DIRECTED ORDERED ACYCLIC GRAPHS

ENUMERATION, UNIFORM SAMPLING, AND LINKS WITH CLASSICAL LABELLED DAGS

Martin Pépin
joint work with Alfredo Viola & Antoine Genitrini

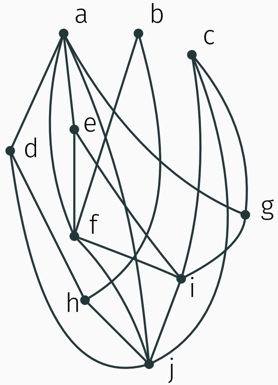
November 8, 2022
Séminaire CALIN

LIPN

Directed Acyclic Graphs

Directed Acyclic Graph (DAG)

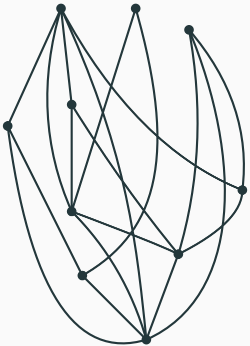
- A finite set of vertices V e.g. $\{a, b, c, \ldots, j\}$;
- a set of directed edges $E \subseteq V \times V$;
- no cycles.



Directed Acyclic Graphs

Directed Acyclic Graph (DAG)

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



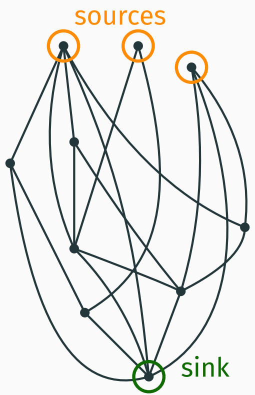
Without labels: **Unlabelled DAGs**

Directed Acyclic Graphs

Directed Acyclic Graph (DAG)

- A finite set of vertices V e.g. $\{a, b, c, \ldots, j\}$;
- 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...

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...

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?
- Unlabelled structures / other ways of breaking symmetries?

Outline of the presentation

Background

Directed ordered acyclic graphs

↳ *Definition and recursive decomposition*

Intermezzo: labelled DAGs

↳ *another way of counting*

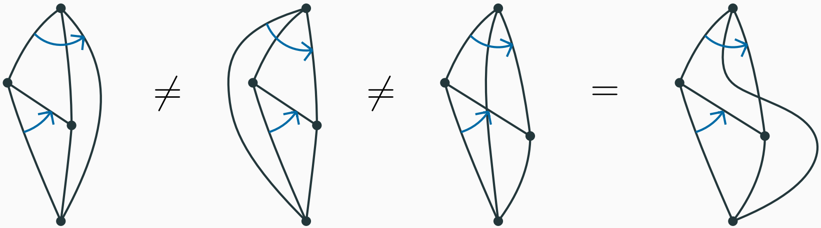
Asymptotic analysis

↳ *Matrix encoding*
↳ *Asymptotic result*
↳ *Faster sampler*

A new kind of DAG

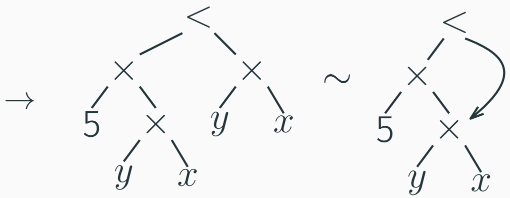
Directed Ordered Acyclic Graphs

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

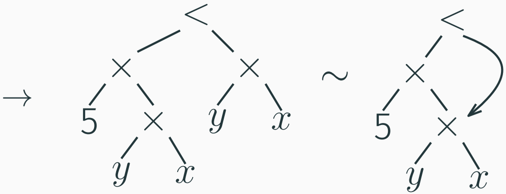


Motivations

- Compacted trees [GGKW20; EFW21] are DOAGs



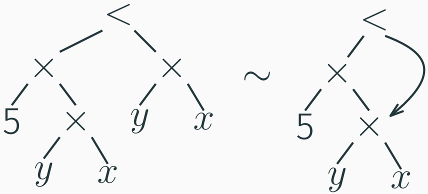
- Compacted trees [GGKW20; EFW21] are DOAGs



(also known as **hash-consing** in functional programming);

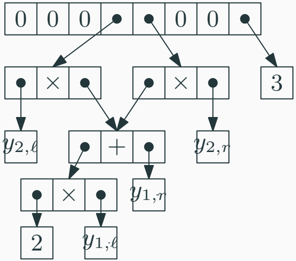
Motivations

- Compacted trees [GGKW20; EFW21] are DOAGs $\rightarrow$



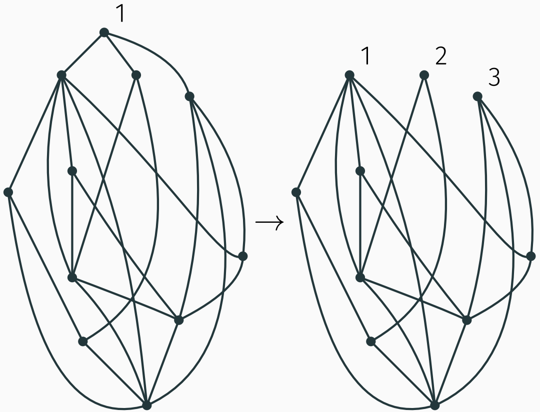
(also known as **hash-consing** in functional programming);

- Real-life implementations of DAGs have an **ordering**; $\rightarrow$



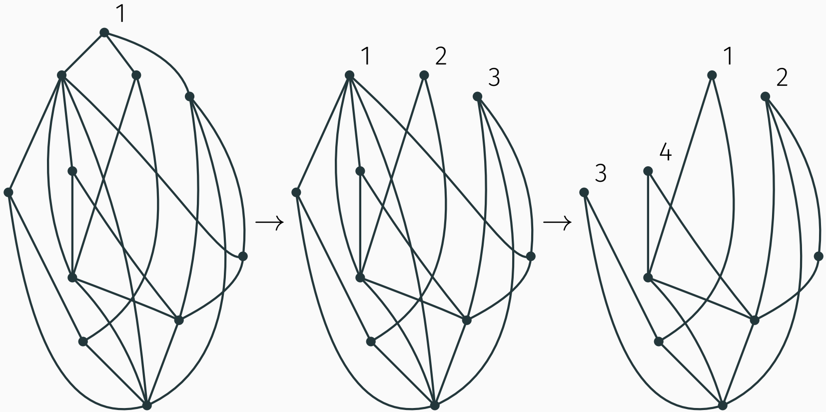
Recursive decomposition

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



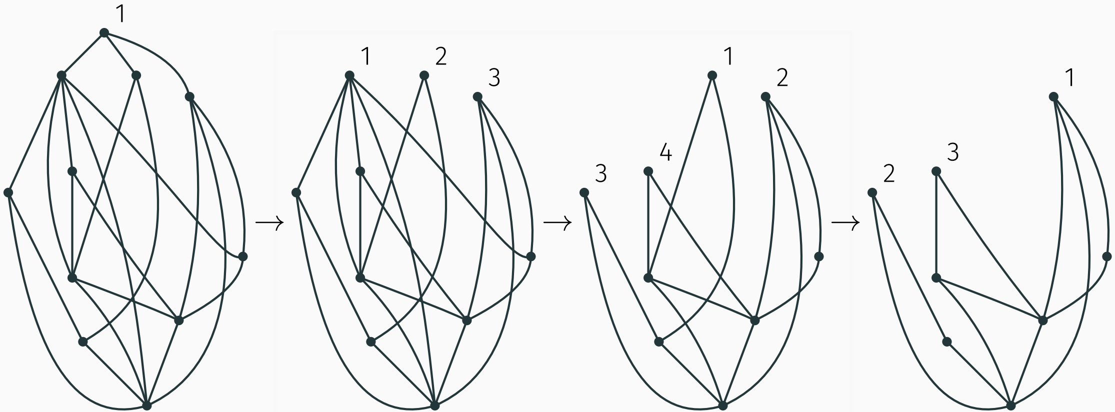
Recursive decomposition

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



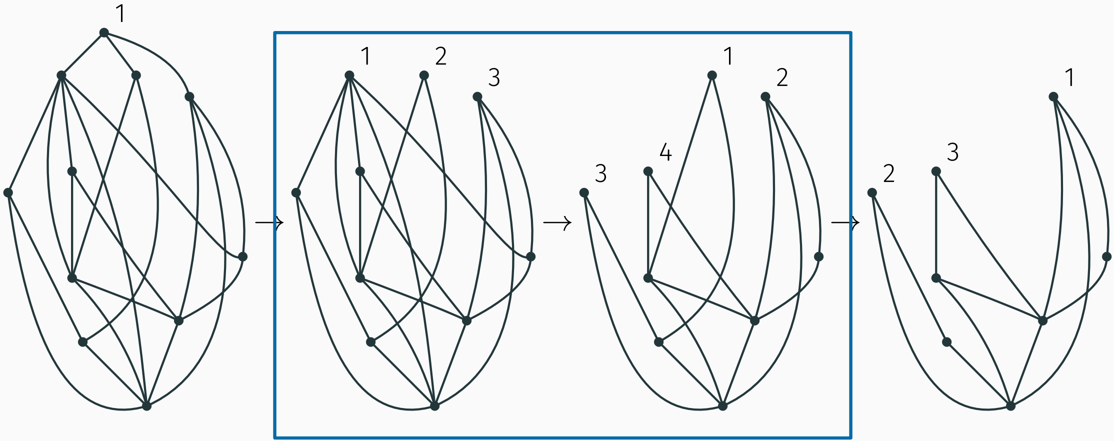
Recursive decomposition

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

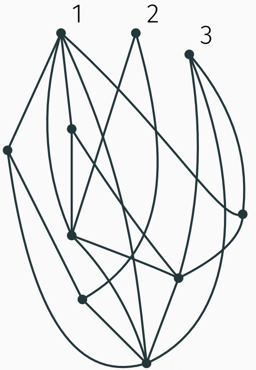


Recursive decomposition

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

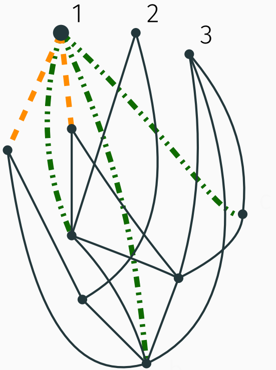


Recursive decomposition



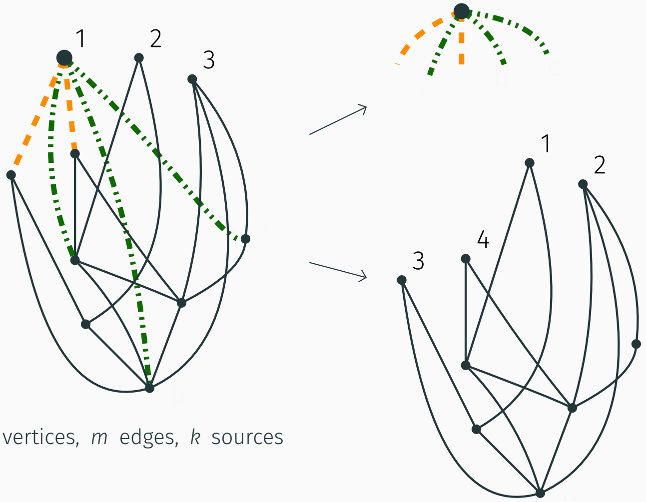
n vertices, m edges, k sources

Recursive decomposition



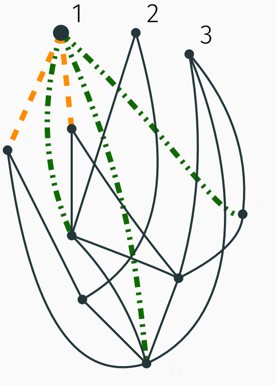
n vertices, m edges, k sources

Recursive decomposition

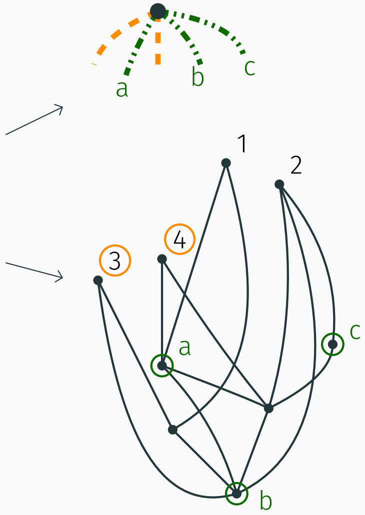


n vertices, m edges, k sources

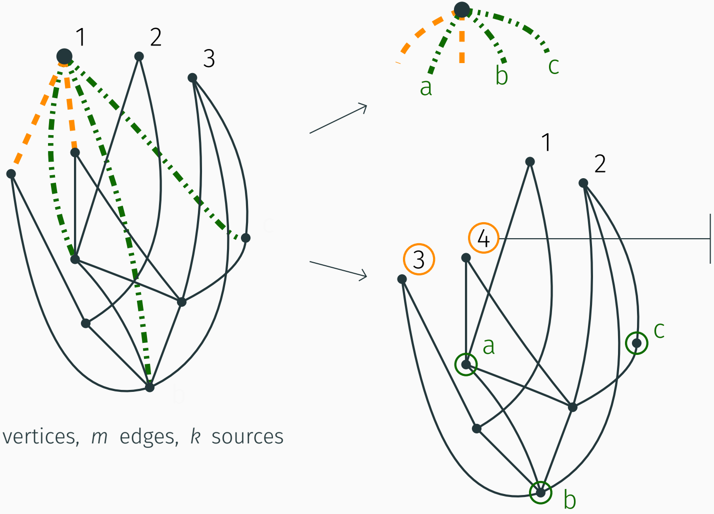
Recursive decomposition



n vertices, m edges, k sources



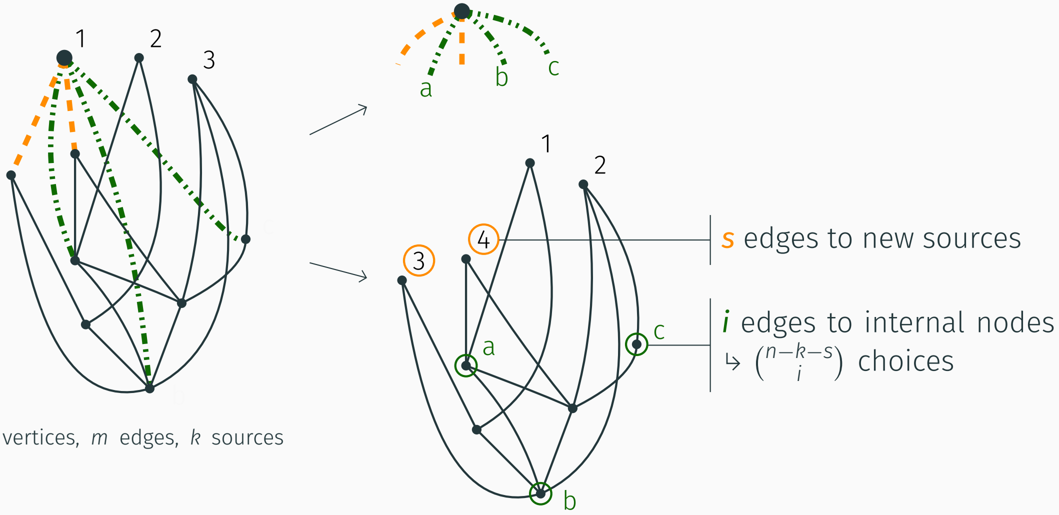
Recursive decomposition



s edges to new sources

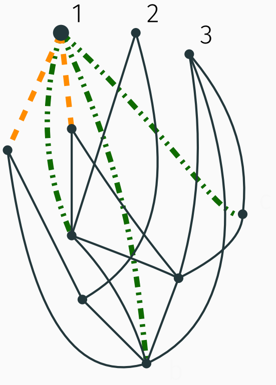
n vertices, m edges, k sources

Recursive decomposition

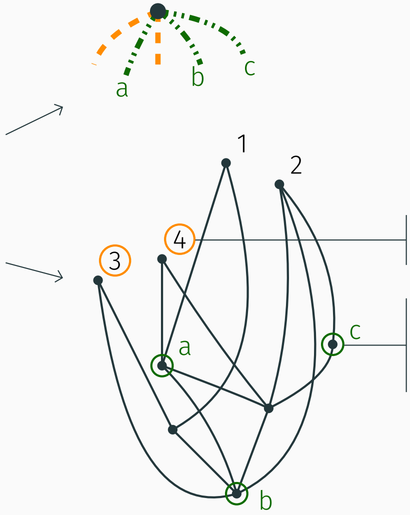


n vertices, m edges, k sources

Recursive decomposition



n vertices, m edges, k sources

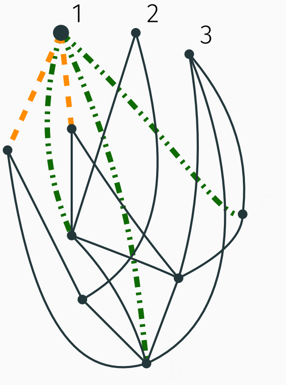


s edges to new sources

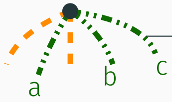
i edges to internal nodes
↳ $\binom{n-k-s}{i}$ choices

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

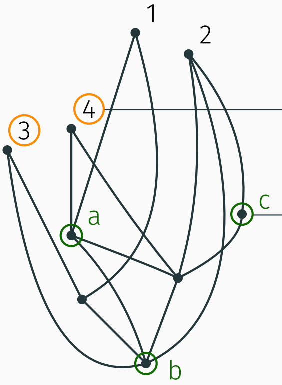
Recursive decomposition



n vertices, m edges, k sources



permute the edges
interleave with
↳ $i!\binom{i+s}{i}$ choices



s edges to new sources

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

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

Recurrence formula

Counting formula

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

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

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

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.

Random sampling = counting

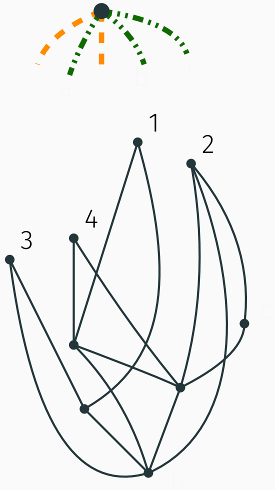
Do the same, but backwards!

Random sampling = ƃuıʇunoɔ

Do the same, but backwards!

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

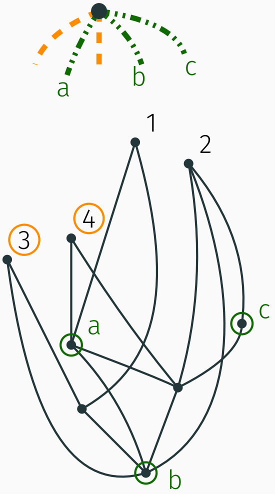
Random sampling = ƃuıʇunoɔ



Do the same, but backwards!

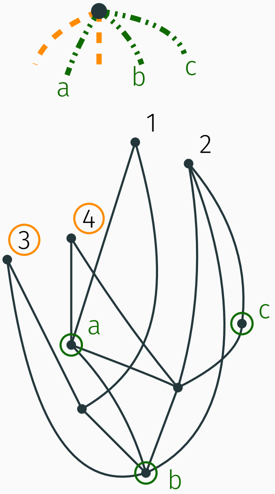
1. Select $({\color{green}i}, {\color{orange}s})$ with probability $\frac{D_{n-1,m-i-s,k+s-1}\binom{n-k-s}{i} i! \binom{i+s}{i}}{D_{n,m,k}}$;
2. sample a $\mathrm{DOAG}_{n-1,m-i-s,k+s-1}$;

Random sampling = counting



Do the same, but backwards!

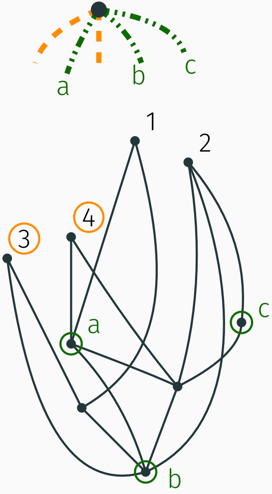
1. Select (i, s) with probability $\frac{D_{n-1,m-i-s,k+s-1}\binom{n-k-s}{i} i! \binom{i+s}{i}}{D_{n,m,k}}$;
2. sample a $\text{DOAG}_{n-1,m-i-s,k+s-1}$;
3. connect the s largest sources;



Do the same, but backwards!

1. Select (i, s) with probability $\frac{D_{n-1,m-i-s,k+s-1}\binom{n-k-s}{i} i! \binom{i+s}{i}}{D_{n,m,k}}$;
2. sample a $\text{DOAG}_{n-1,m-i-s,k+s-1}$;
3. connect the s largest sources;
4. connect i random internal vertices;

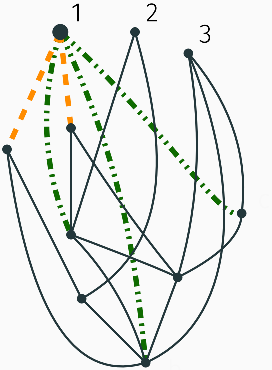
Random sampling = ƃuıʇunoɔ



Do the same, but backwards!

1. Select (i, s) with probability $\frac{D_{n-1,m-i-s,k+s-1}\binom{n-k-s}{i}i!\binom{i+s}{i}}{D_{n,m,k}}$;
2. sample a $\mathrm{DOAG}_{n-1,m-i-s,k+s-1}$;
3. connect the s largest sources;
4. connect i random internal vertices;
5. order the edges.

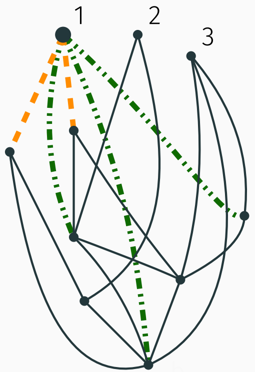
Random sampling = ƃuıʇunoɔ



Do the same, but backwards!

1. Select (i, s) with probability $\frac{D_{n-1,m-i-s,k+s-1}\binom{n-k-s}{i} i! \binom{i+s}{i}}{D_{n,m,k}}$;
2. sample a $\text{DOAG}_{n-1,m-i-s,k+s-1}$;
3. connect the s largest sources;
4. connect i random internal vertices;
5. order the edges.

Random sampling = counting



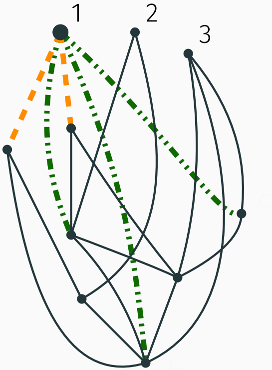
Do the same, but backwards!

1. Select (i, s) with probability $\frac{D_{n-1,m-i-s,k+s-1}\binom{n-k-s}{i}i!\binom{i+s}{i}}{D_{n,m,k}}$;
2. sample a $\text{DOAG}_{n-1,m-i-s,k+s-1}$;
3. connect the s largest sources;
4. connect i random internal vertices;
5. order the edges.

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

↳ out-degree of v

Random sampling = counting



Do the same, but backwards!

1. Select $({\color{green}i}, {\color{orange}s})$ with probability $\frac{D_{n-1,m-i-s,k+s-1}\binom{n-k-s}{i} i! \binom{i+s}{i}}{D_{n,m,k}}$;
2. sample a $\text{DOAG}_{n-1,m-i-s,k+s-1}$;
3. connect the s largest sources;
4. connect i random internal vertices;
5. order the edges.

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.

Conclusion

- New model
- New way of counting
- Control over the number of edges

Antoine Genitrini, Martin Pépin, and Alfredo Viola. “Unlabelled ordered DAGs and labelled DAGs: constructive enumeration and uniform random sampling”. In: *XI Latin and American Algorithms, Graphs and Optimization Symposium*. Eslevier. 2021

Outline of the presentation

Background

Directed ordered acyclic graphs

↳ *Definition and recursive decomposition*

Intermezzo: labelled DAGs

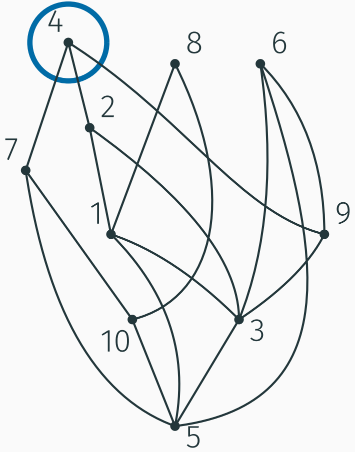
↳ *another way of counting*

Asymptotic analysis

↳ *Matrix encoding*
↳ *Asymptotic result*
↳ *Faster sampler*

What about labelled DAGs?

Idea: mark one source, and remove it.

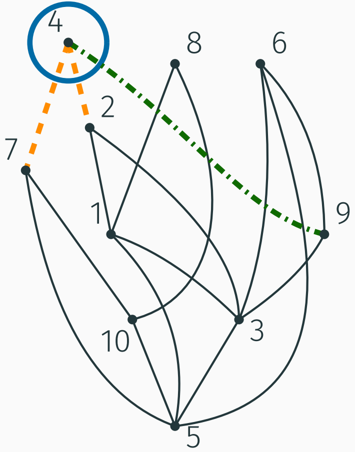


$V_{n,m,k}$ = #DAGs (one sink, k sources)

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

What about labelled DAGs?

Idea: mark one source, and remove it.



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

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

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

Consequences on labelled DAGs

- Counting formula without inclusion-exclusion;
- Effective sampler with fixed number of edges and vertices.

Outline of the presentation

Background

Directed ordered acyclic graphs

↳ *Definition and recursive decomposition*

Intermezzo: labelled DAGs

↳ *another way of counting*

Asymptotic analysis

↳ *Matrix encoding*
↳ *Asymptotic result*
↳ *Faster sampler*

What asymptotics?

Asymptotics: (approximately) how many large DOAGs are there when $n, m \to \infty$?

What asymptotics?

Asymptotics: (approximately) how many large DOAGs are there when $n, m \to \infty$?

Problem: relative growth of n and m?

- dense case: $m \sim c \cdot \binom{n}{2}$?
- sparse case: $m \sim c \cdot n$?
- are there critical values for c?

What asymptotics?

Asymptotics: (approximately) how many large DOAGs are there when $n, m \to \infty$?

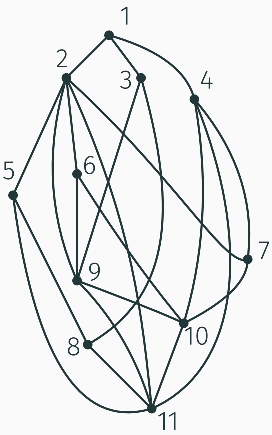
Problem: relative growth of n and m?

- dense case: $m \sim c \cdot \binom{n}{2}$?
- sparse case: $m \sim c \cdot n$?
- are there critical values for c?

Simplification: Drop one parameter: only count by vertices.

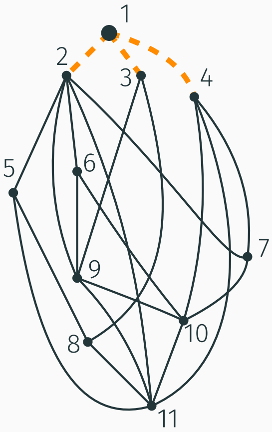
$$D_n \stackrel{\text{def}}{=} \#\{\text{DOAG with } n \text{ vertices, one source.}\}$$

Matrix encoding



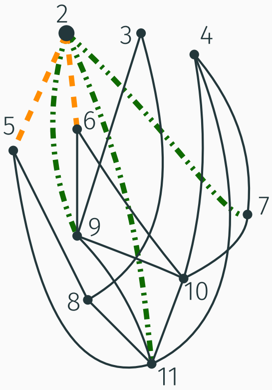
1	2	3	4	5	6	7	8	9	10	11	
											1
											2
											3
											4
											5
											6
											7
											8
											9
											10
											11

Matrix encoding

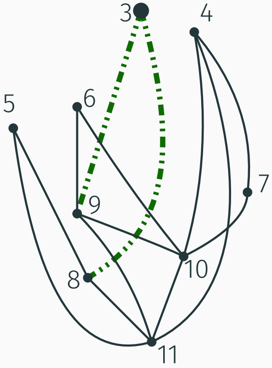


	1	2	3	4	5	6	7	8	9	10	11
1		**1**	**2**	**3**							
2											
3											
4											
5											
6											
7											
8											
9											
10											
11											

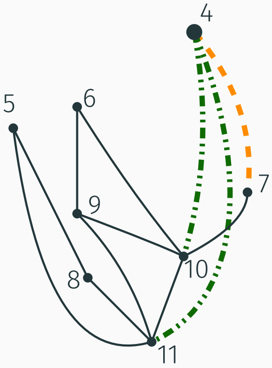
Matrix encoding



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

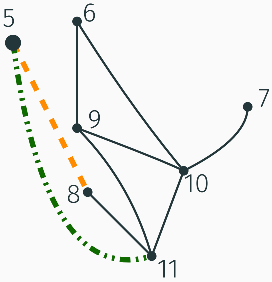


	1	2	3	4	5	6	7	8	9	10	11
1		**1**	**2**	**3**							
2					**1**	**3**	**5**		**2**		**4**
3								**2**	**1**		
4											
5											
6											
7											
8											
9											
10											
11											

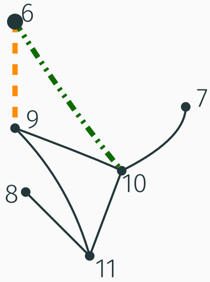


	1	2	3	4	5	6	7	8	9	10	11
1		**1**	**2**	**3**							
2					**1**	**3**	**5**		**2**		**4**
3								**2**	**1**		
4							**3**			**1**	**2**
5											
6											
7											
8											
9											
10											
11											

Matrix encoding

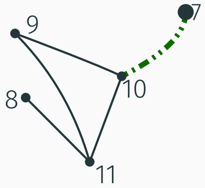


1	2	3	4	5	6	7	8	9	10	11	
	1	**2**	**3**								1
				1	**3**	**5**		**2**		**4**	2
							2	**1**			3
						3			**1**	**2**	4
							2			**1**	5
											6
											7
											8
											9
											10
											11



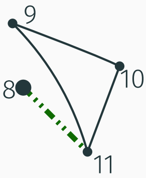
1	2	3	4	5	6	7	8	9	10	11	
	1	**2**	**3**								1
				1	**3**	**5**		**2**		**4**	2
							2	**1**			3
						3			**1**	**2**	4
							2			**1**	5
								1	**2**		6
											7
											8
											9
											10
											11

Matrix encoding



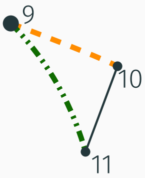
1	2	3	4	5	6	7	8	9	10	11	
	1	**2**	**3**								1
				1	**3**	**5**		**2**		**4**	2
							2	**1**			3
						3			**1**	**2**	4
							2			**1**	5
								1	**2**		6
									1		7
											8
											9
											10
											11

Matrix encoding



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

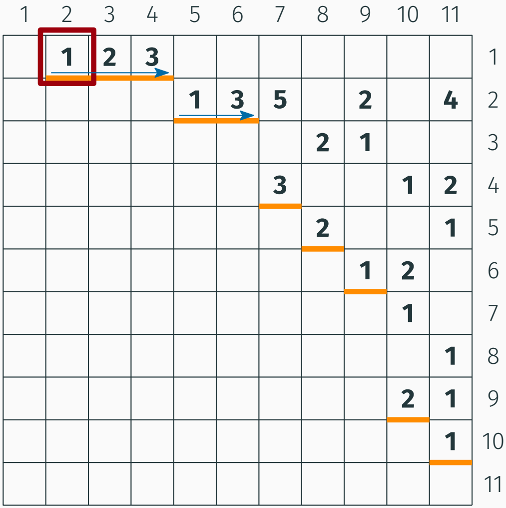
Matrix encoding

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

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

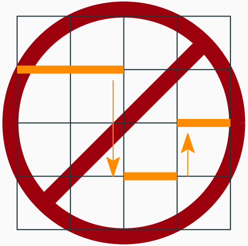
Matrix encoding

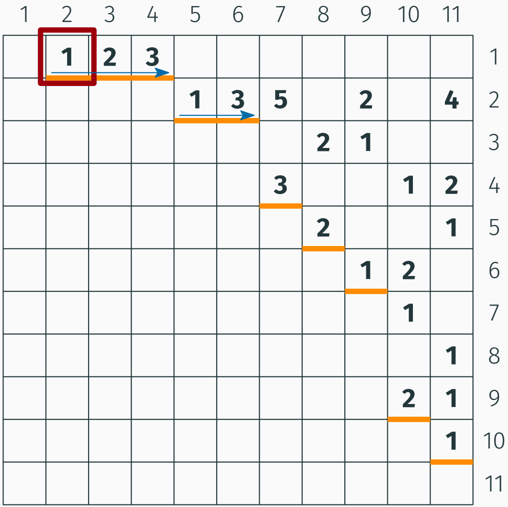
1. strict upper triangular matrix;
2. there is an element at $(1, 2)$;
3. increasing numbers above orange lines;



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.





Asymptotic result

Number of mono-source DOAGs

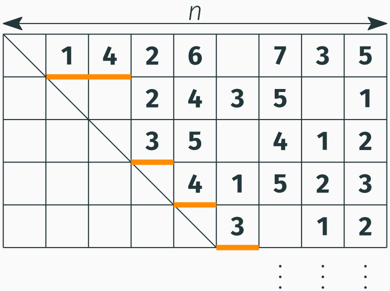
$$D_n \underset{n\to\infty}{\sim} \frac{c}{\sqrt{n}} e^{n-1} \text{¡}n-1!$$

for $c \approx 0.30256$ *and where* $\text{¡}m! = \prod_{k=1}^{m} k!$.

Number of mono-source DOAGs

$$D_n \underset{n\to\infty}{\sim} \frac{c}{\sqrt{n}}\, e^{n-1}\, \text{¡}n-1!$$

for $c \approx 0.30256$ *and where* $\text{¡}m! = \prod_{k=1}^{m} k!$.

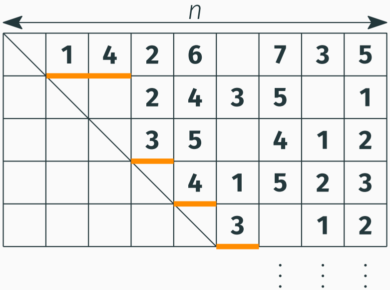


Asymptotic result

Number of mono-source DOAGs

$$D_n \underset{n\to\infty}{\sim} \frac{c}{\sqrt{n}}\, e^{n-1}\, ¡n-1!$$

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



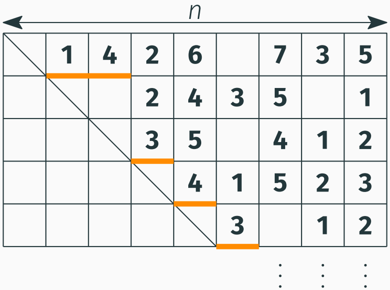
$\approx (n-1)!$ possible rows

Asymptotic result

Number of mono-source DOAGs

$$D_n \underset{n\to\infty}{\sim} \frac{c}{\sqrt{n}}\, e^{n-1}\ \text{¡}n-1!$$

for $c \approx 0.30256$ *and where* $\text{¡}m! = \prod_{k=1}^{m} k!$.



$\approx (n-1)!$ possible rows

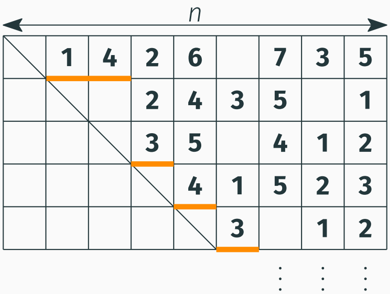
$\approx (n-2)!$ possible rows

Asymptotic result

Number of mono-source DOAGs

$$D_n \underset{n\to\infty}{\sim} \frac{c}{\sqrt{n}} e^{n-1} \text{¡}n-1!$$

for $c \approx 0.30256$ *and where* $\text{¡}m! = \prod_{k=1}^{m} k!$.



$\approx (n-1)!$ possible rows

$\approx (n-2)!$ possible rows

$\approx (n-3)!$ possible rows

$\approx (n-4)!$ possible rows

etc

$\Rightarrow \text{¡}n-1!$

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

6	**1**		**5**		**2**	**4**		**3**	$=$	**6**	**1**	**5**	**2**	**4**	**3**	$\star$			

Proof sketch (1/3)

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

6	1		5		2	4		3

$=$

6	1	5	2	4	3

$\star$

$$\text{Variation} = \textsc{Seq}(\mathcal{Z}) \star \textsc{Set}(\mathcal{Z})$$

Proof sketch (1/3)

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

6	1		5		2	4		3	=	6	1	5	2	4	3	$\star$			

$$\text{Variation} = \textsc{Seq}(\mathcal{Z}) \star \textsc{Set}(\mathcal{Z})$$

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

Proof sketch (1/3)

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

6	1		5		2	4		3

=

6	1	5	2	4	3

$\star$

$$\text{Variation} = \textsc{Seq}(\mathcal{Z}) \star \textsc{Set}(\mathcal{Z})$$

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

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

Proof sketch (1/3)

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

6	1		5		2	4		3

=

6	1	5	2	4	3

$\star$

$$\text{Variation} = \textsc{Seq}(\mathcal{Z}) \star \textsc{Set}(\mathcal{Z})$$

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

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

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

Proof sketch (1/3)

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

6	**1**		**5**		**2**	**4**		**3**	=	**6**	**1**	**5**	**2**	**4**	**3**	$\star$			

$$\text{Variation} = \textsc{Seq}(\mathcal{Z}) \star \textsc{Set}(\mathcal{Z})$$

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

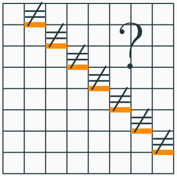
$$v_n = e \cdot n! - o(1)$$

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

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

Proof sketch (2/3)

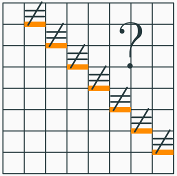
The plan: 1. Upper bound **2. Lower bound** 3. Bootstrapping

{DOAG matrices} $\supseteq$ 

(constraints are automatically satisfied)

Proof sketch (2/3)

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

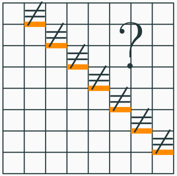
{DOAG matrices} $\supseteq$ 

(constraints are automatically satisfied)

$\#\, \boxed{\neq \mid ? \mid ? \mid ? \mid ? \mid ? \mid ? \mid ? \mid ?} = v_k - v_{k-1}$

Proof sketch (2/3)

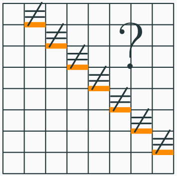
The plan: 1. Upper bound **2. Lower bound** 3. Bootstrapping

{DOAG matrices} $\supseteq$  (constraints are automatically satisfied)

$\#$  $= v_k - v_{k-1} = e \cdot k! \cdot \left(1 - \frac{1}{k} - o\left(\frac{1}{(k-1)!}\right)\right)$

Proof sketch (2/3)

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

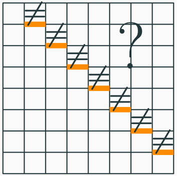
{DOAG matrices} $\supseteq$  (constraints are automatically satisfied)

$$\# \boxed{\neq \mid ? \mid ? \mid ? \mid ? \mid ? \mid ? \mid ? \mid ?} = v_k - v_{k-1} = e \cdot k! \cdot \left(1 - \frac{1}{k} - o\left(\frac{1}{(k-1)!}\right)\right)$$

$$D_n \geq e^{n-1} \text{ ¡}n - 1\text{!} \prod_{k=2}^{n-1} \left(\frac{k-1}{k} + o\left(\frac{1}{(k-1)!}\right)\right)$$

Proof sketch (2/3)

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

{DOAG matrices} $\supseteq$ 

(constraints are automatically satisfied)

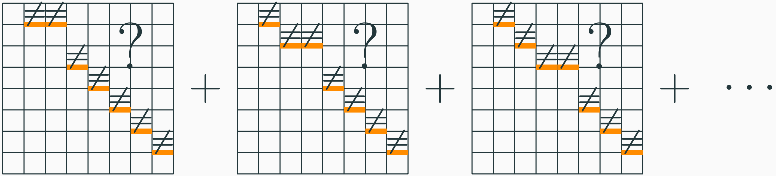
$$\#\,\boxed{\neq}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?}\boxed{?} = v_k - v_{k-1} = e \cdot k! \cdot \left(1 - \frac{1}{k} - o\left(\frac{1}{(k-1)!}\right)\right)$$

$$D_n \geq e^{n-1}\,\text{¡}n-1!\prod_{k=2}^{n-1}\left(\frac{k-1}{k} + o\left(\frac{1}{(k-1)!}\right)\right) \geq e^{n-1}\,\text{¡}n-1!\,\frac{A}{n} \quad \text{for some } A > 0$$

Proof sketch (2’/3)

The plan: 1. Upper bound **2’. Better lower bound** 3. Bootstrapping

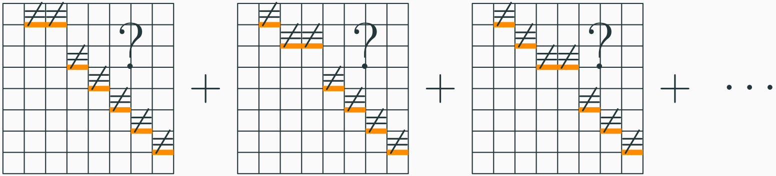
{DOAG matrices} $\supseteq$



Proof sketch (2'/3)

The plan: 1. Upper bound **2'. Better lower bound** 3. Bootstrapping

{DOAG matrices} ⊇

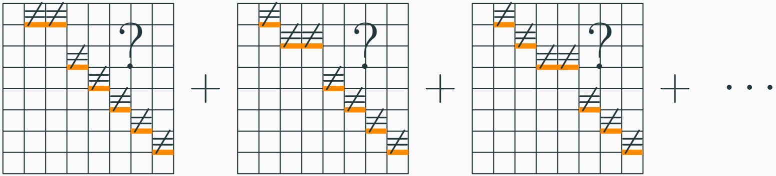


$$D_n \geq \frac{A' \cdot \ln(n)}{n} e^{n-1} \text{¡}n - 1!$$

Proof sketch (2’/3)

The plan: 1. Upper bound **2’. Better lower bound** 3. Bootstrapping

{DOAG matrices} $\supseteq$



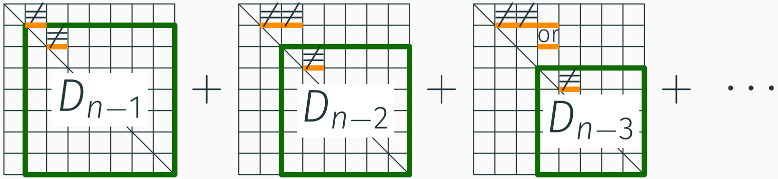
$$D_n \geq \frac{A' \cdot \ln(n)}{n}\, e^{n-1}\, \text{¡}n-1!$$

$$P_n = \frac{D_n}{e^{n-1}\, \text{¡}n-1!} \quad \Rightarrow \quad \frac{A' \cdot \ln(n)}{n} \leq P_n \leq 1$$

Proof sketch (3/3)

The plan: 1. Upper bound 2'. Better lower bound **3. Bootstrapping**

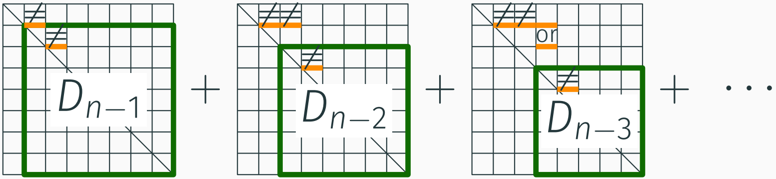
{DOAG matrices} =



Proof sketch (3/3)

The plan: 1. Upper bound 2'. Better lower bound **3. Bootstrapping**

$\{$DOAG matrices$\} =$

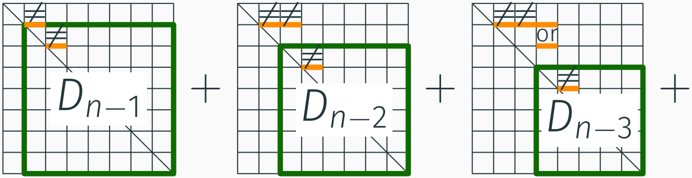


$$D_n = (v_{n-1} - v_{n-2})D_{n-1} + \frac{1}{2}(v_{n-1} - 2v_{n-2} + v_{n-3})v_{n-3}D_{n-2} + \cdots$$

Proof sketch (3/3)

The plan: 1. Upper bound 2'. Better lower bound **3. Bootstrapping**

$\{\text{DOAG matrices}\} =$



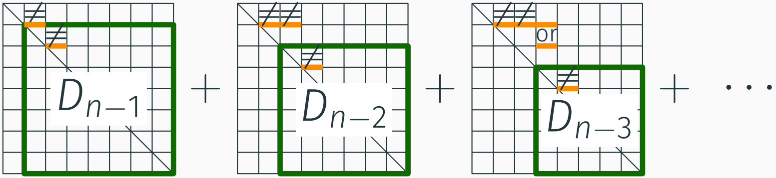
$+ \cdots$

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

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

Proof sketch (3/3)

The plan: 1. Upper bound 2'. Better lower bound **3. Bootstrapping**

{DOAG matrices} = 

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

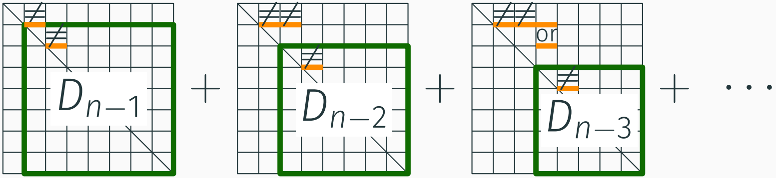
$$P_n = \left(1 - \frac{1}{n-1}\right) P_{n-1} + \frac{1}{2(n-2)} \left(1 - \frac{2}{n-1} + \frac{1}{(n-1)(n-2)}\right) P_{n-2} + \cdots$$

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

Proof sketch (3/3)

The plan: 1. Upper bound 2'. Better lower bound **3. Bootstrapping**

{DOAG matrices} =



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

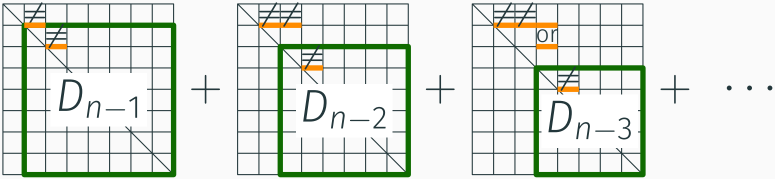
$$P_n = \left(1 - \frac{1}{n-1}\right)P_{n-1} + \frac{1}{2(n-2)}\left(1 - \frac{2}{n-1} + \frac{1}{(n-1)(n-2)}\right)P_{n-2} + \cdots$$

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

Proof sketch (3/3)

The plan: 1. Upper bound 2'. Better lower bound **3. Bootstrapping**

{DOAG matrices} =



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

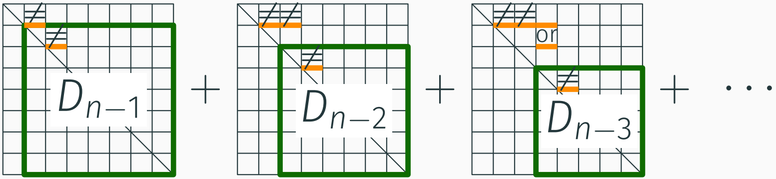
$$P_n = \left(1 - \frac{1}{n-1}\right) P_{n-1} + \frac{1}{2(n-2)} \left(1 - \frac{2}{n-1} + \frac{1}{(n-1)(n-2)}\right) P_{n-2} + \cdots$$

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

Proof sketch (3/3)

The plan: 1. Upper bound 2'. Better lower bound **3. Bootstrapping**

{DOAG matrices} =



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

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

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

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\textit{collections of variations of length } 1, 2, \ldots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\textit{collections of variations of length } 1, 2, \ldots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw collections of variations until they correspond to a valid DOAG matrix.

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\textit{collections of variations of length } 1, 2, \ldots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw collections of variations until they correspond to a valid DOAG matrix.

(Naive) complexity: $\#\text{rejections} \times \text{Cost}(\text{one generation})$

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\textit{collections of variations of length } 1, 2, \ldots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw collections of variations until they correspond to a valid DOAG matrix.

(Naive) complexity: $\#\text{rejections} \times \text{Cost}(\text{one generation})$

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

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\textit{collections of variations of length } 1, 2, \ldots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw collections of variations until they correspond to a valid DOAG matrix.

(Naive) complexity: $O(\sqrt{n} \cdot n^2 \ln(n))$ random bits

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

Random sampling again!

Corollary

$$\frac{D_n}{\#\{\textit{collections of variations of length } 1, 2, \ldots, n-1\}} \sim c \cdot n^{-\frac{1}{2}}$$

Rejection sampling: draw collections of variations until they correspond to a valid DOAG matrix.

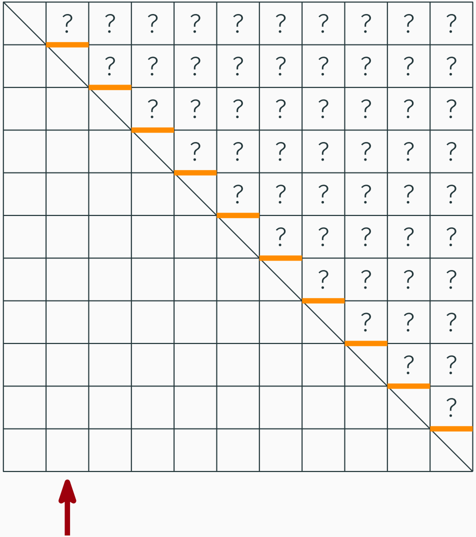
(Naive) complexity: $O(\sqrt{n} \cdot n^2 \ln(n))$ random bits

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

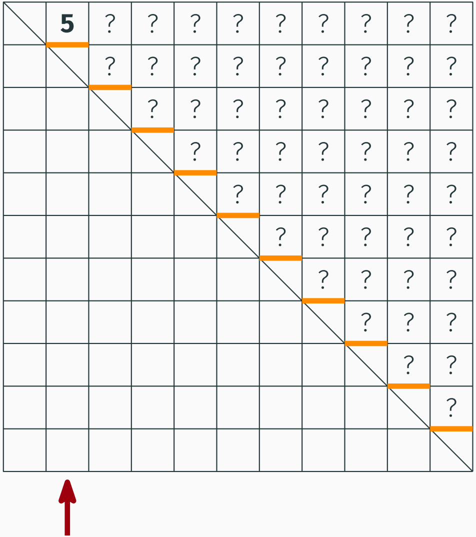
Better complexity:

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

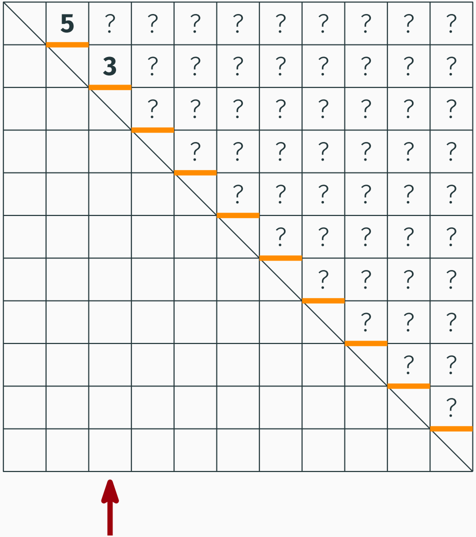
Early rejection



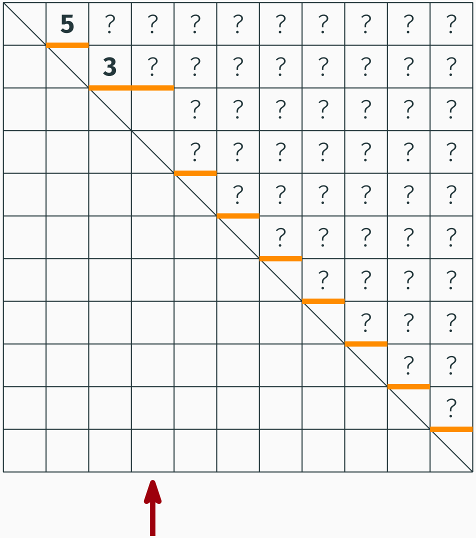
Early rejection



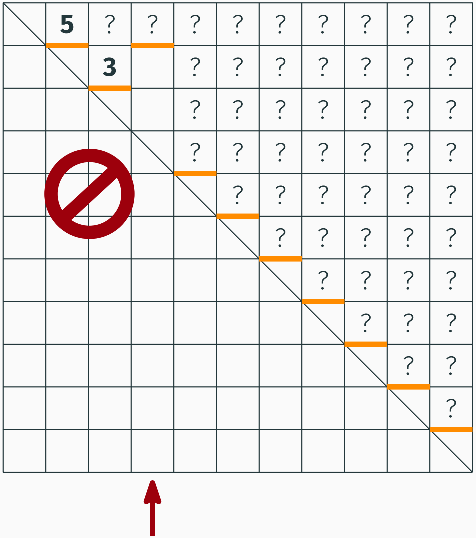
Early rejection



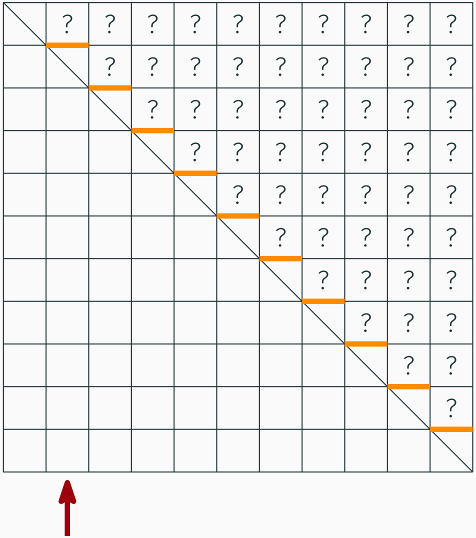
Early rejection



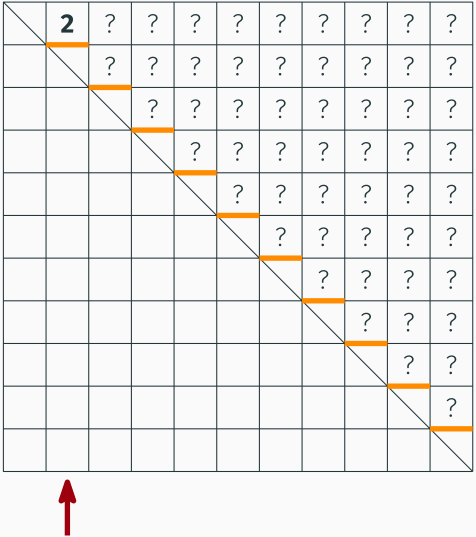
Early rejection



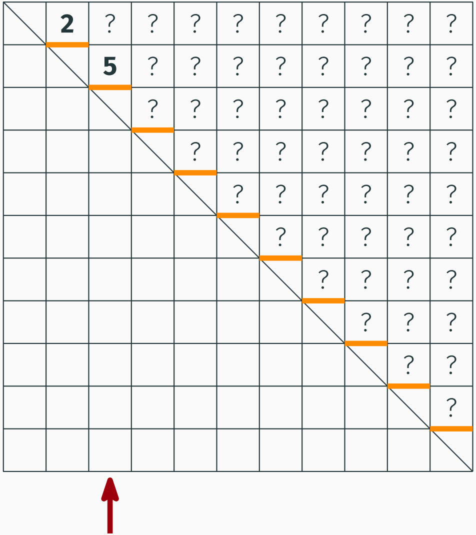
Early rejection



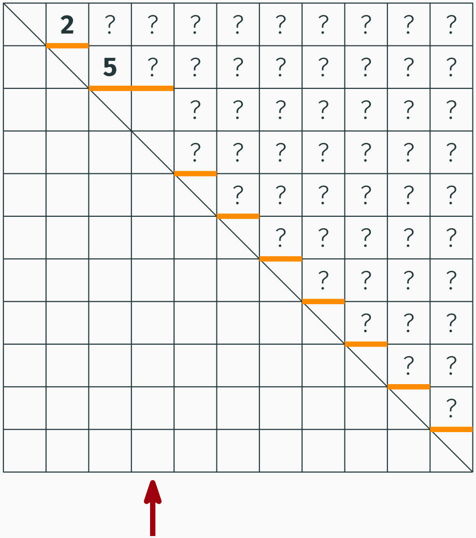
Early rejection



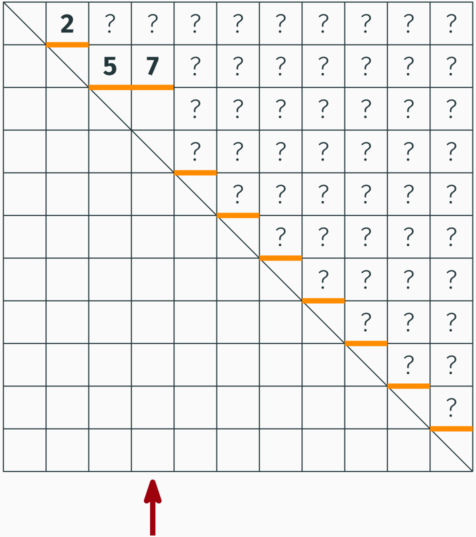
Early rejection



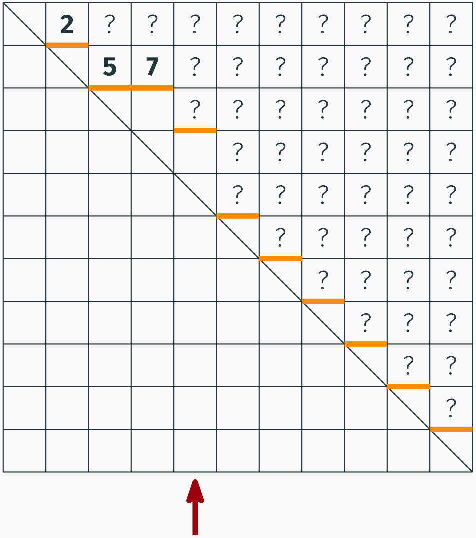
Early rejection



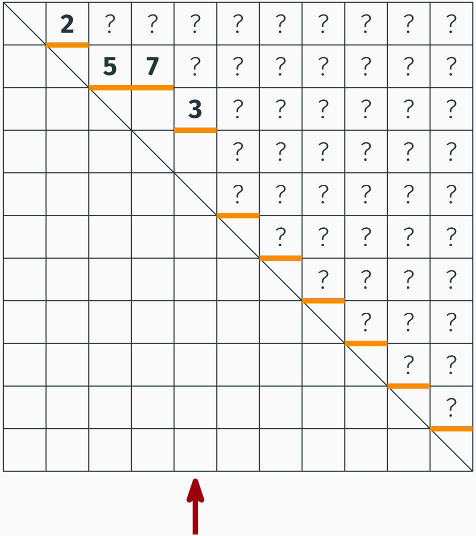
Early rejection



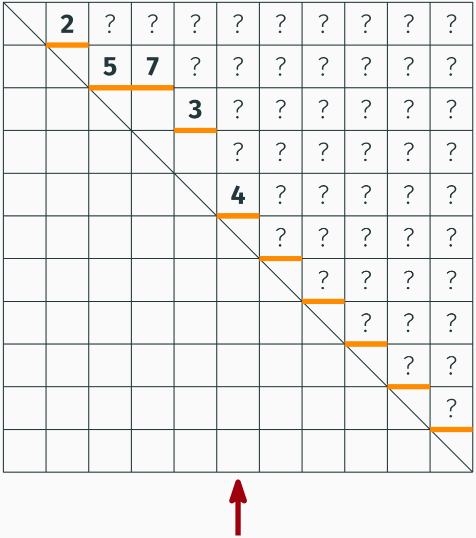
Early rejection



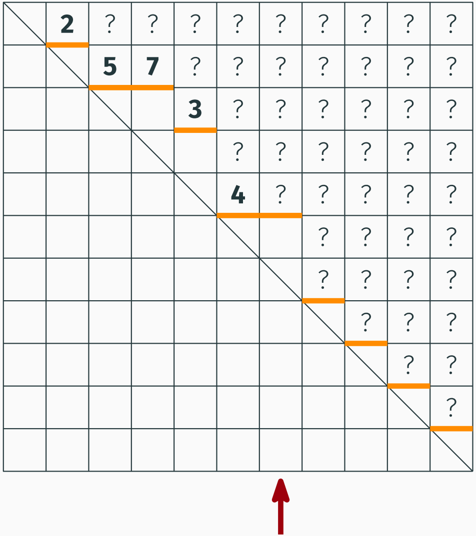
Early rejection



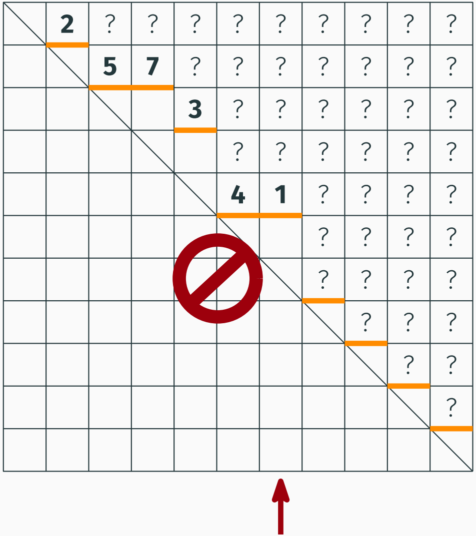
Early rejection



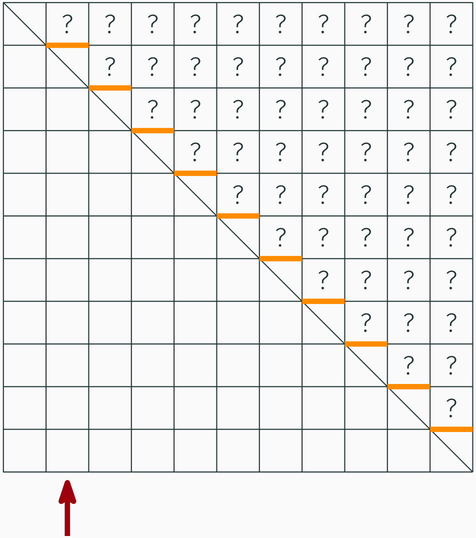
Early rejection



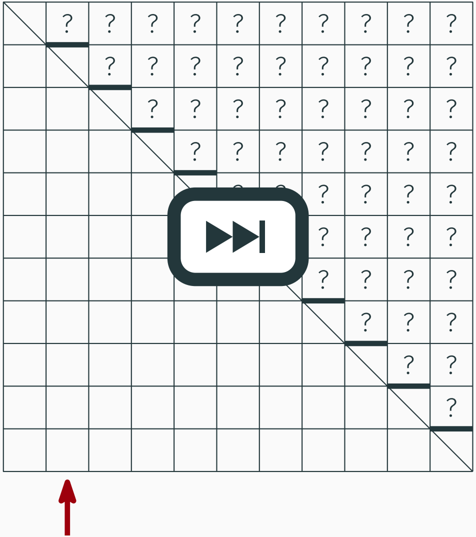
Early rejection



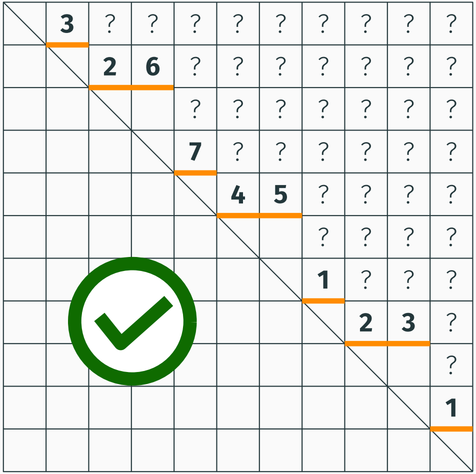
Early rejection



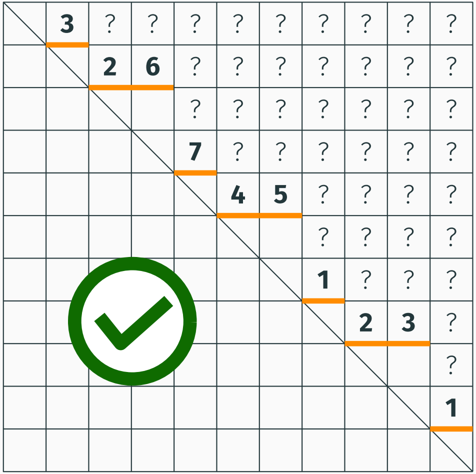
Early rejection



Early rejection



Early rejection



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

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

Perspectives

- Law of the number of edges?

Perspectives

- 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

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: https://doi.org/10.1016/j.jcta.2020.105306. URL: https://www.sciencedirect.com/science/article/pii/S0097316520300984.

[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.

References II

[GPV21] Antoine Genitrini, Martin Pépin, and Alfredo Viola. "Unlabelled ordered DAGs and labelled DAGs: constructive enumeration and uniform random sampling". In: *XI Latin and American Algorithms, Graphs and Optimization Symposium*. Eslevier. 2021.

[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`. URL: `https://doi.org/10.1016/S1571-0653(04)00394-4`.

References III

[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.