

DIRECTED ORDERED ACYCLIC GRAPHS

ASYMPTOTIC ANALYSIS AND EFFICIENT RANDOM SAMPLING

Martin Pépin

joint work with Antoine Genitrini & Alfredo Viola

January 11, 2023

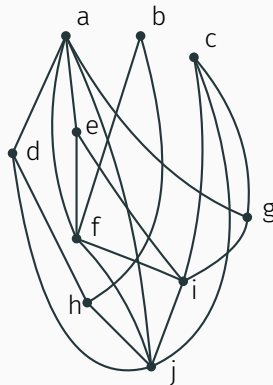
2nd LambdaComb meeting



Directed Acyclic Graphs

Directed Acyclic Graph (DAG)

- A finite set of vertices V e.g. $\{a, b, c, \dots, j\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles.

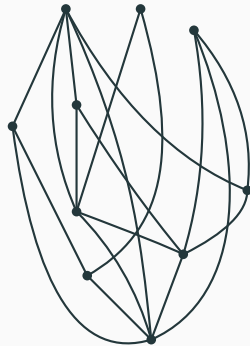


Directed Acyclic Graphs

Directed Acyclic Graph (DAG)

- A finite set of vertices V e.g. $\{a, b, c, \dots, j\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles.

Without labels: **Unlabelled DAGs** 🚫

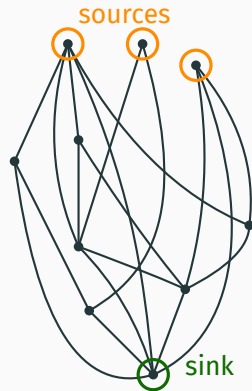


Directed Acyclic Graphs

Directed Acyclic Graph (DAG)

- A finite set of vertices V e.g. $\{a, b, c, \dots, j\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles.

Without labels: Unlabelled DAGs 🚲



Why DAGs?

Omnipresent data structure:

- Encoding partial orders in scheduling problems;
- Git histories;
- Bayesian networks in probabilities;
- genealogy trees (those are not trees!);
- class inheritance in OOP;
- compacted trees (or XML documents);
- hash-consing...

Why DAGs?

Omnipresent data structure:

- **Encoding partial orders in scheduling problems;**
- Git histories;
- Bayesian networks in probabilities;
- genealogy trees (those are not trees!);
- class inheritance in OOP;
- compacted trees (or XML documents);
- hash-consing...

Why DAGs?

Omnipresent data structure:

- Encoding partial orders in scheduling problems;
- Git histories;
- Bayesian networks in probabilities;
- genealogy trees (those are not trees!);
- class inheritance in OOP;
- **compacted trees (or XML documents);**
- **hash-consing...**

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]
- Counting by vertices and edges: [Ges96]

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]
- Counting by vertices and edges: [Ges96]
- Uniform sampling: [MDB01], [KM15]

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]
- Counting by vertices and edges: [Ges96]
- Uniform sampling: [MDB01], [KM15]

Unlabelled DAGs:

- Counting by vertices: [Rob70; Rob77]

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]
- Counting by vertices and edges: [Ges96]
- Uniform sampling: [MDB01], [KM15]

Unlabelled DAGs:

- Counting by vertices: [Rob70; Rob77]

Compacted trees:

- Counting in the binary case: [GGKW20; EFW21]

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]
- Counting by vertices and edges: [Ges96] ●
- Uniform sampling: [MDB01], [KM15]

Unlabelled DAGs:

- Counting by vertices: [Rob70; Rob77] ●

Compacted trees:

- Counting in the binary case: [GGKW20; EFW21]

Problems:

- Inclusion-exclusion

State of the art

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]
- Counting by vertices and edges: [Ges96] ●
- Uniform sampling: [MDB01], [KM15] ●

Unlabelled DAGs:

- Counting by vertices: [Rob70; Rob77] ●

Compacted trees:

- Counting in the binary case: [GGKW20; EFW21]

Problems:

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

State of the art

Labelled DAGs:

- Counting by vertices: [Rob70; Rob73; Sta73]
- Counting by vertices and edges: [Ges96] ●
- Uniform sampling: [MDB01], [KM15] ●

Unlabelled DAGs:

- Counting by vertices: [Rob70; Rob77] ●

Compacted trees:

- Counting in the binary case: [GGKW20; EFW21] ●

Problems:

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

- Finer control over the number of edges?
- What can we say about compacted structures?

Outline of the presentation

Background

Directed ordered acyclic graphs

↳ *definition and recursive decomposition*

Asymptotic analysis

↳ *matrix encoding*

↳ *asymptotic result*

↳ *faster sampler*

Bonus: labelled DAGs

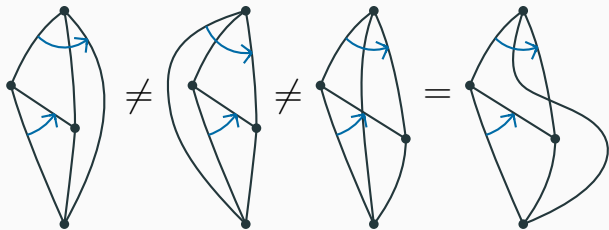
↳ *another way of counting*

A new kind of DAG

Directed Ordered Acyclic Graphs

DOAG = Unlabelled DAG

- + a total order on the **outgoing** edges of each vertex
- + a total order on the sources
- + only one sink

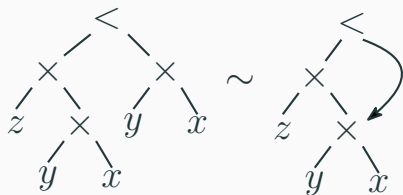
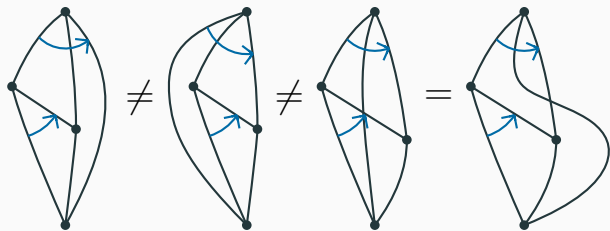


A new kind of DAG

Directed Ordered Acyclic Graphs

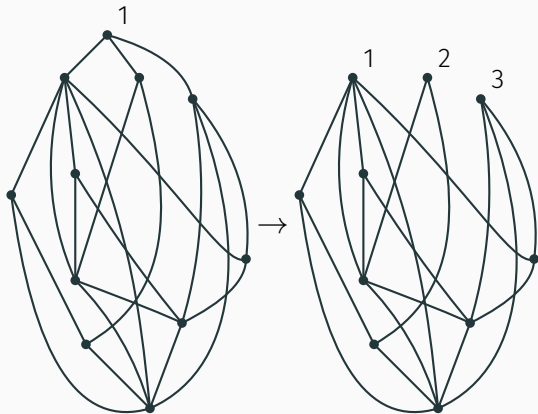
DOAG = Unlabelled DAG

- + a total order on the **outgoing** edges of each vertex
- + a total order on the sources
- + only one sink



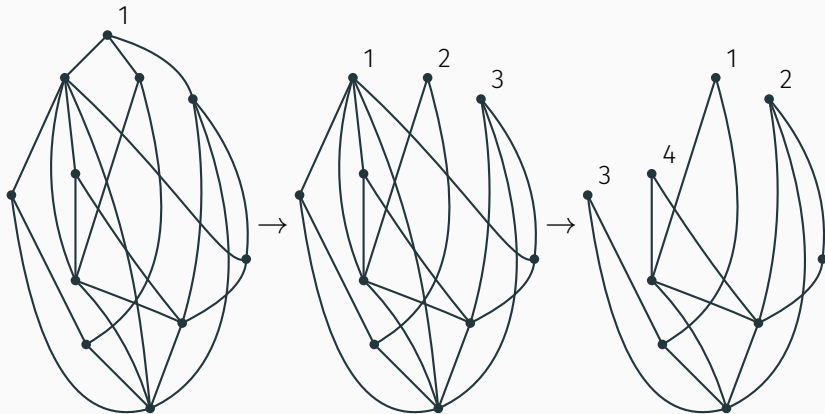
Recursive decomposition

Idea: remove the smallest source and see what is left.



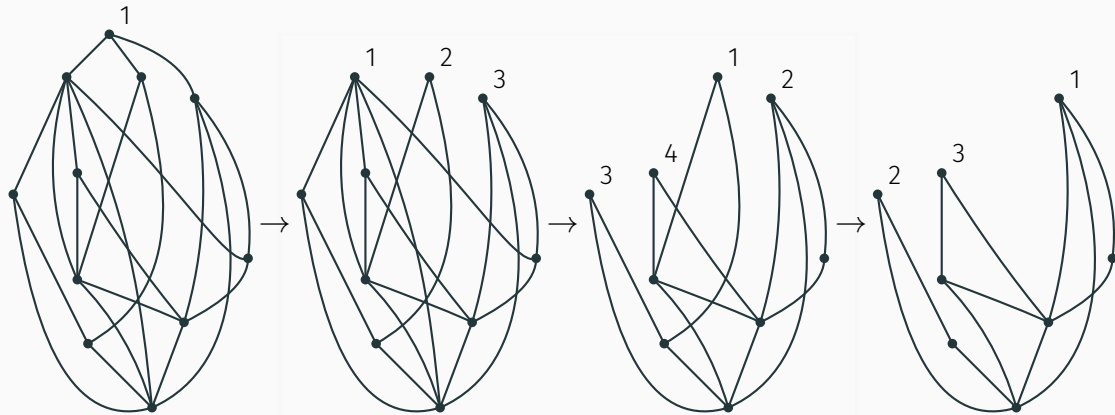
Recursive decomposition

Idea: remove the smallest source and see what is left.



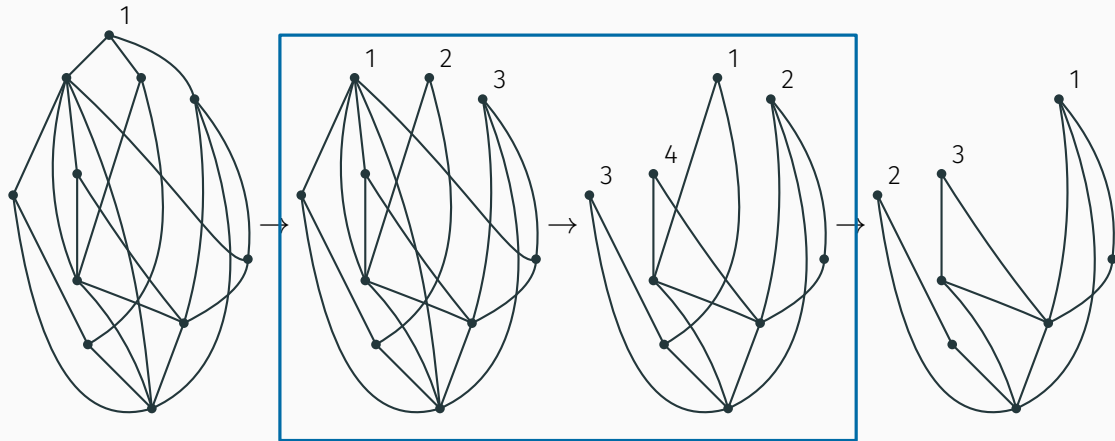
Recursive decomposition

Idea: remove the smallest source and see what is left.

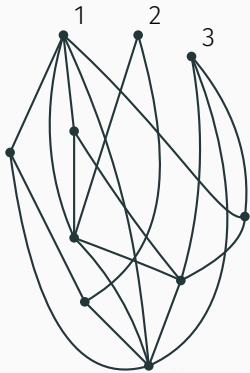


Recursive decomposition

Idea: remove the smallest source and see what is left.

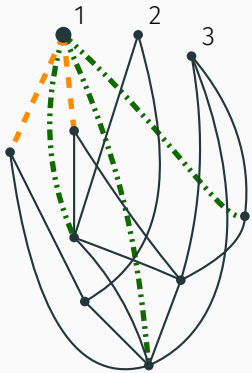


Recursive decomposition



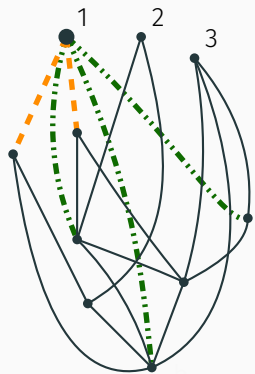
n vertices, m edges, k sources

Recursive decomposition

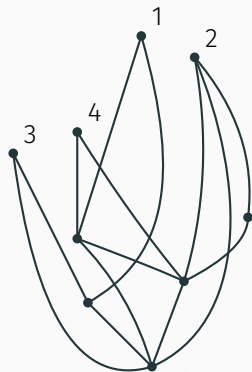


n vertices, m edges, k sources

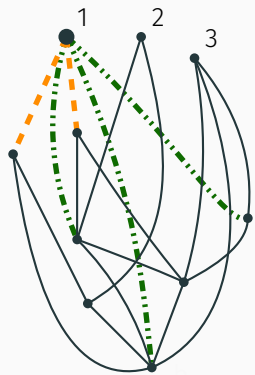
Recursive decomposition



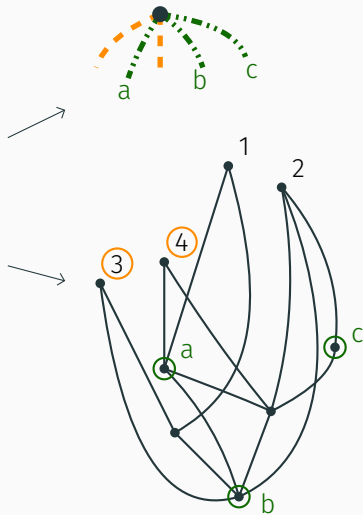
n vertices, m edges, k sources



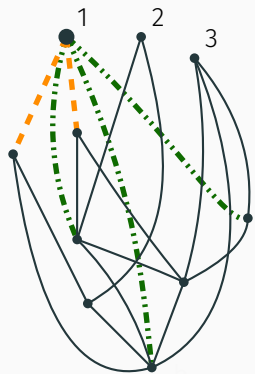
Recursive decomposition



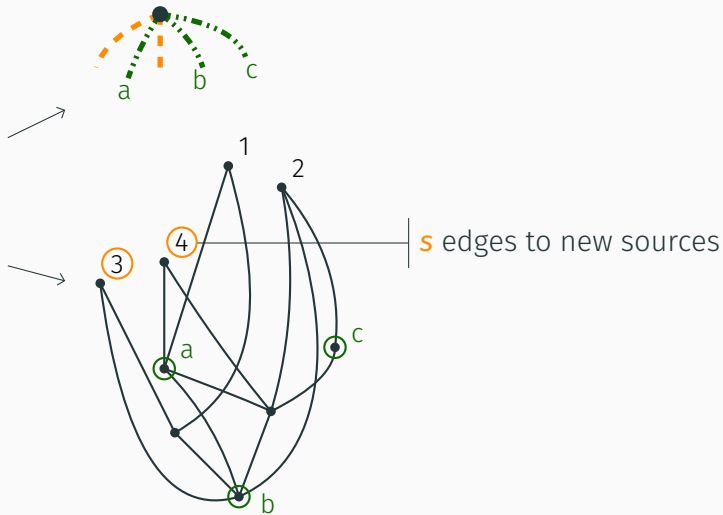
n vertices, m edges, k sources



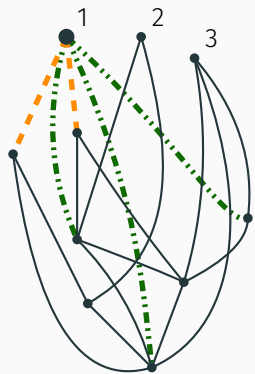
Recursive decomposition



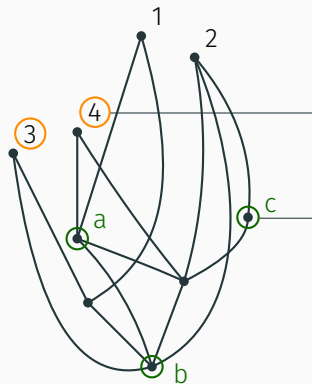
n vertices, m edges, k sources



Recursive decomposition



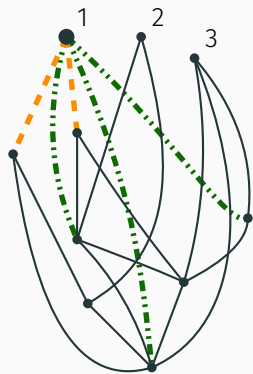
n vertices, m edges, k sources



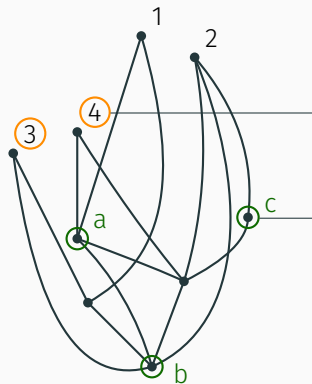
s edges to new sources

i edges to internal nodes
 $\hookrightarrow \binom{n-k-s}{i}$ choices

Recursive decomposition



n vertices, m edges, k sources

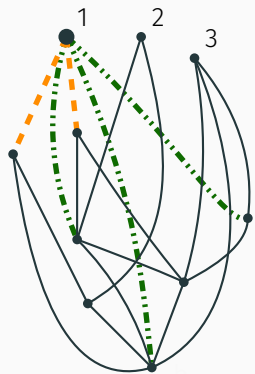


s edges to new sources

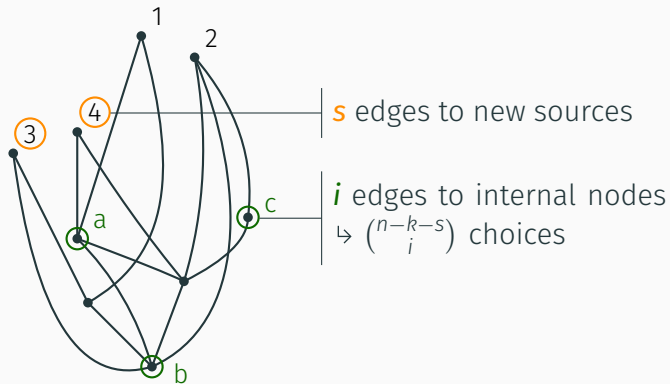
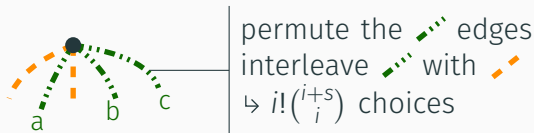
i edges to internal nodes
 $\hookrightarrow \binom{n-k-s}{i}$ choices

$(n-1)$ vertices, $(m-i-s)$ edges, $(k+s-1)$ sources

Recursive decomposition



n vertices, m edges, k sources



$(n - 1)$ vertices, $(m - i - s)$ edges, $(k + s - 1)$ sources

Recurrence formula

Counting formula

$$\begin{aligned} D_{n,m,k} &= \#\{\text{DOAGs with } n \text{ vertices, } m \text{ edges and } k \text{ sources}\} \\ &= \sum_{i+s>0} D_{n-1,m-i-s,k+s-1} \binom{n-k-s}{i} i! \binom{i+s}{i} \end{aligned}$$

Recurrence formula

Counting formula

$$\begin{aligned} D_{n,m,k} &= \#\{\text{DOAGs with } n \text{ vertices, } m \text{ edges and } k \text{ sources}\} \\ &= \sum_{i+s>0} D_{n-1,m-i-s,k+s-1} \binom{n-k-s}{i} i! \binom{i+s}{i} \end{aligned}$$

Complexity: computing all $D_{n,m,k}$ for $n, k \leq N$ and $m \leq M$ costs:
→ $O(N^4 M)$ arithmetic operations;
→ on integers of bit-size $O(M \log M)$.

Recurrence formula

Counting formula

$$\begin{aligned} D_{n,m,k} &= \#\{\text{DOAGs with } n \text{ vertices, } m \text{ edges and } k \text{ sources}\} \\ &= \sum_{i+s>0} D_{n-1,m-i-s,k+s-1} \binom{n-k-s}{i} i! \binom{i+s}{i} \end{aligned}$$

Complexity: computing all $D_{n,m,k}$ for $n, k \leq N$ and $m \leq M$ costs:
→ $O(N^4 M)$ arithmetic operations;
→ on integers of bit-size $O(M \log M)$.

In practice: about 400 edges in a few minutes.

Recursive random sampling = ʘnitnuoɔ

The recursive method [NW78]

If you can count, you can sample: just do the same, but backwards!

Recursive random sampling = ɹɹɹɹɹɹɹɹ

The recursive method [NW78]

If you can count, you can sample: just do the same, but backwards!

→ input = (n, m, k) ;

→ output = uniform DOAG with n vertices, m edges, and k sources.

Recursive random sampling = ʘnitnuoʘ

The recursive method [NW78]

If you can count, you can sample: just do the same, but backwards!

→ input = (n, m, k) ;

→ output = uniform DOAG with n vertices, m edges, and k sources.

Complexity: $O\left(\sum_{v \text{ vertex}} d_v^2\right) = O(M^2)$.
↳ out-degree of v

Recursive random sampling = ʘnitnuoʘ

The recursive method [NW78]

If you can count, you can sample: just do the same, but backwards!

→ input = (n, m, k) ;

→ output = uniform DOAG with n vertices, m edges, and k sources.

Complexity: $O\left(\sum_{v \text{ vertex}} d_v^2\right) = O(M^2)$.
↳ out-degree of v

In practice: about 400 edges in a few ms.

Outline of the presentation

Background

Directed ordered acyclic graphs

↳ *definition and recursive decomposition*

Asymptotic analysis

↳ *matrix encoding*

↳ *asymptotic result*

↳ *faster sampler*

Bonus: labelled DAGs

↳ *another way of counting*

A first asymptotic result

Asymptotics: (approximately) how many large DOAGs are there when $n, m \rightarrow \infty$? (And what about k ?)

A first asymptotic result

Asymptotics: (approximately) how many large DOAGs are there when $n, m \rightarrow \infty$? (And what about k ?)

Simplification: drop parameters, only count by vertices.

$$D_n \stackrel{\text{def}}{=} \#\{\text{DOAG with } n \text{ vertices, one source.}\}$$

A first asymptotic result

Asymptotics: (approximately) how many large DOAGs are there when $n, m \rightarrow \infty$? (And what about k ?)

Simplification: drop parameters, only count by vertices.

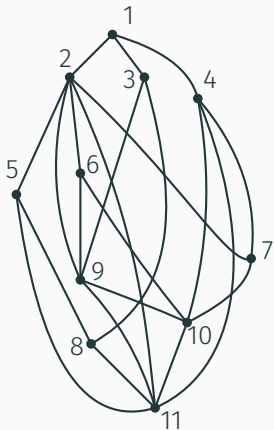
$$D_n \stackrel{\text{def}}{=} \#\{\text{DOAG with } n \text{ vertices, one source.}\}$$

Number of single-source DOAGs

$$D_n \underset{n \rightarrow \infty}{\sim} c \cdot n^{-1/2} \cdot e^{n-1} \cdot jn - 1!$$

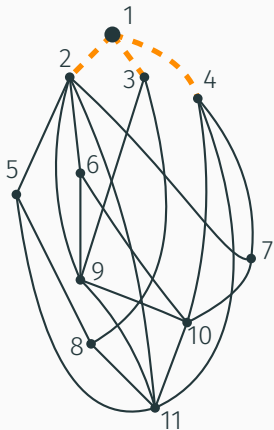
for $c \approx 0.30256$ and where $jn - 1! = \prod_{k=1}^m k!$.

Matrix encoding



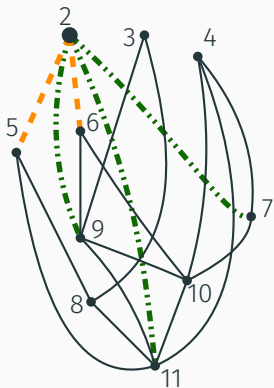
1	2	3	4	5	6	7	8	9	10	11		
												1
												2
												3
												4
												5
												6
												7
												8
												9
												10
												11

Matrix encoding



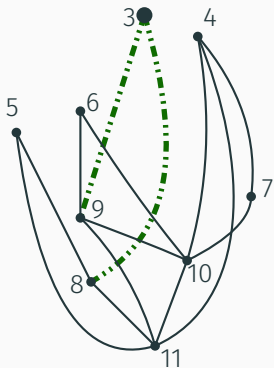
1	2	3	4	5	6	7	8	9	10	11	
	1	2	3								1
											2
											3
											4
											5
											6
											7
											8
											9
											10
											11

Matrix encoding



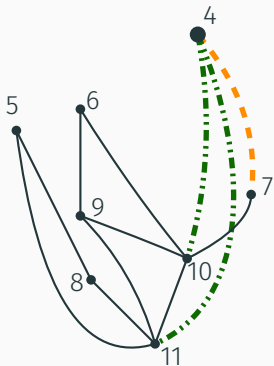
1	2	3	4	5	6	7	8	9	10	11	
	1	2	3								1
				1	3	5		2		4	2
											3
											4
											5
											6
											7
											8
											9
											10
											11

Matrix encoding



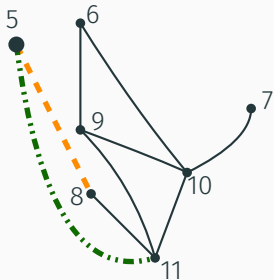
	1	2	3	4	5	6	7	8	9	10	11	
1		1	2	3								1
2					1	3	5		2		4	2
3								2	1			3
4												4
5												5
6												6
7												7
8												8
9												9
10												10
11												11

Matrix encoding



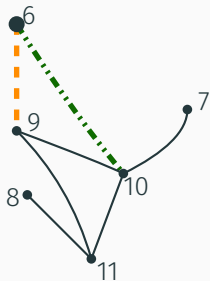
	1	2	3	4	5	6	7	8	9	10	11	
1		1	2	3								1
2					1	3	5		2		4	2
3								2	1			3
4							3			1	2	4
5												5
6												6
7												7
8												8
9												9
10												10
11												11

Matrix encoding



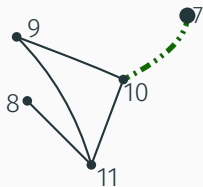
	1	2	3	4	5	6	7	8	9	10	11	
1		1	2	3								1
2					1	3	5		2		4	2
3								2	1			3
4							3			1	2	4
5								2			1	5
6												6
7												7
8												8
9												9
10												10
11												11

Matrix encoding



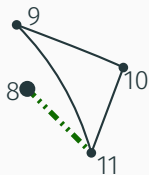
	1	2	3	4	5	6	7	8	9	10	11	
1		1	2	3								1
2					1	3	5		2		4	2
3								2	1			3
4							3			1	2	4
5								2			1	5
6									1	2		6
7												7
8												8
9												9
10												10
11												11

Matrix encoding



	1	2	3	4	5	6	7	8	9	10	11		
1		1	2	3									1
2					1	3	5		2		4		2
3								2	1				3
4							3			1	2		4
5								2				1	5
6									1	2			6
7										1			7
8													8
9													9
10													10
11													11

Matrix encoding



	1	2	3	4	5	6	7	8	9	10	11		
1		1	2	3									1
2					1	3	5		2		4		2
3								2	1				3
4							3			1	2		4
5								2				1	5
6									1	2			6
7										1			7
8												1	8
9													9
10													10
11													11

Matrix encoding



	1	2	3	4	5	6	7	8	9	10	11		
1		1	2	3									1
2					1	3	5		2		4		2
3								2	1				3
4							3			1	2		4
5								2				1	5
6									1	2			6
7										1			7
8											1		8
9										2	1		9
10													10
11													11

Matrix encoding



	1	2	3	4	5	6	7	8	9	10	11		
1		1	2	3									1
2					1	3	5		2		4		2
3								2	1				3
4							3			1	2		4
5								2				1	5
6									1	2			6
7											1		7
8												1	8
9										2	1		9
10												1	10
11													11

Matrix encoding

	1	2	3	4	5	6	7	8	9	10	11	
	1	2	3									1
					1	3	5		2		4	2
								2	1			3
							3			1	2	4
								2			1	5
									1	2		6
										1		7
											1	8
										2	1	9
											1	10
												11

Matrix encoding

1. strict upper triangular matrix;

	1	2	3	4	5	6	7	8	9	10	11		
1		1	2	3									1
2					1	3	5		2		4		2
3								2	1				3
4							3			1	2		4
5								2				1	5
6									1	2			6
7										1			7
8												1	8
9										2	1		9
10												1	10
11													11

Matrix encoding

1. strict upper triangular matrix;
2. there is an element at (1,2);

	1	2	3	4	5	6	7	8	9	10	11		
1		1	2	3									1
2					1	3	5		2		4		2
3								2	1				3
4							3			1	2		4
5								2				1	5
6									1	2			6
7										1			7
8												1	8
9										2	1		9
10												1	10
11													11

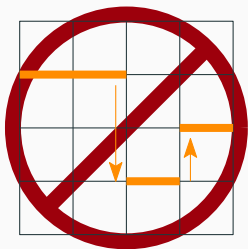
Matrix encoding

1. strict upper triangular matrix;
2. there is an element at (1,2);
3. increasing numbers above orange lines;

	1	2	3	4	5	6	7	8	9	10	11		
1		1	2	3									1
2					1	3	5		2		4		2
3								2	1				3
4							3			1	2		4
5								2				1	5
6									1	2			6
7										1			7
8											1		8
9										2	1		9
10												1	10
11													11

Matrix encoding

1. strict upper triangular matrix;
2. there is an element at (1,2);
3. increasing numbers above orange lines;
4. orange lines go down.



1	2	3	4	5	6	7	8	9	10	11	
	1	2	3								1
				1	3	5		2		4	2
							2	1			3
						3			1	2	4
							2			1	5
								1	2		6
									1		7
										1	8
									2	1	9
										1	10
											11

Proof sketch (1/3)

The plan: 1. Upper bound 2. Lower bound 3. Bootstrapping

Proof sketch (1/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

Proof sketch (1/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & & \mathbf{5} & & \mathbf{2} & \mathbf{4} & & \mathbf{3} \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & \mathbf{5} & \mathbf{2} & \mathbf{4} & \mathbf{3} \\ \hline \end{array} \star \begin{array}{|c|c|c|} \hline & & \\ \hline \end{array}$$

Proof sketch (1/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & & \mathbf{5} & & \mathbf{2} & \mathbf{4} & & \mathbf{3} \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & \mathbf{5} & \mathbf{2} & \mathbf{4} & \mathbf{3} \\ \hline \end{array} \star \begin{array}{|c|c|c|} \hline & & \\ \hline \end{array}$$

Variation

$$= \text{SEQ}(\mathcal{Z}) \star \text{SET}(\mathcal{Z})$$

Proof sketch (1/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & & \mathbf{5} & & \mathbf{2} & \mathbf{4} & & \mathbf{3} \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & \mathbf{5} & \mathbf{2} & \mathbf{4} & \mathbf{3} \\ \hline \end{array} \star \begin{array}{|c|c|c|} \hline & & \\ \hline \end{array}$$

Variation

$$= \text{SEQ}(\mathcal{Z}) \star \text{SET}(\mathcal{Z})$$

$V(z)$

$$= (1 - z)^{-1} e^z$$

Proof sketch (1/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\boxed{6} \boxed{1} \boxed{} \boxed{5} \boxed{} \boxed{2} \boxed{4} \boxed{} \boxed{3} = \boxed{6} \boxed{1} \boxed{5} \boxed{2} \boxed{4} \boxed{3} \star \boxed{} \boxed{} \boxed{}$$

$$\text{Variation} = \text{SEQ}(\mathcal{Z}) \star \text{SET}(\mathcal{Z})$$

$$V(z) = (1 - z)^{-1} e^z$$

$$V_n = e \cdot n! - o(1)$$

Proof sketch (1/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & & \mathbf{5} & & \mathbf{2} & \mathbf{4} & & \mathbf{3} \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|c|} \hline \mathbf{6} & \mathbf{1} & \mathbf{5} & \mathbf{2} & \mathbf{4} & \mathbf{3} \\ \hline \end{array} \star \begin{array}{|c|c|c|} \hline & & \\ \hline \end{array}$$

$$\text{Variation} = \text{SEQ}(\mathcal{Z}) \star \text{SET}(\mathcal{Z})$$

$$V(z) = (1 - z)^{-1} e^z$$

$$V_n = e \cdot n! - o(1)$$

$$\#\{\text{DOAG matrices}\} = \#\{\text{collections of rows}\} \leq \#\{\text{collections of variations}\}$$

Proof sketch (1/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\begin{array}{|c|c|c|c|c|c|c|} \hline 6 & 1 & & 5 & & 2 & 4 & & 3 \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|c|} \hline 6 & 1 & 5 & 2 & 4 & 3 \\ \hline \end{array} \star \begin{array}{|c|c|c|} \hline & & \\ \hline \end{array}$$

$$\text{Variation} = \text{SEQ}(\mathcal{Z}) \star \text{SET}(\mathcal{Z})$$

$$V(z) = (1 - z)^{-1} e^z$$

$$V_n = e \cdot n! - o(1)$$

$$\#\{\text{DOAG matrices}\} = \#\{\text{collections of rows}\} \leq \#\{\text{collections of variations}\}$$

$$D_n \leq \prod_{k=1}^{n-1} v_k \leq e^{n-1} (n-1)!$$

Proof sketch (2/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

{DOAG matrices} \supseteq

The diagram illustrates the decomposition of a set of DOAG matrices. It shows three 6x6 grids, each representing a matrix. Each grid has a diagonal of orange blocks and a question mark in the top-right corner. The grids are separated by plus signs, and the sequence ends with an ellipsis. The first grid has a question mark in the top-right corner. The second grid has a question mark in the top-right corner. The third grid has a question mark in the top-right corner.

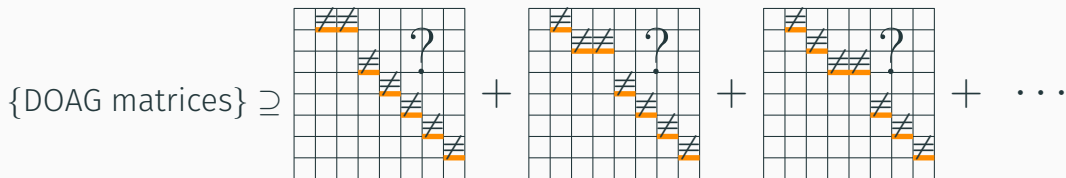
Proof sketch (2/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping



$$D_n \geq \frac{c' \cdot \ln(n)}{n} e^{n-1} (n-1)!$$

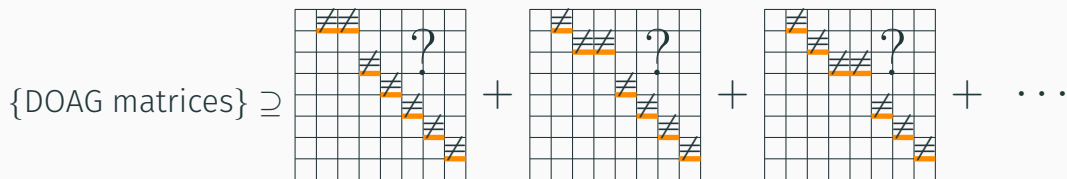
Proof sketch (2/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping



$$D_n \geq \frac{c' \cdot \ln(n)}{n} e^{n-1} i_{n-1}!$$

$$P_n = \frac{D_n}{e^{n-1} i_{n-1}!}$$

\Rightarrow

$$\frac{c' \cdot \ln(n)}{n} \leq P_n \leq 1$$

Proof sketch (3/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\{\text{DOAG matrices}\} = \begin{matrix} \begin{matrix} \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \end{matrix} \\ \begin{matrix} \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \end{matrix} \end{matrix} D_{n-1} + \begin{matrix} \begin{matrix} \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \end{matrix} \\ \begin{matrix} \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \end{matrix} \end{matrix} D_{n-2} + \begin{matrix} \begin{matrix} \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \\ \text{///} \end{matrix} \\ \begin{matrix} \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \\ \text{ } \end{matrix} \end{matrix} D_{n-3} + \dots$$

$$D_n = (v_{n-1} - v_{n-2})D_{n-1} + \frac{1}{2}(v_{n-1} - 2v_{n-2} + v_{n-3})v_{n-3}D_{n-2} + \dots$$

$$P_n = \left(1 - \frac{1}{n-1}\right) P_{n-1} + \frac{1}{2(n-2)} \left(1 - \frac{2}{n-1} + \frac{1}{(n-1)(n-2)}\right) P_{n-2} + \dots$$

$$P_n \sim P_{n-1}$$

Proof sketch (3/3)

The plan:

1. Upper bound

2. Lower bound

3. Bootstrapping

$$\{\text{DOAG matrices}\} = \begin{array}{|c|} \hline \begin{array}{c} \text{//} \\ \text{//} \\ \text{//} \\ \hline D_{n-1} \\ \hline \end{array} \\ \hline \end{array} + \begin{array}{|c|} \hline \begin{array}{c} \text{//} \\ \text{//} \\ \text{//} \\ \hline D_{n-2} \\ \hline \end{array} \\ \hline \end{array} + \begin{array}{|c|} \hline \begin{array}{c} \text{//} \\ \text{//} \\ \text{//} \\ \hline \text{or} \\ \hline D_{n-3} \\ \hline \end{array} \\ \hline \end{array} + \dots$$

$$D_n = (v_{n-1} - v_{n-2})D_{n-1} + \frac{1}{2}(v_{n-1} - 2v_{n-2} + v_{n-3})v_{n-3}D_{n-2} + \dots$$

$$P_n = \left(1 - \frac{1}{n-1}\right) P_{n-1} + \frac{1}{2(n-2)} \left(1 - \frac{2}{n-1} + \frac{1}{(n-1)(n-2)}\right) P_{n-2} + \dots$$

$$P_n = P_{n-1} \left(1 - \frac{1}{2n} + O(n^{-2})\right) \Rightarrow P_n \sim c \cdot n^{-1/2}$$

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\text{matrices of variations of length } 1, 2, \dots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\text{matrices of variations of length } 1, 2, \dots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw matrices of variations until they correspond to a DOAG

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\text{matrices of variations of length } 1, 2, \dots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw matrices of variations until they correspond to a DOAG

Generating one variation: $\sim n \log_2(n)$ random bits.

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\text{matrices of variations of length } 1, 2, \dots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw matrices of variations until they correspond to a DOAG

Generating one variation: $\sim n \log_2(n)$ random bits.

Complexity:

$$\begin{aligned} & \text{Cost}(\text{successful generation}) + \#\text{rejections} \times \text{Cost}(\text{failed generation}) \\ &= \frac{n^2}{2} \log_2(n) + O(\sqrt{n} \cdot \mathbf{Cost}(\text{one failed generation})) \end{aligned}$$

Early rejection

	?	?	?	?	?	?	?	?	?
		?	?	?	?	?	?	?	?
			?	?	?	?	?	?	?
				?	?	?	?	?	?
					?	?	?	?	?
						?	?	?	?
							?	?	?
								?	?
									?



Early rejection

	5	?	?	?	?	?	?	?	?	?
		?	?	?	?	?	?	?	?	?
			?	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	5	?	?	?	?	?	?	?	?	?
		3	?	?	?	?	?	?	?	?
			?	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?

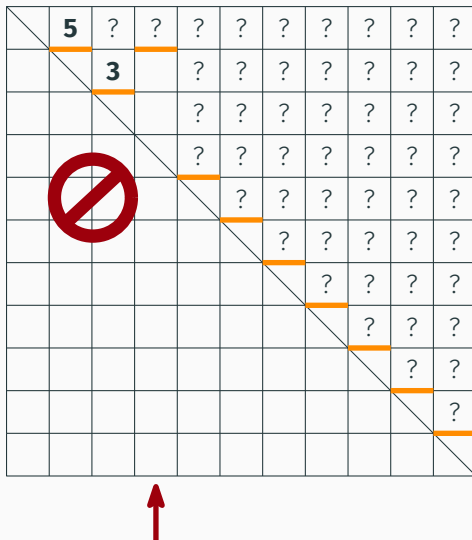


Early rejection

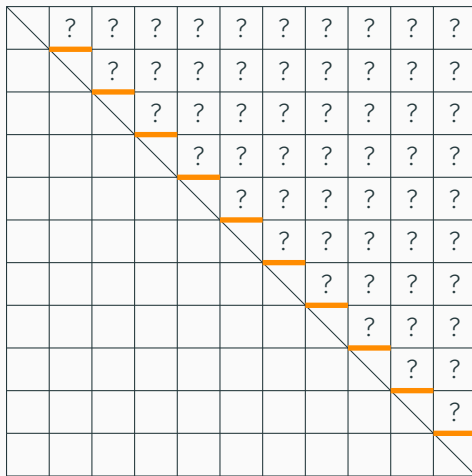
	5	?	?	?	?	?	?	?	?	?
		3	?	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection



Early rejection



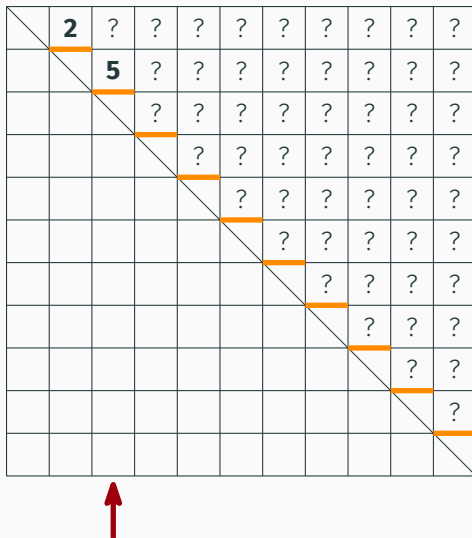
Early rejection

	2	?	?	?	?	?	?	?	?	?
		?	?	?	?	?	?	?	?	?
			?	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	?	?	?	?	?	?	?	?
			?	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	?	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	7	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	7	?	?	?	?	?	?	?
				?	?	?	?	?	?	?
					?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



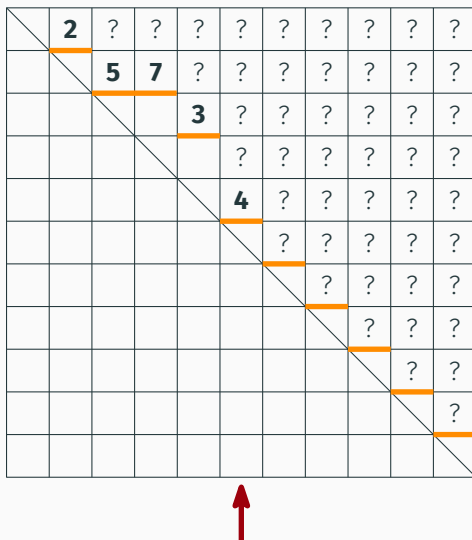
Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	7	?	?	?	?	?	?	?
				3	?	?	?	?	?	?
					?	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	7	?	?	?	?	?	?	?
				3	?	?	?	?	?	?
					?	?	?	?	?	?
					4	?	?	?	?	?
						?	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	7	?	?	?	?	?	?	?
				3	?	?	?	?	?	?
					?	?	?	?	?	?
					4	?	?	?	?	?
							?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?



Early rejection

	2	?	?	?	?	?	?	?	?	?
		5	7	?	?	?	?	?	?	?
				3	?	?	?	?	?	?
					?	?	?	?	?	?
					4	1	?	?	?	?
							?	?	?	?
								?	?	?
									?	?
										?

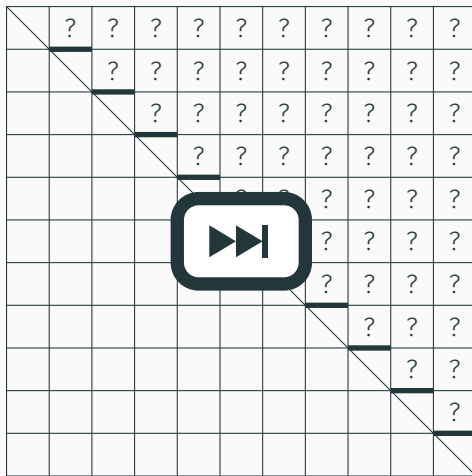
A red prohibition sign is overlaid on the cell (4,4). A red arrow points to the bottom of the grid at column 5.

Early rejection

	?	?	?	?	?	?	?	?	?
		?	?	?	?	?	?	?	?
			?	?	?	?	?	?	?
				?	?	?	?	?	?
					?	?	?	?	?
						?	?	?	?
							?	?	?
								?	?
									?




Early rejection



Early rejection

	3	?	?	?	?	?	?	?	?	?	
		2	6	?	?	?	?	?	?	?	
				?	?	?	?	?	?	?	
				7	?	?	?	?	?	?	
					4	5	?	?	?	?	
							?	?	?	?	
								1	?	?	
									2	3	?
											?
											1



Early rejection

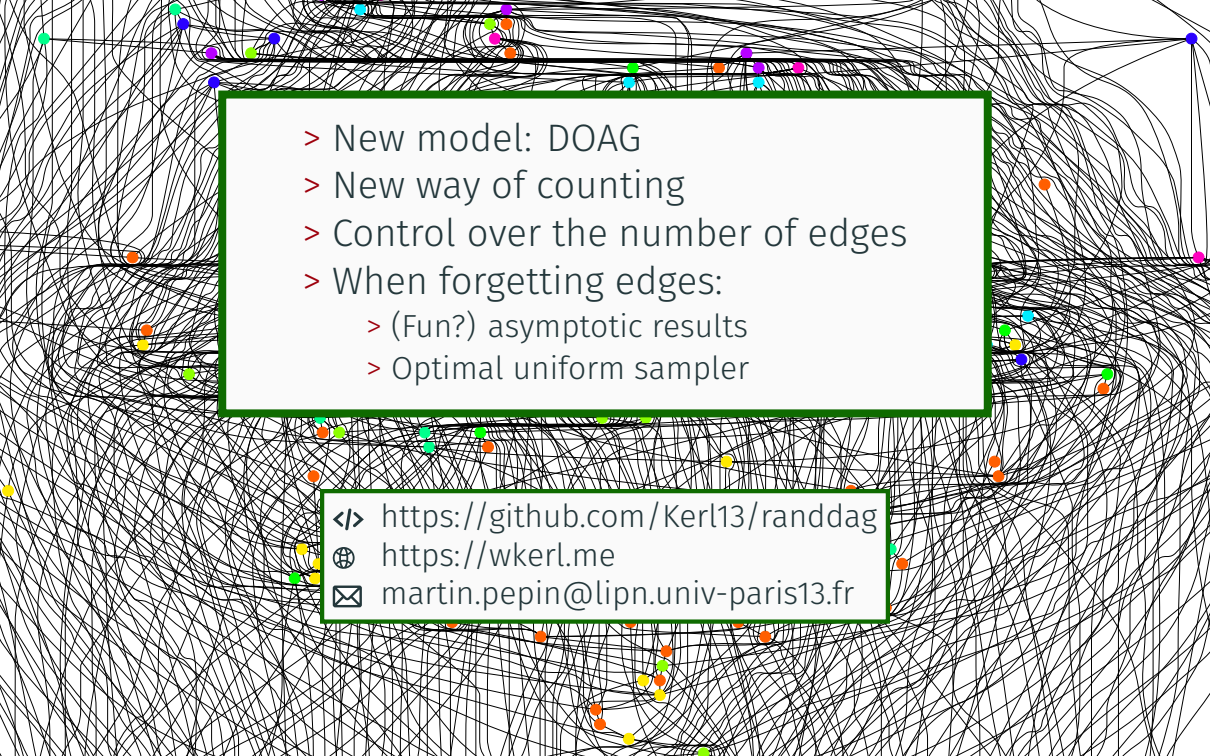
	3	?	?	?	?	?	?	?	?
		2	6	?	?	?	?	?	?
				?	?	?	?	?	?
				7	?	?	?	?	?
					4	5	?	?	?
							?	?	?
							1	?	?
								2	3
									?
									1

Complexity = $O(n \ln(n))$

Total complexity = $\frac{n^2}{2} \log_2(n) + O(\sqrt{n} \cdot n \ln(n))$

- Law of the number of edges?

- Law of the number of edges?
- Multigraph equivalent: DOAMG
 - Identical to compacted plane trees
 - We have to count by edges
 - Simpler recurrence relation
 - No asymptotics (yet)
 - Collaborations with Alfredo Viola (Montevideo) and Michael Wallner (TU Wien)

- 
- > New model: DOAG
 - > New way of counting
 - > Control over the number of edges
 - > When forgetting edges:
 - > (Fun?) asymptotic results
 - > Optimal uniform sampler

</> <https://github.com/Kerl13/randdag>

🌐 <https://wkerl.me>

✉ martin.pepin@lipn.univ-paris13.fr

Outline of the presentation

Background

Directed ordered acyclic graphs

↳ *definition and recursive decomposition*

Asymptotic analysis

↳ *matrix encoding*

↳ *asymptotic result*

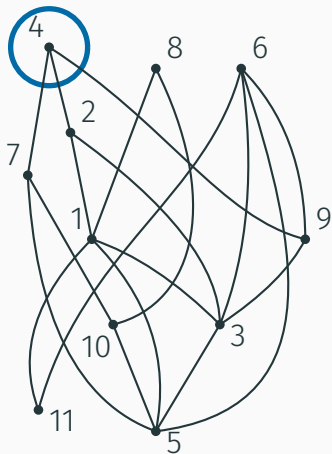
↳ *faster sampler*

Bonus: labelled DAGs

↳ *another way of counting*

What about labelled DAGs?

Idea: mark one source, and remove it.

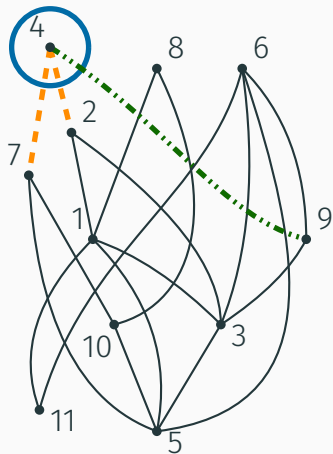


$$A_{n,m,k} = \#\text{DAGs } (n \text{ vertices, } m \text{ edges, } k \text{ sources})$$

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

What about labelled DAGs?

Idea: mark one source, and remove it.

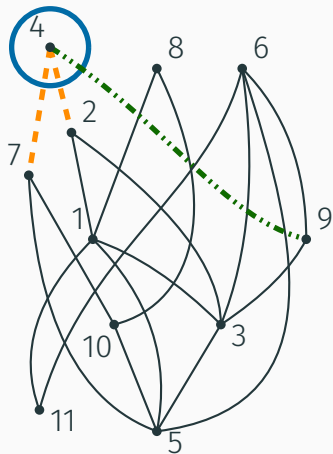


$A_{n,m,k} = \# \text{DAGs (} n \text{ vertices, } m \text{ edges, } k \text{ sources)}$

$$k A_{n,m,k} = n \sum_{i,s} A_{n-1,m-i-s,k+s-1} \binom{k+s-1}{s} \binom{n-s-k}{i}$$

What about labelled DAGs?

Idea: mark one source, and remove it.



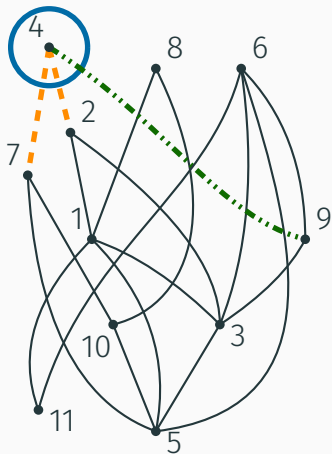
$A_{n,m,k} = \# \text{DAGs (} n \text{ vertices, } m \text{ edges, } k \text{ sources)}$

$$k A_{n,m,k} = n \sum_{i,s} A_{n-1,m-i-s,k+s-1} \binom{k+s-1}{s} \binom{n-s-k}{i}$$

→ New counting formula for DAGs;

What about labelled DAGs?

Idea: mark one source, and remove it.



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

$$k A_{n,m,k} = n \sum_{i,s} A_{n-1,m-i-s,k+s-1} \binom{k+s-1}{s} \binom{n-s-k}{i}$$

→ New counting formula for DAGs;

→ Effective sampler with fixed number of edges and vertices.

References I

- [EFW21] Andrew Elvey Price, Wenjie Fang, and Michael Wallner. “Compacted binary trees admit a stretched exponential”. In: *Journal of Combinatorial Theory, Series A* 177 (2021), page 105306. ISSN: 0097-3165. DOI: [10.1016/j.jcta.2020.105306](https://doi.org/10.1016/j.jcta.2020.105306).
- [Ges96] Ira Martin Gessel. “Counting acyclic digraphs by sources and sinks”. In: *Discrete Mathematics* 160.1 (1996), pages 253–258. ISSN: 0012-365X.
- [GGKW20] Antoine Genitrini et al. “Asymptotic enumeration of compacted binary trees of bounded right height”. In: *Journal of Combinatorial Theory, Series A* 172 (2020), page 105177. ISSN: 0097-3165. DOI: <https://doi.org/10.1016/j.jcta.2019.105177>.
- [KM15] Jack Kuipers and Giusi Moffa. “Uniform random generation of large acyclic digraphs”. In: *Statistics and Computing* 25.2 (2015), pages 227–242.
- [MDB01] Guy Melançon, Isabelle Dutour, and Mireille Bousquet-Mélou. “Random Generation of Directed Acyclic Graphs”. In: *Electronic Notes in Discrete Mathematics* 10 (2001), pages 202–207. DOI: [10.1016/S1571-0653\(04\)00394-4](https://doi.org/10.1016/S1571-0653(04)00394-4). URL: [https://doi.org/10.1016/S1571-0653\(04\)00394-4](https://doi.org/10.1016/S1571-0653(04)00394-4).

References II

- [NW78] Albert Nijenhuis and Herbert Wilf. *Combinatorial Algorithms: For Computers and Hand Calculators*. 2nd. USA: Academic Press, Inc., 1978. ISBN: 0125192606.
- [Rob70] Robert William Robinson. “Enumeration of acyclic digraphs”. In: *Proceedings of The Second Chapel Hill Conference on Combinatorial Mathematics and its Applications (Univ. North Carolina, Chapel Hill, NC, 1970)*, Univ. North Carolina, Chapel Hill, NC (University of North Carolina at Chapel Hill, North Carolina, May 8–13, 1970). 1970, pages 391–399.
- [Rob73] Robert William Robinson. “Counting labeled acyclic digraphs”. In: *New Directions in the Theory of Graphs* (1973), pages 239–273.
- [Rob77] Robert William Robinson. “Counting unlabeled acyclic digraphs”. In: *Combinatorial Mathematics V*. Lecture Notes in Mathematics. Springer, 1977, pages 28–43.
- [Sta73] Richard Peter Stanley. “Acyclic orientations of graphs”. In: *Discrete Mathematics* 5.2 (1973), pages 171–178.