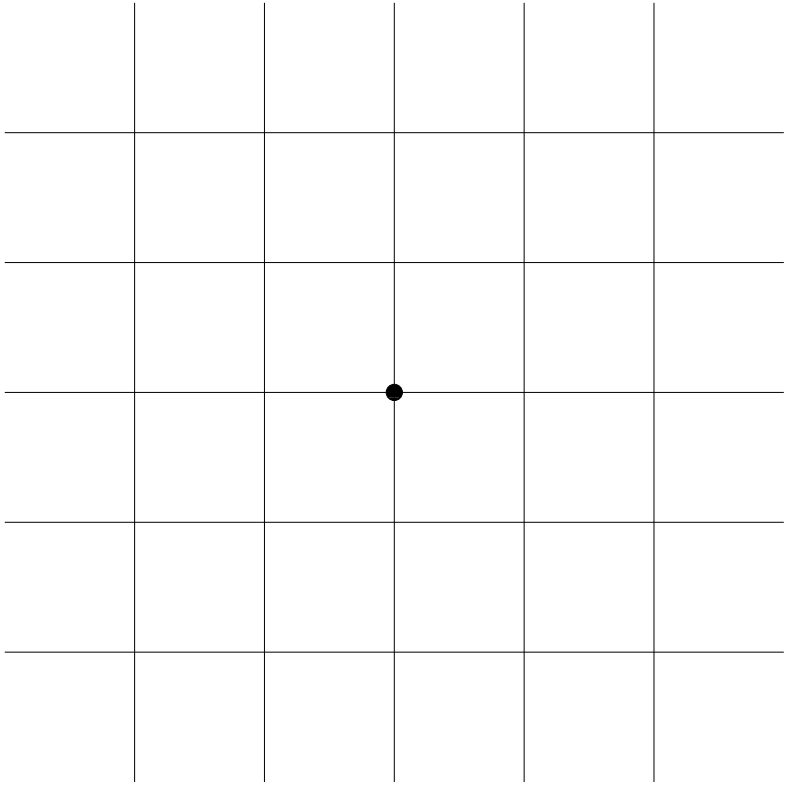
Science Atlantic
Wolfville

Random graphs and random trees

Louigi Addario-Berry (McGill)

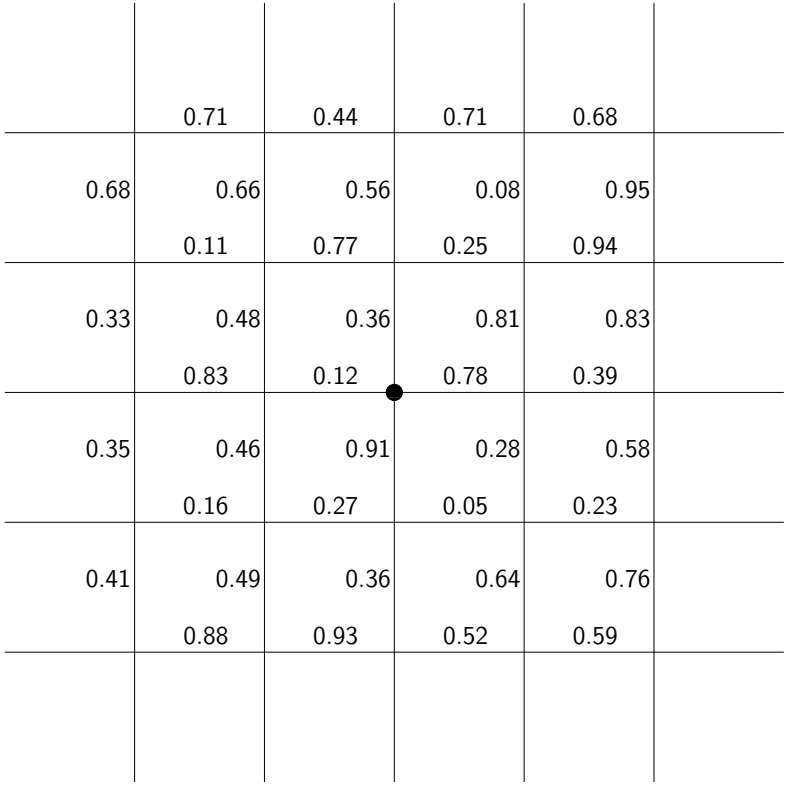
I. Prehistory. A) Euclidean percolation

- Example: Nearest neighbour graph on $\mathbb{Z}^2$.



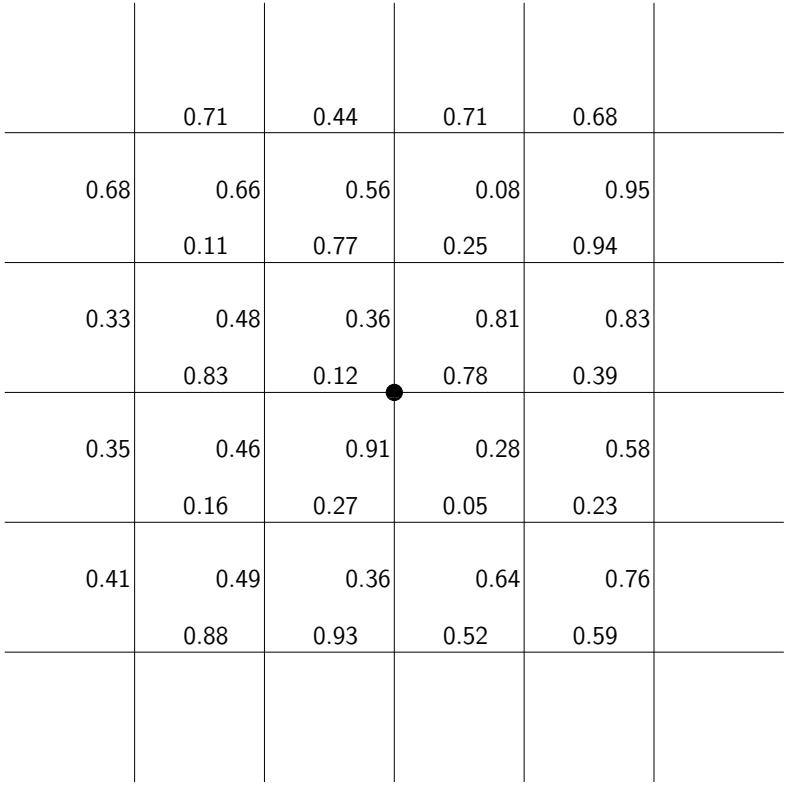
I. Prehistory. A) Euclidean percolation

- Example: Nearest neighbour graph on $\mathbb{Z}^2$.
- Edges augmented with i.i.d. Uniform $[0, 1]$ r.v.s.



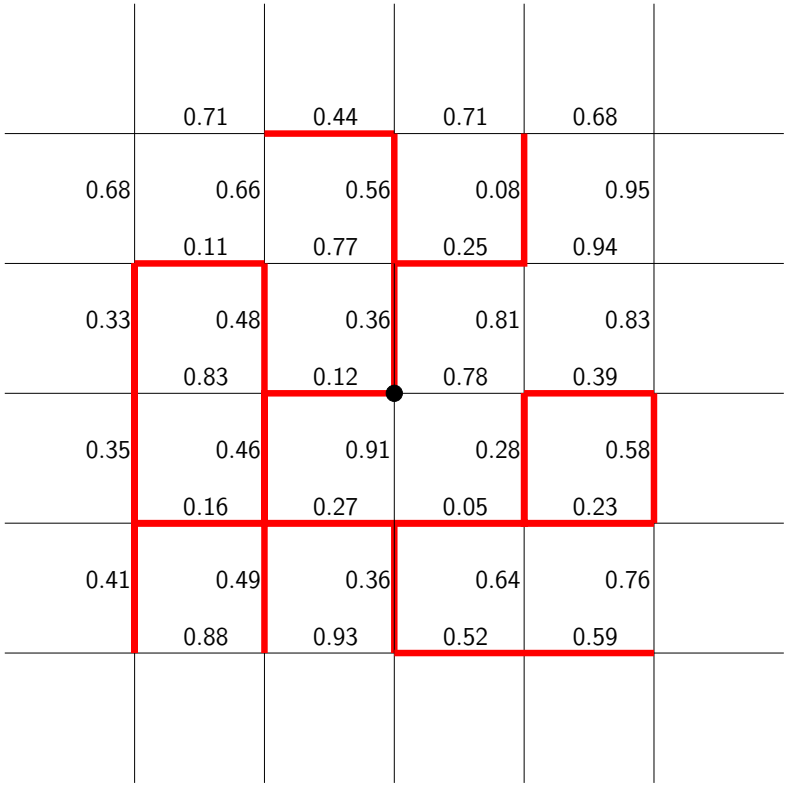
I. Prehistory. A) Euclidean percolation

- Example: Nearest neighbour graph on $\mathbb{Z}^2$.
- Edges augmented with i.i.d. Uniform $[0,1]$ r.v.s.
- Fix p, $0 < p < 1$, and keep only edges of weight at most p. (Call the result $\mathbb{Z}^2_p$.)



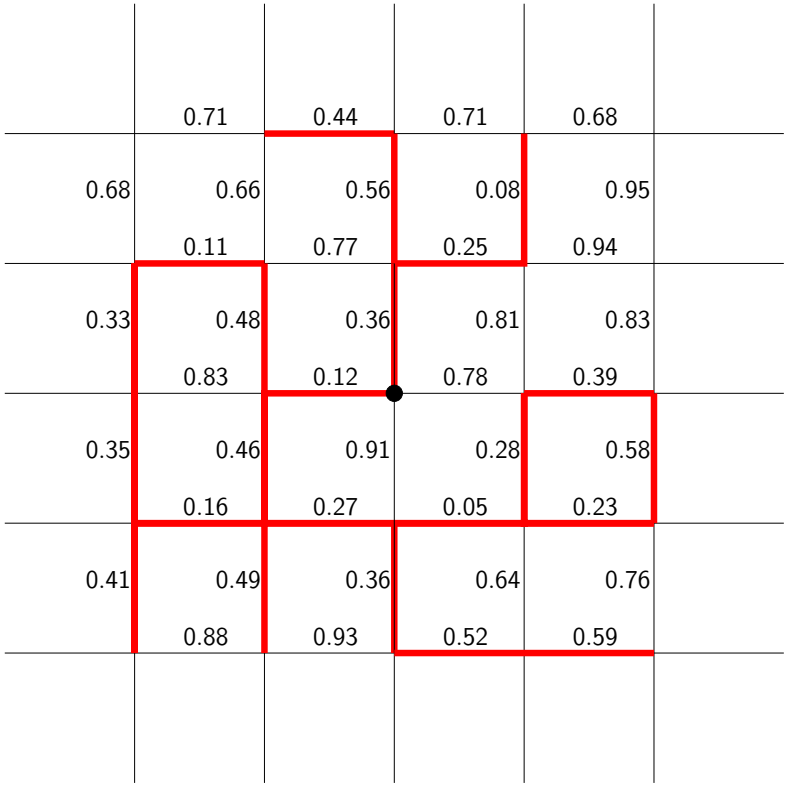
I. Prehistory. A) Euclidean percolation

- Example: Nearest neighbour graph on $\mathbb{Z}^2$.
- Edges augmented with i.i.d. Uniform $[0, 1]$ r.v.s.
- Fix p, $0 < p < 1$, and keep only edges of weight at most p. (Call the result $\mathbb{Z}^2_p$.)



I. Prehistory. A) Euclidean percolation

- Example: Nearest neighbour graph on $\mathbb{Z}^2$.
- Edges augmented with i.i.d. Uniform $[0, 1]$ r.v.s.
- Fix p, $0 < p < 1$, and keep only edges of weight at most p. (Call the result $\mathbb{Z}^2_p$.)
- Here $p = 0.6$.



I. Prehistory. A) Euclidean percolation

- A phase transition occurs at $p = 1/2$.

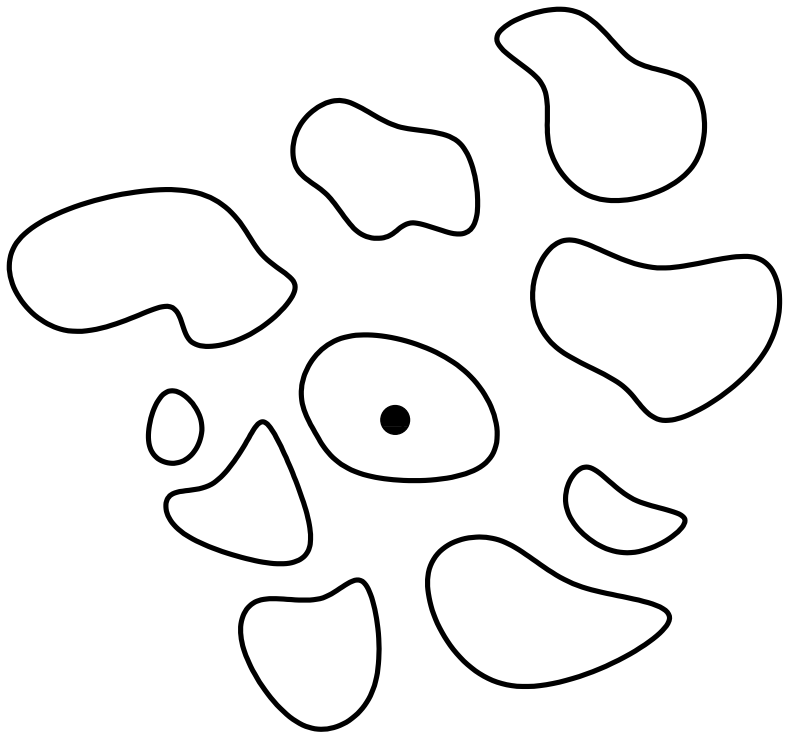
I. Prehistory. A) Euclidean percolation

- A phase transition occurs at $p = 1/2$.
- Write $\mathcal{C}_p$ for the component of $\mathbb{Z}_p^2$ containing the origin.

I. Prehistory. A) Euclidean percolation

- A phase transition occurs at $p = 1/2$.
- Write $\mathcal{C}_p$ for the component of $\mathbb{Z}_p^2$ containing the origin.
- $p < 1/2$: all components are finite, and $\mathbb{E}|\mathcal{C}_p| < \infty$.

  (Subcritical case)

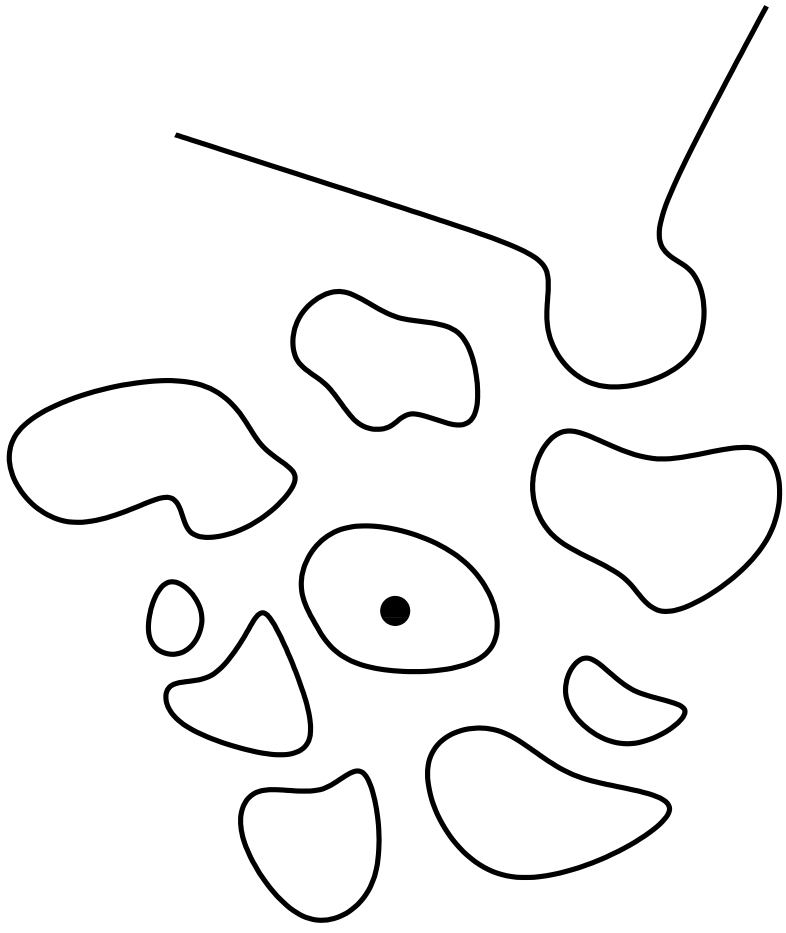


$p < \frac{1}{2}$

I. Prehistory. A) Euclidean percolation

- A phase transition occurs at $p = 1/2$.
- Write $\mathcal{C}_p$ for the component of $\mathbb{Z}_p^2$ containing the origin.
- $p < 1/2$: all components are finite, and $\mathbb{E}|\mathcal{C}_p| < \infty$.

  (Subcritical case)
- $p > 1/2$: with probability 1 there is a unique infinite component, which contains a positive proportion of the vertices. (Supercritical case)



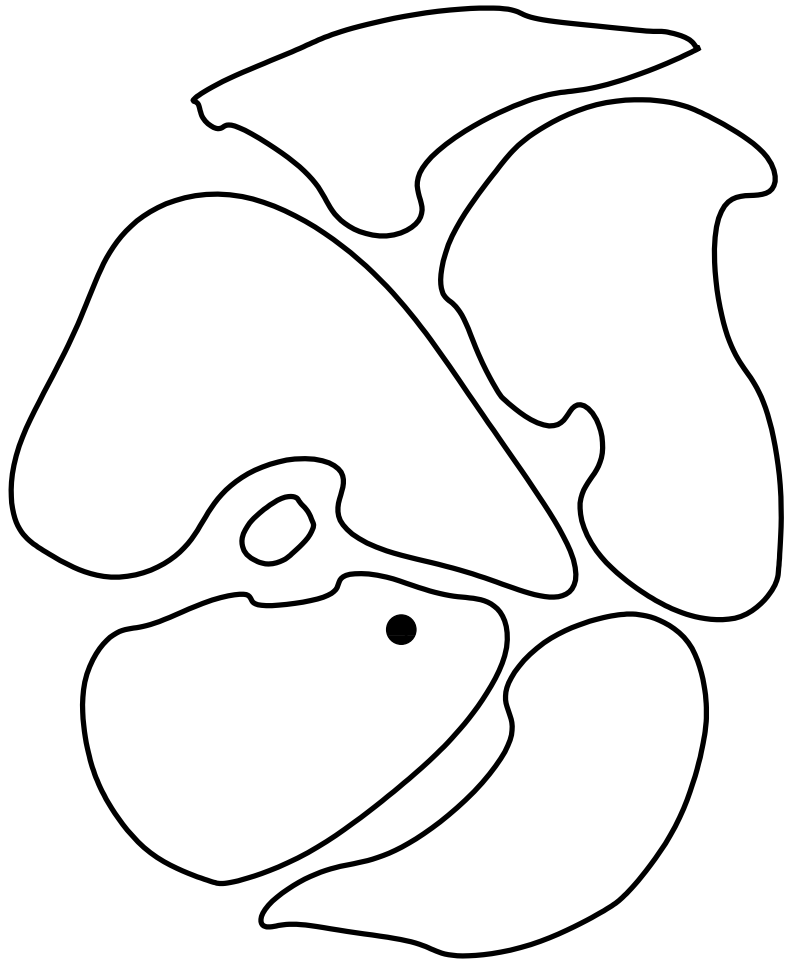
$p > \frac{1}{2}$

I. Prehistory. A) Euclidean percolation

- A phase transition occurs at $p = 1/2$.
- Write $\mathcal{C}_p$ for the component of $\mathbb{Z}_p^2$ containing the origin.
- $p < 1/2$: all components are finite, and $\mathbb{E}|\mathcal{C}_p| < \infty$.

  (Subcritical case)
- $p > 1/2$: with probability 1 there is a unique infinite component, which contains a positive proportion of the vertices. (Supercritical case)
- $p = 1/2$: there is no infinite component* but $\mathbb{E}|\mathcal{C}_p| = \infty$.

  (Critical case)



$p = \frac{1}{2}$

I. Prehistory. A) Euclidean percolation

- Last 20 years: breakthroughs on understanding *component interfaces* in 2D critical percolation.

I. Prehistory. A) Euclidean percolation

- Last 20 years: breakthroughs on understanding *component interfaces* in 2D critical percolation.
- Key role: Schramm-Loewner Evolution SLE(σ): Loewner evolution driven by Brownian motion with diffusivity σ.

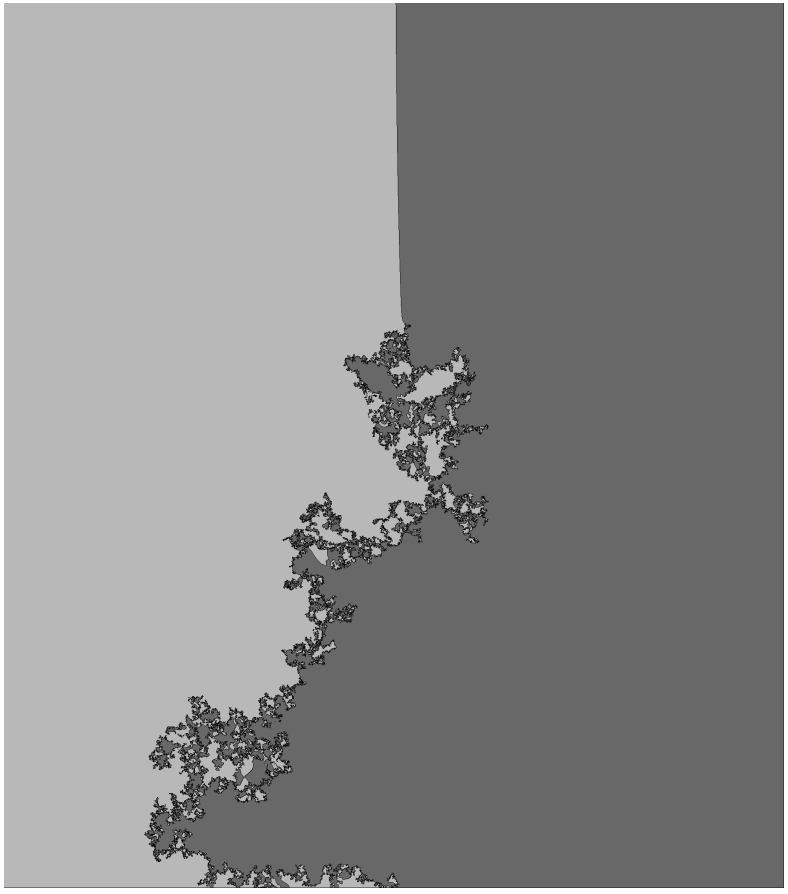


Image by Vincent Beffara

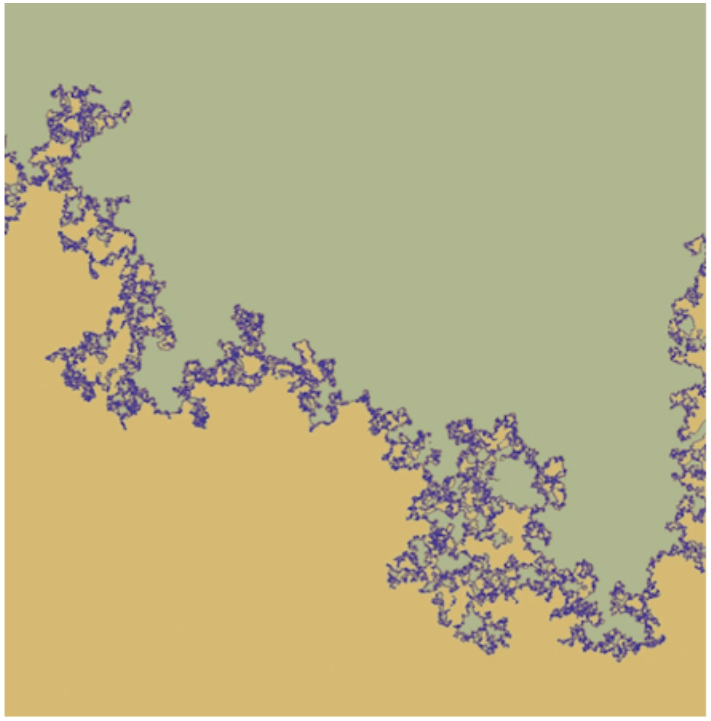
I. Prehistory. A) Euclidean percolation

- Last 20 years: breakthroughs on understanding *component interfaces* in 2D critical percolation.
- Key role: Schramm-Loewner Evolution SLE(σ): Loewner evolution driven by Brownian motion with diffusivity σ.
- Percolation interfaces described by SLE(6).

Animation by Jason Miller

I. Prehistory. A) Euclidean percolation

- Last 20 years: breakthroughs on understanding *component interfaces* in 2D critical percolation.
- Key role: Schramm-Loewner Evolution SLE(σ): Loewner evolution driven by Brownian motion with diffusivity σ.
- Percolation interfaces described by SLE(6).
- Universality: choice of lattice should not matter



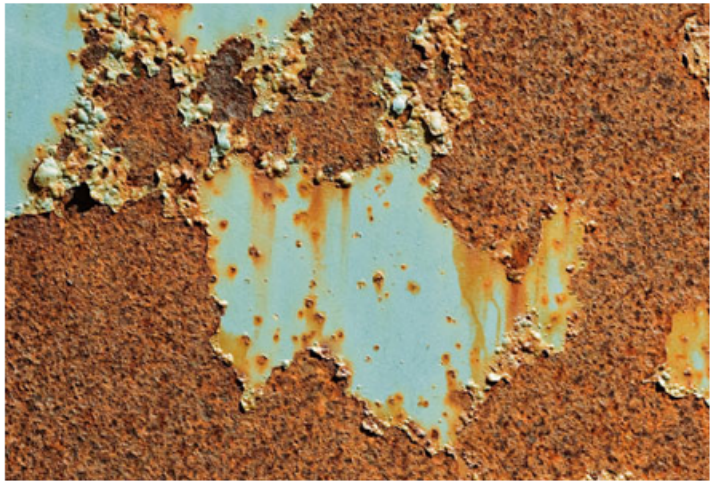
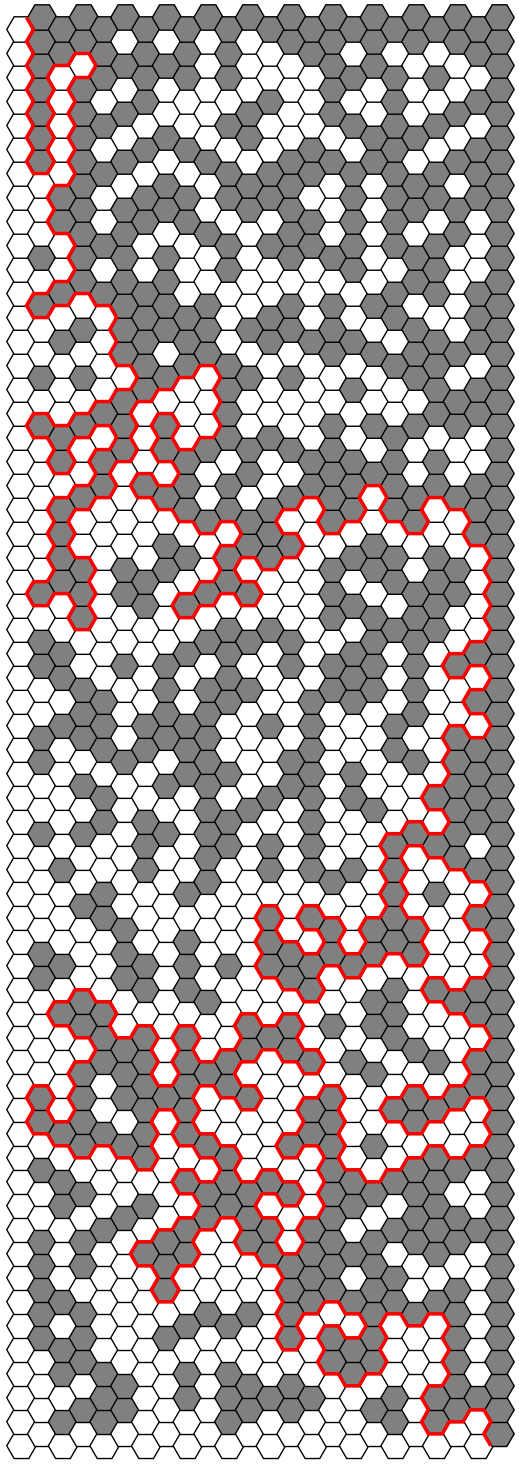


Image from `http://images.math.cnrs.fr/La-percolation-jeu-de-pavages-aleatoires.html`.

I. Prehistory. A) Euclidean percolation

- Last 20 years: breakthroughs on understanding *component interfaces* in 2D critical percolation.
- Key role: Schramm-Loewner Evolution SLE(σ): Loewner evolution driven by Brownian motion with diffusivity σ.
- Percolation interfaces described by SLE(6).
- Universality: choice of lattice should not matter
- But so far we only know the SLE(6) story for the hexagonal lattice, which has special "conformal" structure.



I. Prehistory. A) Euclidean Percolation

(Some open problems.)

- Proving that critical percolation on $\mathbb{Z}^2$ behaves like critical percolation on the hexagonal lattice would be a big deal.

I. Prehistory. A) Euclidean Percolation

(Some open problems.)

- Proving that critical percolation on $\mathbb{Z}^2$ behaves like critical percolation on the hexagonal lattice would be a big deal.
- In all dimensions $d \geq 2$ there is a critical probability $p_c = p_c(\mathbb{Z}^d) \in (0,1)$ such that for $p < p_c$ there is no infinite component and for $p > p_c$ there is an infinite component.

I. Prehistory. A) Euclidean Percolation

(Some open problems.)

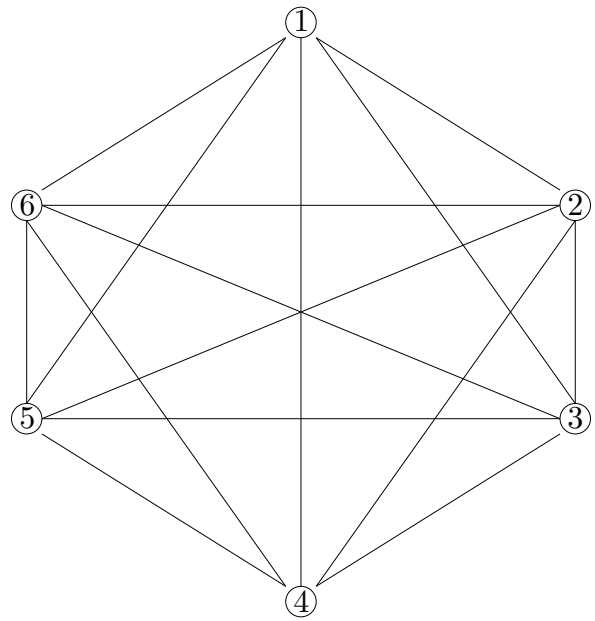
- Proving that critical percolation on $\mathbb{Z}^2$ behaves like critical percolation on the hexagonal lattice would be a big deal.
- In all dimensions $d \geq 2$ there is a critical probability $p_c = p_c(\mathbb{Z}^d) \in (0,1)$ such that for $p < p_c$ there is no infinite component and for $p > p_c$ there is an infinite component.
- Probably the most famous open problem in probability: **Prove (or disprove) that**

$$\mathbb{P}(\mathbb{Z}^d_{p_c} \text{ contains an infinite component}) = 0\,.$$

I. Prehistory. B) The Erdős–Rényi random graph

$G(n,p)$: Take vertices labelled $1, 2, \ldots, n$ and fix $p \in [0,1]$. For each pair of vertices, put an edge between them with probability p, independently for different pairs.

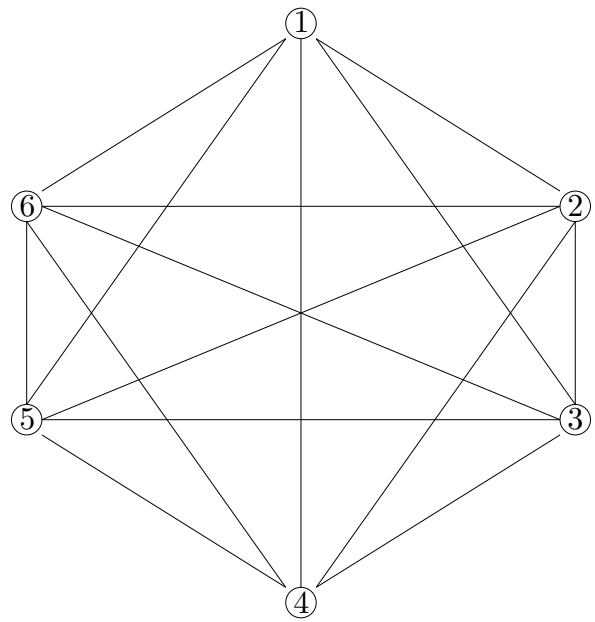
This is percolation on the complete graph at parameter p.



I. Prehistory. B) The Erdős–Rényi random graph

$G(n,p)$: Take vertices labelled $1, 2, \ldots, n$ and fix $p \in [0,1]$. For each pair of vertices, put an edge between them with probability p, independently for different pairs.

This is percolation on the complete graph at parameter p.

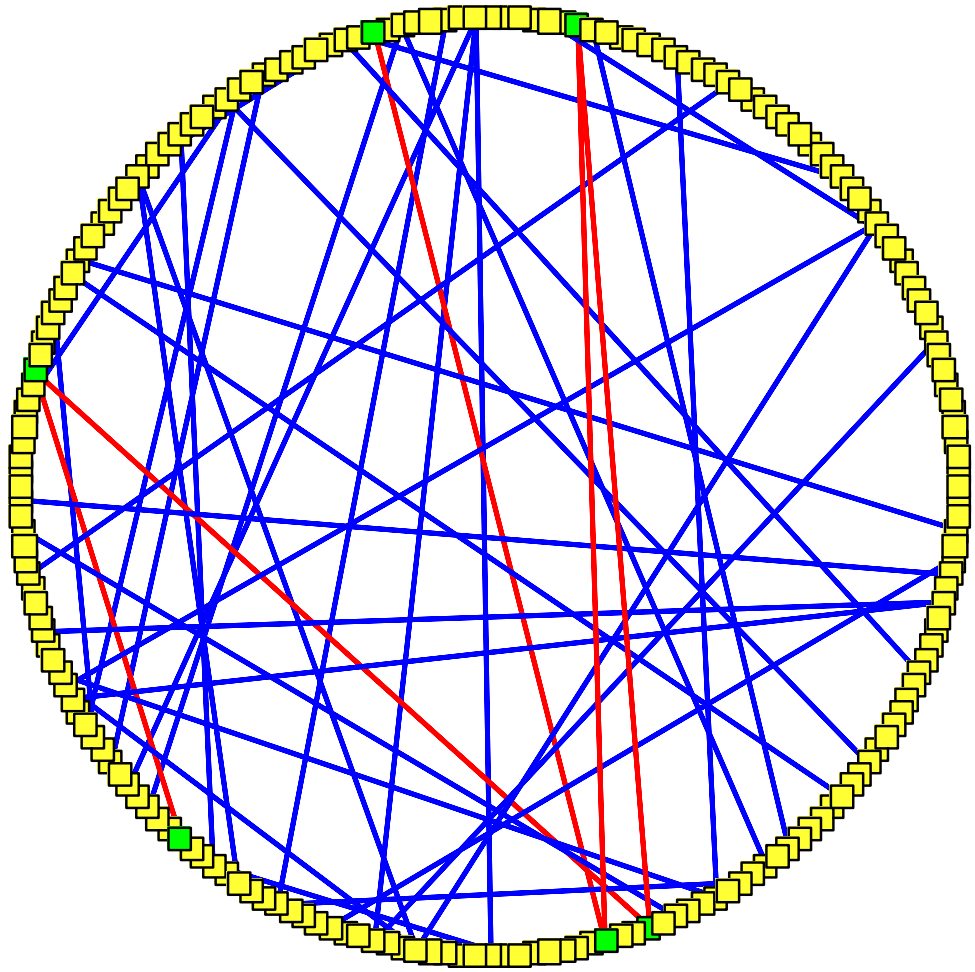


We will consider the case where $p = c/n$ for a constant $c > 0$, so that a single vertex has a Binomial$(n-1, c/n)$ number of neighbours, with mean approximately c for large n.

II. Erdős–Rényi random graph

Let $p = c/n$ and consider the largest component (vertices in green, edges in red).

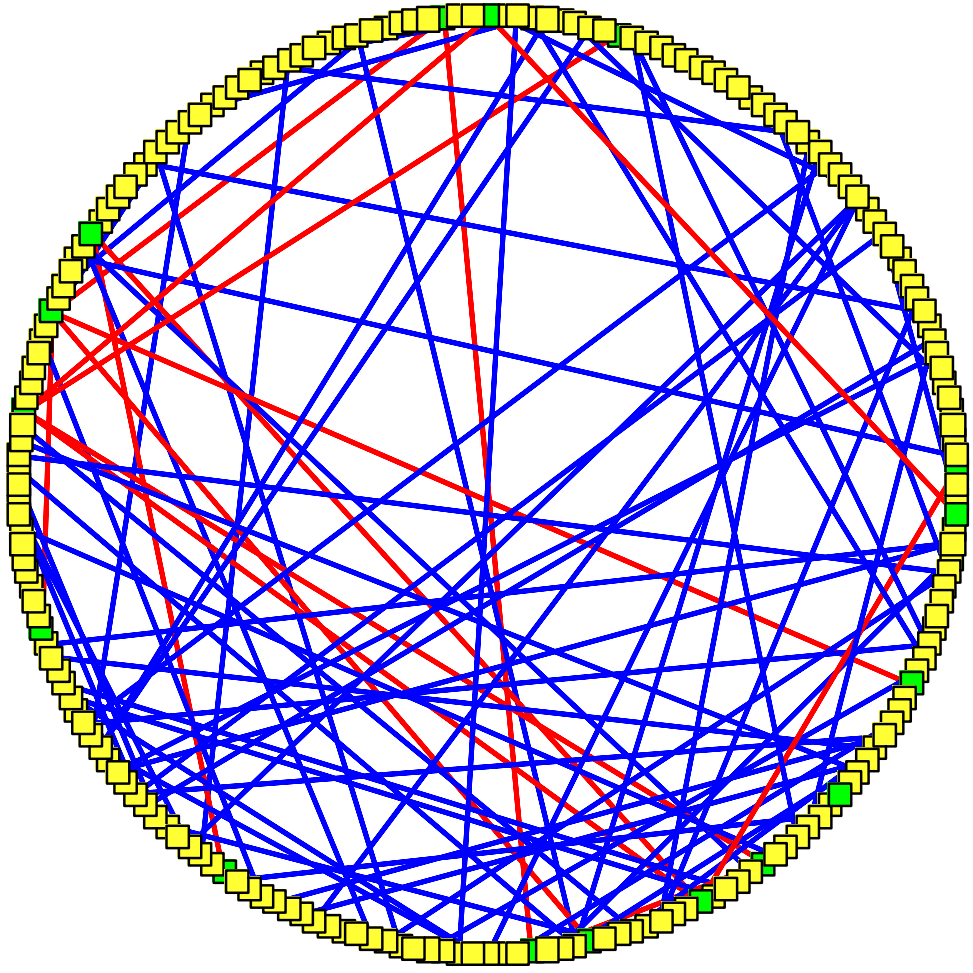
$n = 200,\ c = 0.4$



II. Erdős–Rényi random graph

Let $p = c/n$ and consider the largest component (vertices in green, edges in red).

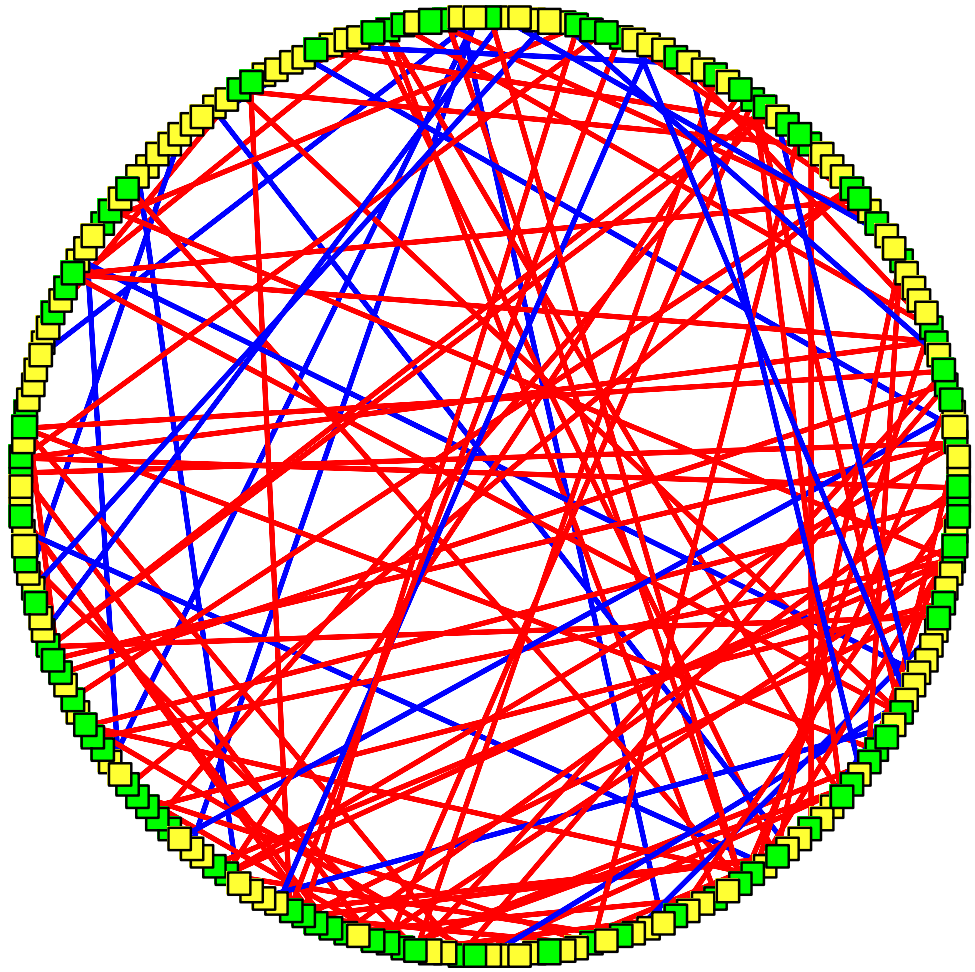
$n = 200,\ c = 0.8$



II. Erdős–Rényi random graph

Let $p = c/n$ and consider the largest component (vertices in green, edges in red).

$n = 200,\ c = 1.2$



II. The phase transition

Theorem. (Erdős and Rényi (1960))
For $G(n, c/n)$ the following statements hold with probability tending to 1 as $n \to \infty$.

- If $c < 1$, the largest component contains $O(\log n)$ vertices.
- If $c > 1$, the largest component contains $\Theta(n)$ vertices and the rest contain $O(\log n)$ vertices each.

$p = 1/n$ is the critical point. (Erdős-Rényi analogue of critical percolation; expect fractal behaviour.)

II. The phase transition

Theorem. (Erdős and Rényi (1960))
For $G(n, c/n)$ the following statements hold with probability tending to 1 as $n \to \infty$.

- If $c < 1$, the largest component contains $O(\log n)$ vertices.
- If $c > 1$, the largest component contains $\Theta(n)$ vertices and the rest contain $O(\log n)$ vertices each.

$p = 1/n$ is the critical point. (Erdős-Rényi analogue of critical percolation; expect fractal behaviour.)

I learned about the Erdős-Rényi phase transition in (I think) January 2002, in the first year of my M.Sc., and have been studying it on-and-off since then.

II. The phase transition

Theorem. (Erdős and Rényi (1960))
For $G(n, c/n)$ the following statements hold with probability tending to 1 as $n \to \infty$.

- If $c < 1$, the largest component contains $O(\log n)$ vertices.
- If $c > 1$, the largest component contains $\Theta(n)$ vertices and the rest contain $O(\log n)$ vertices each.

$p = 1/n$ is the critical point. (Erdős-Rényi analogue of critical percolation; expect fractal behaviour.)

I learned about the Erdős-Rényi phase transition in (I think) January 2002, in the first year of my M.Sc., and have been studying it on-and-off since then.

In 2007 I started studying the structure of the critical random graph viewed as a *random metric space* (distance given by graph distance). Prototype question: what is the expected distance between two uniformly random vertices in the largest component?

III. Convergence of metric measure spaces

Let $(M_n, d_n, \mu_n)_{n\geq 1}$ be a sequence of (random) metric spaces (M_n, d_n) each endowed with a probability measure μ_n on (the Borel sets of) M_n.

III. Convergence of metric measure spaces

Let $(M_n, d_n, \mu_n)_{n\geq 1}$ be a sequence of (random) metric spaces (M_n, d_n) each endowed with a probability measure μ_n on (the Borel sets of) M_n.

We want to give a notion of convergence for such a sequence. In order to do this, we will sample points from M_n according to μ_n and then look at the matrix of pairwise distances between them.

III. Convergence of metric measure spaces

Let $(M_n, d_n, \mu_n)_{n\geq 1}$ be a sequence of (random) metric spaces (M_n, d_n) each endowed with a probability measure μ_n on (the Borel sets of) M_n.

We want to give a notion of convergence for such a sequence. In order to do this, we will sample points from M_n according to μ_n and then look at the matrix of pairwise distances between them.

Idea: as we select more and more points, we get better and better (finite) approximations to the true space (M_n, d_n). If each sequence of finite approximations converges, we say that the spaces do.

III. Convergence of metric measure spaces

Fix $k \geq 2$ and let $U_1^n, U_2^n, \ldots, U_k^n$ be independent random variables, each with distribution μ_n. For $1 \leq i, j \leq k$, let

$$D_{i,j}^{n,k} = d_n(U_i^n, U_j^n)$$

and let $D^{n,k} = (D_{i,j}^{n,k})_{1 \leq i,j \leq k}$ be the pairwise distance matrix.

III. Convergence of metric measure spaces

Fix $k \geq 2$ and let $U_1^n, U_2^n, \ldots, U_k^n$ be independent random variables, each with distribution μ_n. For $1 \leq i, j \leq k$, let

$$D_{i,j}^{n,k} = d_n(U_i^n, U_j^n)$$

and let $D^{n,k} = (D_{i,j}^{n,k})_{1 \leq i,j \leq k}$ be the pairwise distance matrix.

We will say

$$(M_n, d_n, \mu_n) \longrightarrow (M, d, \mu)$$

as $n \to \infty$ if, for each $k \geq 2$,

$$D^{n,k} \longrightarrow D^k$$

in distribution as $n \to \infty$.

III. Convergence of metric measure spaces

Fix $k \geq 2$ and let $U_1^n, U_2^n, \ldots, U_k^n$ be independent random variables, each with distribution μ_n. For $1 \leq i, j \leq k$, let

$$D_{i,j}^{n,k} = d_n(U_i^n, U_j^n)$$

and let $D^{n,k} = (D_{i,j}^{n,k})_{1 \leq i,j \leq k}$ be the pairwise distance matrix.

We will say

$$(M_n, d_n, \mu_n) \longrightarrow (M, d, \mu)$$

as $n \to \infty$ if, for each $k \geq 2$,

$$D^{n,k} \longrightarrow D^k$$

in distribution as $n \to \infty$. **N.B.** If the metric spaces (M_n, d_n, μ_n) are random then the matrices $D^{n,k}$ have "two levels" of randomness – this requires care to set up formally.

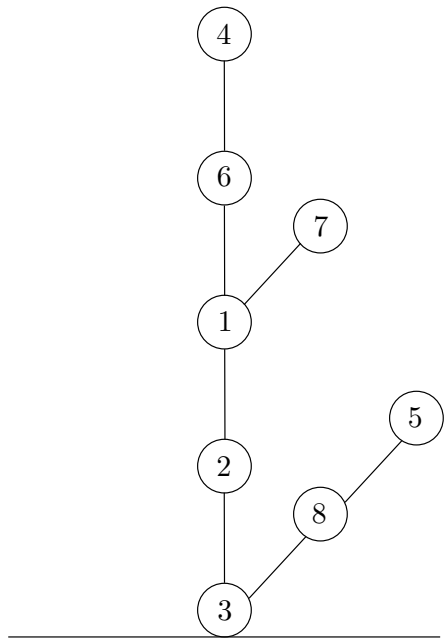
IV. A key example: random trees

Let $\mathbb{T}_n$ be the set of rooted trees with vertices labelled $1, 2, \ldots, n$.
Cayley's formula: $|\mathbb{T}_n| = n^{n-1}$.

IV. A key example: random trees

Let $\mathbb{T}_n$ be the set of rooted trees with vertices labelled $1, 2, \ldots, n$.
Cayley's formula: $|\mathbb{T}_n| = n^{n-1}$.

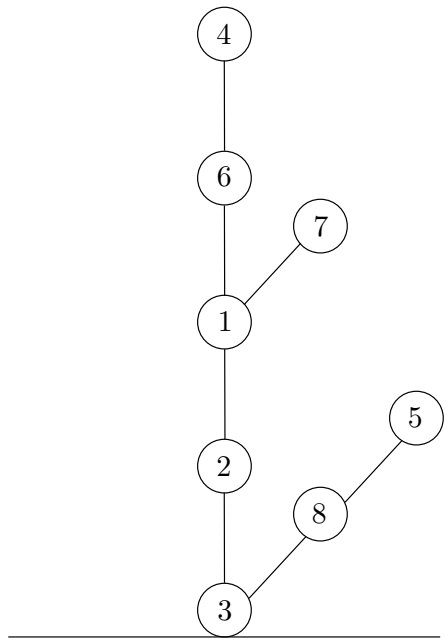
Choose a tree T_n uniformly at random from $\mathbb{T}_n$.



IV. A key example: random trees

Let $\mathbb{T}_n$ be the set of rooted trees with vertices labelled $1, 2, \ldots, n$.
Cayley's formula: $|\mathbb{T}_n| = n^{n-1}$.

Choose a tree T_n uniformly at random from $\mathbb{T}_n$.



What does the tree look like as n gets large?

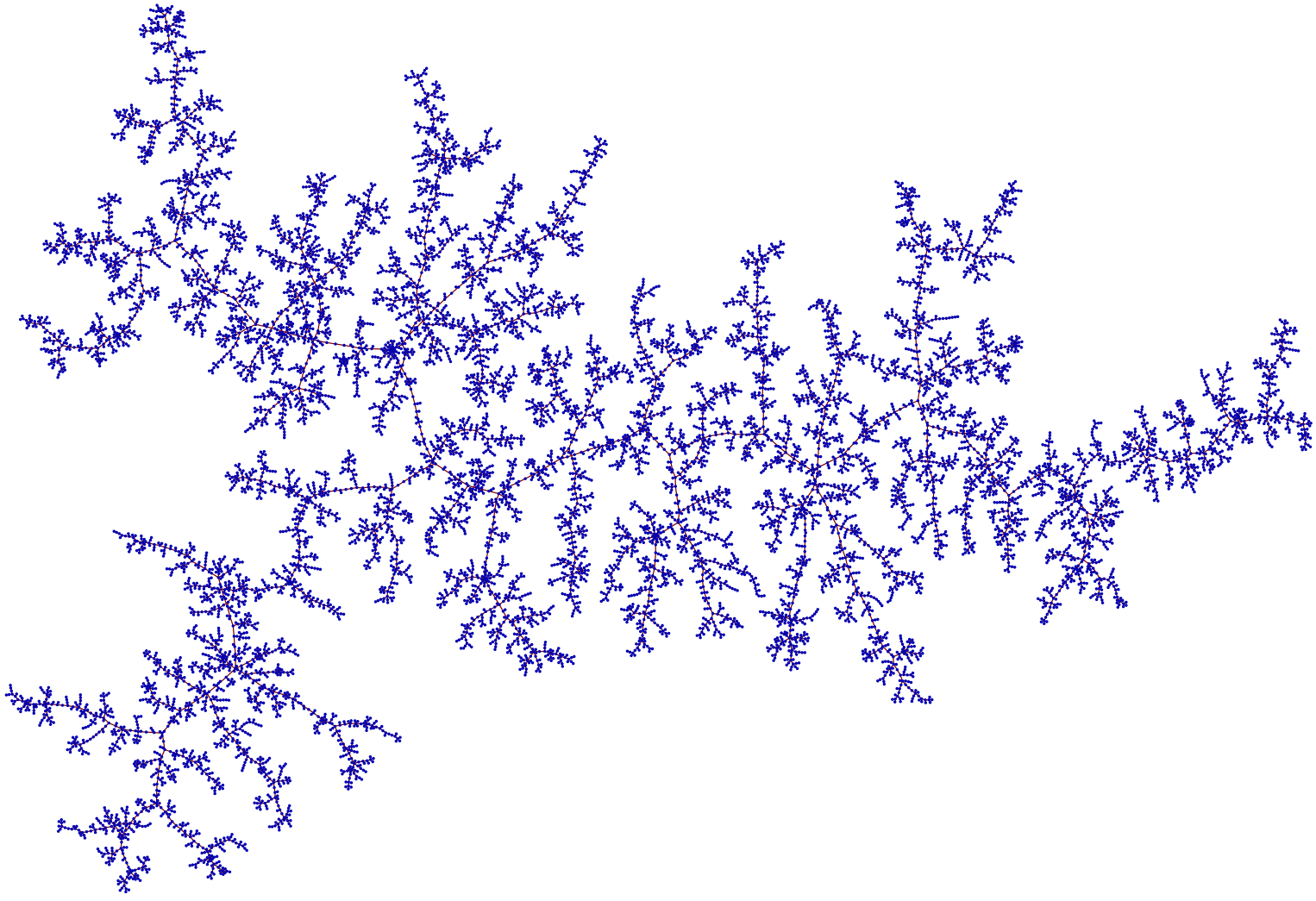
IV. Convergence of the uniform random tree

Theorem. (Aldous (1991))
As $n \to \infty$,

$$\frac{1}{\sqrt{n}} T_n \xrightarrow{d} \mathcal{T}$$

where $\mathcal{T}$ is the Brownian continuum random tree (CRT).

IV. The Brownian continuum random tree

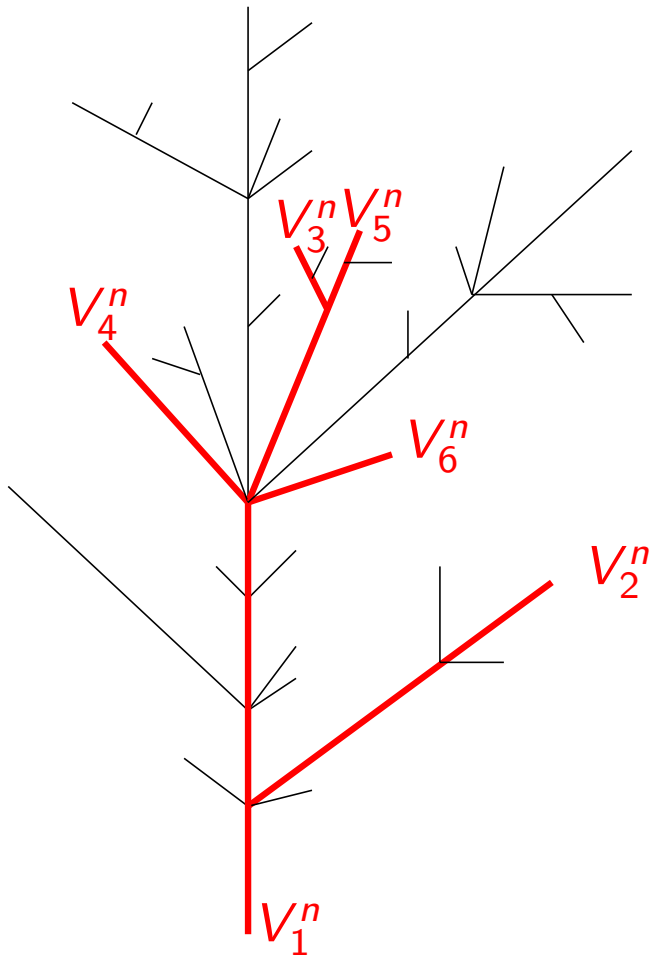


[Picture by Igor Kortchemski]

IV. Convergence of the uniform random tree

Fix $k \geq 2$, take the uniform random tree T_n and pick k independent uniform random vertices, $V_1^n, V_2^n, \ldots, V_k^n$.

$k = 6$:



IV. Convergence of the uniform random tree

The matrix of pairwise distances between these independent uniform points is

$$D^{n,k} = \begin{pmatrix} 0 & \mathrm{dist}(V_1^n, V_2^n) & \cdots & \mathrm{dist}(V_1^n, V_k^n) \\ \mathrm{dist}(V_2^n, V_1^n) & 0 & \cdots & \mathrm{dist}(V_2^n, V_k^n) \\ \vdots & \vdots & & \vdots \\ \mathrm{dist}(V_k^n, V_1^n) & \mathrm{dist}(V_k^n, V_2^n) & \cdots & 0 \end{pmatrix}$$

IV. Convergence of the uniform random tree

The matrix of pairwise distances between these independent uniform points is

$$D^{n,k} = \begin{pmatrix} 0 & \mathrm{dist}(V_1^n, V_2^n) & \cdots & \mathrm{dist}(V_1^n, V_k^n) \\ \mathrm{dist}(V_2^n, V_1^n) & 0 & \cdots & \mathrm{dist}(V_2^n, V_k^n) \\ \vdots & \vdots & & \vdots \\ \mathrm{dist}(V_k^n, V_1^n) & \mathrm{dist}(V_k^n, V_2^n) & \cdots & 0 \end{pmatrix}$$

Aldous' theorem says that for each $k \geq 2$, as $n \to \infty$,

$$\frac{1}{\sqrt{n}} D^{n,k} \xrightarrow{d} D^k$$

where D^k is the matrix of pairwise distances between k uniform points in $\mathcal{T}$.

IV. Convergence of the uniform random tree

The matrix of pairwise distances between these independent uniform points is

$$D^{n,k} = \begin{pmatrix} 0 & \mathrm{dist}(V_1^n, V_2^n) & \cdots & \mathrm{dist}(V_1^n, V_k^n) \\ \mathrm{dist}(V_2^n, V_1^n) & 0 & \cdots & \mathrm{dist}(V_2^n, V_k^n) \\ \vdots & \vdots & & \vdots \\ \mathrm{dist}(V_k^n, V_1^n) & \mathrm{dist}(V_k^n, V_2^n) & \cdots & 0 \end{pmatrix}$$

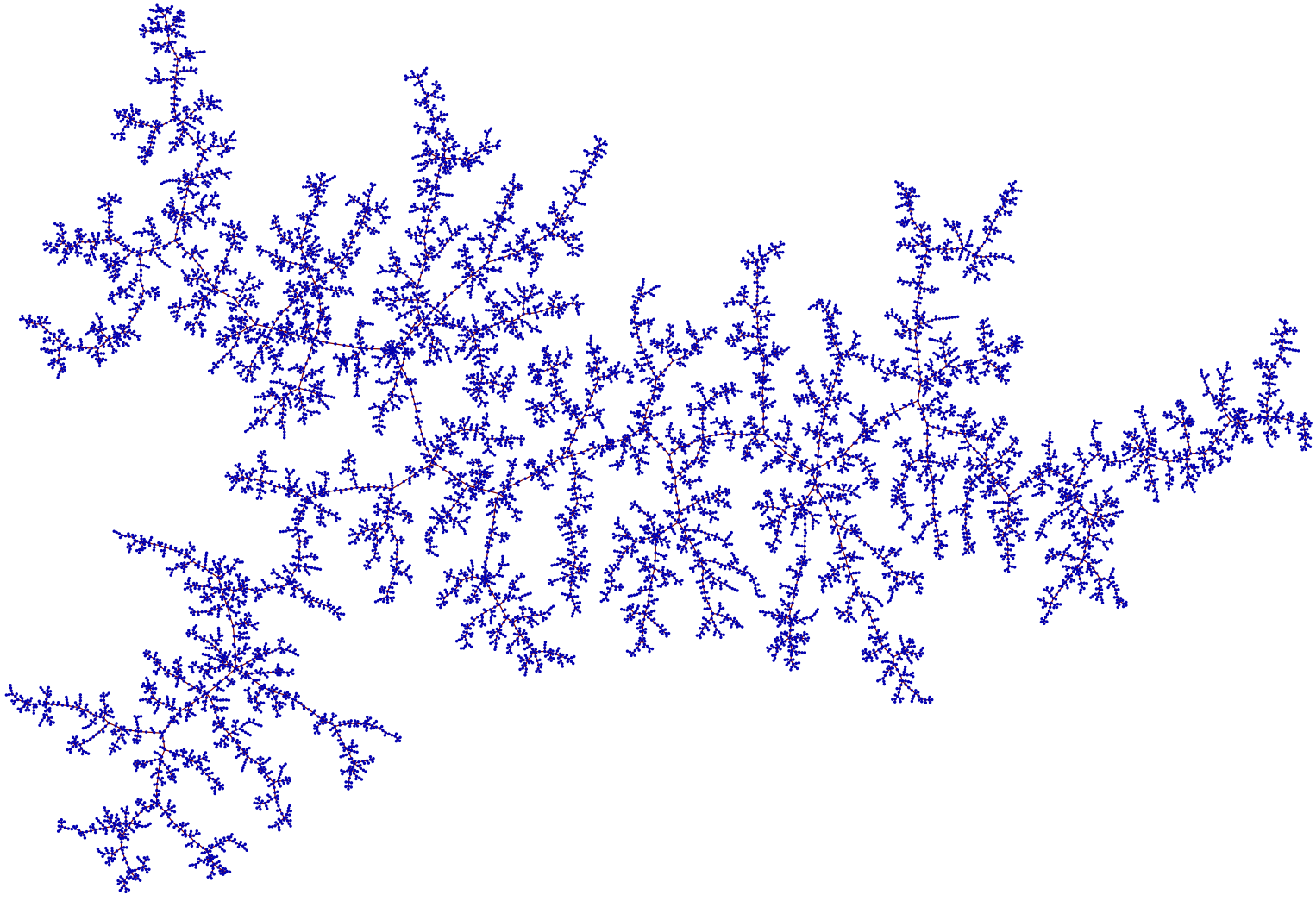
Aldous' theorem says that for each $k \geq 2$, as $n \to \infty$,

$$\frac{1}{\sqrt{n}} D^{n,k} \xrightarrow{d} D^k$$

where D^k is the matrix of pairwise distances between k uniform points in $\mathcal{T}$.

The $1/\sqrt{n}$ scaling says T_n is in some sense 2-dimensional: n points, diameter of order $\sqrt{n}$.

IV. The Brownian continuum random tree



[Picture by Igor Kortchemski]

V. Discrete construction (reversed Prüfer code)

We can understand the finite-dimensional distributions (distances between randomly sampled points) using a *reverse Prüfer code*.

V. Discrete construction (reversed Prüfer code)

We can understand the finite-dimensional distributions (distances between randomly sampled points) using a *reverse Prüfer code*.

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.

V. Discrete construction (reversed Prüfer code)

We can understand the finite-dimensional distributions (distances between randomly sampled points) using a *reverse Prüfer code*.

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.

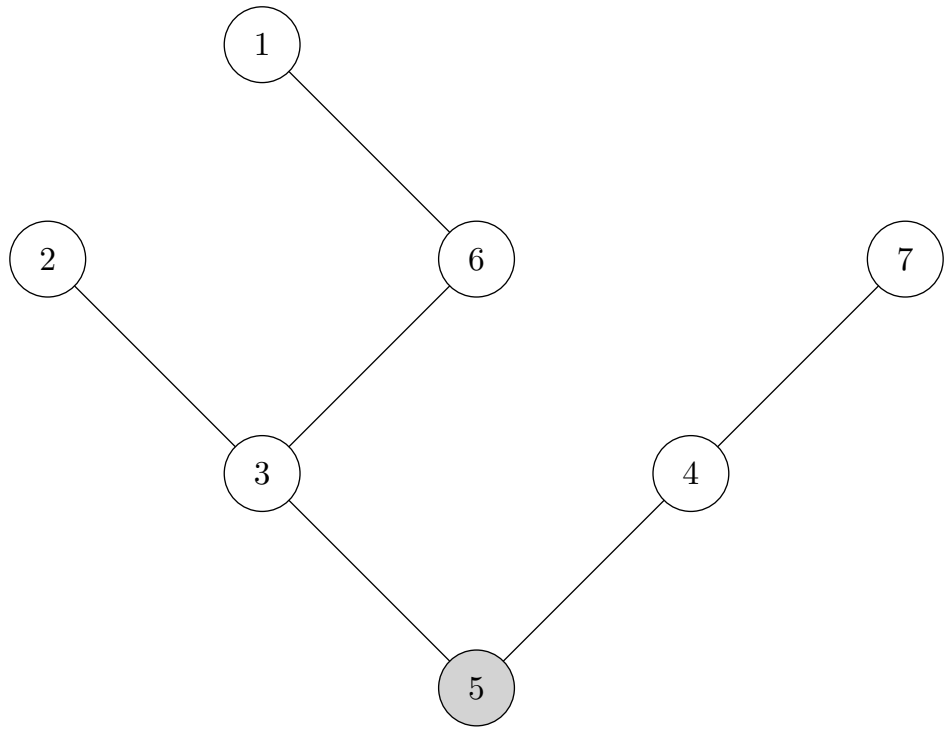
V. Discrete construction (reversed Prüfer code)

We can understand the finite-dimensional distributions (distances between randomly sampled points) using a *reverse Prüfer code*.

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.

V. Discrete construction (reversed Prüfer code)

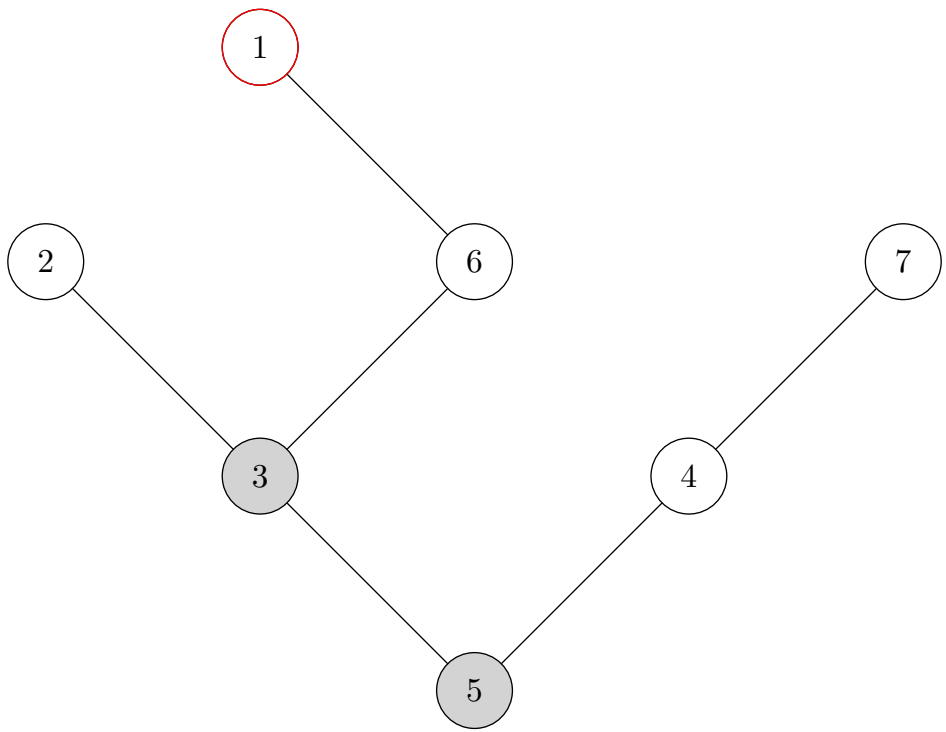
- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.



V. Discrete construction (reversed Prüfer code)

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.

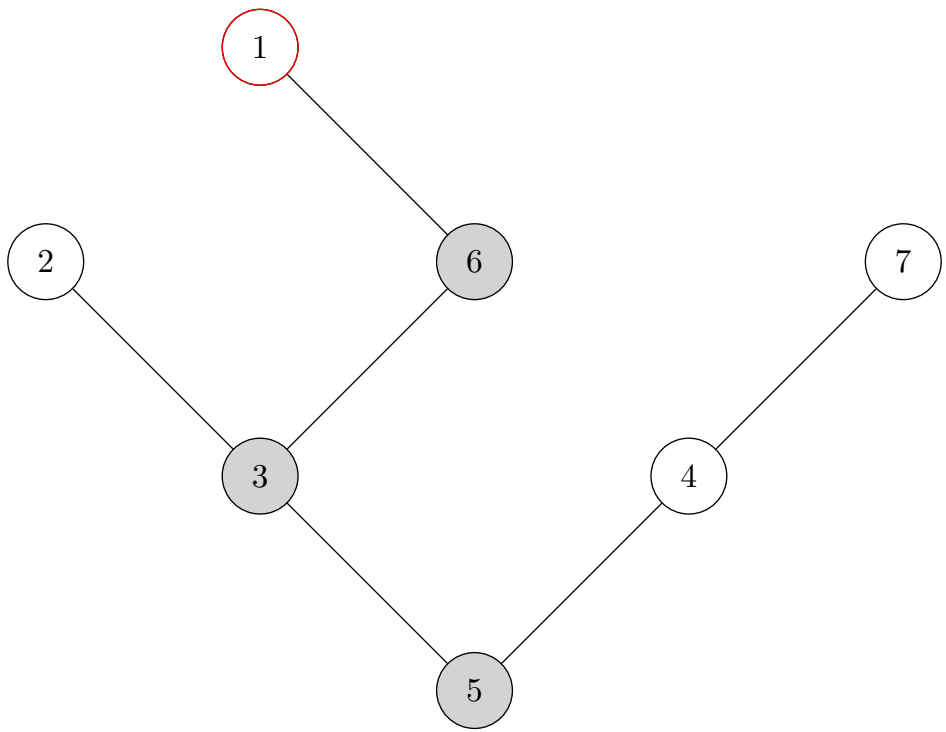
(5,



V. Discrete construction (reversed Prüfer code)

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.

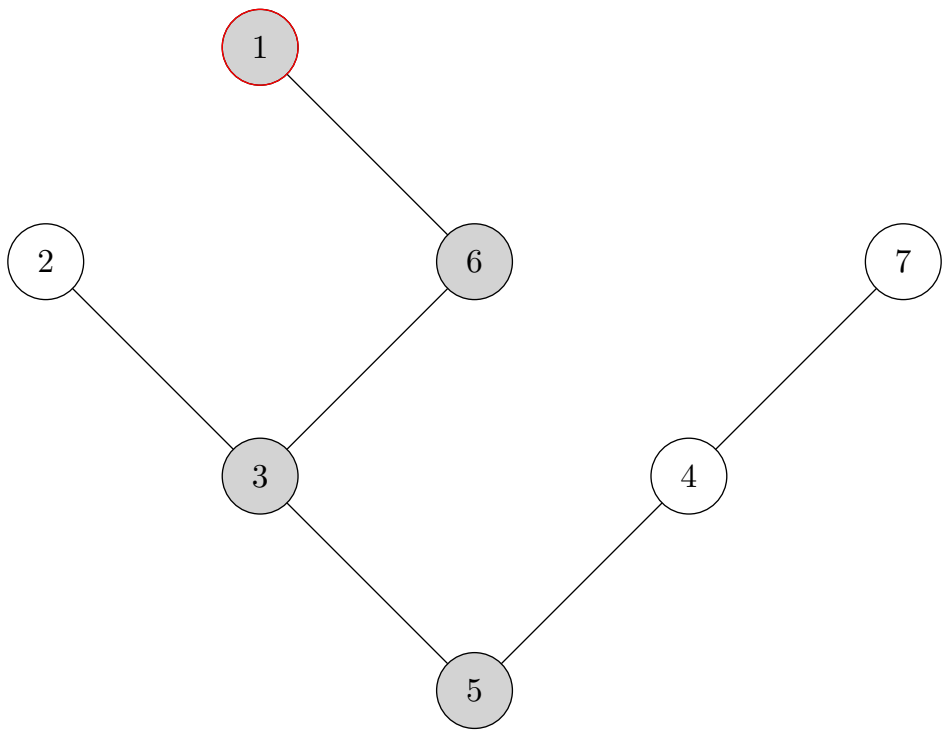
$(5, 3,$



V. Discrete construction (reversed Prüfer code)

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.

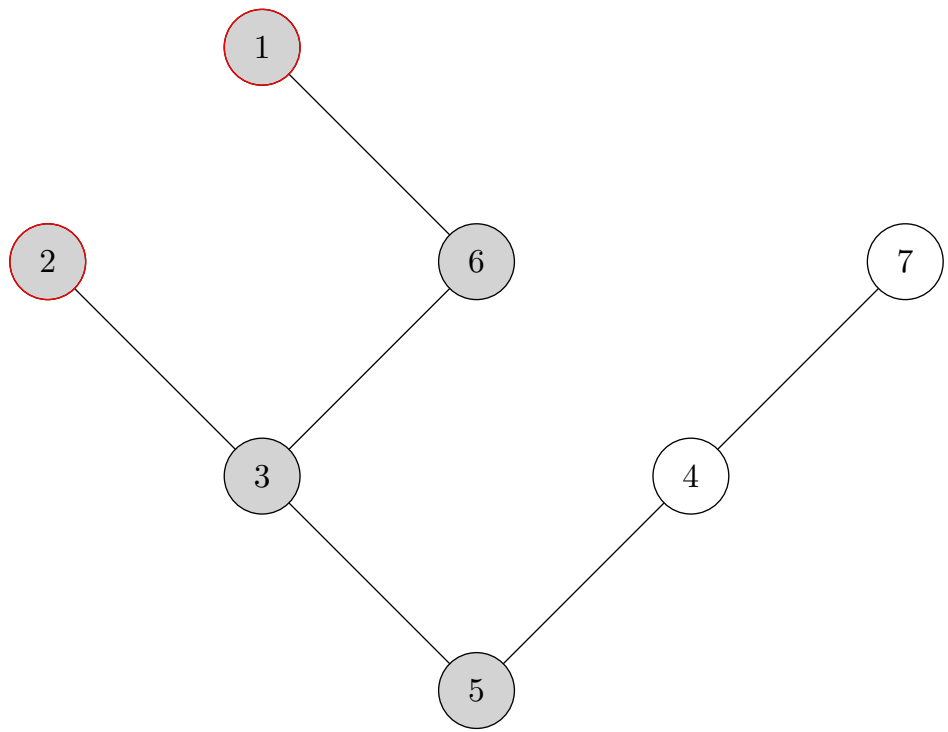
$(5, 3, 6,$



V. Discrete construction (reversed Prüfer code)

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.

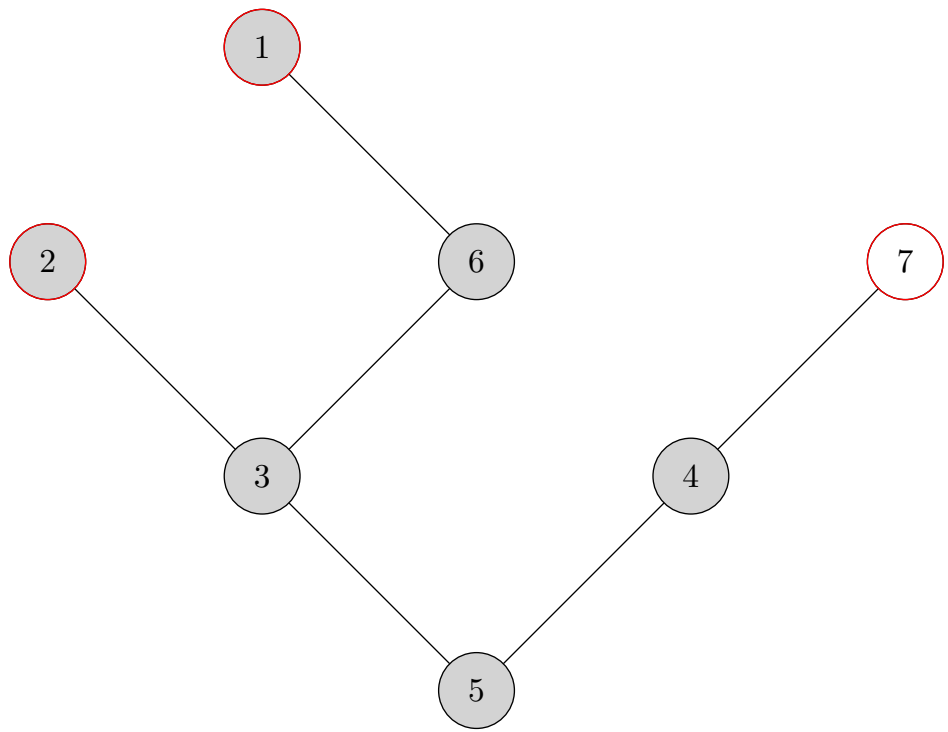
$(5, 3, 6, 3,$



V. Discrete construction (reversed Prüfer code)

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.

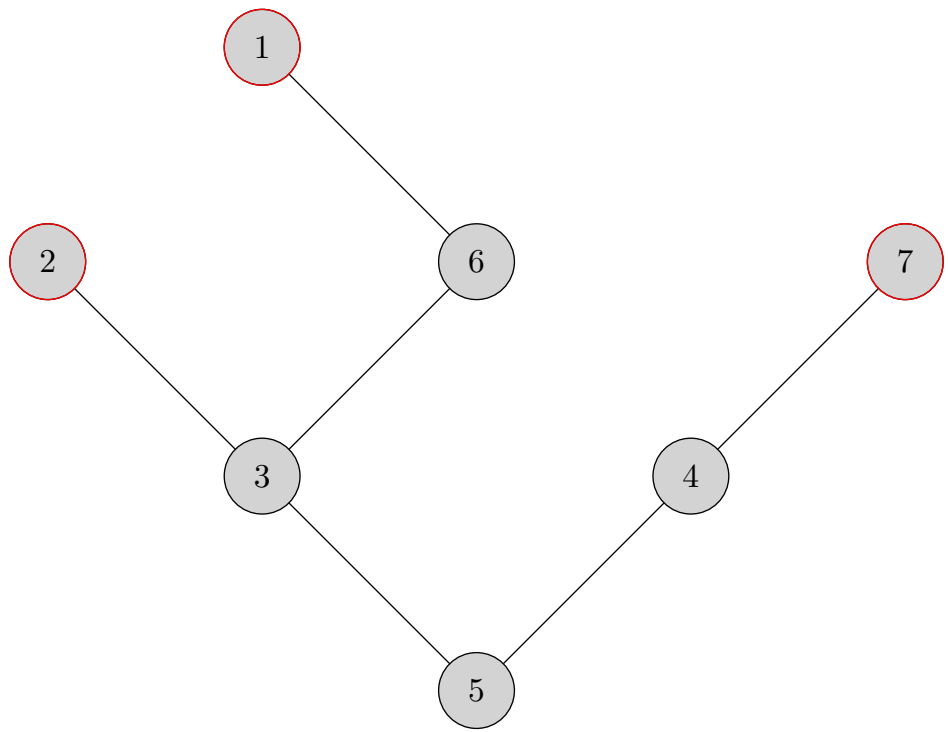
$(5, 3, 6, 3, 5,$



V. Discrete construction (reversed Prüfer code)

- Start from the root. At any step in the construction, other vertices have either *already been selected* or have not *yet been selected*.
- Among the unselected vertices, some have the smallest unselected vertex as a descendant. Among these, select the closest to the root.
- To derive the code, each time we select a vertex other than the root, record its parent, until all vertices have been selected.

$(5, 3, 6, 3, 5, 4)$



V. Random finite-dimensional distributions from Prüfer codes

We generate a labelled, unordered tree, with vertex labels 1,2,...,n, by sampling a uniformly random element $(S_1, S_2, \ldots, S_{n-1})$ from the set $[n]^{n-1} = \{(s_1, \ldots, s_{n-1}) : s_i \in [n] \text{ for } 1 \leq i \leq n-1\}$, and *inverting* the above encoding. The resulting tree is rooted at the vertex with label S_1. We build the rest of the tree step-by-step.

V. Random finite-dimensional distributions from Prüfer codes

We generate a labelled, unordered tree, with vertex labels 1,2,...,n, by sampling a uniformly random element $(S_1, S_2, \ldots, S_{n-1})$ from the set $[n]^{n-1} = \{(s_1, \ldots, s_{n-1}) : s_i \in [n] \text{ for } 1 \leq i \leq n-1\}$, and *inverting* the above encoding. The resulting tree is rooted at the vertex with label S_1. We build the rest of the tree step-by-step.

At step $2 \leq j \leq n-2$:

V. Random finite-dimensional distributions from Prüfer codes

We generate a labelled, unordered tree, with vertex labels 1,2,...,n, by sampling a uniformly random element $(S_1, S_2, \ldots, S_{n-1})$ from the set $[n]^{n-1} = \{(s_1, \ldots, s_{n-1}) : s_i \in [n] \text{ for } 1 \leq i \leq n-1\}$, and *inverting* the above encoding. The resulting tree is rooted at the vertex with label S_1. We build the rest of the tree step-by-step.

At step $2 \leq j \leq n-2$:

- If S_j is distinct from all those vertices which have gone before, add an edge from vertex S_j to S_{j-1}.
  (We say such a step extends a stick.)

V. Random finite-dimensional distributions from Prüfer codes

We generate a labelled, unordered tree, with vertex labels 1,2,...,n, by sampling a uniformly random element $(S_1, S_2, \ldots, S_{n-1})$ from the set $[n]^{n-1} = \{(s_1, \ldots, s_{n-1}) : s_i \in [n] \text{ for } 1 \leq i \leq n-1\}$, and *inverting* the above encoding. The resulting tree is rooted at the vertex with label S_1. We build the rest of the tree step-by-step.

At step $2 \leq j \leq n-2$:

- If S_j is distinct from all those vertices which have gone before, add an edge from vertex S_j to S_{j-1}.
  (We say such a step extends a stick.)
- If we have already seen S_j, instead add an edge from vertex S_{j-1} to the next-lowest vertex which has not yet appeared.
  (We say such a step creates a cut and starts a new stick.)

V. Random finite-dimensional distributions from Prüfer codes

We generate a labelled, unordered tree, with vertex labels 1,2,...,n, by sampling a uniformly random element $(S_1, S_2, \ldots, S_{n-1})$ from the set $[n]^{n-1} = \{(s_1, \ldots, s_{n-1}) : s_i \in [n] \text{ for } 1 \leq i \leq n-1\}$, and *inverting* the above encoding. The resulting tree is rooted at the vertex with label S_1. We build the rest of the tree step-by-step.

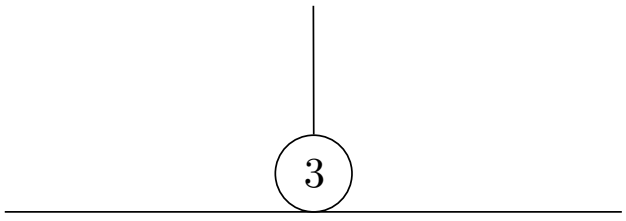
At step $2 \leq j \leq n-2$:

- If S_j is distinct from all those vertices which have gone before, add an edge from vertex S_j to S_{j-1}.
  (We say such a step extends a stick.)
- If we have already seen S_j, instead add an edge from vertex S_{j-1} to the next-lowest vertex which has not yet appeared.
  (We say such a step creates a cut and starts a new stick.)

Finish by adding a vertex corresponding to the final leftover label.

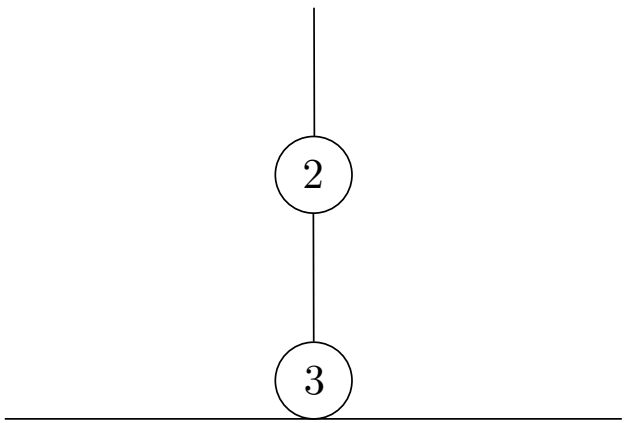
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$



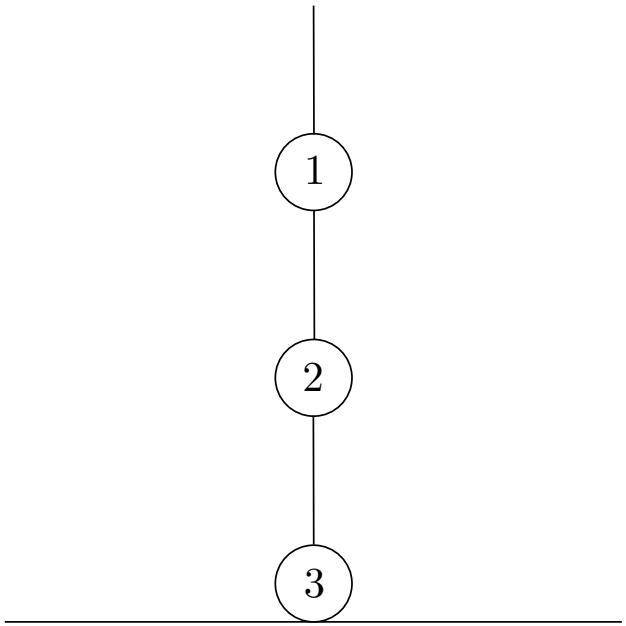
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$



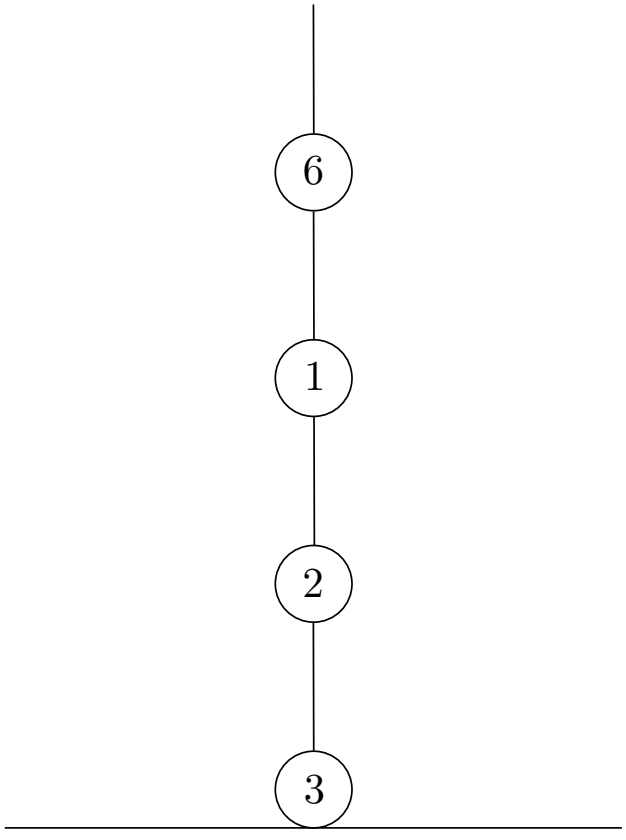
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$



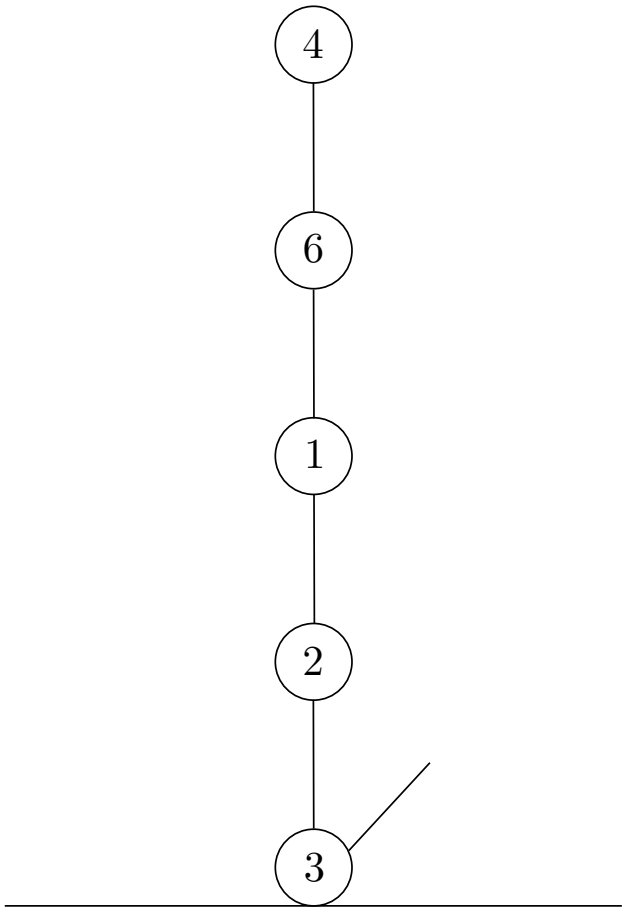
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$



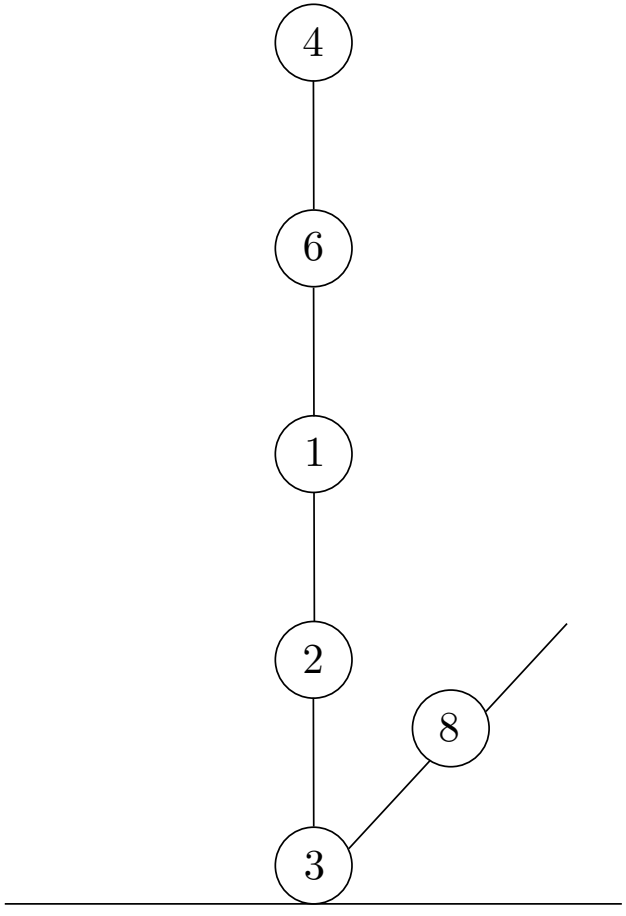
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$ Stick 1: 32164



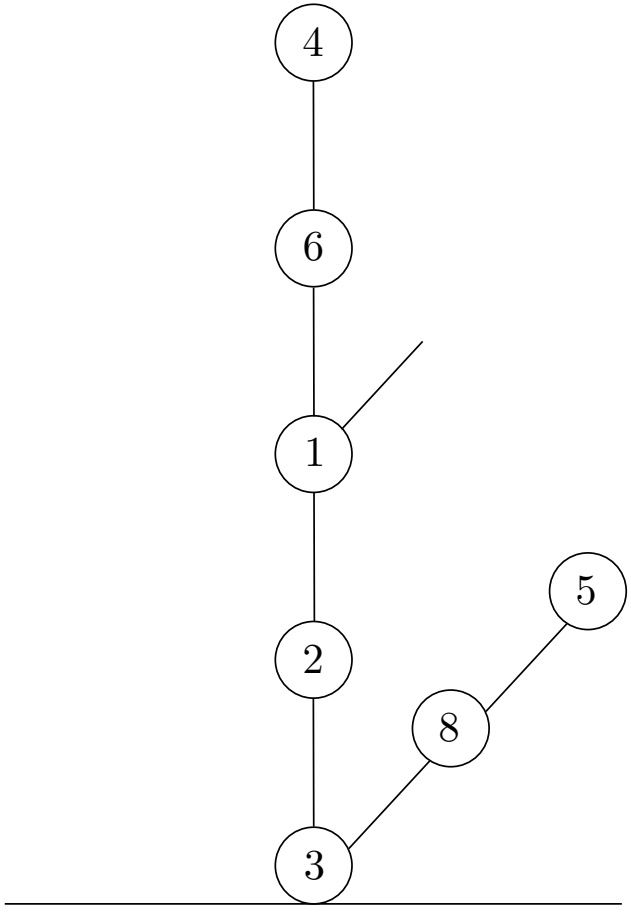
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$ Stick 1: 32164



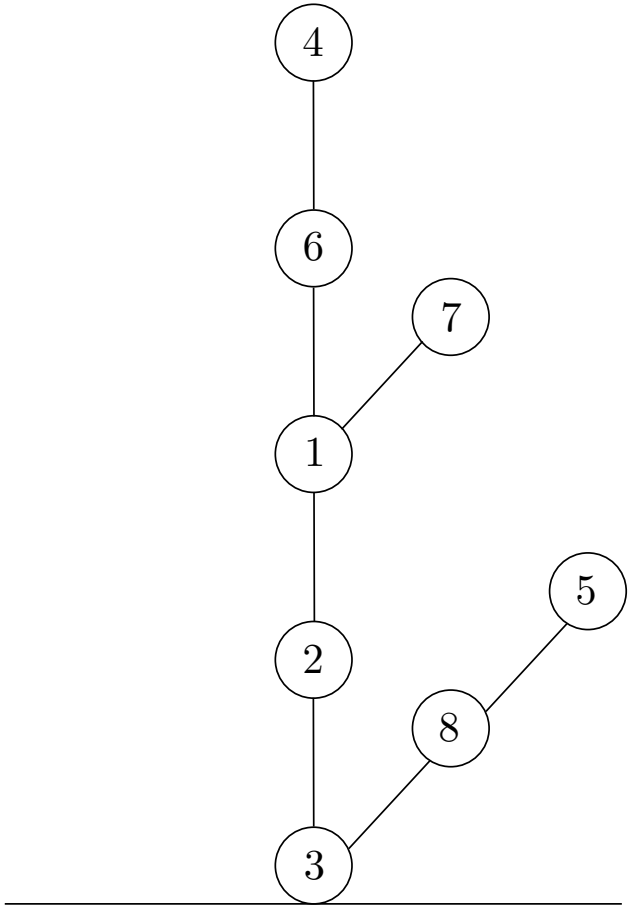
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$ Stick 1: 32164; Stick 2: 385



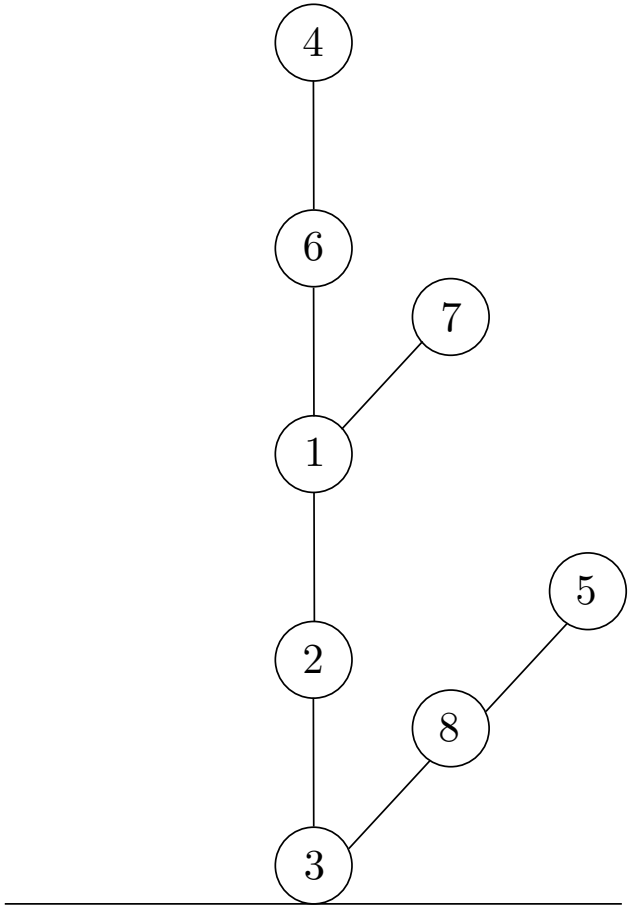
V. Random finite-dimensional distributions from Prüfer codes

$(3, 2, 1, 6, 3, 8, 1)$ Stick 1: 32164; Stick 2: 385; Stick 3: 17



V. Random finite-dimensional distributions from Prüfer codes

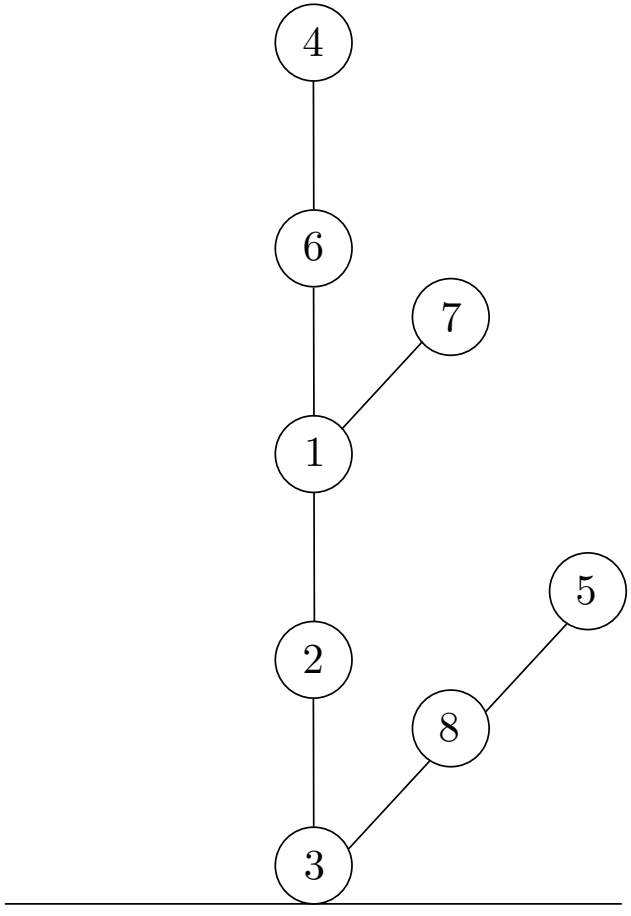
$(3, 2, 1, 6, 3, 8, 1)$ Stick 1: 32164; Stick 2: 385; Stick 3: 17



- Sticks get shorter on average as the process goes along since repetitions become more likely.

V. Random finite-dimensional distributions from Prüfer codes

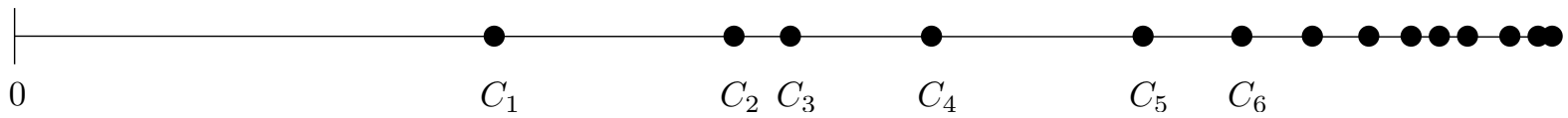
$(3, 2, 1, 6, 3, 8, 1)$ Stick 1: 32164; Stick 2: 385; Stick 3: 17



- Sticks get shorter on average as the process goes along since repetitions become more likely.
- Birthday paradox: for fixed k, for large n, the first k sticks have length of order $n^{1/2}$.

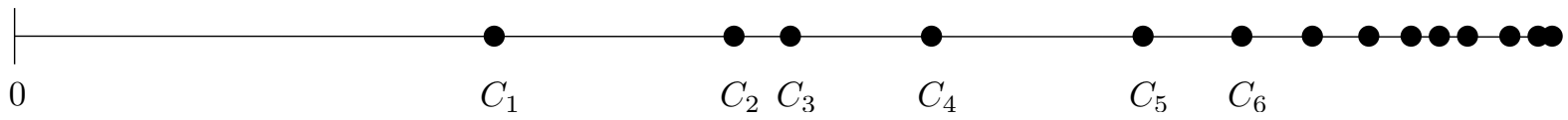
V. Continuous version of Prüfer encoding: Line-breaking

Take $E_1, E_2, \ldots, E_k$ independent and identically distributed Exponential(1/2) random variables, i.e. $\mathbb{P}(E_1 > x) = e^{-x/2}$. For $j \geq 1$, let $C_j = \sqrt{\sum_{i=1}^{j} E_i}$.



V. Continuous version of Prüfer encoding: Line-breaking

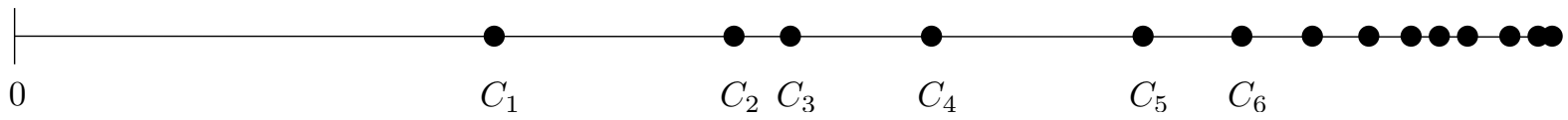
Take $E_1, E_2, \ldots, E_k$ independent and identically distributed Exponential(1/2) random variables, i.e. $\mathbb{P}(E_1 > x) = e^{-x/2}$. For $j \geq 1$, let $C_j = \sqrt{\sum_{i=1}^{j} E_i}$.



Consider the line-segments $[0, C_1), [C_1, C_2), \ldots, [C_{k-1}, C_k)$.

V. Continuous version of Prüfer encoding: Line-breaking

Take $E_1, E_2, \ldots, E_k$ independent and identically distributed Exponential(1/2) random variables, i.e. $\mathbb{P}(E_1 > x) = e^{-x/2}$. For $j \geq 1$, let $C_j = \sqrt{\sum_{i=1}^{j} E_i}$.

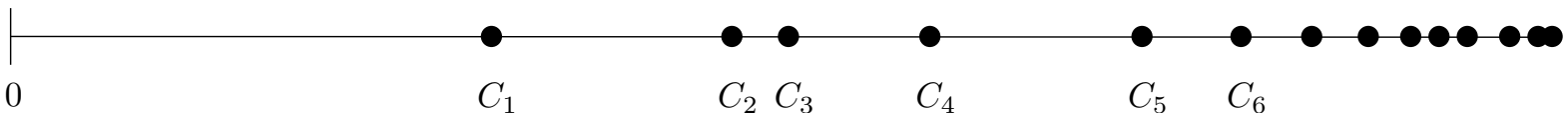


Consider the line-segments $[0, C_1)$, $[C_1, C_2), \ldots, [C_{k-1}, C_k)$.

Start from $[0, C_1)$ and proceed inductively.

V. Continuous version of Prüfer encoding: Line-breaking

Take $E_1, E_2, \ldots, E_k$ independent and identically distributed Exponential(1/2) random variables, i.e. $\mathbb{P}(E_1 > x) = e^{-x/2}$. For $j \geq 1$, let $C_j = \sqrt{\sum_{i=1}^{j} E_i}$.

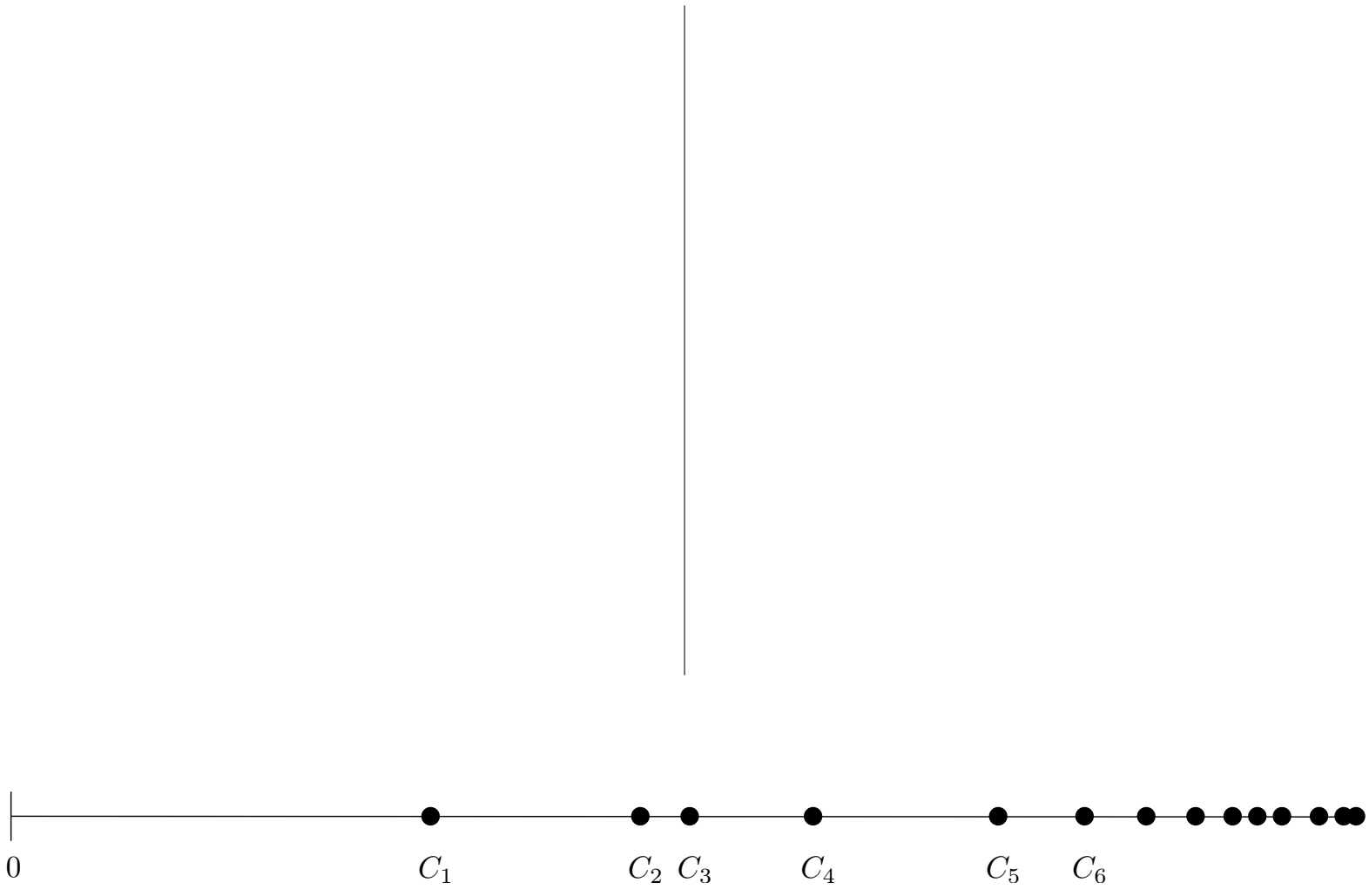


Consider the line-segments $[0, C_1), [C_1, C_2), \ldots, [C_{k-1}, C_k)$.

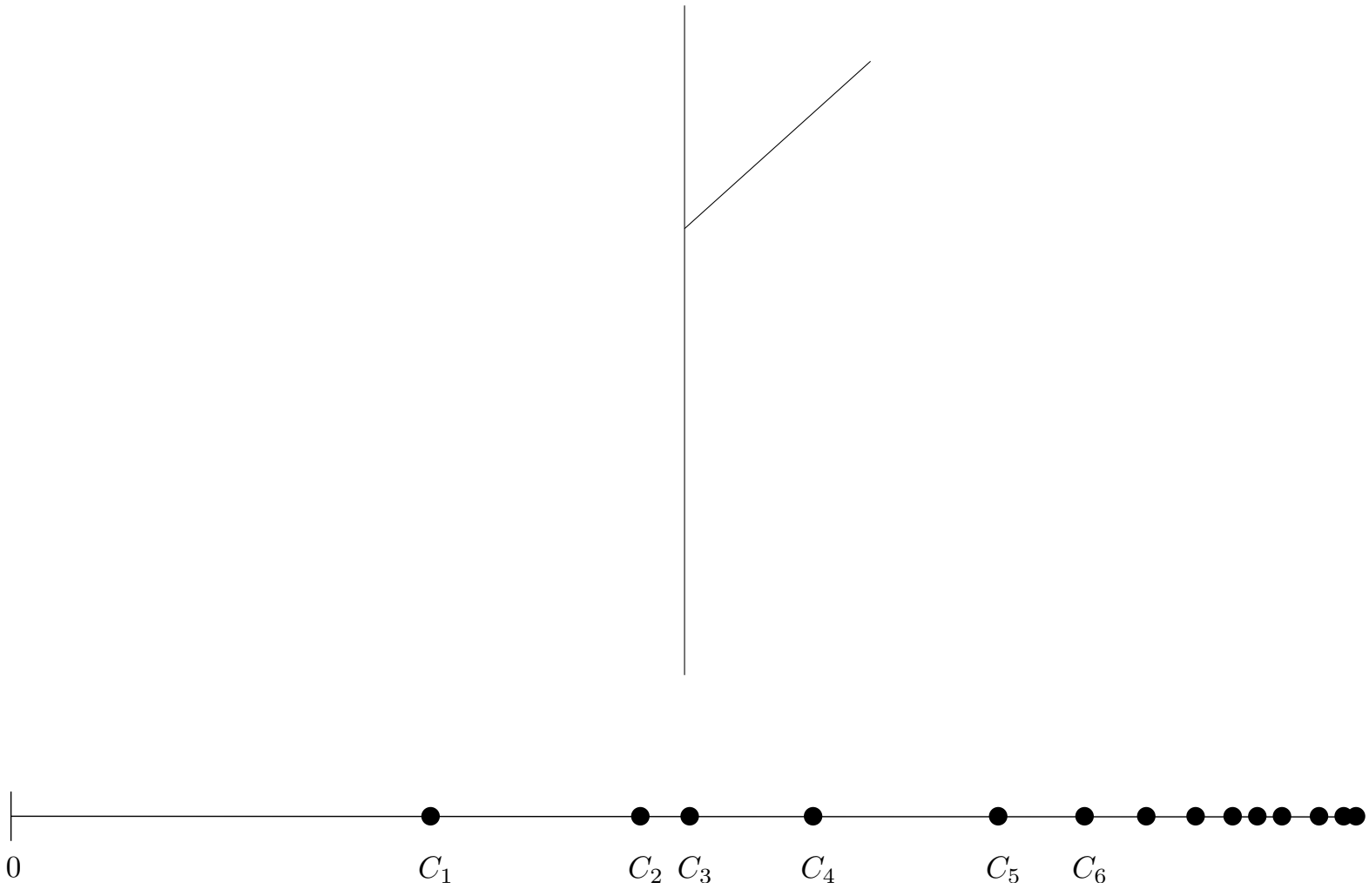
Start from $[0, C_1)$ and proceed inductively.

For $2 \leq i \leq k$, attach $[C_{i-1}, C_i)$ at a random point chosen uniformly over the existing tree.

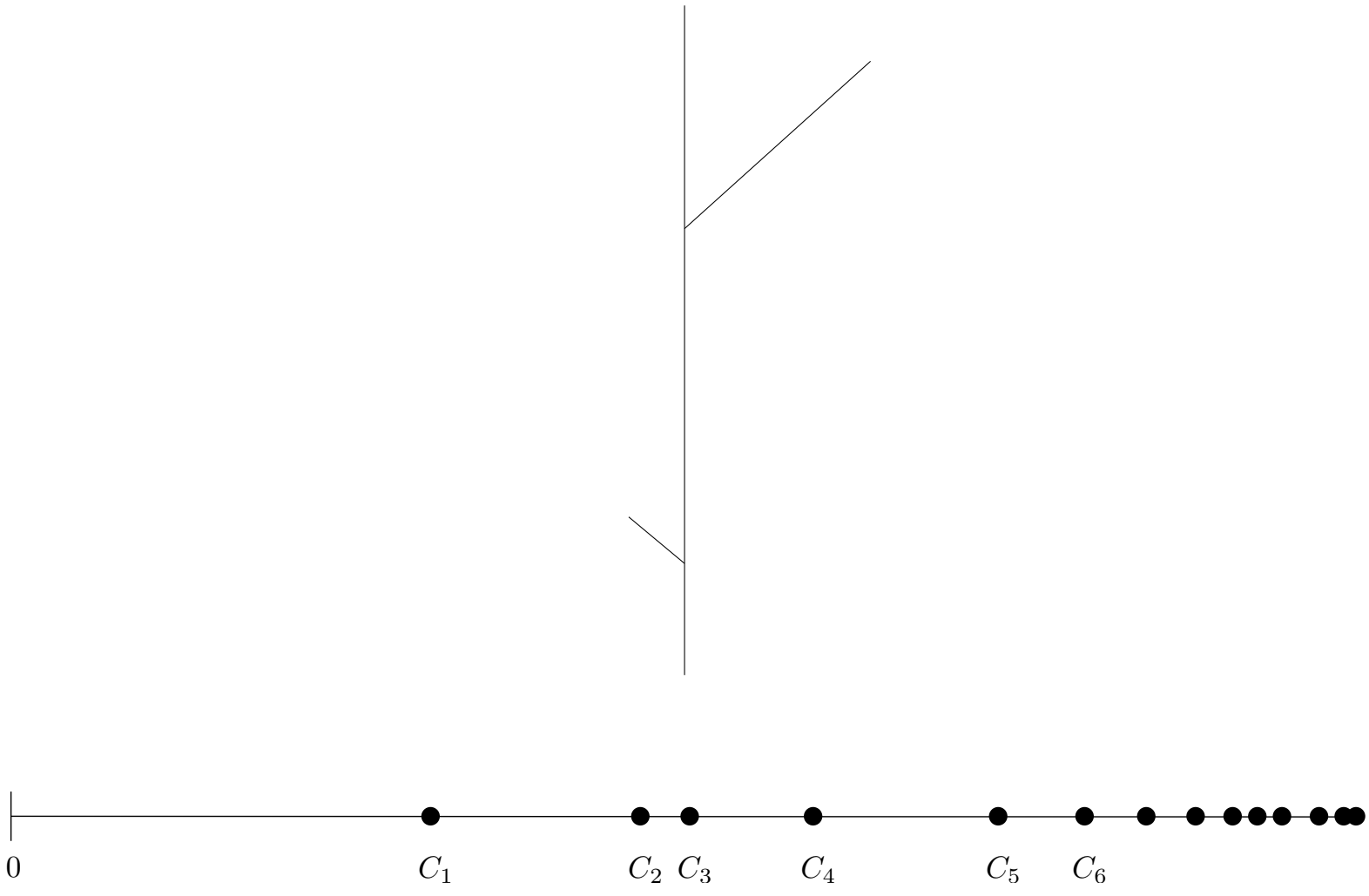
V. Line-breaking



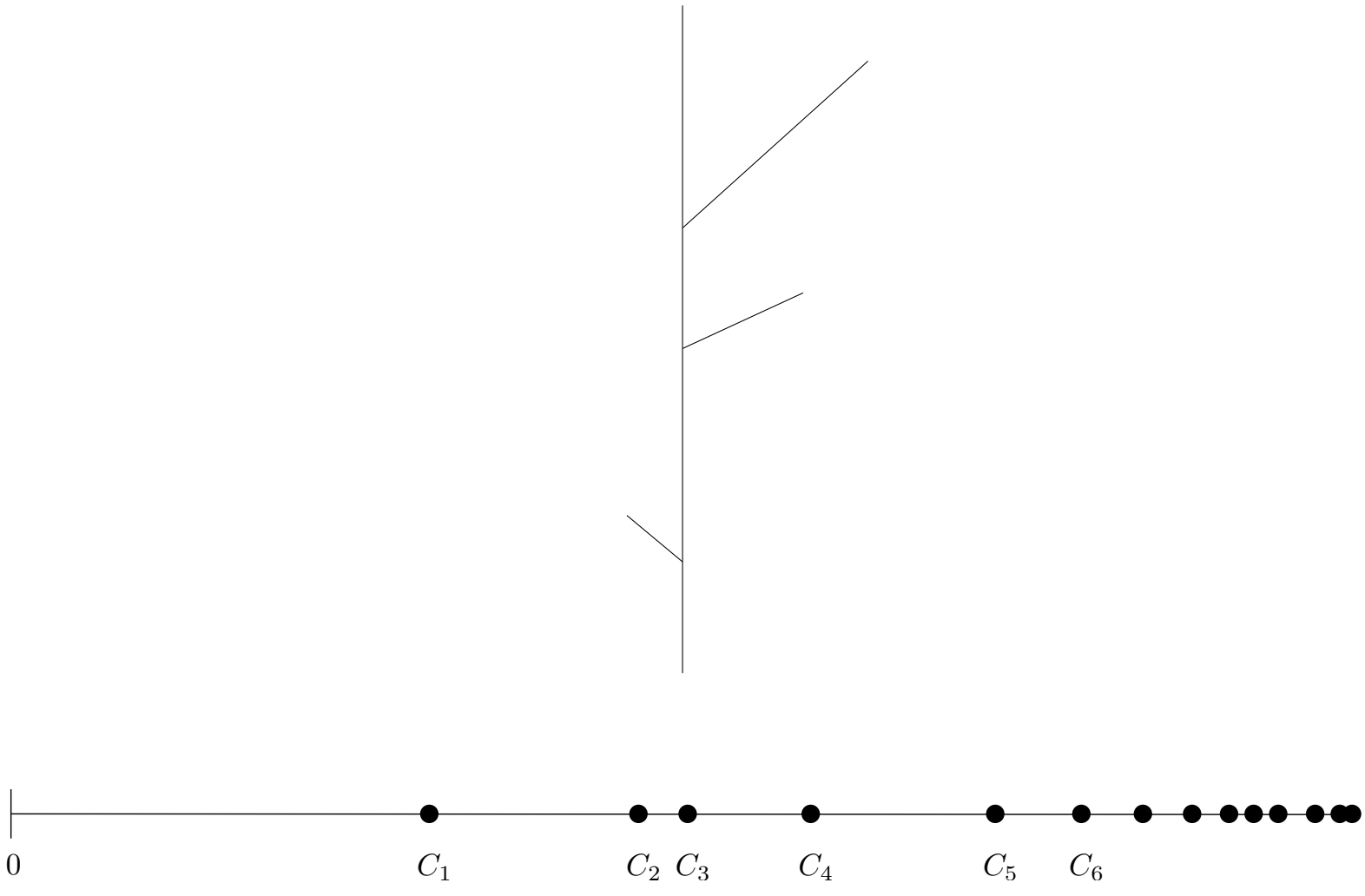
Line-breaking



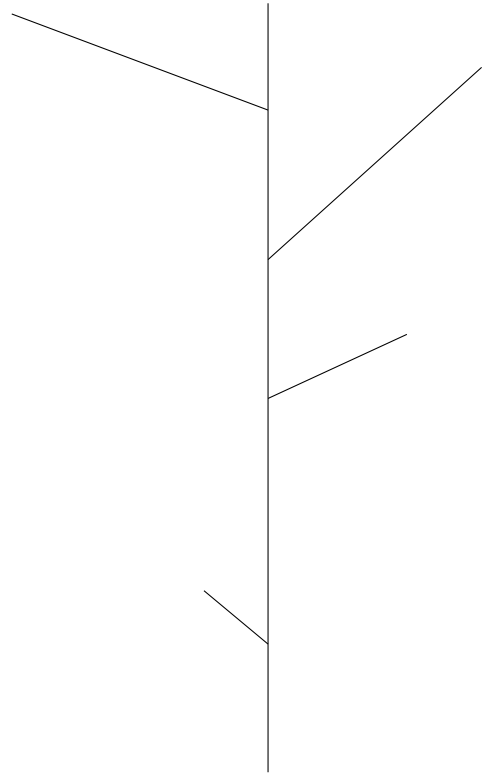
V. Line-breaking

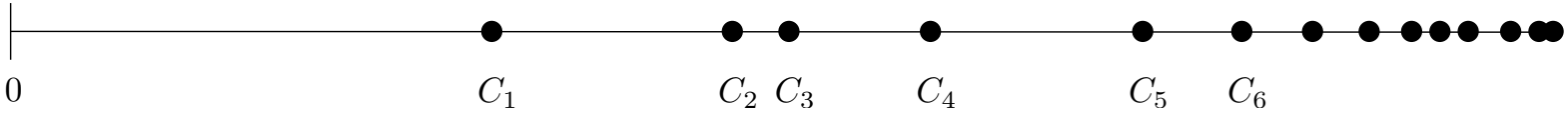


V. Line-breaking

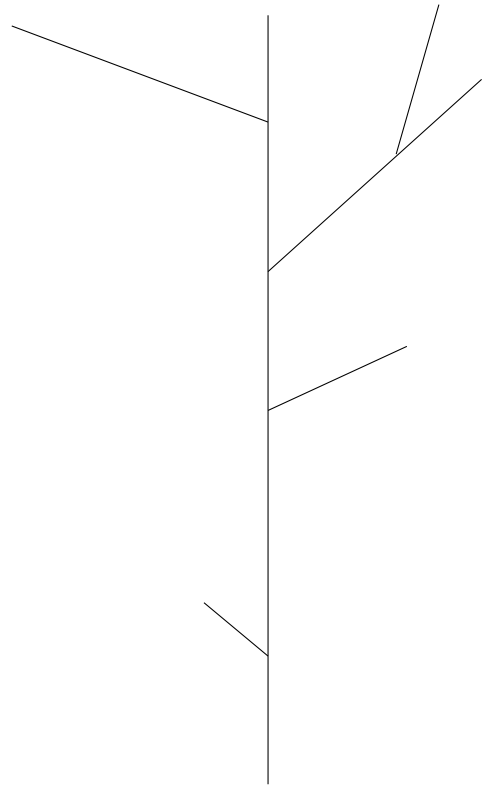


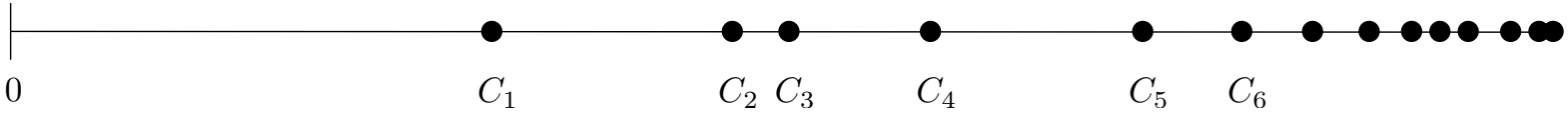
V. Line-breaking



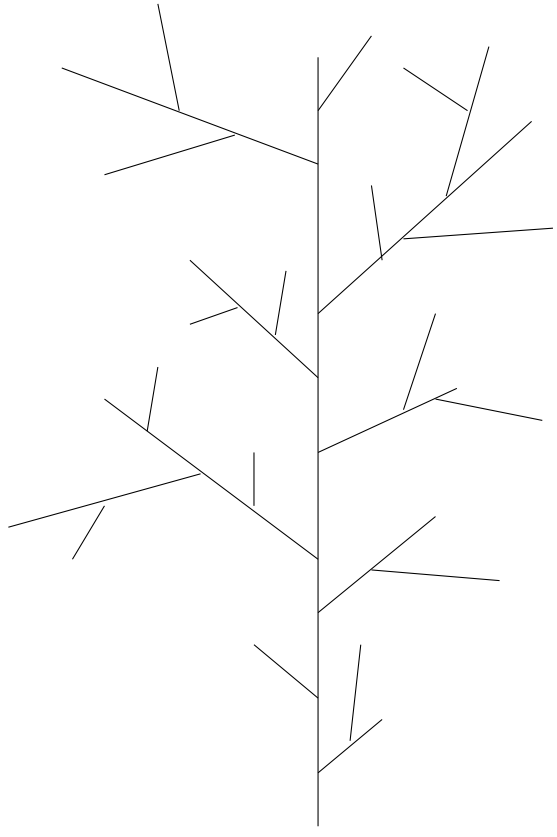


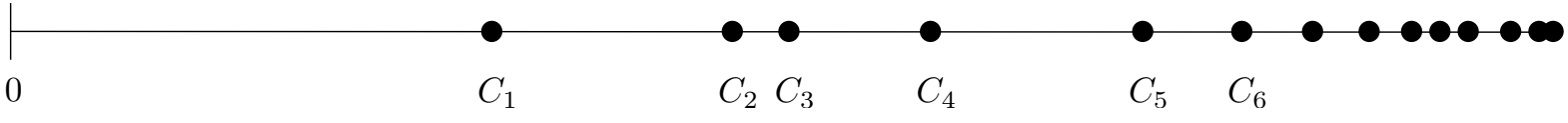
V. Line-breaking





V. Line-breaking





V. Line-breaking

D^k is the matrix of pairwise distances between the first k leaves in the line-breaking construction.

V. Line-breaking

D^k is the matrix of pairwise distances between the first k leaves in the line-breaking construction.

For increasing k, the line-breaking construction gives a sequence of better and better approximations to the Brownian Continuum Random Tree. One way to *define* the Brownian CRT is then as the metric space completion of the increasing limit as $k \to \infty$ of the sequence of approximations.

Aside: Universality

The convergence result is, in fact, much more general, in that (up to a constant scaling factor) very many random trees have the Brownian CRT as their scaling limit.

- uniform unordered unlabelled rooted trees
- uniform unordered unlabelled unrooted trees
- Branching process trees with offspring mean 1 and finite offspring variance, conditioned to have size n
- random trees with a prescribed degree sequence satisfying certain conditions
- random dissections of a polygon.
- random graphs from "subcritical combinatorial classes".

Aside: Universality

The convergence result is, in fact, much more general, in that (up to a constant scaling factor) very many random trees have the Brownian CRT as their scaling limit.

- uniform unordered unlabelled rooted trees
- uniform unordered unlabelled unrooted trees
- Branching process trees with offspring mean 1 and finite offspring variance, conditioned to have size n
- random trees with a prescribed degree sequence satisfying certain conditions
- random dissections of a polygon.
- random graphs from "subcritical combinatorial classes".

In particular, all these objects are in some sense "2-dimensional".

Critical random graphs in Oxford

In 2007, when I was a postdoc, two friends and I realized we could apply metric space convergence theory to the setting of the critical Erdős–Rényi random graph. (Left: Christina Goldschmidt, Oxford. Right: Nicolas Broutin, Sorbonne.)

VI. The critical Erdős–Rényi random graph

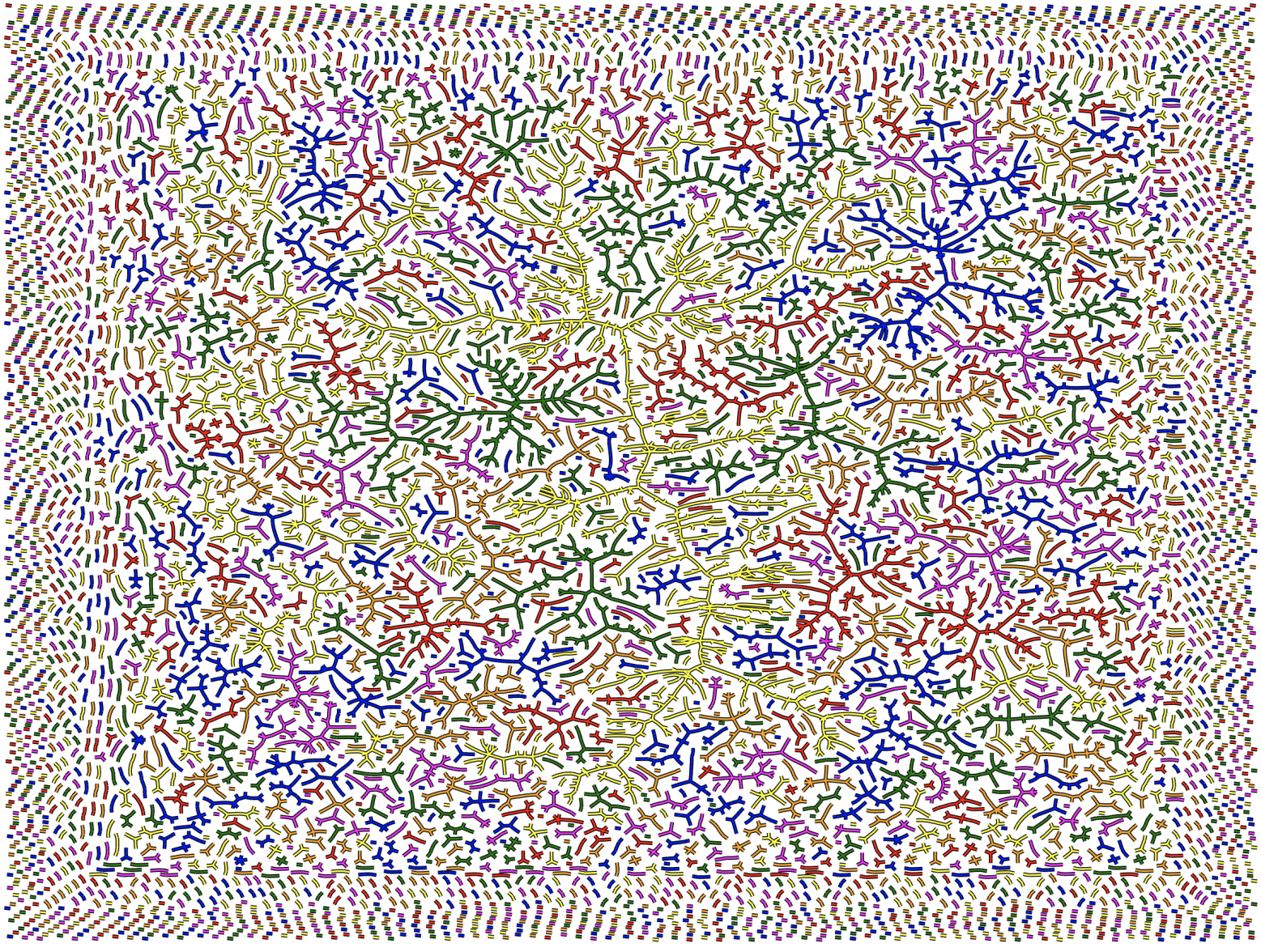


Image by Nicolas Broutin.

VI. The critical Erdős–Rényi random graph

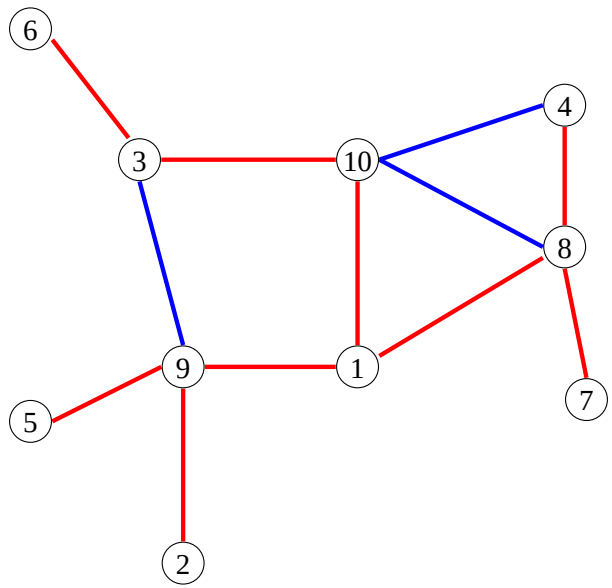
We saw earlier that the Erdős–Rényi random graph $G(n, p)$ undergoes a phase transition when $p = 1/n$. What happens exactly at the critical point?

VI. The critical Erdős–Rényi random graph

We saw earlier that the Erdős–Rényi random graph $G(n, p)$ undergoes a phase transition when $p = 1/n$. What happens exactly at the critical point?

We start by considering the sizes of the components. We will also be interested in the surplus of a component, that is the number of edges more than a tree that it has.

A component with surplus 3:



VI. The critical Erdős–Rényi random graph

Let $C_1^n, C_2^n, \ldots$ be the ordered component sizes of $G(n, 1/n)$.
Let $S_1^n, S_2^n, \ldots$ be the corresponding surpluses.

Theorem. (Aldous (1997))
As $n \to \infty$, we have jointly that

$$\frac{1}{n^{2/3}}(C_1^n, C_2^n, \ldots) \xrightarrow{d} (C_1, C_2, \ldots)$$

and

$$(S_1^n, S_2^n, \ldots) \xrightarrow{d} (S_1, S_2, \ldots),$$

where $C_1, C_2, \ldots \in \mathbb{R}_+$ and $S_1, S_2, \ldots \in \mathbb{Z}_+$ are random variables with an explicit (but complicated) distribution, such that $\mathbb{P}(S_k = 0) \to 1$ as $k \to \infty$.

Component convergence

This theorem tells us the limiting rescaled sizes of the components, and that they are quite close to being trees. Indeed, most of them really are trees!

Component convergence

This theorem tells us the limiting rescaled sizes of the components, and that they are quite close to being trees. Indeed, most of them really are trees!

Suppose, for the moment, we know that the vertices $v_1, v_2, \ldots, v_m$ form a tree-component of $G(n, 1/n)$. Then that tree is uniform its vertex-set (which we may as well think of as $\{1, 2, \ldots, m\}$).

Component convergence

This theorem tells us the limiting rescaled sizes of the components, and that they are quite close to being trees. Indeed, most of them really are trees!

Suppose, for the moment, we know that the vertices $v_1, v_2, \ldots, v_m$ form a tree-component of $G(n, 1/n)$. Then that tree is uniform its vertex-set (which we may as well think of as $\{1, 2, \ldots, m\}$).

So we should be thinking about rescaling by $m^{-1/2}$ in order to see a limit, which will be the Brownian CRT.

Component convergence

This theorem tells us the limiting rescaled sizes of the components, and that they are quite close to being trees. Indeed, most of them really are trees!

Suppose, for the moment, we know that the vertices $v_1, v_2, \ldots, v_m$ form a tree-component of $G(n, 1/n)$. Then that tree is uniform its vertex-set (which we may as well think of as $\{1, 2, \ldots, m\}$).

So we should be thinking about rescaling by $m^{-1/2}$ in order to see a limit, which will be the Brownian CRT.

Since components have sizes on the order of $n^{2/3}$, we should rescale by $n^{-1/3}$.

Component convergence

What about components which aren't trees? Again, if we know that $v_1, v_2, \ldots, v_m$ form a component of $G(n, 1/n)$ with surplus $s \geq 1$, that component is uniform among the possibilities.

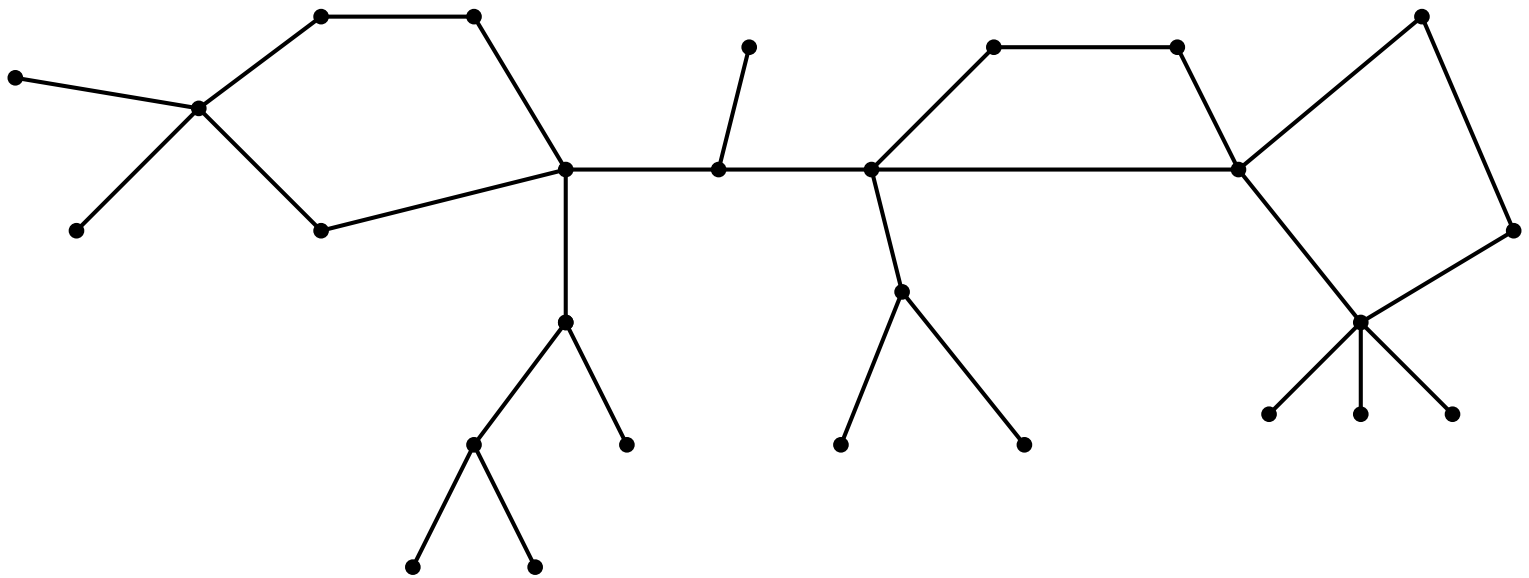
Component convergence

What about components which aren't trees? Again, if we know that $v_1, v_2, \ldots, v_m$ form a component of $G(n, 1/n)$ with surplus $s \geq 1$, that component is uniform among the possibilities.

So we need to understand the large-m behaviour of a uniform random connected graph on m vertices with surplus s.

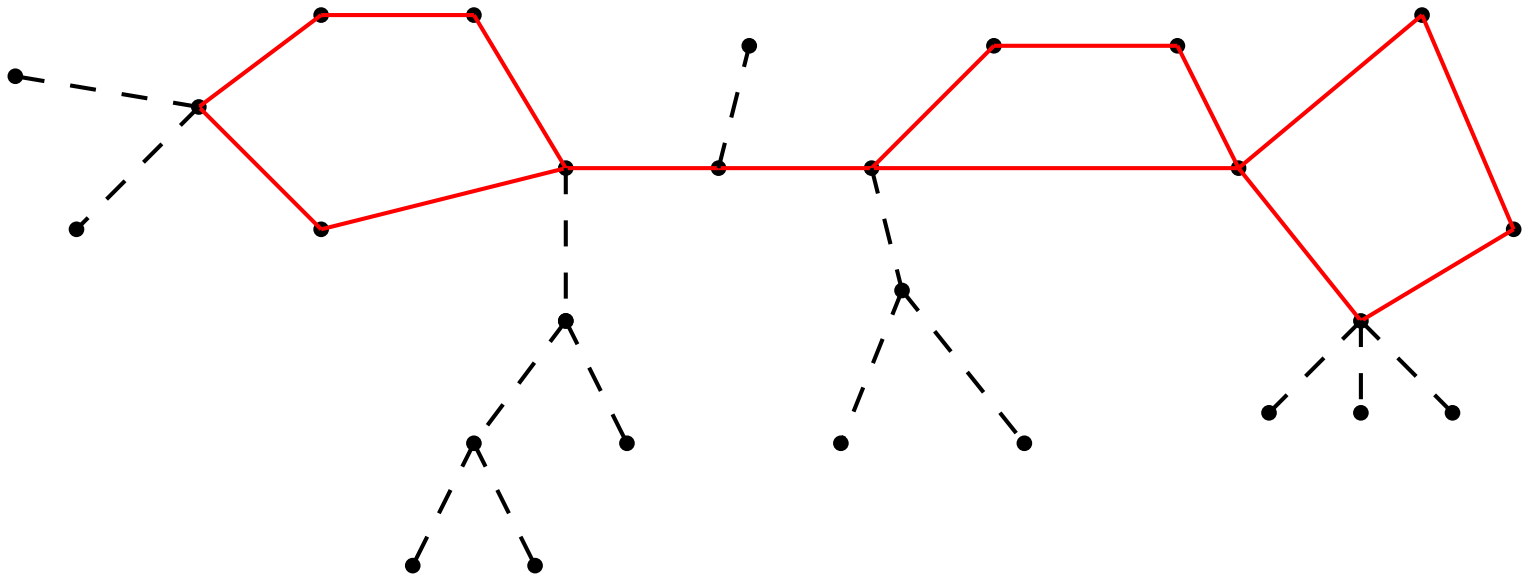
Cycle structure

Let's first look into the cycle structure.



Cycle structure

Core $C(G)$



Cycle structure

The kernel $K(G)$ is the multigraph which gives the "shape of the core": take the vertices of the core of degree 3 or more; contract the paths between them to a single edge.

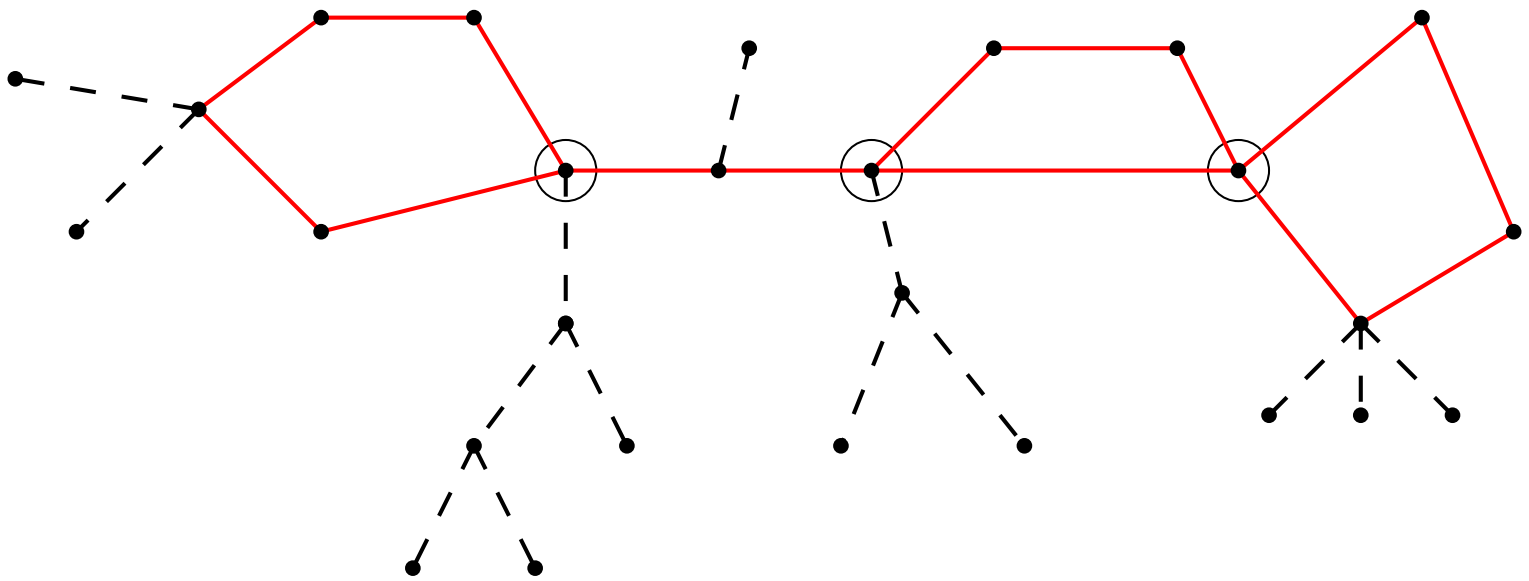
Cycle structure

The kernel $K(G)$ is the multigraph which gives the "shape of the core": take the vertices of the core of degree 3 or more; contract the paths between them to a single edge.

(By convention, the kernel of a tree or unicyclic component is empty.)

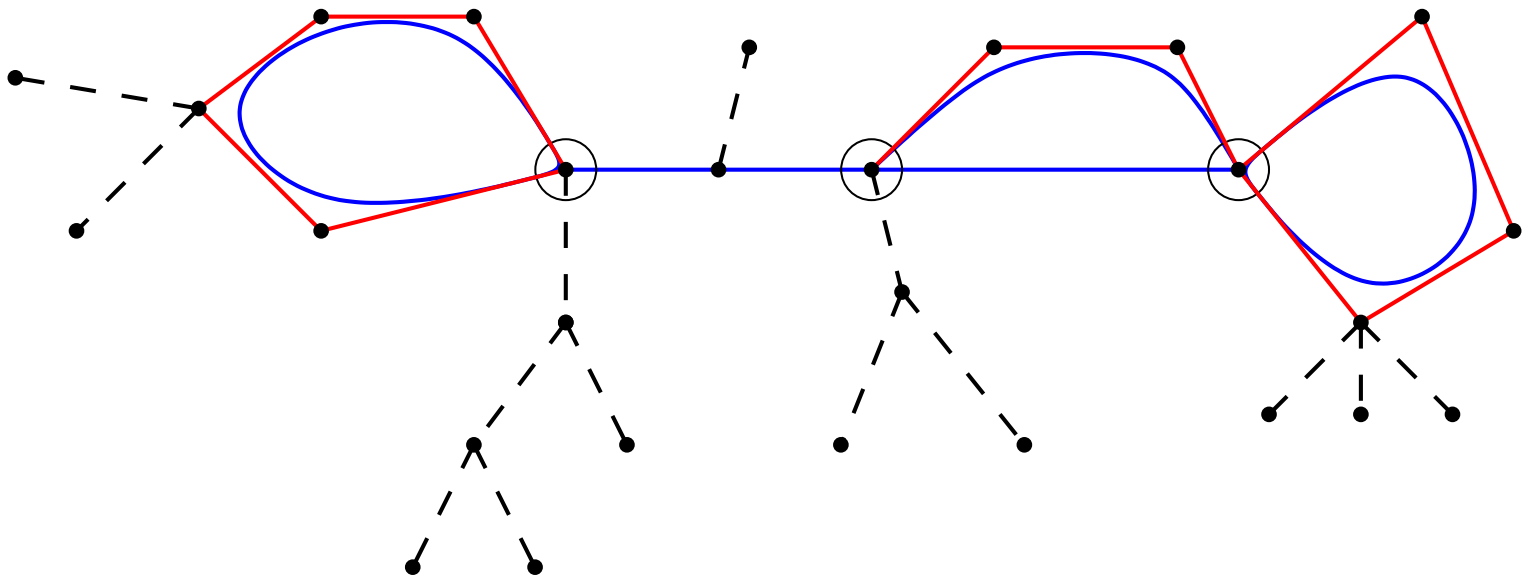
Cycle structure

Vertices of degree at least 3 in the core



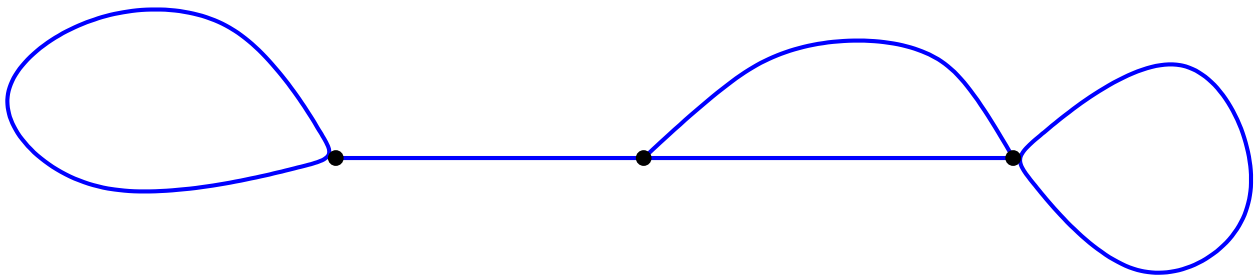
Cycle structure

Contract paths between them



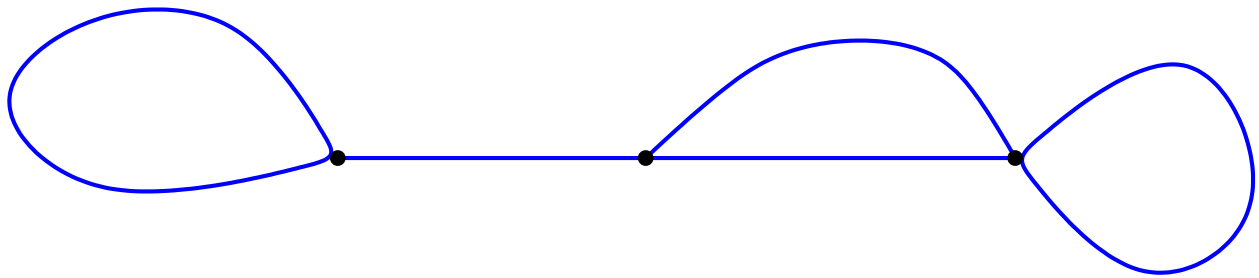
Cycle structure

Kernel $K(G)$



Cycle structure

Kernel $K(G)$



Note that the kernel has the same surplus as the original graph.

Scaling limits for uniform connected graphs

Theorem. (A.-B., Broutin and Goldschmidt. (2010, 2012))
Let U_m^s be a uniform connected graph with vertices labelled by $1, 2, \ldots, m$ and surplus $s \geq 1$. Then

$$\frac{1}{\sqrt{m}} U_m^s \xrightarrow{d} \mathcal{U}^s,$$

for a random limit object we will construct in a moment.

Scaling limits for uniform connected graphs

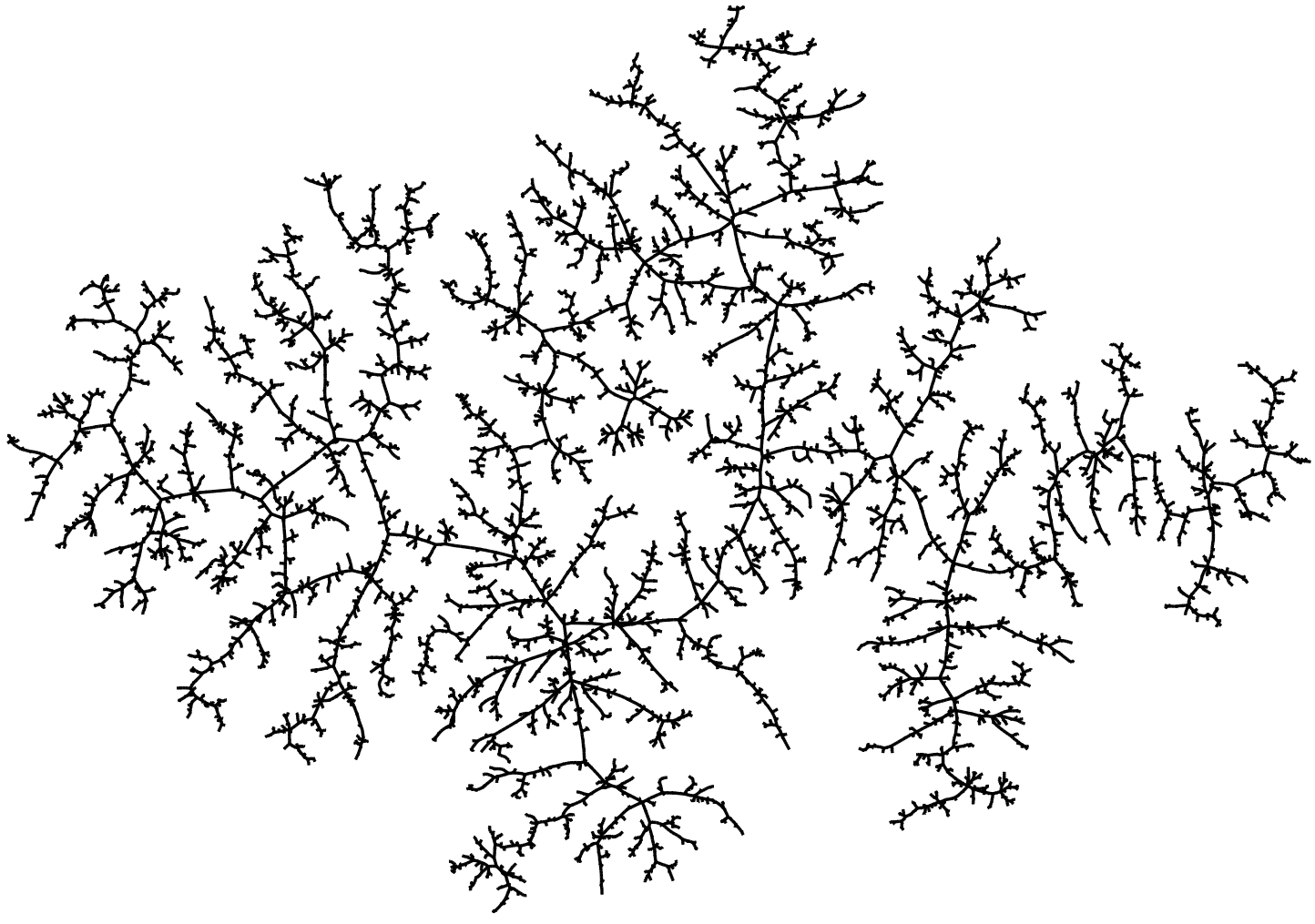


Image by Nicolas Broutin.

Scaling limits for uniform connected graphs

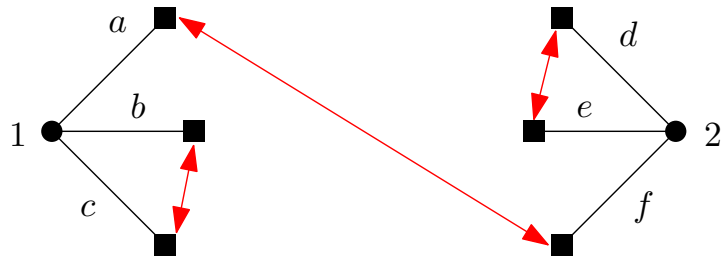
Again, this convergence means that if we sample k uniform random points and look at the matrix of pairwise distances between them, that matrix converges on rescaling by $m^{-1/2}$.

Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 1: The core.

- Take $2s-2$ labelled vertices and assign each one 3 half-edges.

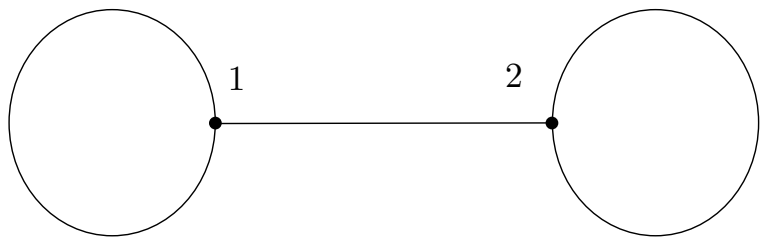
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 1: The core.

- Take $2s - 2$ labelled vertices and assign each one 3 half-edges.
- Pair the half-edges up uniformly at random to make full edges, conditionally on the multigraph obtained being connected.



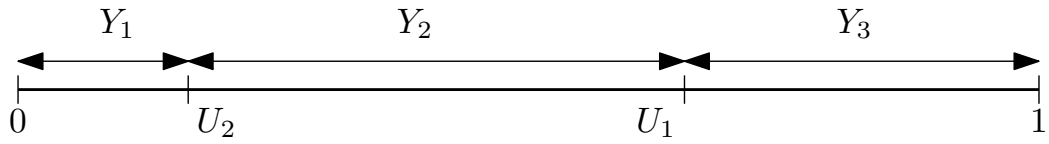
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 1: The core.

- Take $2s-2$ labelled vertices and assign each one 3 half-edges.
- Pair the half-edges up uniformly at random to make full edges, conditionally on the multigraph obtained being connected.



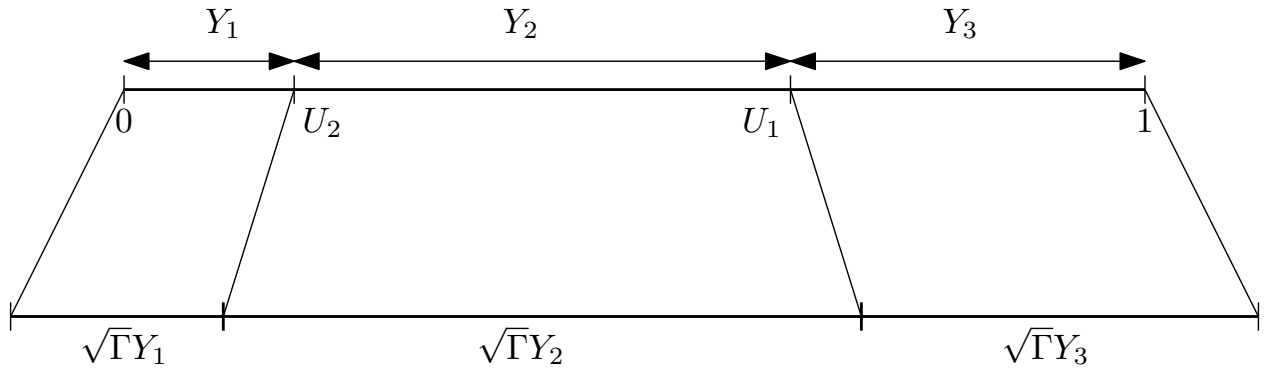
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 1: The core.

- Take $2s-2$ labelled vertices and assign each one 3 half-edges.
- Pair the half-edges up uniformly at random to make full edges, conditionally on the multigraph obtained being connected.
- Let $(Y_1, Y_2, \ldots, Y_{3s-3})$ be the lengths of the subintervals into which $[0,1]$ is split by throwing down $3s-4$ independent uniform random variables.



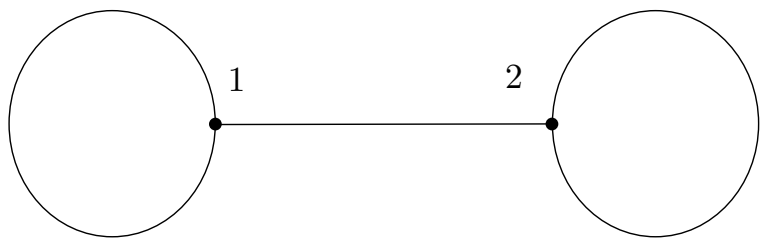
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 1: The core.

- Take $2s-2$ labelled vertices and assign each one 3 half-edges.
- Pair the half-edges up uniformly at random to make full edges, conditionally on the multigraph obtained being connected.
- Let $(Y_1, Y_2, \ldots, Y_{3s-3})$ be the lengths of the subintervals into which $[0,1]$ is split by throwing down $3s-4$ independent uniform random variables.
- Sample $\Gamma \sim \text{Gamma}(\frac{3s-2}{2}, \frac{1}{2})$ and scale the ith edge of the multigraph so that it gets length $\sqrt{\Gamma} Y_i$.



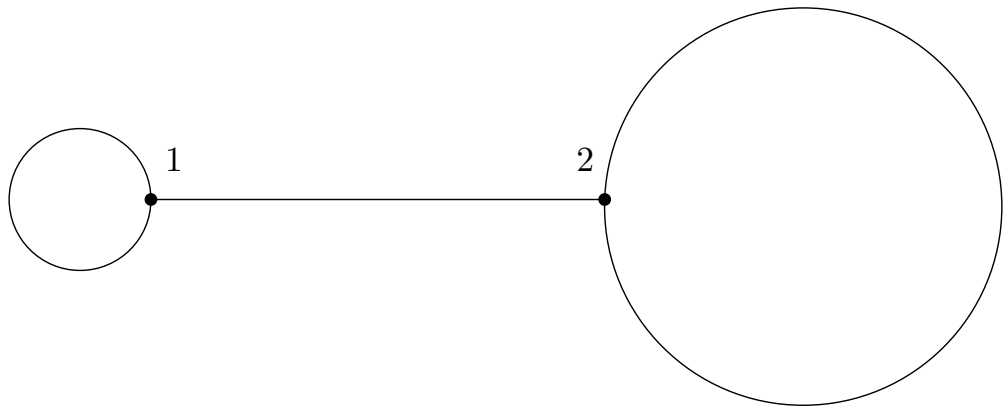
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 1: The core.

- Take $2s-2$ labelled vertices and assign each one 3 half-edges.
- Pair the half-edges up uniformly at random to make full edges, conditionally on the multigraph obtained being connected.
- Let $(Y_1, Y_2, \ldots, Y_{3s-3})$ be the lengths of the subintervals into which $[0,1]$ is split by throwing down $3s-4$ independent uniform random variables.
- Sample $\Gamma \sim \text{Gamma}(\frac{3s-2}{2}, \frac{1}{2})$ and scale the ith edge of the multigraph so that it gets length $\sqrt{\Gamma} Y_i$.



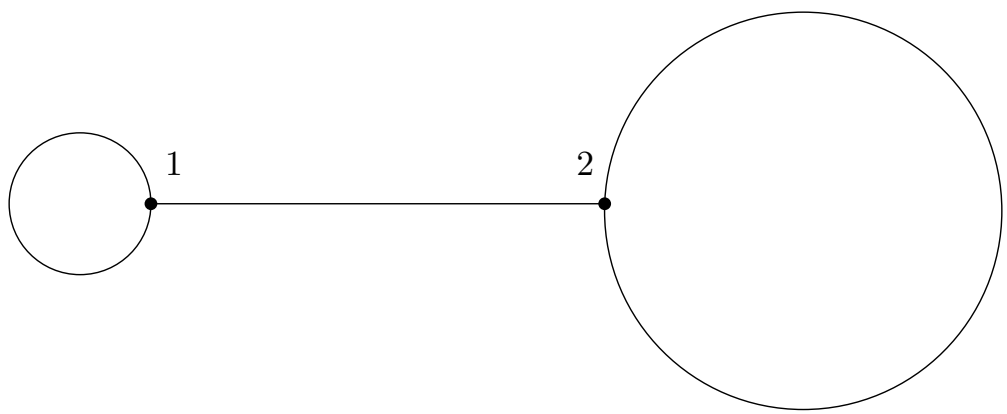
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 1: The core.

- Take $2s-2$ labelled vertices and assign each one 3 half-edges.
- Pair the half-edges up uniformly at random to make full edges, conditionally on the multigraph obtained being connected.
- Let $(Y_1, Y_2, \ldots, Y_{3s-3})$ be the lengths of the subintervals into which $[0,1]$ is split by throwing down $3s-4$ independent uniform random variables.
- Sample $\Gamma \sim \text{Gamma}(\frac{3s-2}{2}, \frac{1}{2})$ and scale the ith edge of the multigraph so that it gets length $\sqrt{\Gamma} Y_i$.



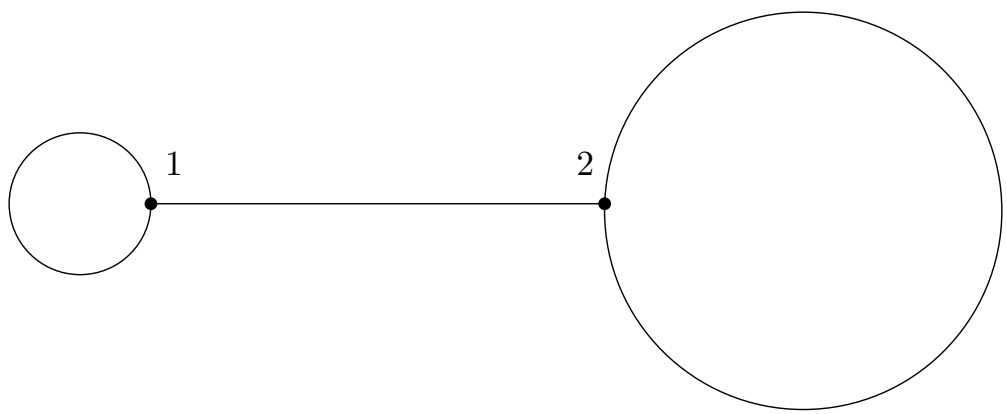
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



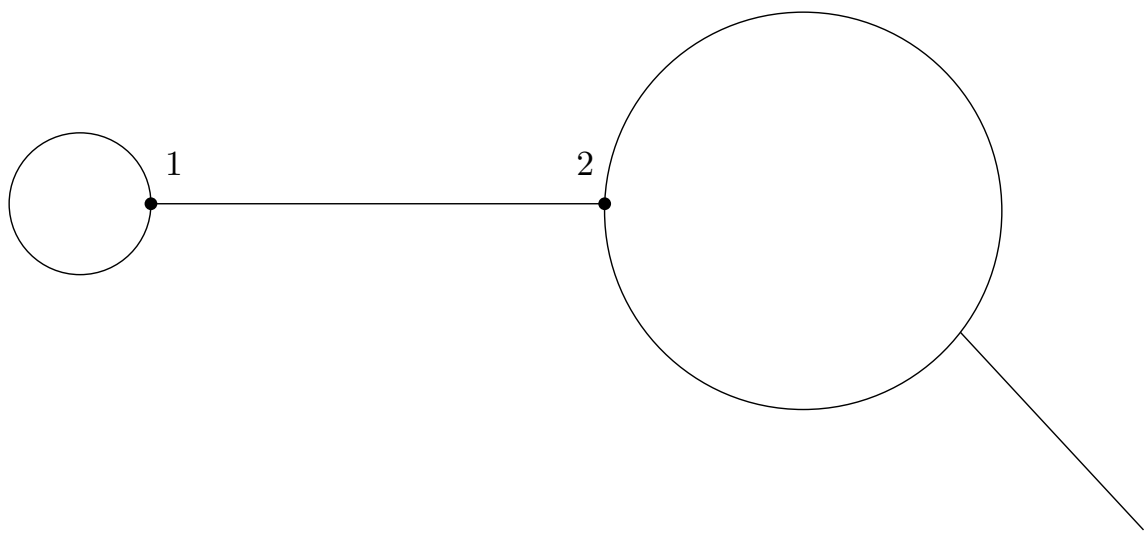
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



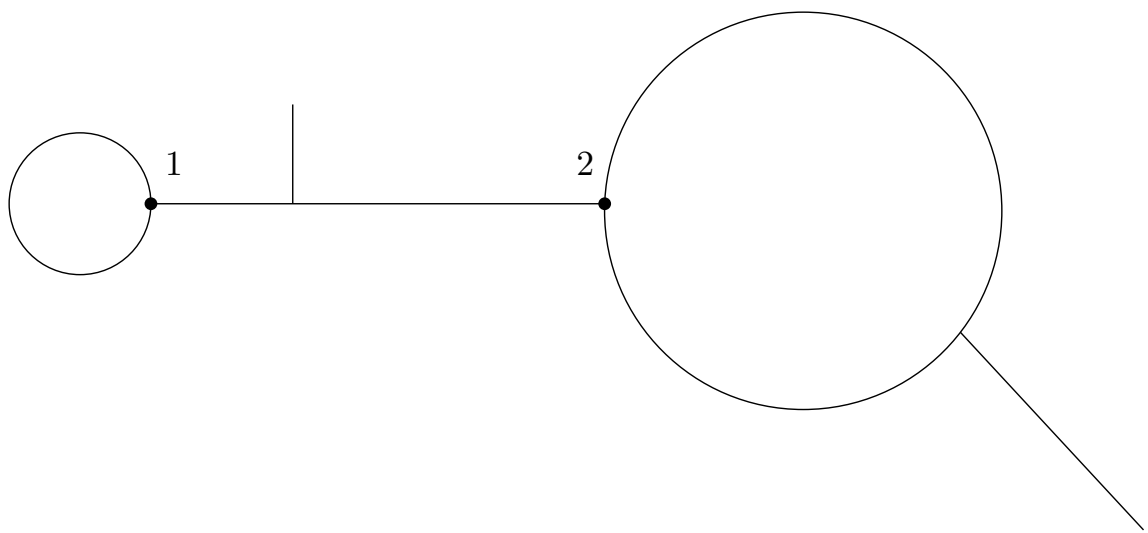
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



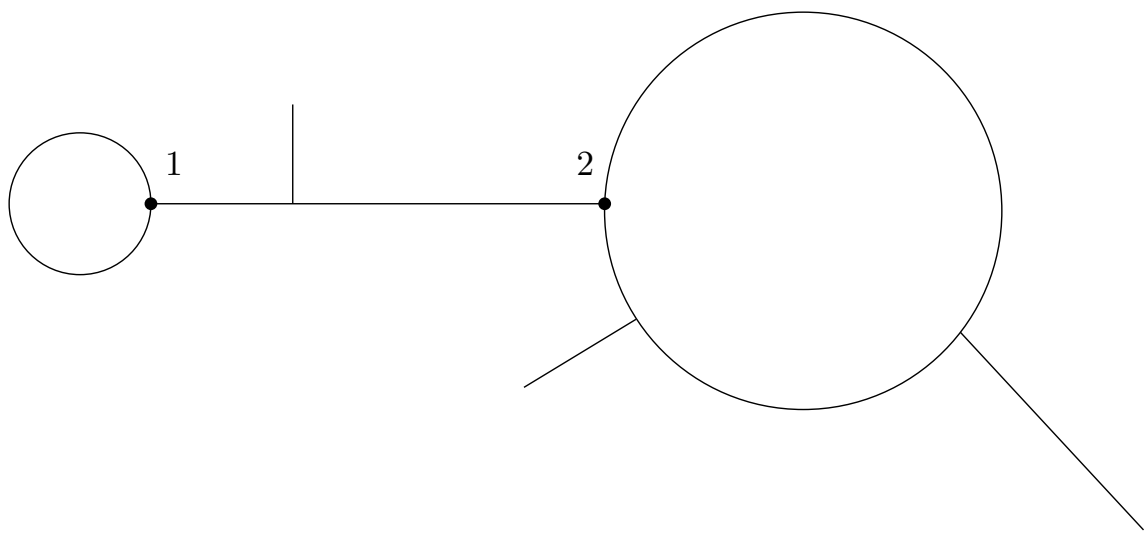
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



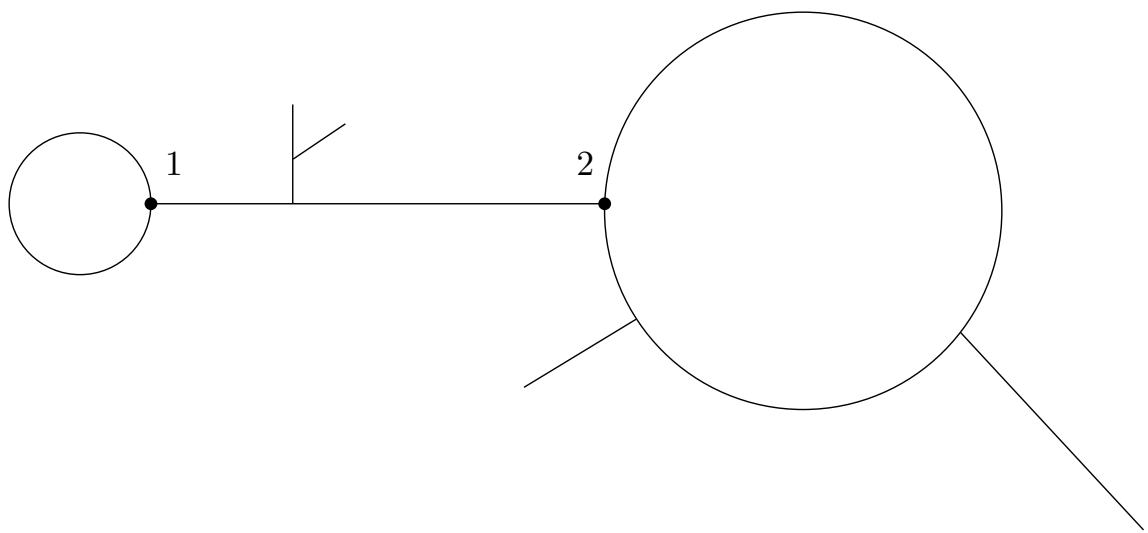
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



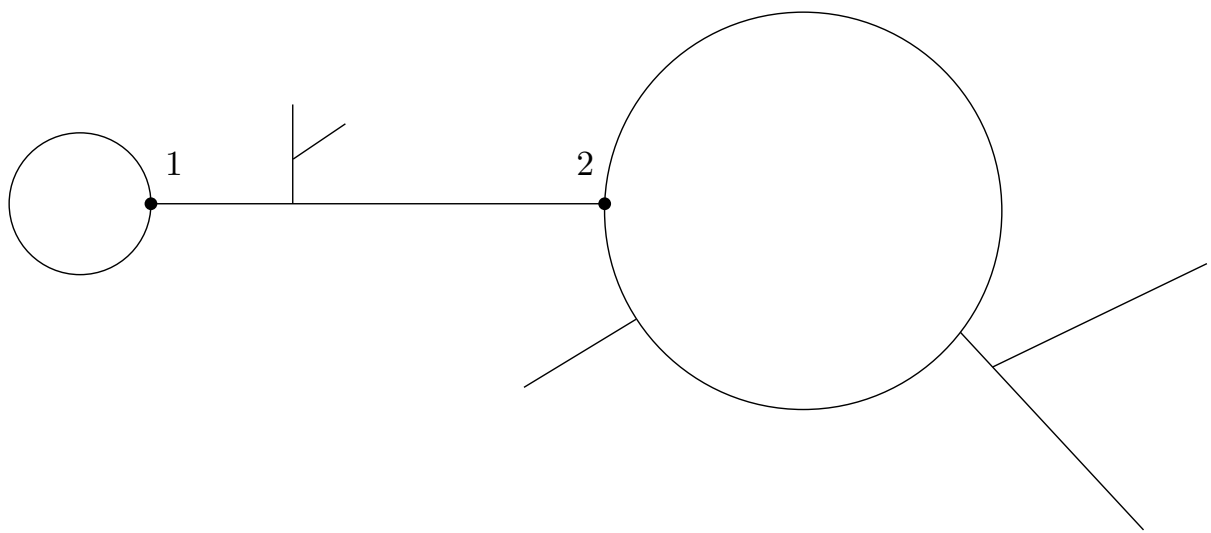
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



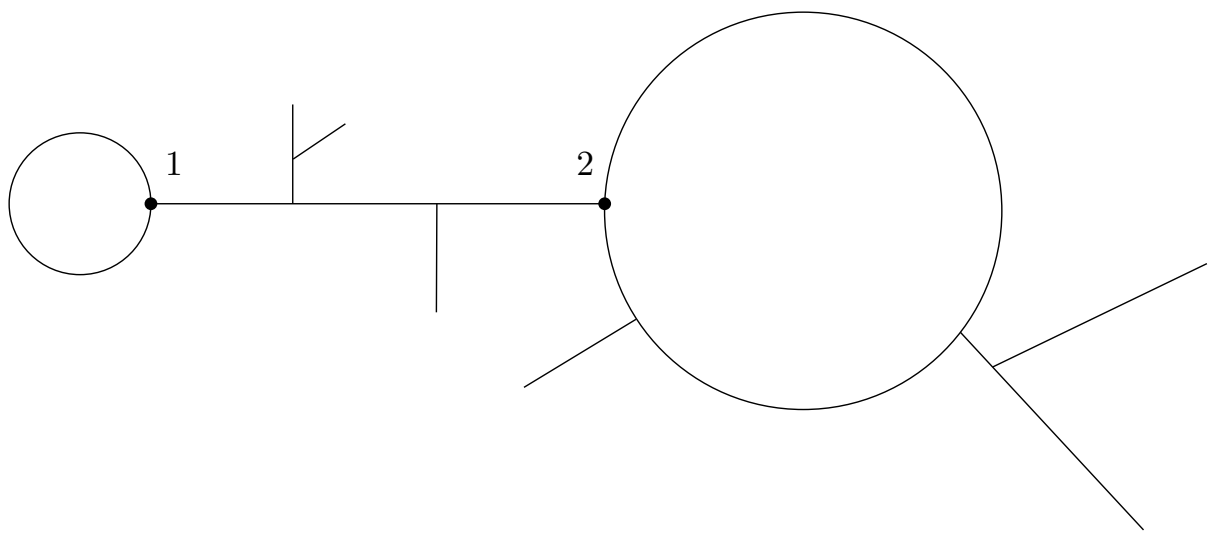
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



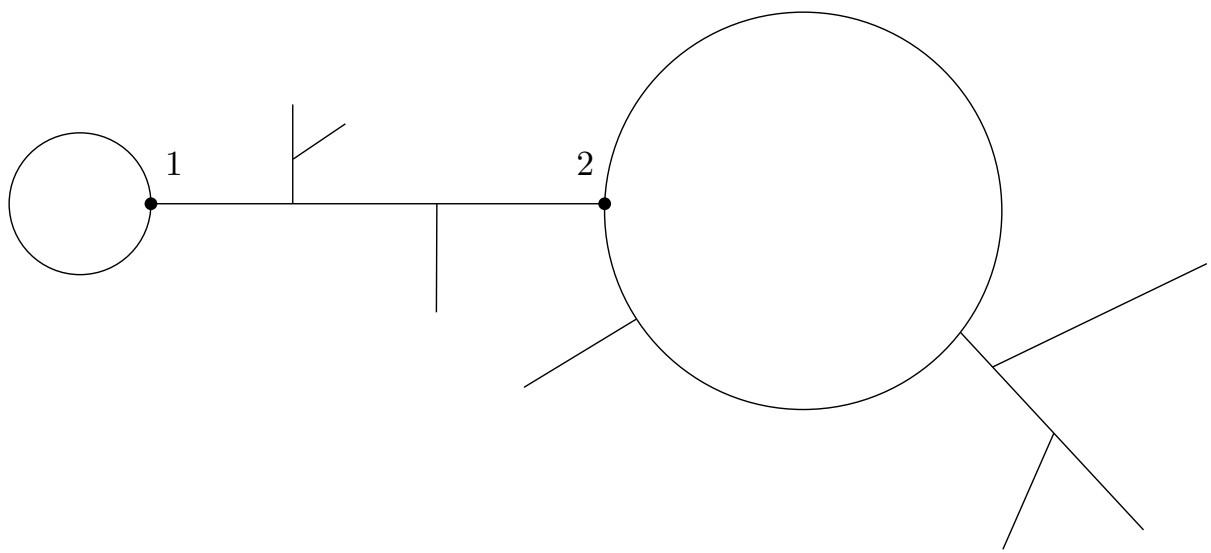
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



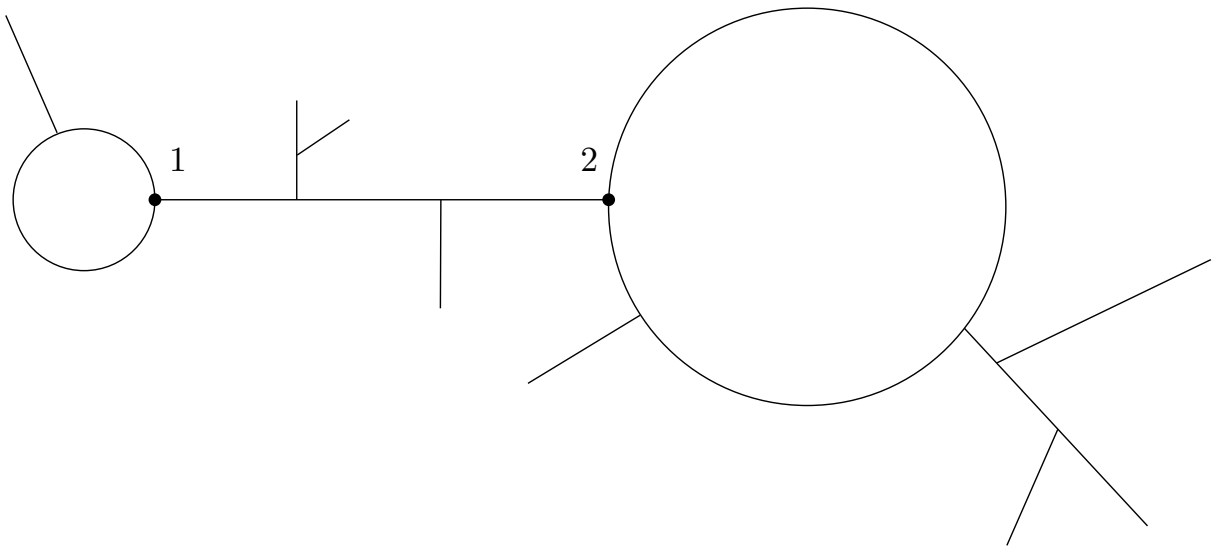
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



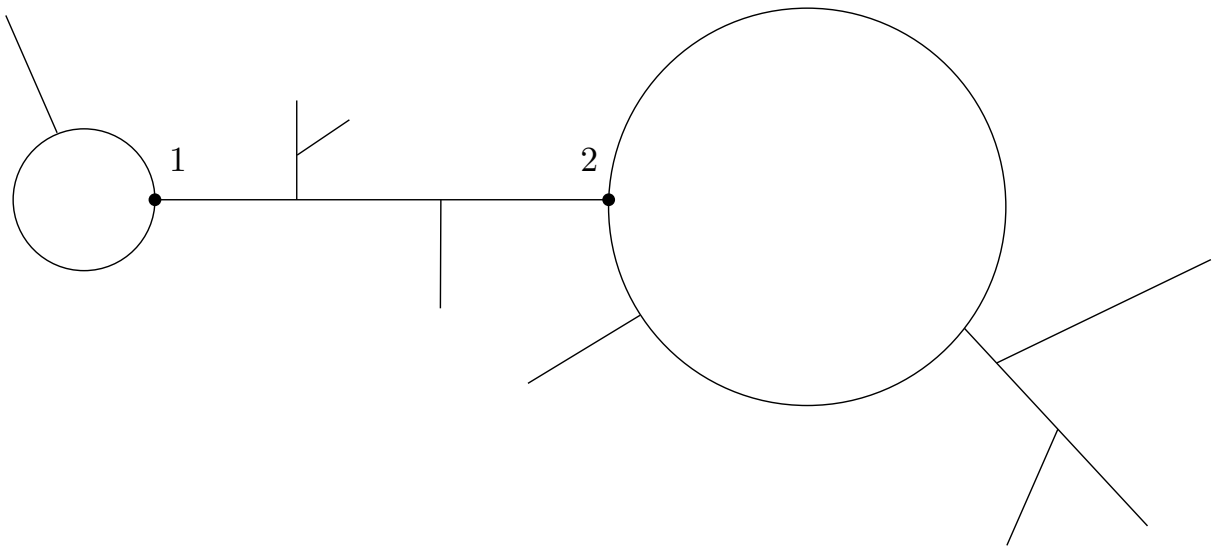
Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.



Construction of $\mathcal{U}^s$ for $s \geq 2$. Step 2: Line-breaking.

- Sample $E_1, E_2, \ldots$ independent Exp(1/2) random variables. Let $C_0 = \sqrt{\Gamma}$ and, for $j \geq 1$, let $C_j = \sqrt{\Gamma + \sum_{i=1}^{j} E_i}$.
- Now perform line-breaking starting from the core, and recursively gluing line-segments $[C_j, C_{j+1}]$ uniformly over the structure built so far.
- Do this forever, then take the completion of the limit; the result is $\mathcal{U}^s$.



Construction of $\mathcal{U}_s$ for $s = 1$.

Needs to be handled separately because the kernel is empty.

Step 1: The core

- Start from a lollipop shape  of total length $\sqrt{\Gamma}$ where $\Gamma \sim \text{Gamma}(\frac{3}{2}, \frac{1}{2})$.
- Split that length uniformly between the cycle and the stick.

Step 2: Line-breaking

- This step works exactly as before.

Scaling limit

Putting this together with Aldous' theorem yields a scaling limit for the whole critical Erdős–Rényi random graph.

Let $G_1^n, G_2^n, \ldots$ be the components of $G(n, 1/n)$ in decreasing order of size.

Theorem. (A.-B., Broutin and G. (2010,2012))
As $n \to \infty$,

$$\frac{1}{n^{1/3}}(G_1^n, G_2^n, \ldots) \xrightarrow{d} (\mathcal{G}_1, \mathcal{G}_2, \ldots)$$

where $\mathcal{G}_1, \mathcal{G}_2, \ldots$ are conditionally independent given the sizes $(C_1, C_2, \ldots)$ and surpluses $(S_1, S_2, \ldots)$ with $\mathcal{G}_i \stackrel{d}{=} \sqrt{C_i}\, \mathcal{U}^{S_i}$.

Probab. Theory Relat. Fields (2012) 152:367–406
DOI 10.1007/s00440-010-0325-4

The continuum limit of critical random graphs

L. Addario-Berry · N. Broutin · C. Goldschmidt

Received: 30 November 2009 / Revised: 30 June 2010 / Published online: 12 October 2010

Abstract We consider the Erdős–Rényi random graph $G(n, p)$ inside the critical window, that is when $p = 1/n + \lambda n^{-4/3}$, for some fixed $\lambda \in \mathbb{R}$. We prove that the sequence of connected components of $G(n, p)$, considered as metric spaces using the graph distance rescaled by $n^{-1/3}$, converges towards a sequence of continuous compact metric spaces. The result relies on a bijection between graphs and certain marked random walks, and the theory of continuum random trees. Our result gives access to the answers to a great many questions about distances in critical random graphs. In particular, we deduce that the diameter of $G(n, p)$ rescaled by $n^{-1/3}$ converges in distribution to an absolutely continuous random variable with finite mean.

Keywords Random graphs · Gromov–Hausdorff distance · Scaling limits · Continuum random tree · Diameter

Mathematics Subject Classification (2000) 05C80 · 60C05

Electronic Journal of Probability

Vol. 15 (2010), Paper no. 25, pages 741–775.

Journal URL
http://www.math.washington.edu/~ejpecp/

Critical random graphs: limiting constructions and distributional properties*†

L. Addario-Berry
Department of Mathematics and Statistics
McGill University
805 Sherbrooke W.
Montréal, QC, H3A 2K6, Canada
louigi@gmail.com

N. Broutin
Projet Algorithms
INRIA Rocquencourt
78153 Le Chesnay
France
nicolas.broutin@inria.fr

C. Goldschmidt
Department of Statistics
University of Warwick
Coventry CV4 7AL UK
C.A.Goldschmidt@warwick.ac.uk

Abstract

We consider the Erdős–Rényi random graph $G(n,p)$ inside the critical window, where $p = 1/n + \lambda n^{-4/3}$ for some $\lambda \in \mathbb{R}$. We proved in [1] that considering the connected components of $G(n,p)$ as a sequence of metric spaces with the graph distance rescaled by $n^{-1/3}$ and letting $n \to \infty$ yields a non-trivial sequence of limit metric spaces $\mathscr{C} = (\mathscr{C}_1, \mathscr{C}_2, \dots)$. These limit metric spaces can be constructed from certain random real trees with vertex-identifications. For a single such metric space, we give here two equivalent constructions, both of which are in terms of more standard probabilistic objects. The first is a global construction using Dirichlet random variables and Aldous' Brownian continuum random tree. The second is a recursive construction from an inhomogeneous Poisson point process on $\mathbb{R}_+$. These constructions allow us to characterize the distributions of the masses and lengths in the constituent parts of a limit component when it is decomposed according to its cycle structure. In particular, this strengthens results of Łuczak et al. [29] by providing precise distributional convergence for the lengths of paths between kernel vertices and the length of a shortest cycle, within any fixed limit component.

MSC 2000 subject classifications: primary 05C80; secondary 60C05.

†L.A.B. was supported by an NSERC Discovery Grant throughout the research and writing of this paper. C.G. was funded by EPSRC Postdoctoral Fellowship EP/D065755/1.

Timeline

- Jean-François Le Gall's seminar in October 2007.
- Project crystallized in spring 2008.
- LAB moved back to Montréal in summer 2008.
- Major progress in September 2008 during a visit of NB and CG to LAB at Université de Montréal.
- Finished first paper in March 2009. Less than 2 years! Math is not fast, but this actually feels like it did come together quickly.
- (On the other hand, that paper didn't appear in print in a journal until 2012!)
- Finished follow-up paper in March 2010.

Universality

Our scaling limit has subsequently been shown to be universal, in that a whole host of different critical random graph models have essentially the same limit.

VII. Minimum Spanning Trees

- Introduced by Borůvka in 1926.

VII. Minimum Spanning Trees

- Introduced by Borůvka in 1926.
- His formulation: Given the *complete graph* K_n together with distinct non-negative *edge weights* $\{w_e, e \in E(K_n)\}$.

VII. Minimum Spanning Trees

- Introduced by Borůvka in 1926.
- His formulation: Given the *complete graph* K_n together with distinct non-negative *edge weights* $\{w_e, e \in E(K_n)\}$.
- Seek the *unique connected graph* T that minimizes the total length $\sum_{e \in E(T)} w_e$.

VII. Minimum Spanning Trees

- Introduced by Borůvka in 1926.
- His formulation: Given the *complete graph* K_n together with distinct non-negative *edge weights* $\{w_e, e \in E(K_n)\}$.
- Seek the *unique connected graph* T that minimizes the total length $\sum_{e \in E(T)} w_e$.
- Necessarily T is a tree: the *minimum spanning tree* of K_n with these weights.

VII. Minimum Spanning Trees

- Introduced by Borůvka in 1926.
- His formulation: Given the *complete graph* K_n together with distinct non-negative *edge weights* $\{w_e, e \in E(K_n)\}$.
- Seek the *unique connected graph* T that minimizes the total length $\sum_{e \in E(T)} w_e$.
- Necessarily T is a tree: the *minimum spanning tree* of K_n with these weights.
- For *random edge weights*, the structure of the minimum spanning tree of K_n is intimately connected to that of the critical Erdős-Renyi random graph.

VII. Minimum Spanning Trees

- Introduced by Borůvka in 1926.
- His formulation: Given the *complete graph* K_n together with distinct non-negative *edge weights* $\{w_e, e \in E(K_n)\}$.
- Seek the *unique connected graph* T that minimizes the total length $\sum_{e \in E(T)} w_e$.
- Necessarily T is a tree: the *minimum spanning tree* of K_n with these weights.
- For *random edge weights*, the structure of the minimum spanning tree of K_n is intimately connected to that of the critical Erdős-Renyi random graph.
- We used this connection to study the metric space limit of the minimum spanning tree.

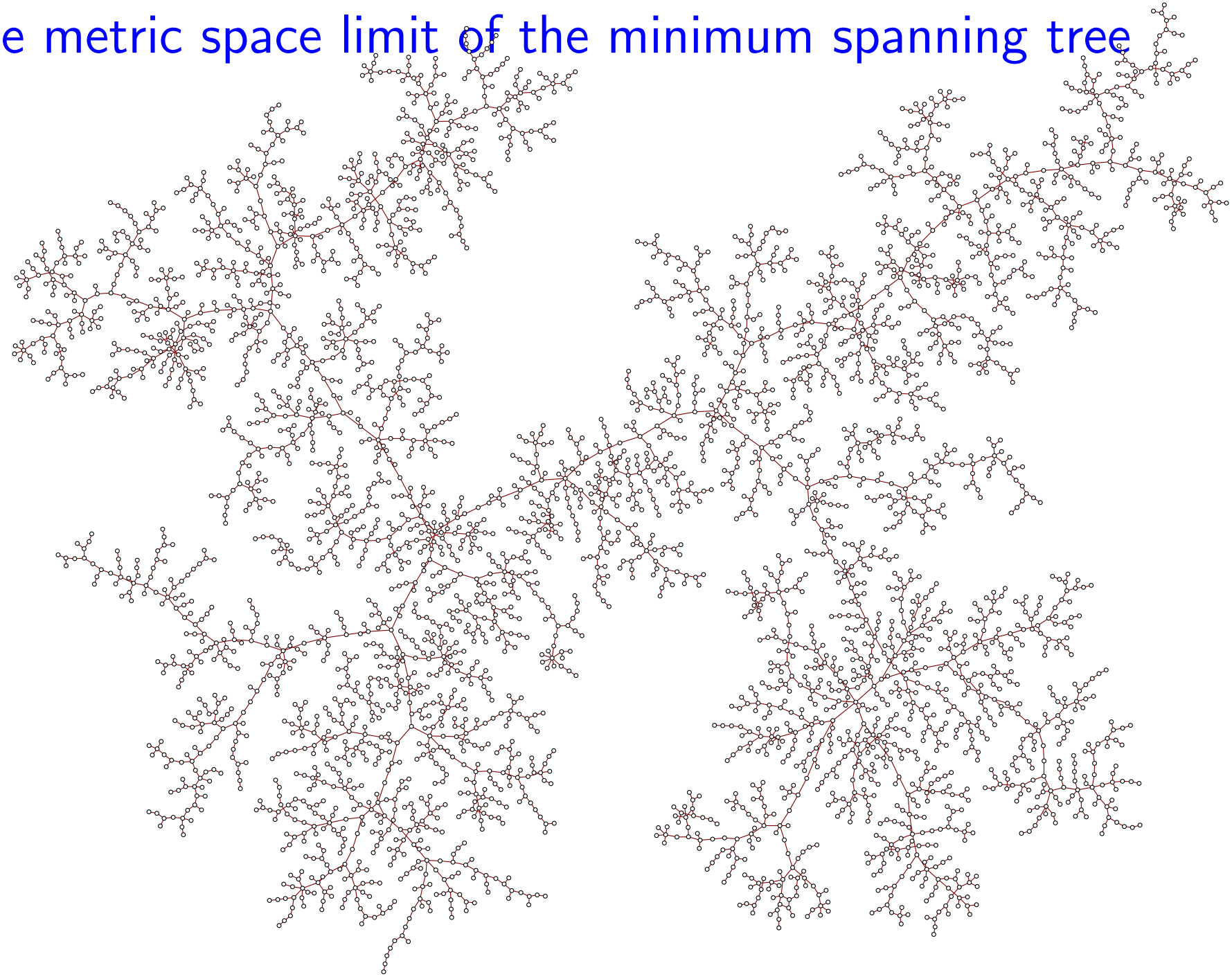
The metric space limit of the minimum spanning tree

- Set-up: K_n, independent Uniform$[0,1]$ edge weights $\{U_e, e \in E(K_n)\}$.

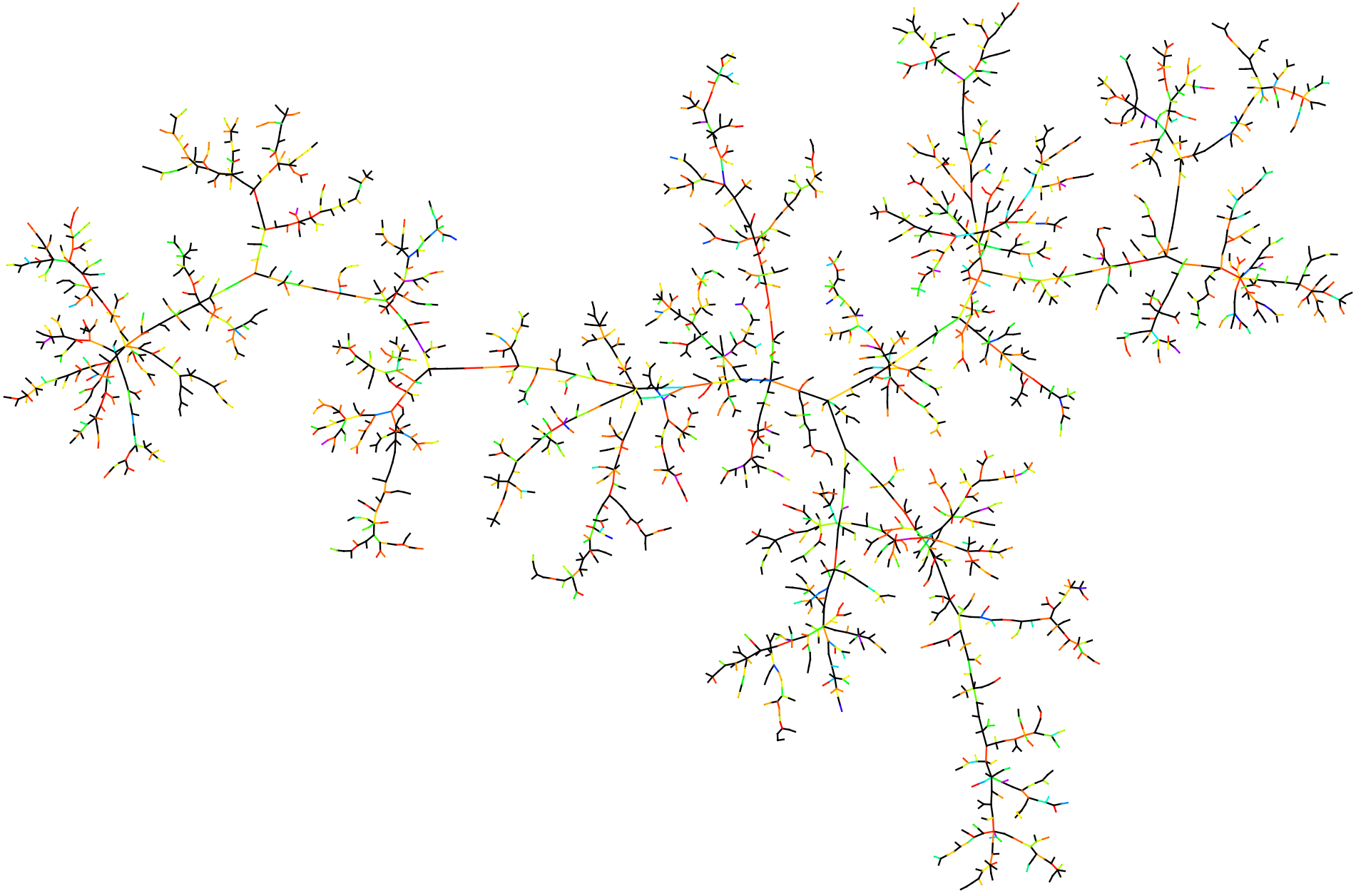
The metric space limit of the minimum spanning tree

- Set-up: K_n, independent Uniform$[0,1]$ edge weights $\{U_e, e \in E(K_n)\}$.
- Rescale the resulting MST so each edge has length $n^{-1/3}$.

The metric space limit of the minimum spanning tree



The metric space limit of the minimum spanning tree



The metric space limit of the minimum spanning tree

- Set-up: K_n, independent Uniform$[0,1]$ edge weights $\{U_e, e \in E(K_n)\}$.
- Rescale the resulting MST so each edge has length $n^{-1/3}$. Call the result M_n.

Theorem (Addario-Berry, Broutin, Goldschmidt, Miermont 2013)

As $n \to \infty$, $M_n \longrightarrow \mathcal{M}$ in distribution, for a limiting random measured metric space $\mathcal{M}$.

The metric space limit of the minimum spanning tree

- Set-up: K_n, independent Uniform$[0,1]$ edge weights $\{U_e, e \in E(K_n)\}$.
- Rescale the resulting MST so each edge has length $n^{-1/3}$. Call the result M_n.

Theorem (Addario-Berry, Broutin, Goldschmidt, Miermont 2013)

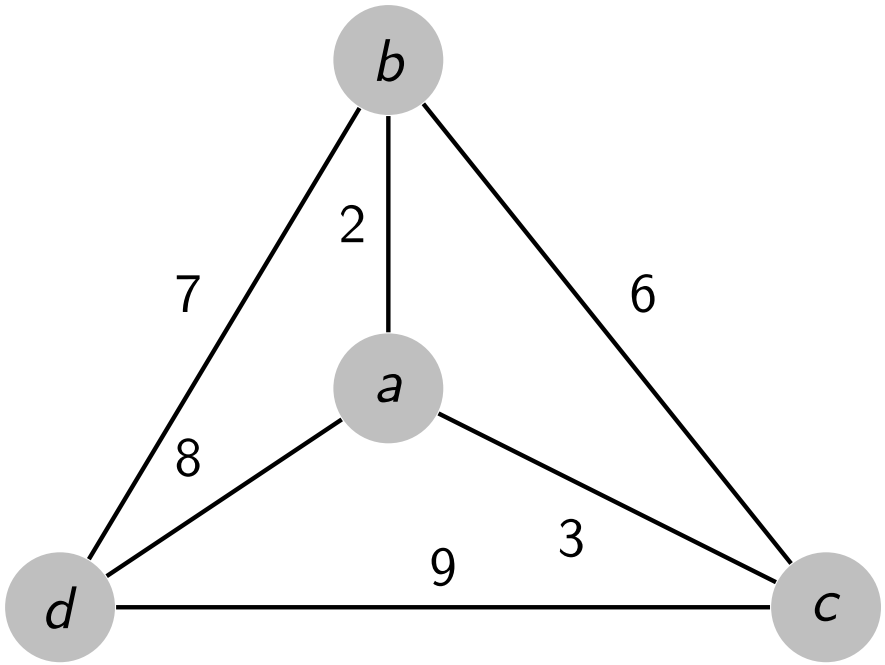
As $n \to \infty$, $M_n \longrightarrow \mathcal{M}$ in distribution, for a limiting random measured metric space $\mathcal{M}$.

- Work started in 2011; published in 2017

Grégory Miermont

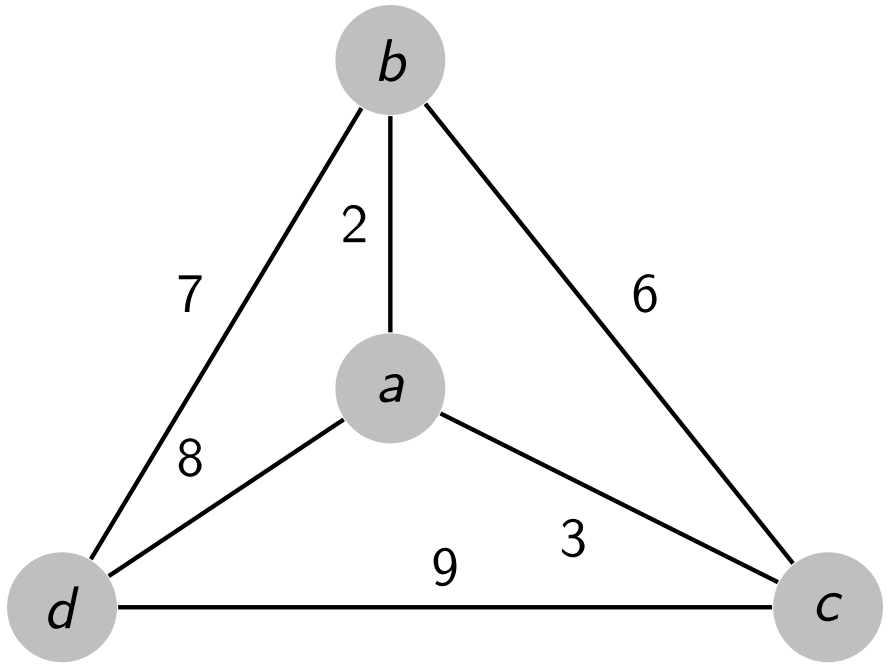
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.



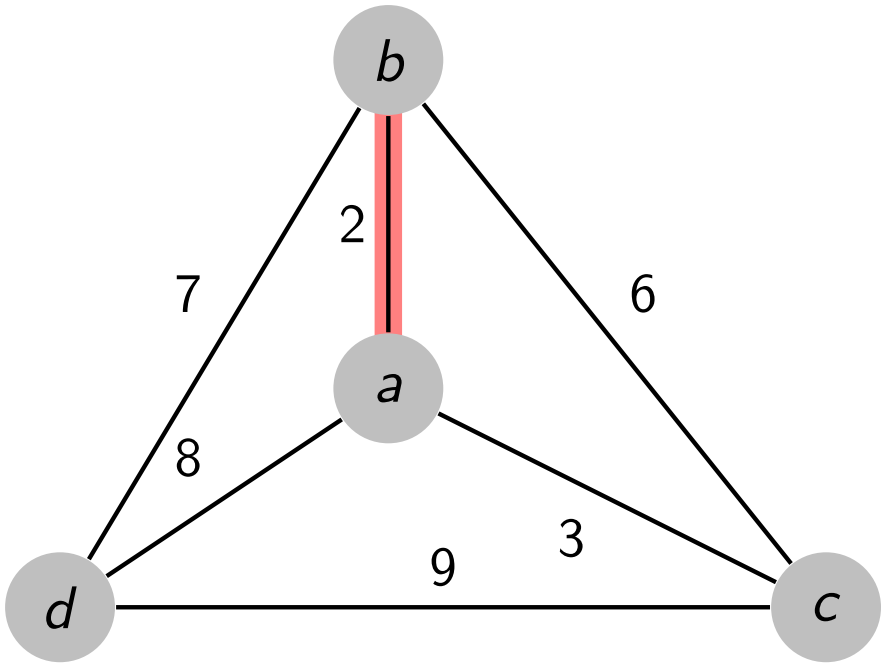
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



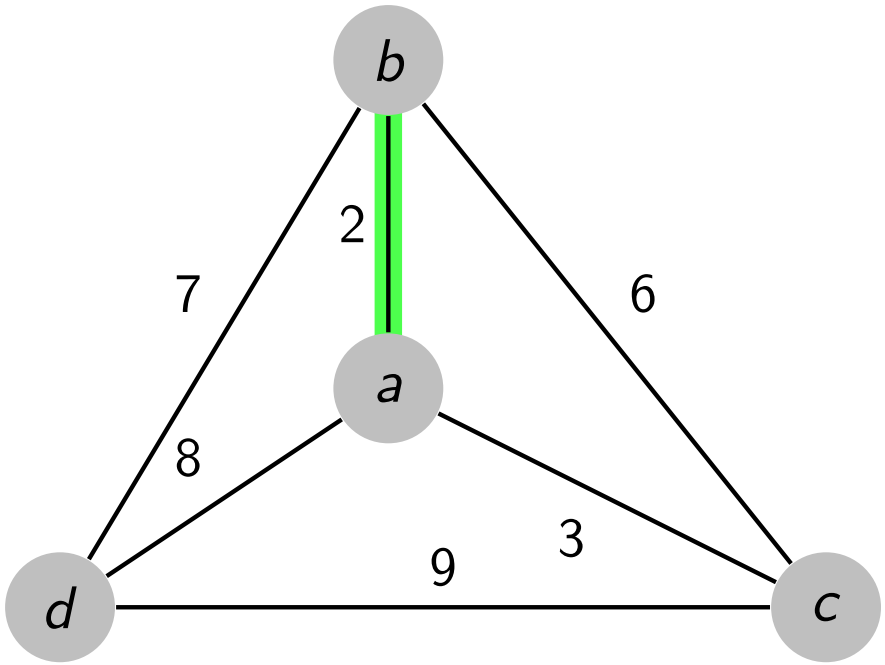
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



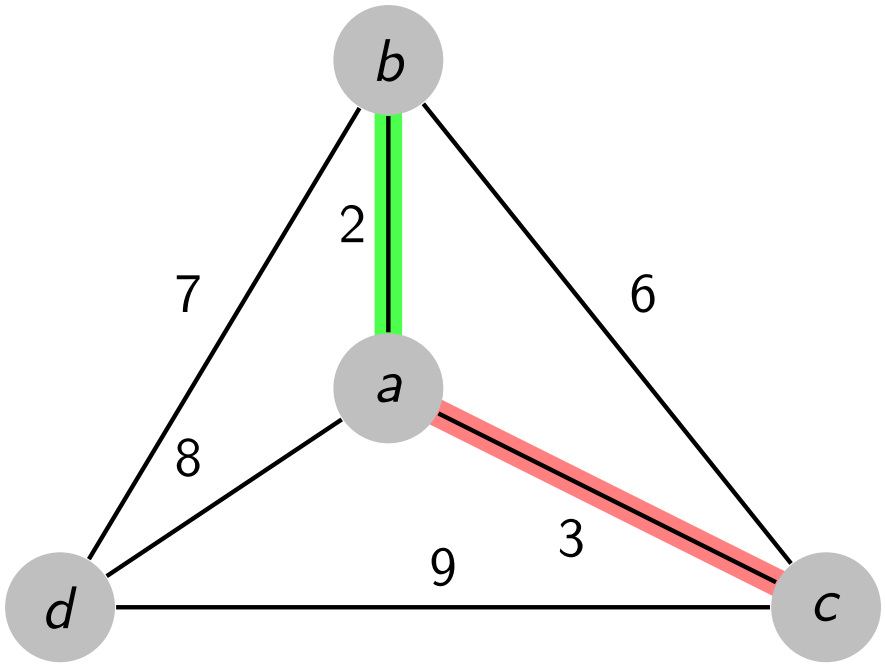
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



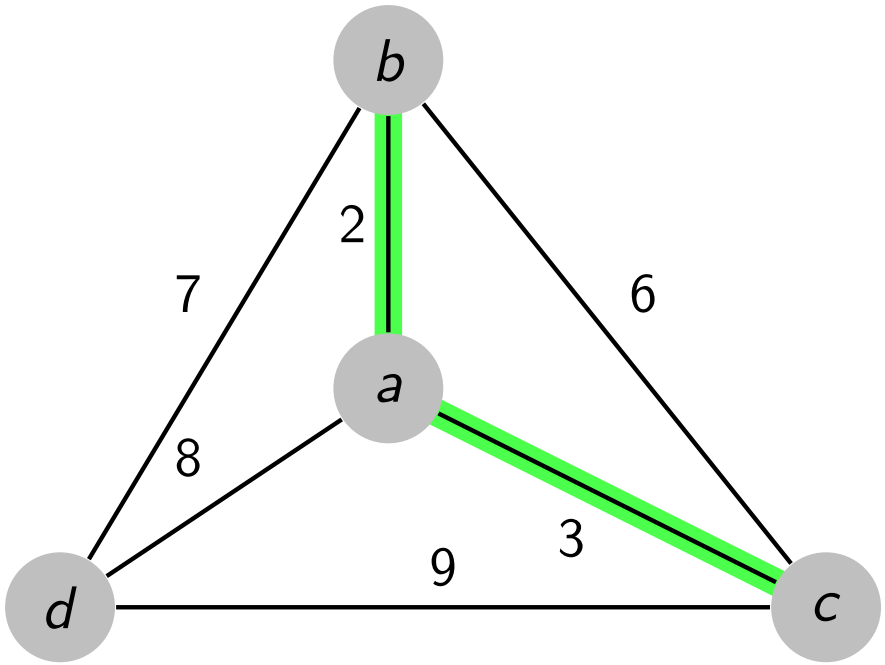
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



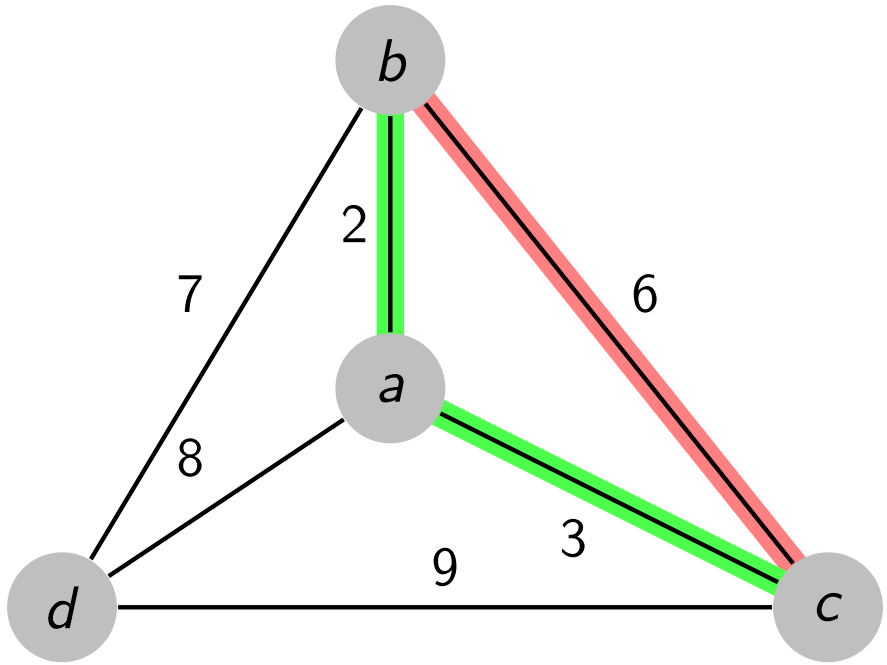
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



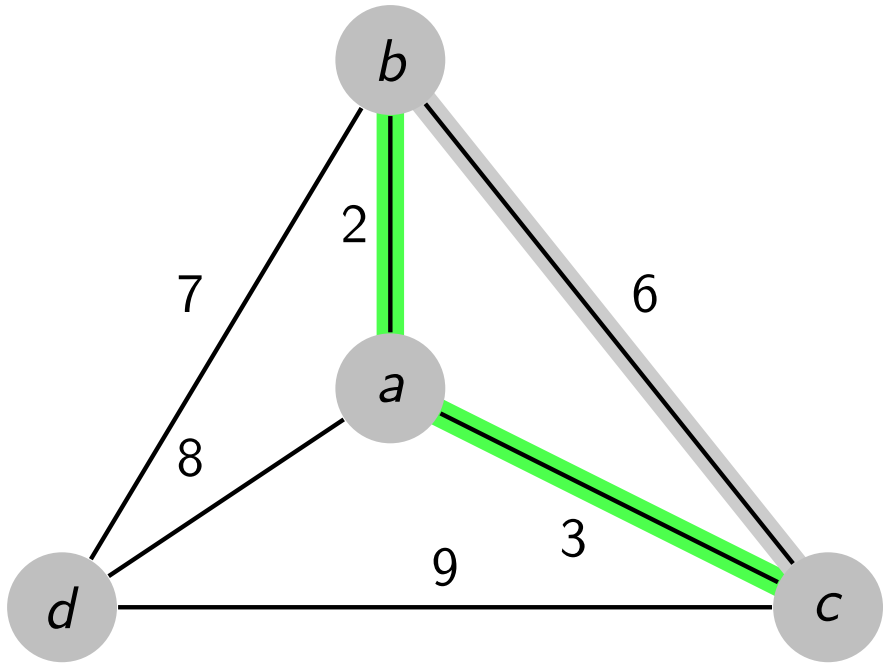
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



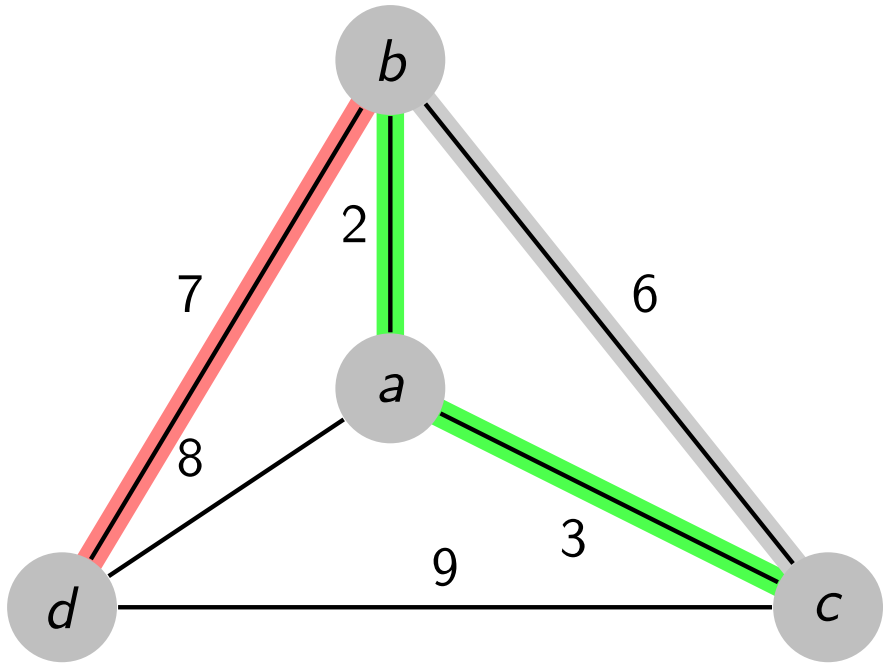
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



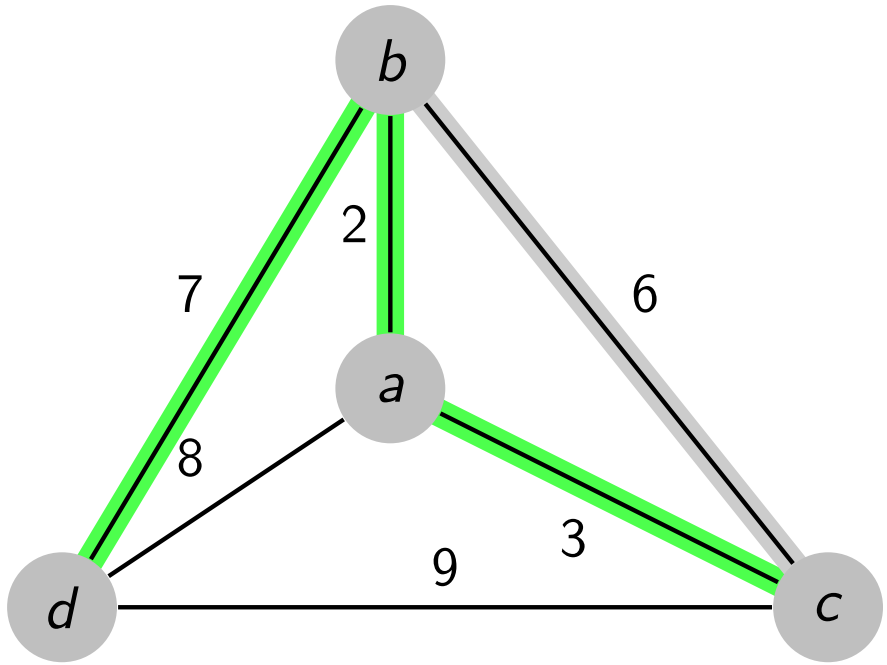
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



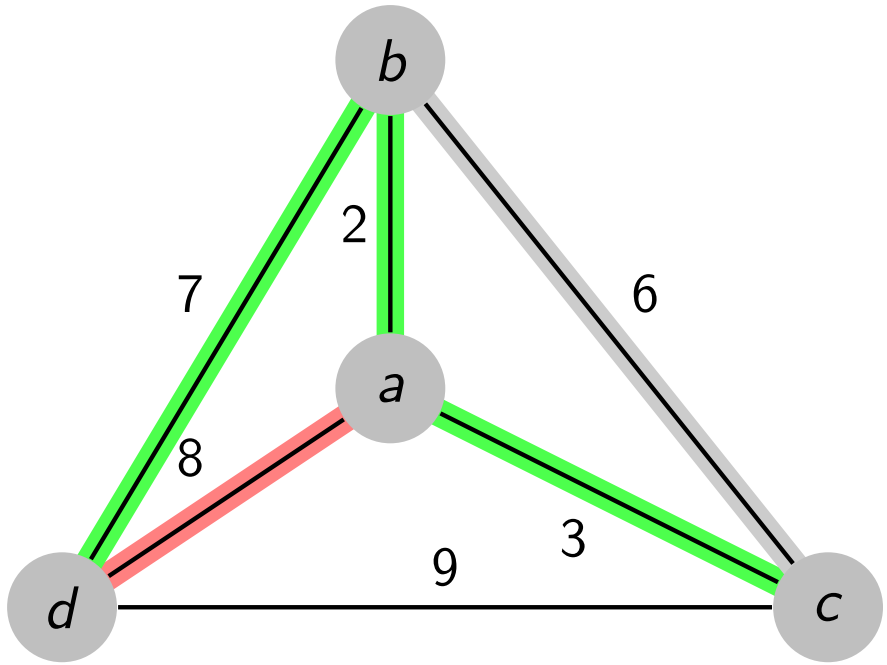
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



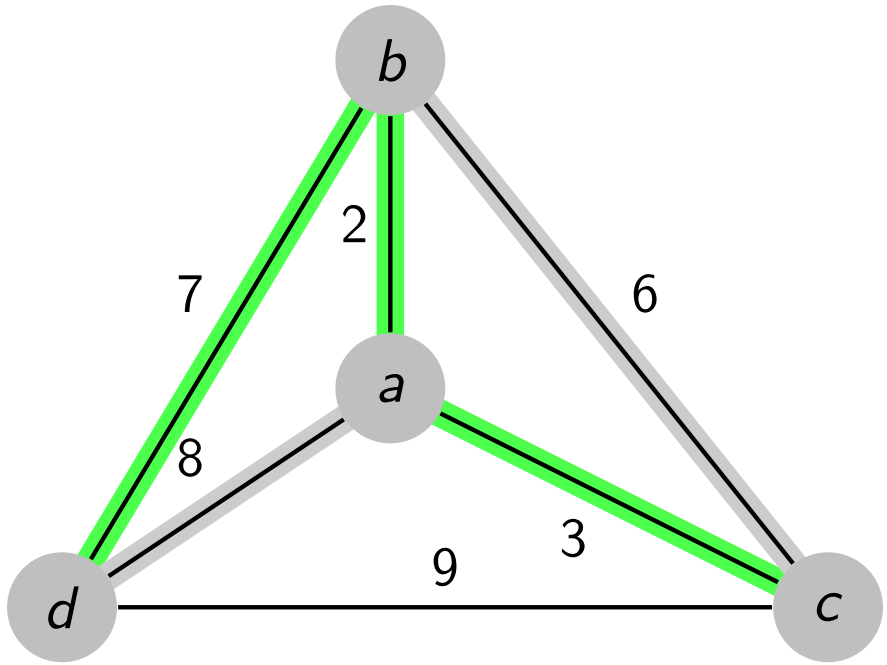
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



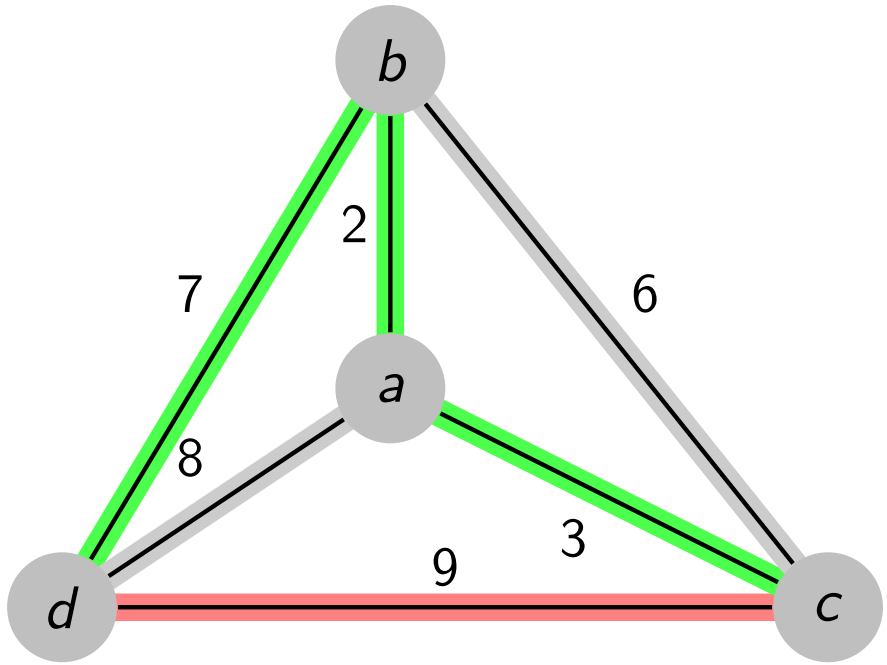
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



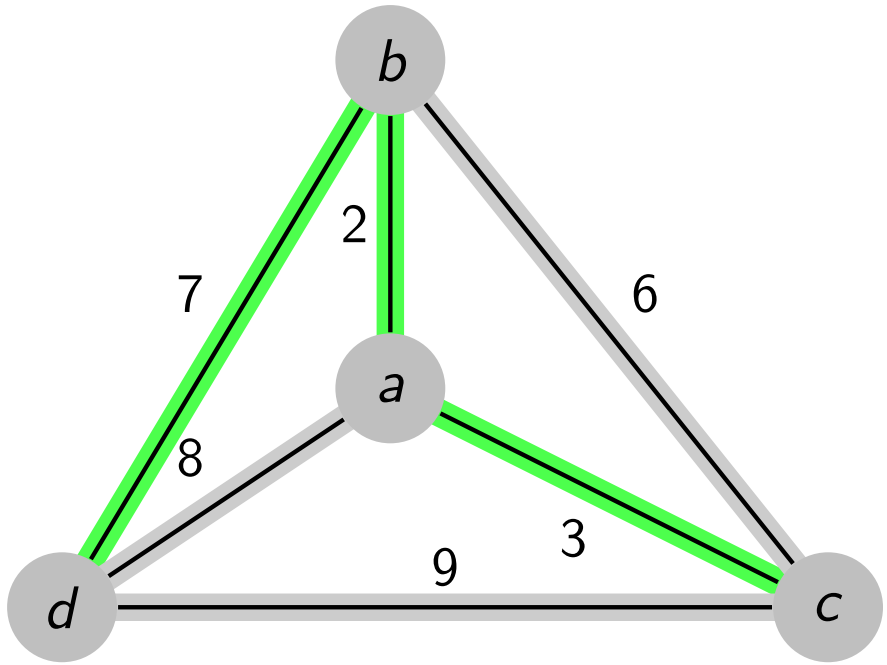
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.



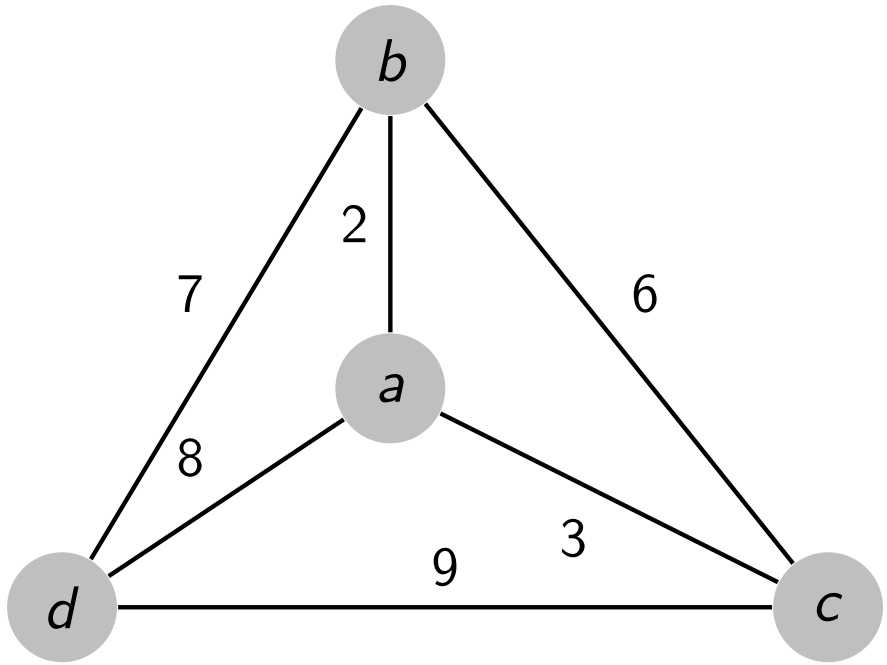
Kruskal's minimum spanning tree algorithm

- Order edges in increasing order of weight as $e_1, \ldots, e_m$.
- For each i, add edge e_i unless doing so would create a cycle.
- The result is the minimum spanning tree.



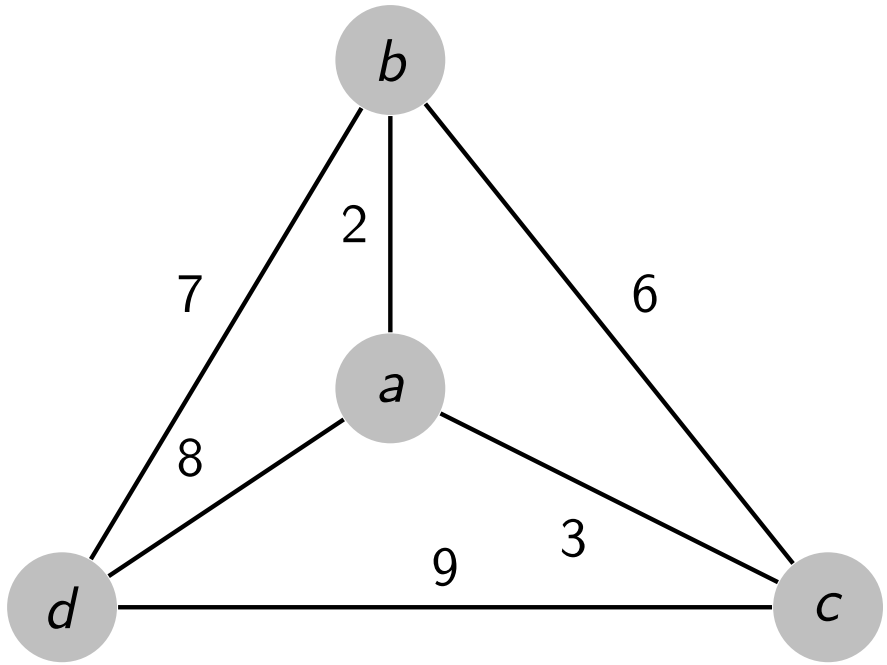
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.



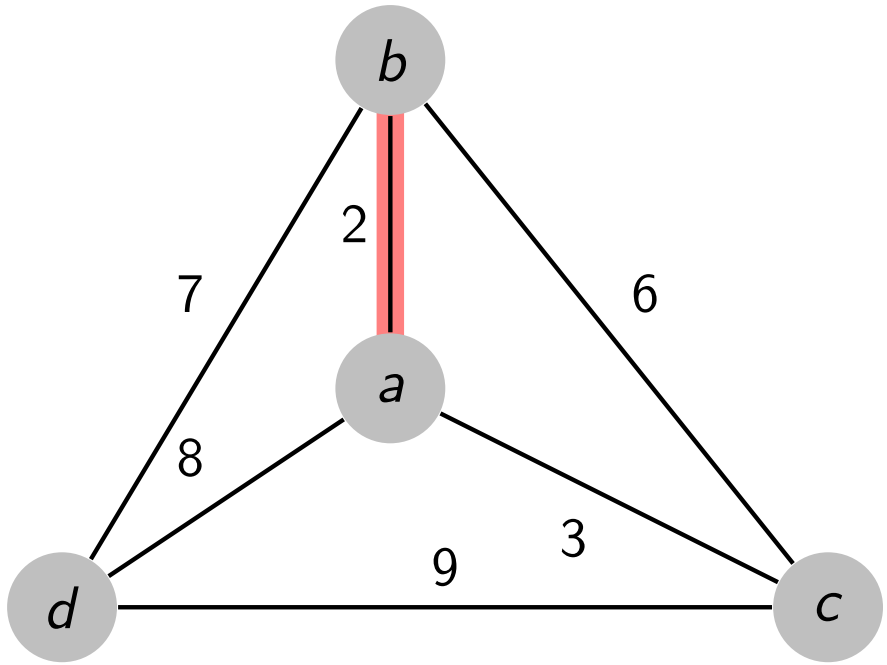
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



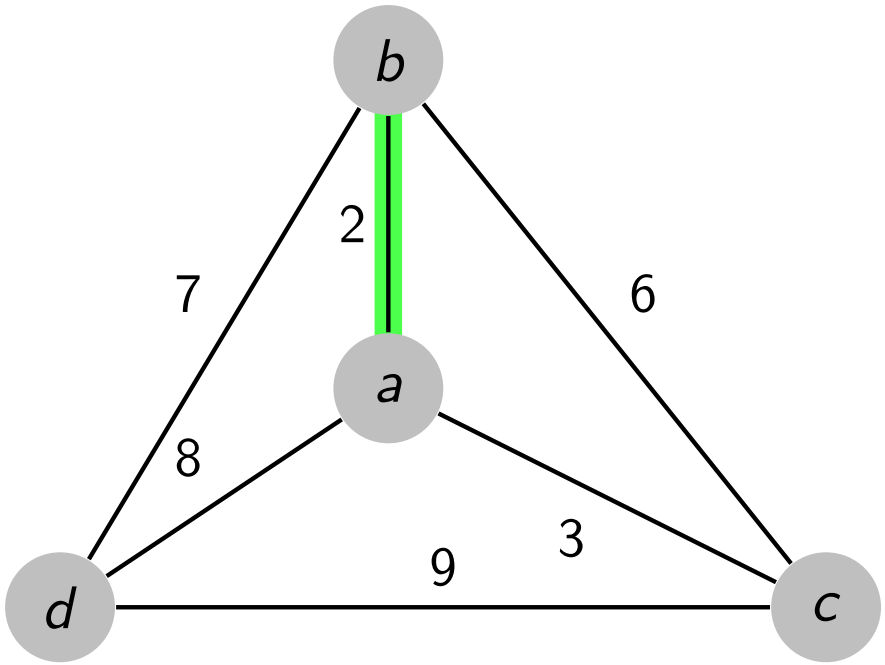
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



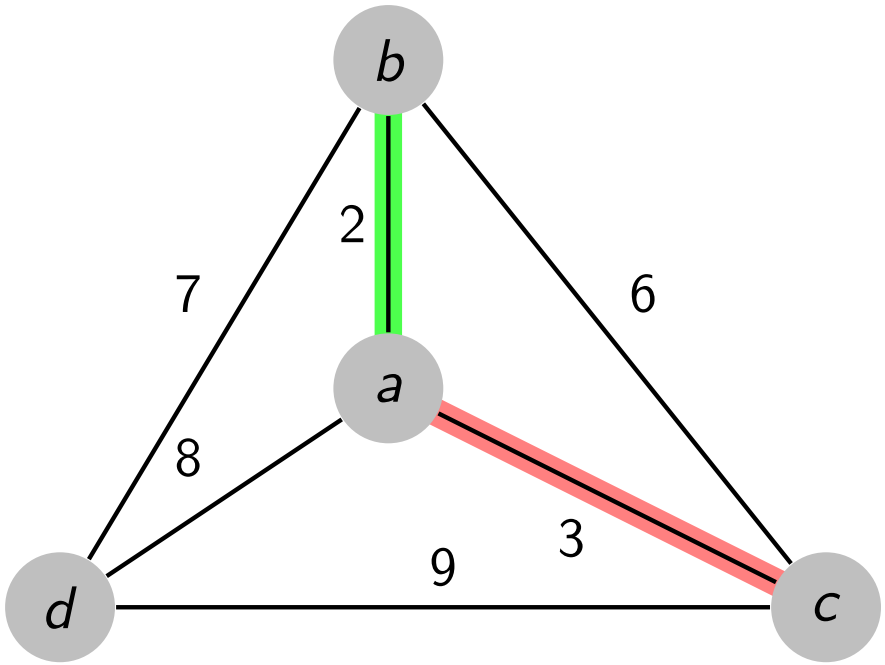
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



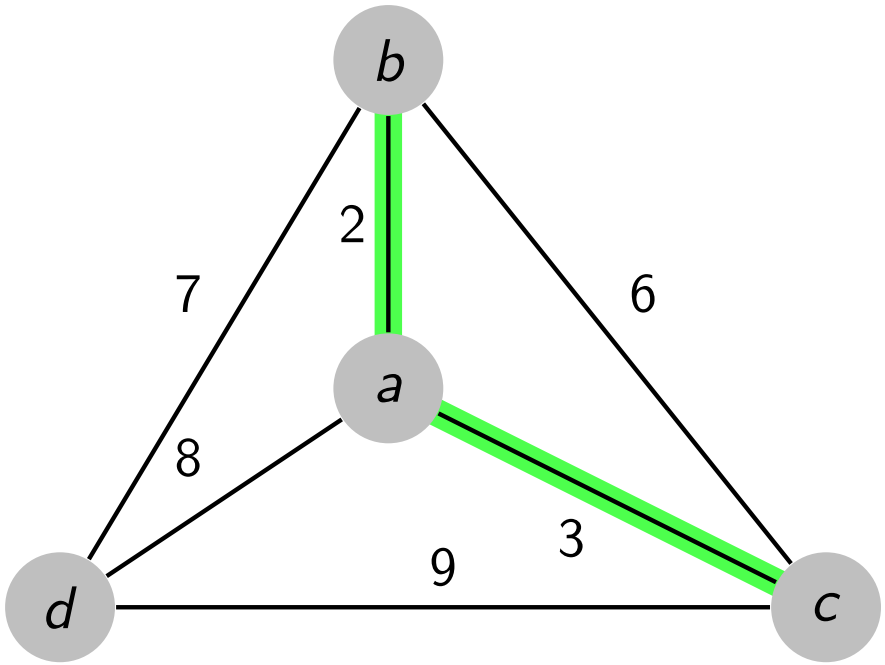
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



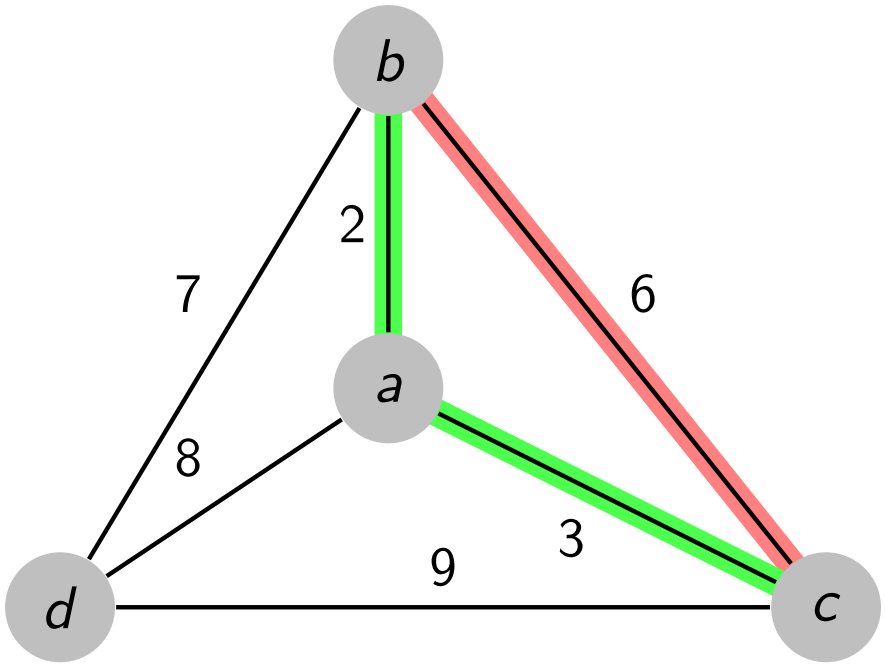
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



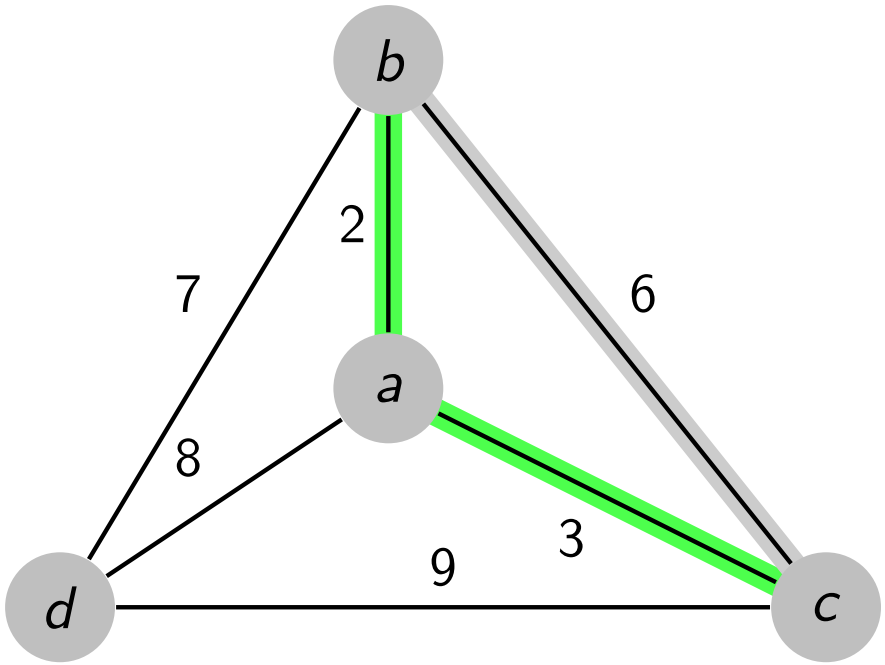
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



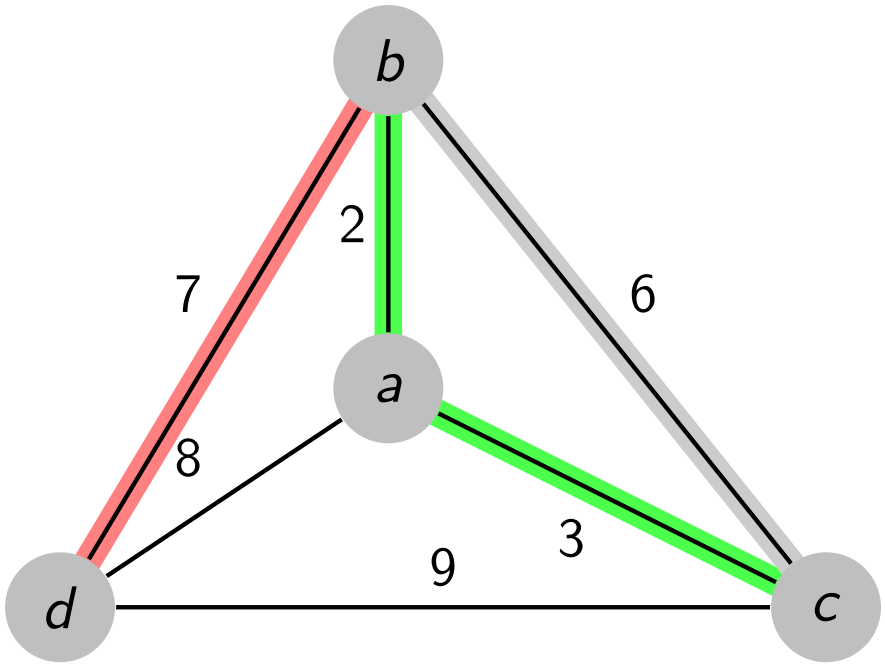
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



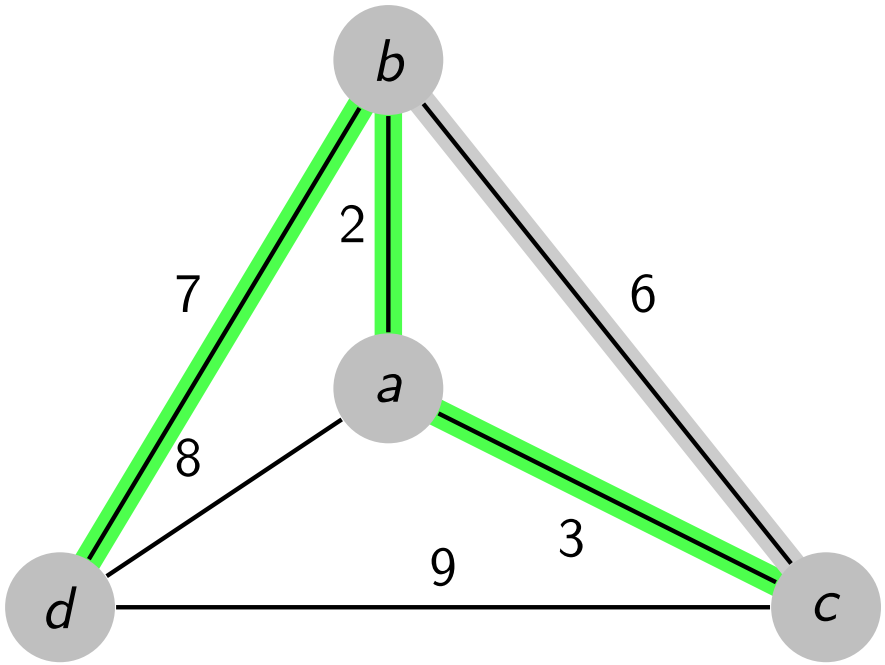
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



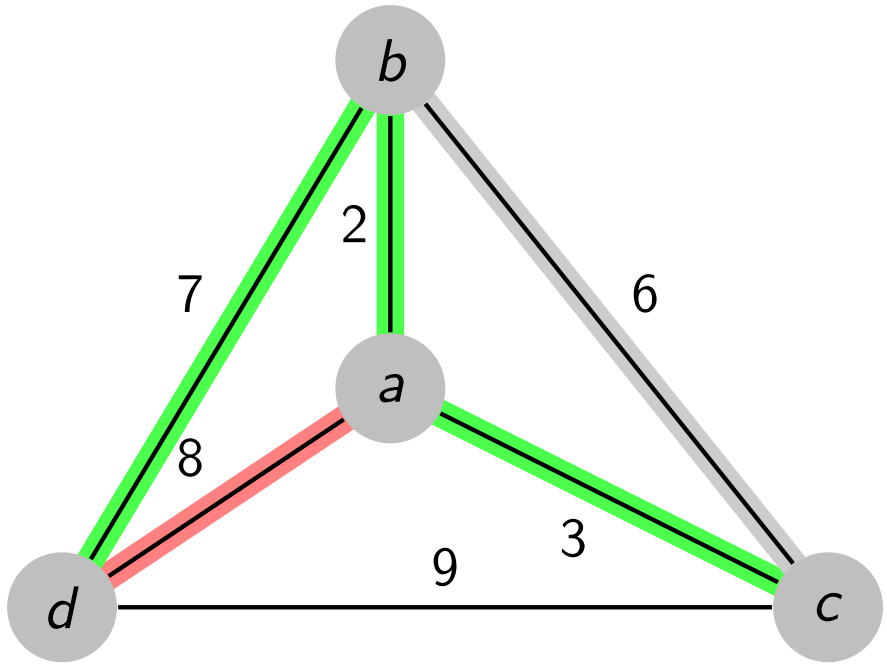
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



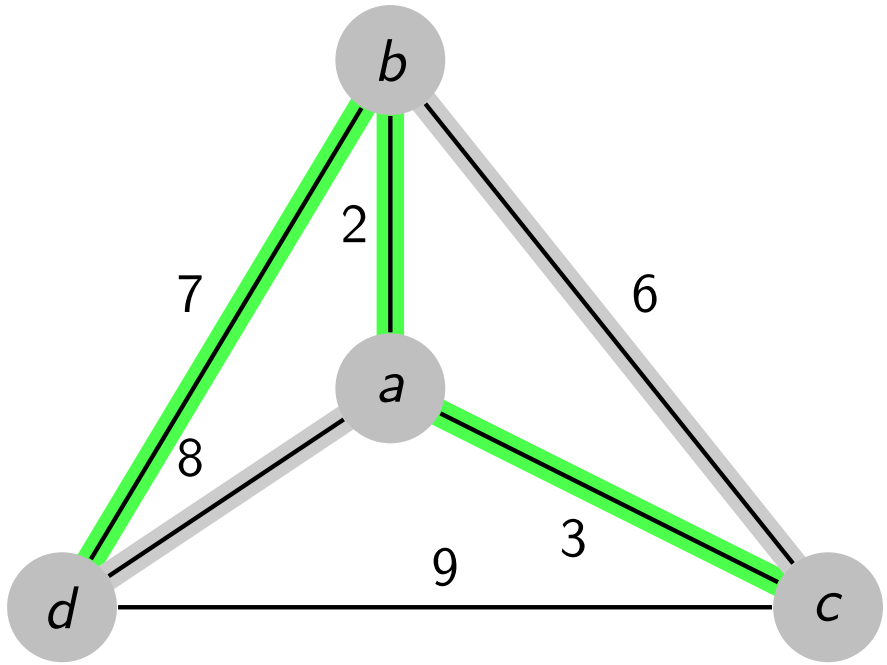
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



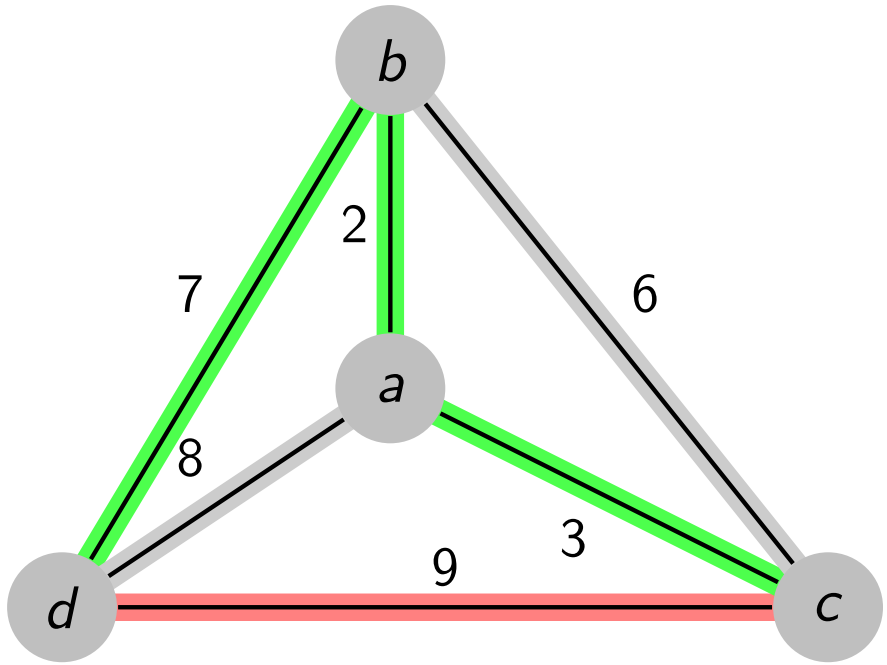
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



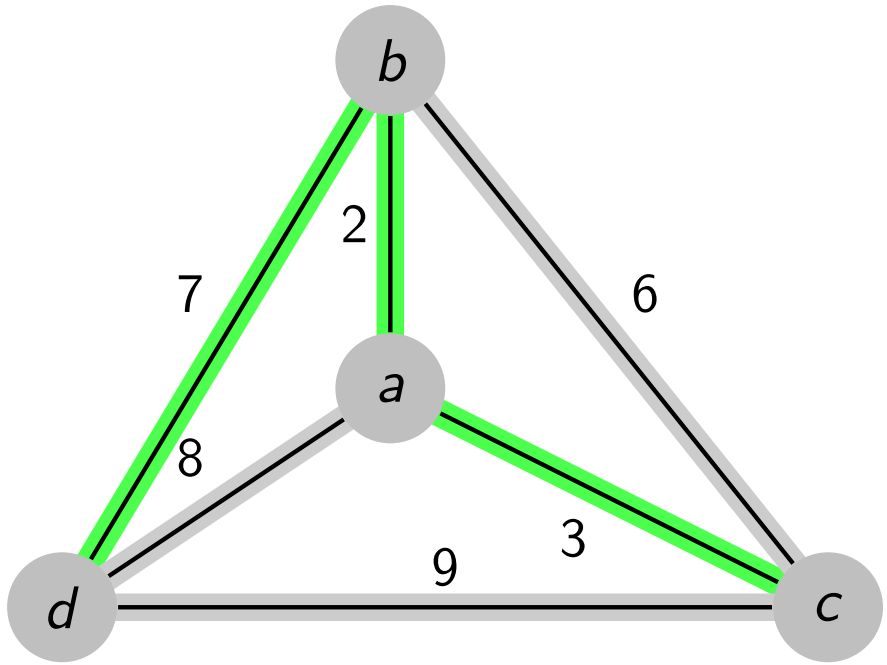
Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



Kruskal's minimum spanning tree algorithm

- Any time an edge is not added, its endpoints are in the same component.
- So at every step, the components are the same as if we added *all* edges.



Kruskal - Percolation coupling

Recall set-up: K_n, weights $\{U_e, e \in E(K_n)\}$ are iid Uniform$[0,1]$.

Kruskal - Percolation coupling

Recall set-up: K_n, weights $\{U_e, e \in E(K_n)\}$ are iid Uniform$[0,1]$.

- $G(n,p)$: only keep edges of K_n of weight less than p.

Kruskal - Percolation coupling

Recall set-up: K_n, weights $\{U_e, e \in E(K_n)\}$ are iid Uniform$[0,1]$.

- $G(n,p)$: only keep edges of K_n of weight less than p.
- $M(n,p)$: only keep edges of M_n of weight less than p.

Kruskal - Percolation coupling

Recall set-up: K_n, weights $\{U_e, e \in E(K_n)\}$ are iid Uniform$[0,1]$.

- $G(n,p)$: only keep edges of K_n of weight less than p.
- $M(n,p)$: only keep edges of M_n of weight less than p.
- Key property: $G(n,p)$ and $M(n,p)$ have the same components.

Kruskal - Percolation coupling

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

Kruskal - Percolation coupling

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

Everything interesting occurs when $p \sim 1/n$.

Kruskal - Percolation coupling

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

Everything interesting occurs when $p \sim 1/n$.

- $G(n, 0.99/n) \to$ all components have size $O(\log n)$.

Kruskal - Percolation coupling

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

Everything interesting occurs when $p \sim 1/n$.

- $G(n, 0.99/n) \to$ all components have size $O(\log n)$.
- $G(n, 1.01/n) \to$ unique "giant" component, of linear size; large scale structure already formed.

Kruskal - Percolation coupling

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

Everything interesting occurs when $p \sim 1/n$.

- $G(n, 0.99/n) \to$ all components have size $O(\log n)$.
- $G(n, 1.01/n) \to$ unique "giant" component, of linear size; large scale structure already formed.
- $G(n, 1/n)$: the breakpoint. MST structure is built here.

Proof idea

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

Proof idea

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

- GHP limit of components of $G(n,1/n)$ already understood.

Proof idea

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

- GHP limit of components of $G(n,1/n)$ already understood.
- Components of $G(n,1/n)$ all have $O(1)$ cycles (most are already trees).

Proof idea

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

- GHP limit of components of $G(n,1/n)$ already understood.
- Components of $G(n,1/n)$ all have $O(1)$ cycles (most are already trees).
- The largest components of $G(n,1/n)$ have size $\Theta(n^{2/3})$ and diameter $\Theta(n^{1/3})$ – this is the reason for the scaling in the theorem.

Proof idea

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

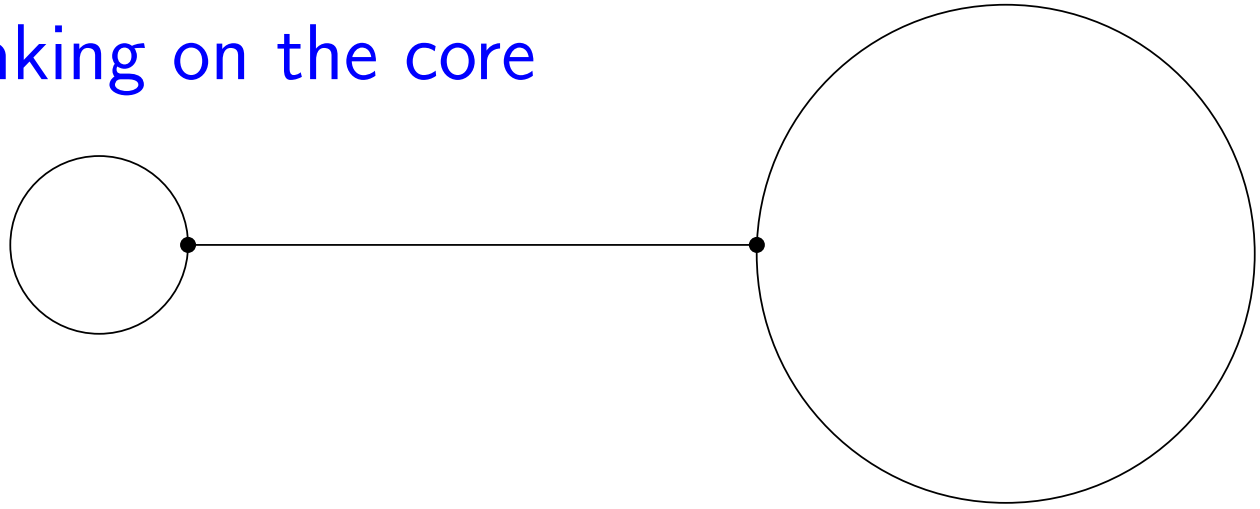
- GHP limit of components of $G(n,1/n)$ already understood.
- Components of $G(n,1/n)$ all have $O(1)$ cycles (most are already trees).
- The largest components of $G(n,1/n)$ have size $\Theta(n^{2/3})$ and diameter $\Theta(n^{1/3})$ – this is the reason for the scaling in the theorem.
- Can use an MST algorithm called *cycle breaking* to relate the structure of $M(n,1/n)$ to the (already understood) structure of $M(n,1/n)$.

Proof idea

Key property: $M(n,p)$ and $G(n,p)$ have the same components.

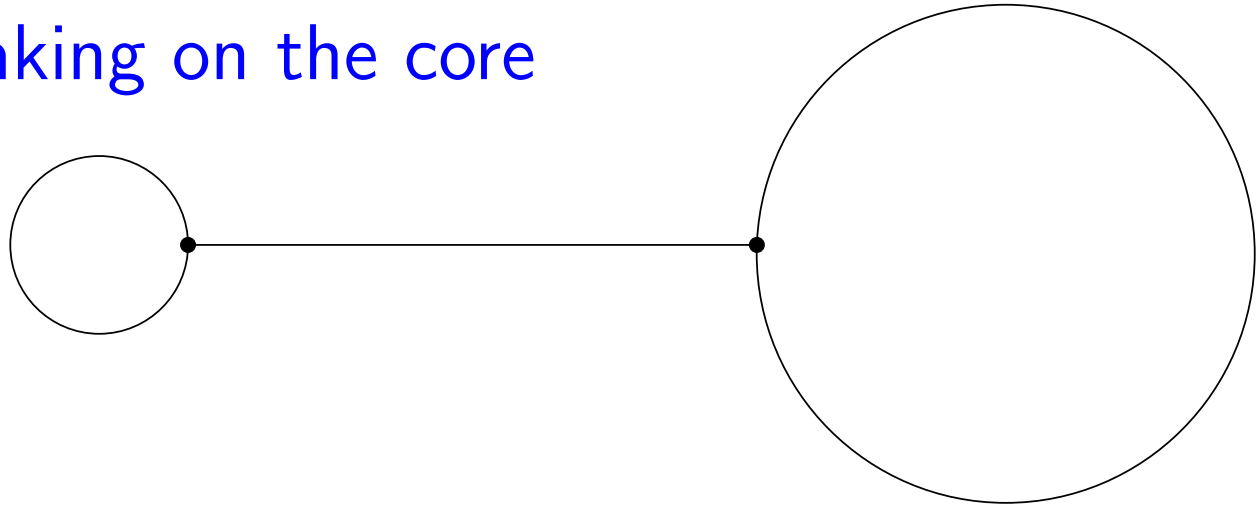
- GHP limit of components of $G(n,1/n)$ already understood.
- Components of $G(n,1/n)$ all have $O(1)$ cycles (most are already trees).
- The largest components of $G(n,1/n)$ have size $\Theta(n^{2/3})$ and diameter $\Theta(n^{1/3})$ – this is the reason for the scaling in the theorem.
- Can use an MST algorithm called *cycle breaking* to relate the structure of $M(n,1/n)$ to the (already understood) structure of $M(n,1/n)$.
- Cycle breaking is *dual* to Kruskal: instead of adding edges (unless they would create cycles), remove edges (unless they would cause disconnections).

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.

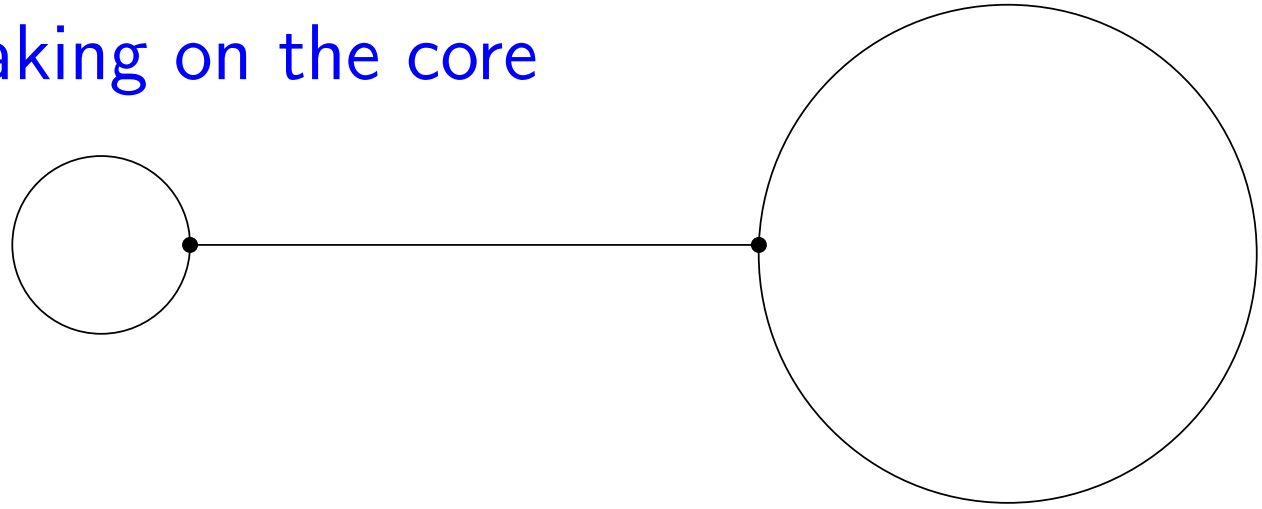
Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

Cycle breaking on the core

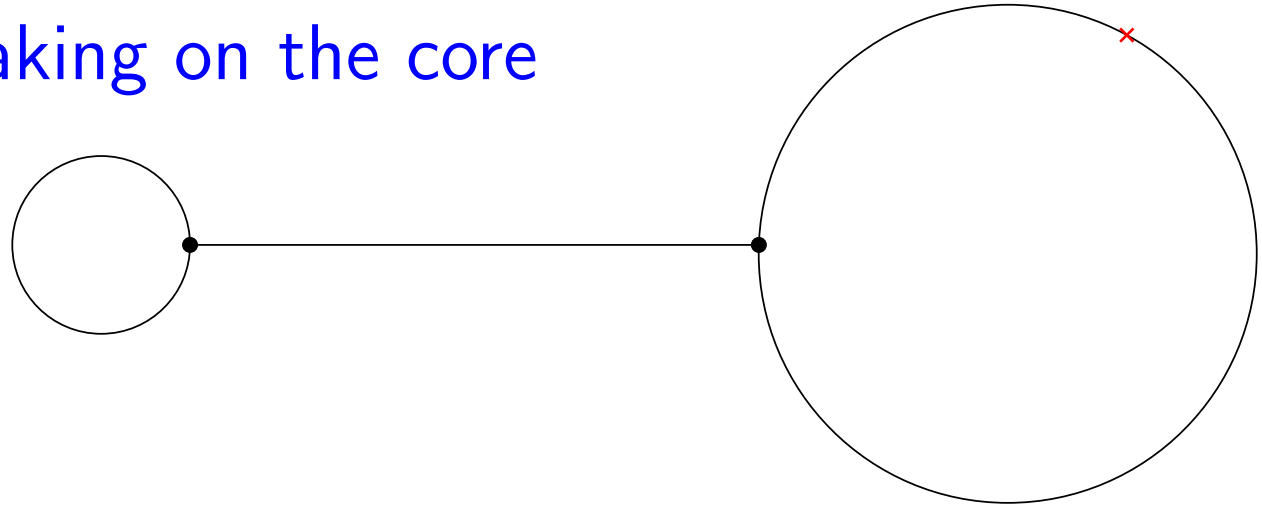


⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

⋆ In the limit, we're choosing a random point on the core.

Cycle breaking on the core

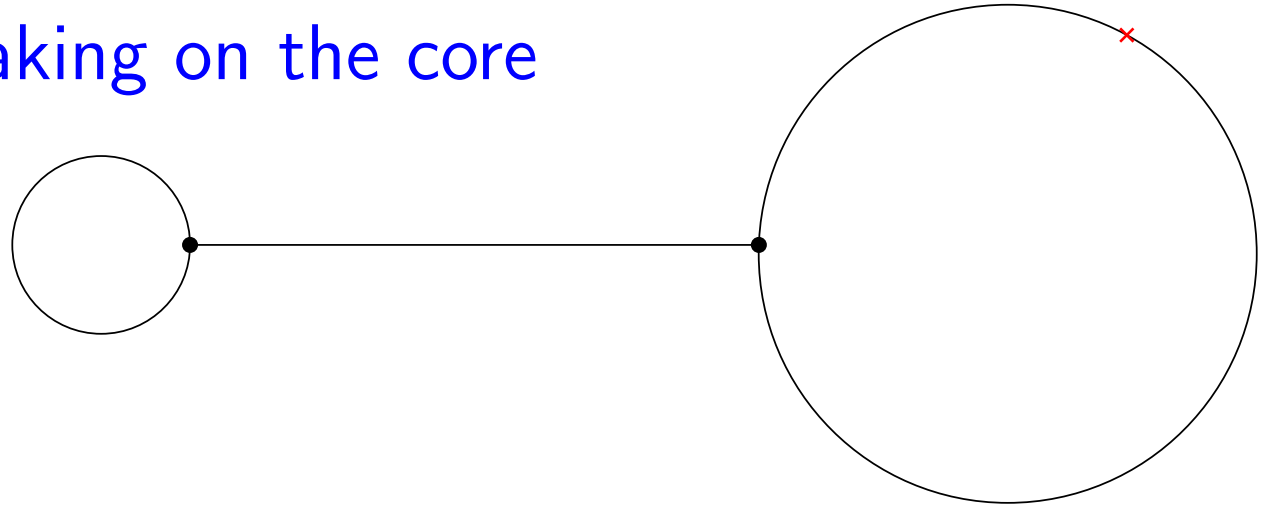


⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

⋆ In the limit, we're choosing a random point on the core.

Cycle breaking on the core



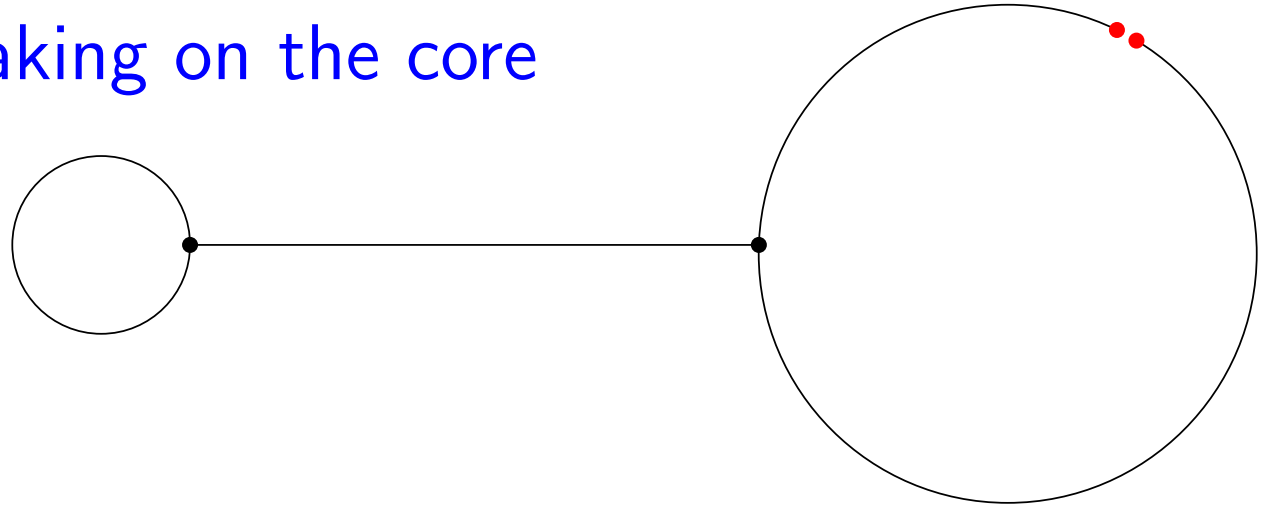
⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

⋆ In the limit, we're choosing a random point on the core.

⋆ This operation may or may not break a cycle.

Cycle breaking on the core



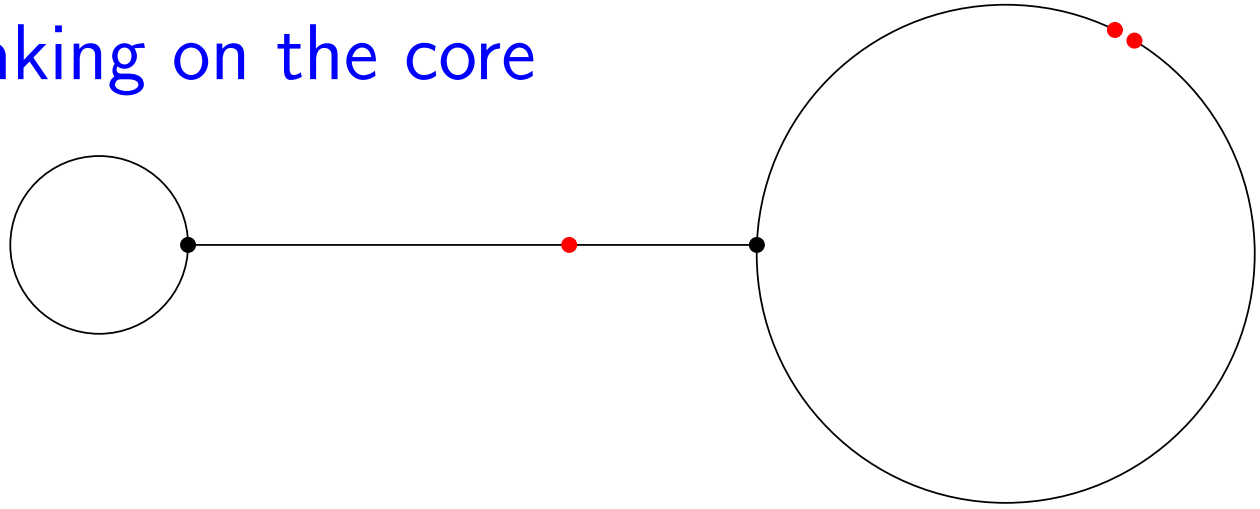
⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

⋆ In the limit, we're choosing a random point on the core.

⋆ This operation may or may not break a cycle.

Cycle breaking on the core



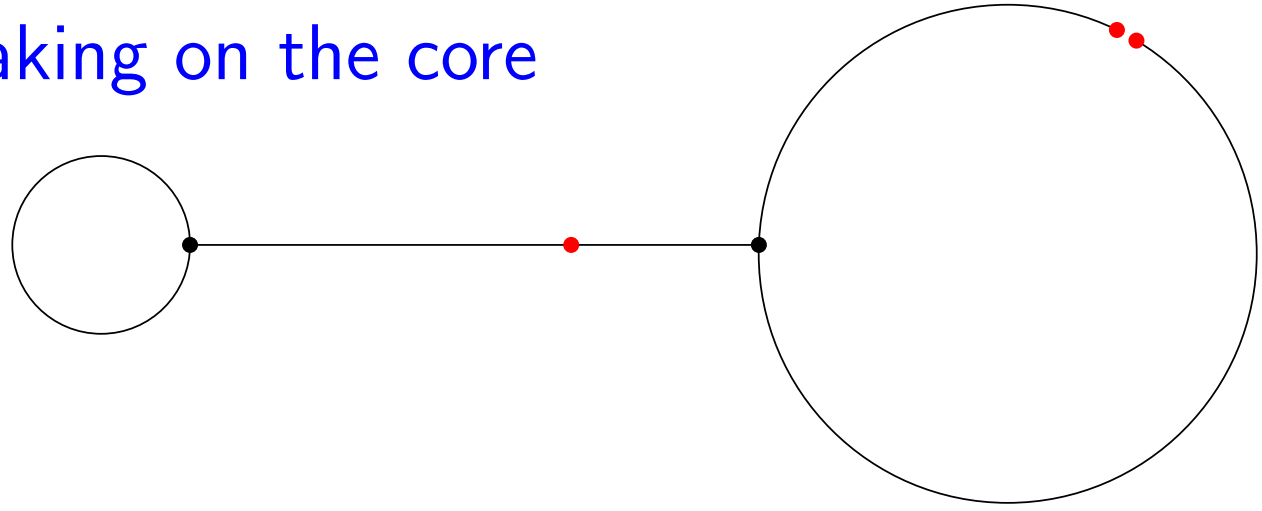
⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

⋆ In the limit, we're choosing a random point on the core.

⋆ This operation may or may not break a cycle.

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.

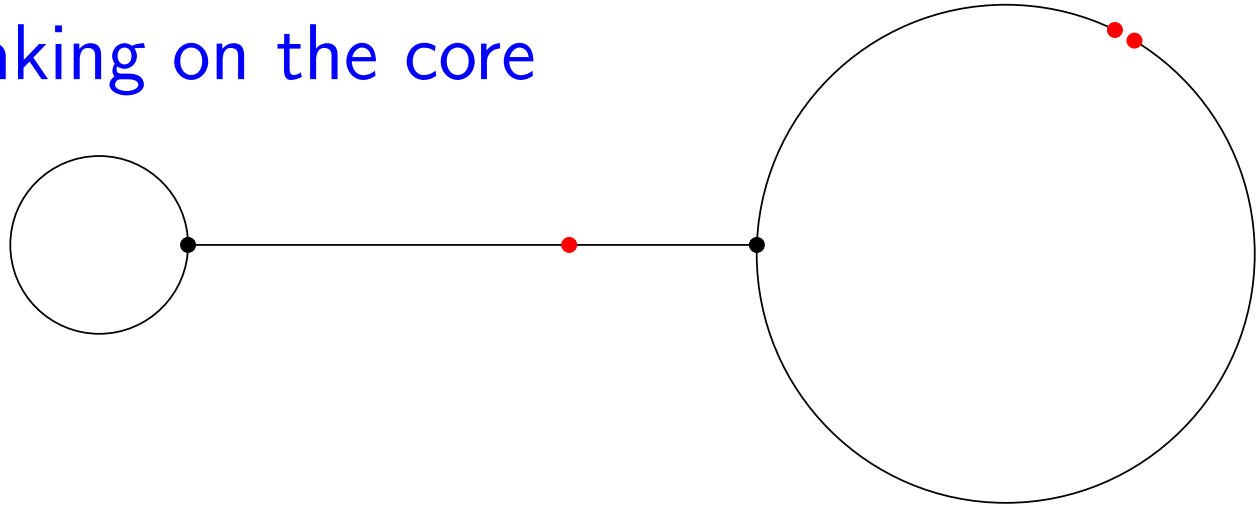
⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

⋆ In the limit, we're choosing a random point on the core.

⋆ This operation may or may not break a cycle.

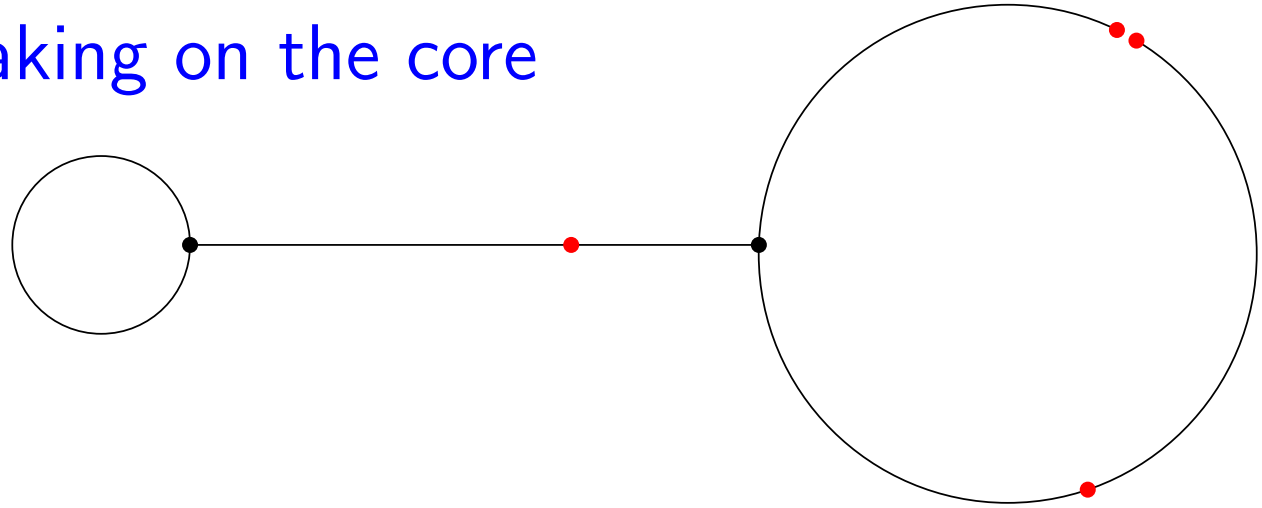
⋆ But it certainly breaks a path in two.

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.
⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.
⋆ In the limit, we're choosing a random point on the core.
⋆ This operation may or may not break a cycle.
⋆ But it certainly breaks a path in two.
⋆ Repeat until all cycles are broken.

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

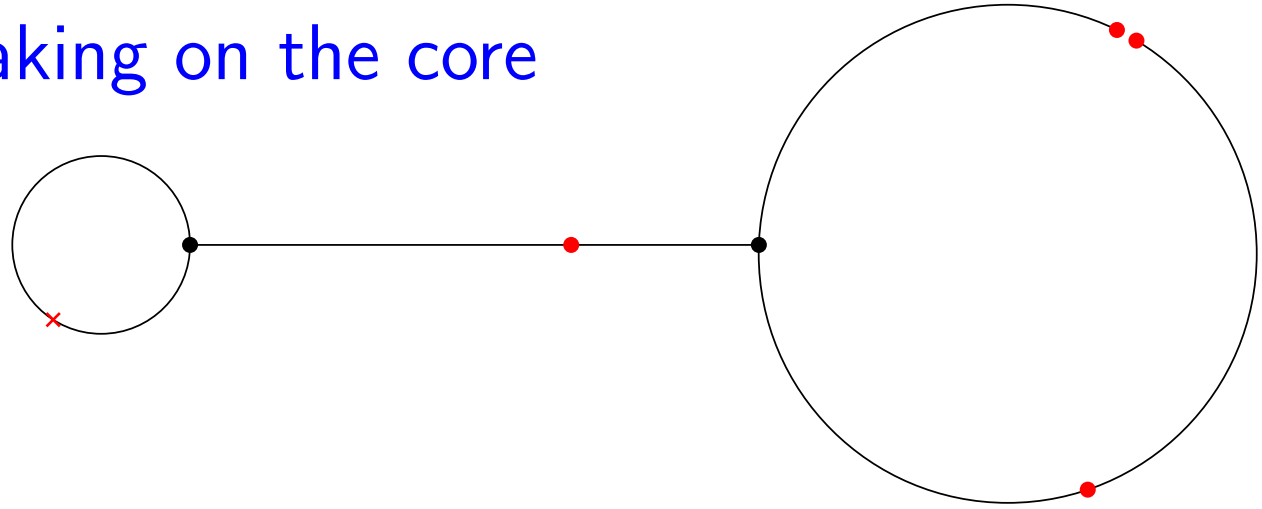
⋆ In the limit, we're choosing a random point on the core.

⋆ This operation may or may not break a cycle.

⋆ But it certainly breaks a path in two.

⋆ Repeat until all cycles are broken.

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

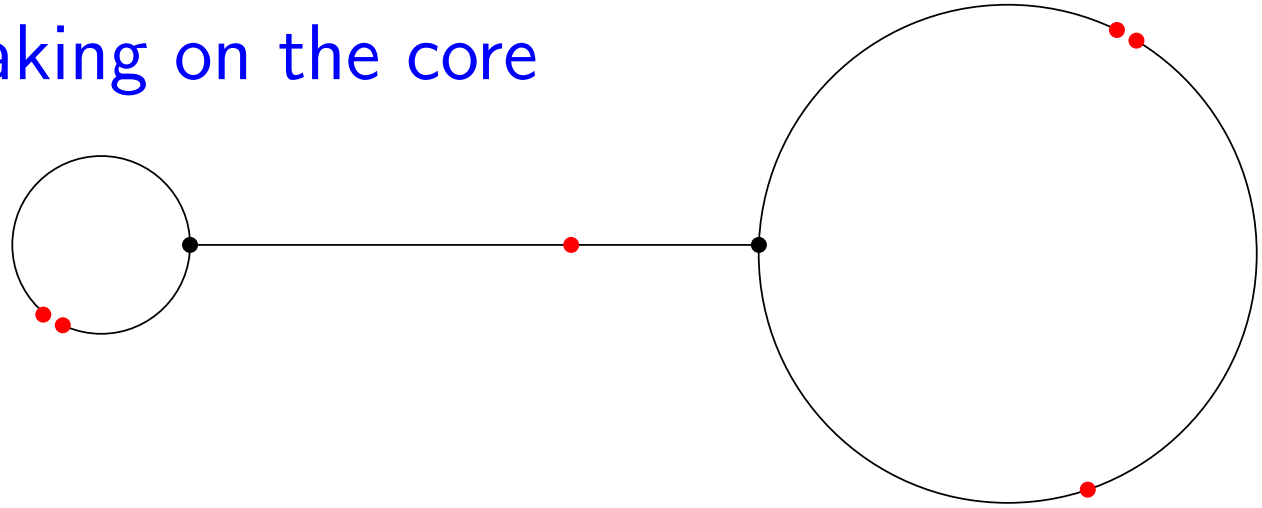
⋆ In the limit, we're choosing a random point on the core.

⋆ This operation may or may not break a cycle.

⋆ But it certainly breaks a path in two.

⋆ Repeat until all cycles are broken.

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

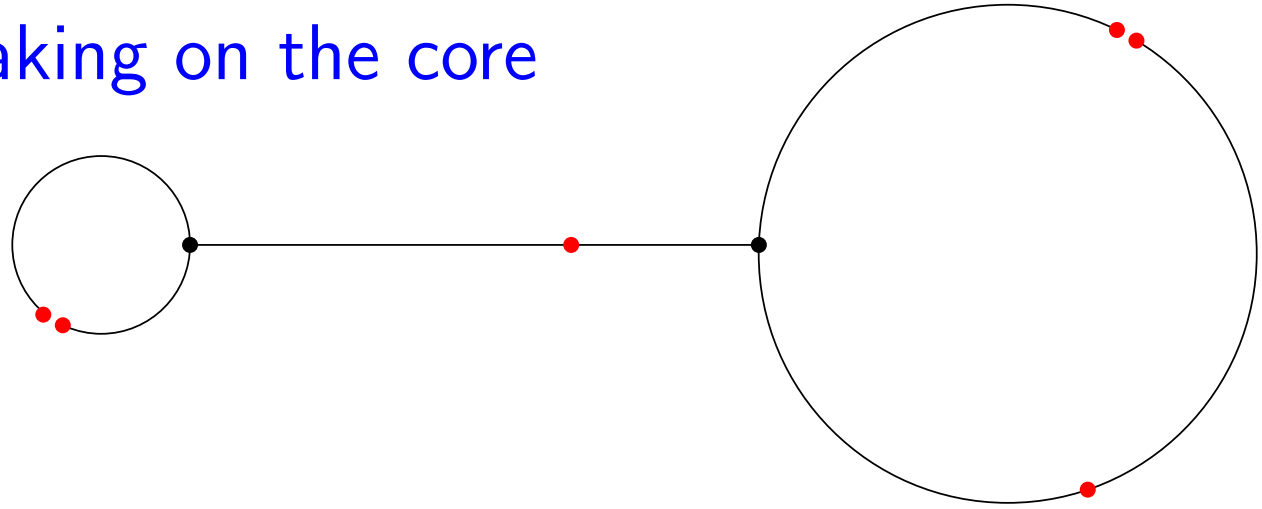
⋆ In the limit, we're choosing a random point on the core.

⋆ This operation may or may not break a cycle.

⋆ But it certainly breaks a path in two.

⋆ Repeat until all cycles are broken.

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.

⋆ It turns out that for components of $G(n, p)$, this edge is equally likely to be any edge of the core.

⋆ In the limit, we're choosing a random point on the core.

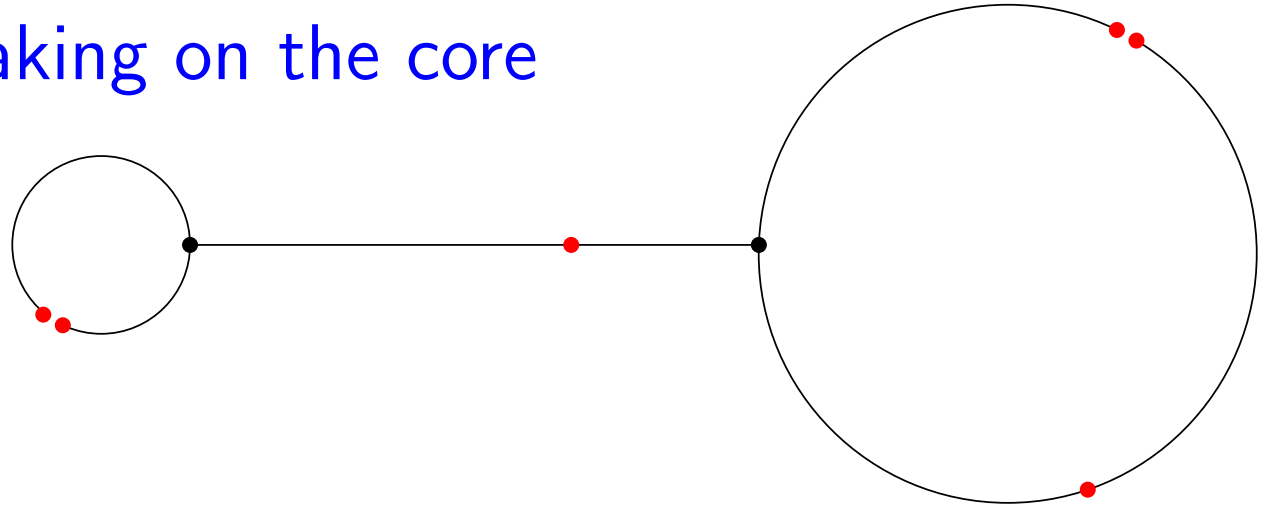
⋆ This operation may or may not break a cycle.

⋆ But it certainly breaks a path in two.

⋆ Repeat until all cycles are broken.

⋆ Now glue all the pendant trees back on.

Cycle breaking on the core



⋆ Choose the largest weight edge in the core, try to remove it.
⋆ It turns out that for components of $G(n,p)$, this edge is equally likely to be any edge of the core.
⋆ In the limit, we're choosing a random point on the core.
⋆ This operation may or may not break a cycle.
⋆ But it certainly breaks a path in two.
⋆ Repeat until all cycles are broken.
⋆ Now glue all the pendant trees back on.

Result: MST of a component of $G(n,1/n)$ can be built (in the limit) by gluing randomly rescaled Brownian continuum random trees along the edges of a random discrete tree.

Wrapping up

Result: MST of a component of $G(n, 1/n)$ can be built (in the limit) by gluing randomly rescaled Brownian continuum random trees along the edges of a random discrete tree.

Wrapping up

Result: MST of a component of $G(n, 1/n)$ can be built (in the limit) by gluing randomly rescaled Brownian continuum random trees along the edges of a random discrete tree.

Because $p = 1/n$ is the critical point for the Erdős-Rényi random graph, and due to the coupling between the random graph (percolation) process and Kruskal's algorithm, proving convergence for the minimum spanning trees of components of $G(n, 1/n)$, as sketched on the previous slide, is the main step in proving convergence of the full minimum spanning tree M_n to a limit $\mathcal{M}$.

The Annals of Probability
2017, Vol. 45, No. 5, 3075–3144
DOI: 10.1214/16-AOP1132

THE SCALING LIMIT OF THE MINIMUM SPANNING TREE OF THE COMPLETE GRAPH

BY LOUIGI ADDARIO-BERRY[1,2,*], NICOLAS BROUTIN[3,†], CHRISTINA GOLDSCHMIDT[2,4,‡] AND GRÉGORY MIERMONT[5,§]

McGill University[*], *Inria Rocquencourt-Paris*[†], *University of Oxford*[‡] *and École Normale Supérieure de Lyon/Institut universitaire de France*[§]

Consider the minimum spanning tree (MST) of the complete graph with n vertices, when edges are assigned independent random weights. Endow this tree with the graph distance renormalized by $n^{1/3}$ and with the uniform measure on its vertices. We show that the resulting space converges in distribution as $n \to \infty$ to a random compact measured metric space in the Gromov–Hausdorff–Prokhorov topology. We additionally show that the limit is a random binary $\mathbb{R}$-tree and has Minkowski dimension 3 almost surely. In particular, its law is mutually singular with that of the Brownian continuum random tree or any rescaled version thereof. Our approach relies on a coupling between the MST problem and the Erdős–Rényi random graph. We exploit the explicit description of the scaling limit of the Erdős–Rényi random graph in the so-called critical window, established in [*Probab. Theory Related Fields* **152** (2012) 367–406], and provide a similar description of the scaling limit for a "critical minimum spanning forest" contained within the MST. In order to accomplish this, we introduce the notion of $\mathbb{R}$-graphs, which generalise $\mathbb{R}$-trees, and are of independent interest.

Universality of the minimum spanning tree limit. . .

. . . still an open problem in general!