $v$ ranges over the vertices of the output and $d_v$ is the out-degree of $v$.

In practice it takes a few milliseconds.

# Outline

- Background
- Directed Ordered Acyclic Graphs
- Extensions
- Conclusion and future work

# Bounded degree sampling

What if we want DOAGs with maximum out-degree $d$?

# Bounded degree sampling

What if we want DOAGs with maximum out-degree $d$?

$$D_{n,m,k} = \sum_{0<s+q} D_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s!$$

# Bounded degree sampling

What if we want DOAGs with maximum out-degree $d$?

$$D^{(d)}_{n,m,k} = \sum_{0<s+q\leq d} D^{(d)}_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s!$$

# Bounded degree sampling

What if we want DOAGs with maximum out-degree $d$?

$$D^{(d)}_{n,m,k} = \sum_{0<s+q\leq d} D^{(d)}_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s!$$

- Counting: $O(N^2 d^4)$ arithmetic operations.

# Bounded degree sampling

What if we want DOAGs with maximum out-degree $d$?

$$D^{(d)}_{n,m,k} = \sum_{0<s+q\leq d} D^{(d)}_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s!$$

- Counting: $O(N^2 d^4)$ arithmetic operations.
- Sampling $O(Nd^2)$ arithmetic operations.

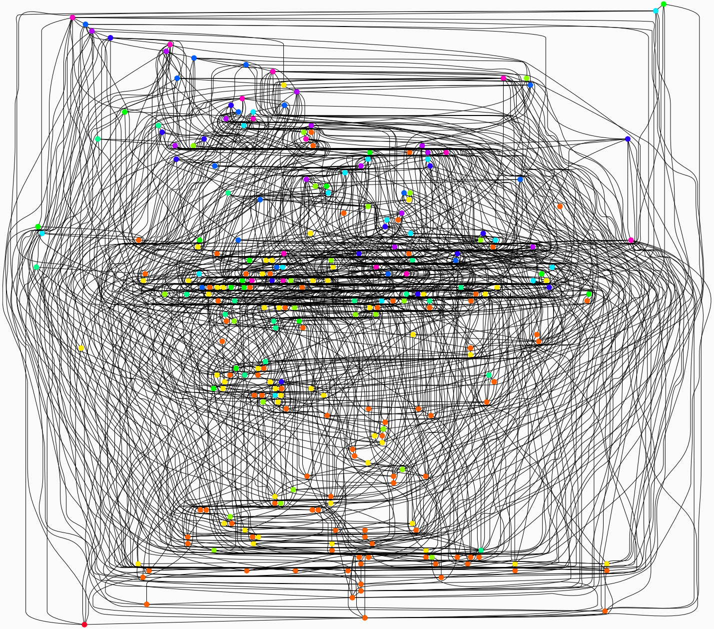
# Bounded degree sampling

What if we want DOAGs with maximum out-degree $d$?

$$D^{(d)}_{n,m,k} = \sum_{0<s+q\leq d} D^{(d)}_{n-1,m-s-q,k-1+q} \binom{s+q}{s} \binom{n-k-q}{s} s!$$

- Counting: $O(N^2 d^4)$ arithmetic operations.
- Sampling $O(N d^2)$ arithmetic operations.
- In practice we reached $m = 1500$ with $d = 2$ and $m = 1000$ with $d = 10$.

# Your next favourite wallpaper



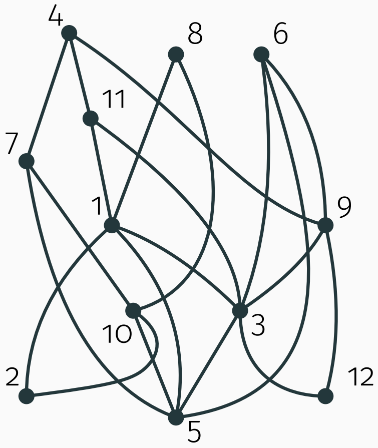
Uniform DOAG with $m = 1000$ edges and with out-degree bounded by $d = 10$.

# Back to labelled DAGs

The classical way to count is by a *layer-by-layer* approach.

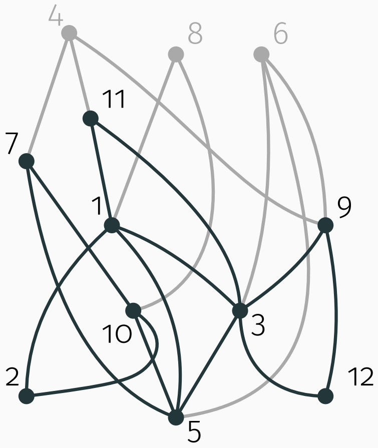
# Back to labelled DAGs

The classical way to count is by a *layer-by-layer* approach.



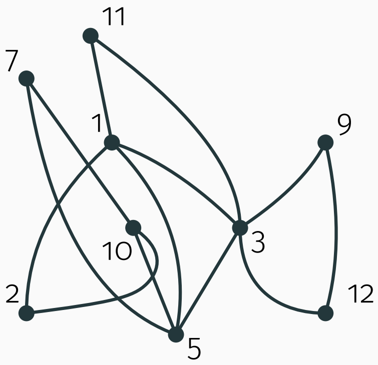
# Back to labelled DAGs

The classical way to count is by a *layer-by-layer* approach.



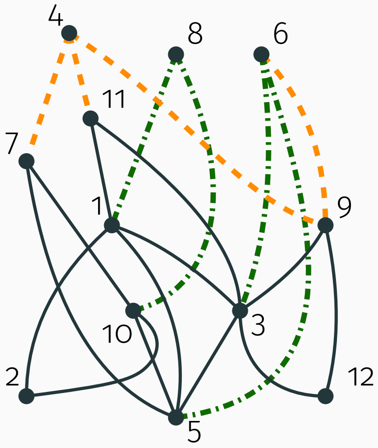
# Back to labelled DAGs

The classical way to count is by a *layer-by-layer* approach.



# Back to labelled DAGs

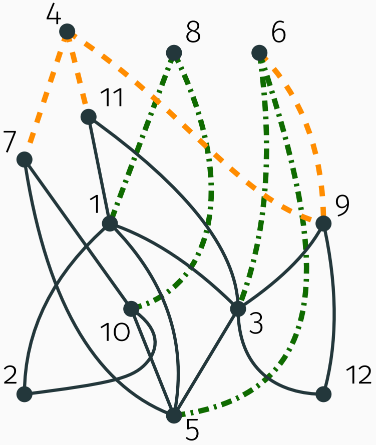
The classical way to count is by a *layer-by-layer* approach.




$A_{n,k}$ = #DAGs with $n$ vertices, $k$ sources

# Back to labelled DAGs

The classical way to count is by a *layer-by-layer* approach.

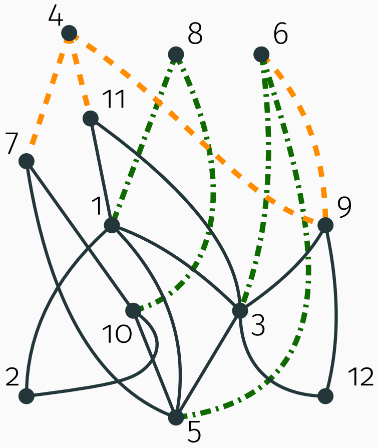



$A_{n,k} = \#\text{DAGs with } n \text{ vertices, } k \text{ sources}$

$$A_{n,k} = \binom{n}{k} \sum_{j>0} A_{n-k,j} \cdot (2^k - 1)^j \cdot 2^{k(n-k-j)}$$

# Back to labelled DAGs

The classical way to count is by a *layer-by-layer* approach.



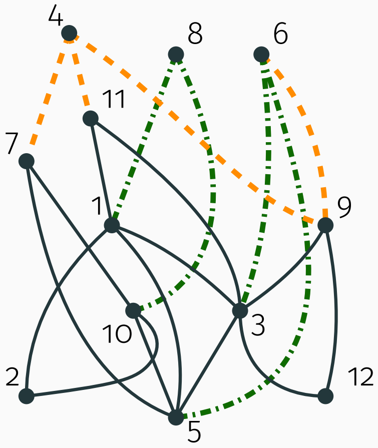

$$A_{n,k} = \#\text{DAGs with } n \text{ vertices, } k \text{ sources}$$

$$A_{n,k} = \binom{n}{k} \sum_{j>0} A_{n-k,j} \cdot (2^k - 1)^j \cdot 2^{k(n-k-j)}$$

- How to count by number of edges?
- How to enforce connectivity (e.g. with one sink and one source)?

# Back to labelled DAGs

The classical way to count is by a *layer-by-layer* approach.




$A_{n,k}$ = #DAGs with $n$ vertices, $k$ sources

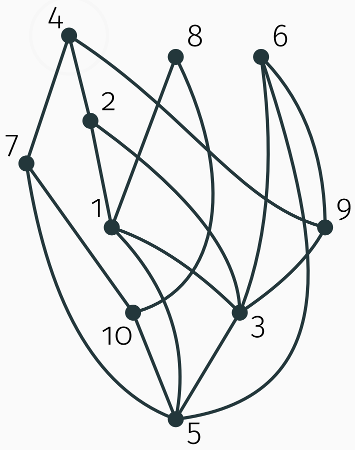
$$A_{n,k} = \binom{n}{k} \sum_{j>0} A_{n-k,j} \cdot (2^k - 1)^j \cdot 2^{k(n-k-j)}$$

- How to count by number of edges?
- How to enforce connectivity (e.g. with one sink and one source)?

→ Use our approach!

# Vertex-by-vertex decomposition of labelled DAGs

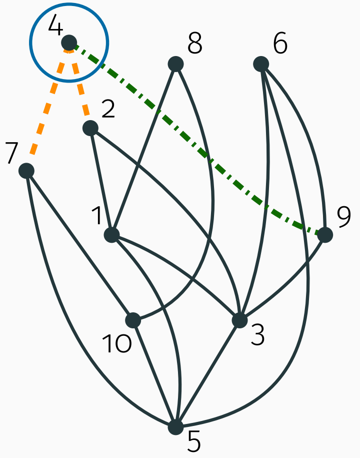
Idea: mark one source, and remove it.




$A_{n,m,k} = \#\text{DAGs (one sink, } k \text{ sources)}$

# Vertex-by-vertex decomposition of labelled DAGs

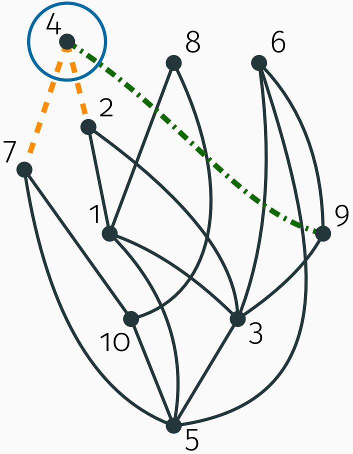
Idea: mark one source, and remove it.




$A_{n,m,k} = \#\text{DAGs (one sink, } k \text{ sources)}$

# Vertex-by-vertex decomposition of labelled DAGs

Idea: mark one source, and remove it.




$A_{n,m,k}$ = #DAGs (one sink, $k$ sources)

$$k \cdot A_{n,m,k} =$$

$$n \cdot \sum_{s+q>0} A_{n-1,m-s-q,k-1+q} \binom{k-1+q}{q} \binom{n-q-k}{s}$$

# Outline

Background

Directed Ordered Acyclic Graphs

Extensions

Conclusion and future work

# Conclusion

Initial questions:

- Finer control over the number of edges?
- Sampling of unlabelled structures?

# Conclusion

Initial questions:

- Finer control over the number of edges? ✓
- Sampling of unlabelled structures?

# Conclusion

Initial questions:

- Finer control over the number of edges? ✓
- Sampling of unlabelled structures? ➡ We made one step forward

# Conclusion

Initial questions:

- Finer control over the number of edges? ✓
- Sampling of unlabelled structures? ➡ We made one step forward

We presented:

- a new model of DAGs: DOAGs;
- a new way of counting.

# Future work

- Can we get rid of the one-sink-one-source constraint while retaining weak connectivity?
- Is there a symbolic method operator hidden behind the vertex-by-vertex decomposition?
- Asymptotics?
- Can we get closer to sampling regular unlabelled DAGs?

Thank you for your attention!

I. M. Gessel.
**Counting acyclic digraphs by sources and sinks.**
*Discrete Mathematics*, 160(1):253 – 258, 1996.

J. Kuipers and G. Moffa.
**Uniform random generation of large acyclic digraphs.**
*Stat. and Computing*, 25(2):227–242, 2015.

G. Melançon, I. Dutour, and M. Bousquet-Mélou.
**Random generation of directed acyclic graphs.**
*Electron. Notes Discret. Math.*, 10:202–207, 2001.

# References ii

R.W. Robinson.
**Counting labeled acyclic digraphs.**
*New Directions in the Theory of Graphs*, pages 239–273, 1973.

R. W. Robinson.
**Counting unlabeled acyclic digraphs.**
In *Combinatorial Mathematics V*, Lecture Notes in Mathematics, pages 28–43. Springer, 1977.