Minimum spanning arborescence

Gourab Ray

Probdyn Seminar, U.Vic

Joint work with: Arnab Sen (U Minnesota)

University of Victoria

January 2024

The model

- Take a finite graph $G = (V, E)$. Orient all the edges in both directions. Call this $\vec{E}$.
- Fix a boundary vertex, call it ∂.

The model

- Take a finite graph $G = (V, E)$. Orient all the edges in both directions. Call this $\vec{E}$.
- Fix a boundary vertex, call it ∂.

Definition

An **spanning arborescence** of (G, ∂) is a subset of $\vec{E}$ such that

- Every edge has exactly one outgoing edge, except ∂ which has none.
- There are no cycles.

Minimum spanning arborescence

Definition

An **spanning arborescence** of (G, ∂) is a subset of $\vec{E}$ such that

- Every edge has exactly one outgoing edge, except ∂ which has none.
- There are no cycles.

Minimum spanning arborescence

Definition

An **spanning arborescence** of (G, ∂) is a subset of $\vec{E}$ such that

- Every edge has exactly one outgoing edge, except ∂ which has none.
- There are no cycles.

- Put i.i.d. weights coming from a continuous distribution (e.g. Exponential (1)) on every $\vec{e} \in \vec{E}$.
- The spanning arborescence with minimum weight (the a.s. unique one) is called the minimum spanning arborescence.

Goal of this work

- Take an infinite graph G.
- Take an exhaustion $G_1 \subset G_2 \subset \dots$ such that $\cup G_n = G$.
- Identify every vertex in the complement of G_n into a single vertex ∂. Call this G_n^{w}
- Take T_n^{w}, the MSA of G_n^{w}.
- Observe that each T_n^{w} is a measurable function of the weights in G_n^{w}.

Question

Does the weak limit of T_n^{w} exist as $n \to \infty$? If yes, what can we say about the geometry of such limits?

Definition

End of a tree is the number of distinct, infinite disjoint paths we can draw on it.

Definition

End of a tree is the number of distinct, infinite disjoint paths we can draw on it.

Example

Regular tree has infinitely many ends. $\mathbb{Z}$ has two ends.

Theorem (R., Sen, 24.)

Take a 'bounded subdivision' of a regular tree of degree at least 3. Then the wired, weak limit exists of the MSA exists. There are infinitely many infinite components and each component is one ended almost surely.

Theorem (R., Sen, 24.)

Take a 'bounded subdivision' of a regular tree of degree at least 3. Then the wired, weak limit exists of the MSA exists. There are infinitely many infinite components and each component is one ended almost surely.

Theorem (R., Sen, 24.)

Same result as above for Galton–Watson trees with zero probability of producing a single offspring.

Theorem (R., Sen, 24.)

Take a 'bounded subdivision' of a regular tree of degree at least 3. Then the wired, weak limit exists of the MSA exists. There are infinitely many infinite components and each component is one ended almost surely.

Theorem (R., Sen, 24.)

Same result as above for Galton–Watson trees with zero probability of producing a single offspring.

Theorem (R., Sen, 24.)

In fact, same result is true in any nonamenable, unimodular graph once we assume that the wired MSA limit exists.

Theorem (R., Sen, 24.)

Take a 'bounded subdivision' of a regular tree of degree at least 3. Then the wired, weak limit exists of the MSA exists. There are infinitely many infinite components and each component is one ended almost surely.

Theorem (R., Sen, 24.)

Same result as above for Galton–Watson trees with zero probability of producing a single offspring.

Theorem (R., Sen, 24.)

In fact, same result is true in any nonamenable, unimodular graph once we assume that the wired MSA limit exists.

It turns out that a sufficient condition for the existence of the wired MSA limit is the 'transience' of a certain stochastic process called 'Loop contracting random walk'.

Motivation

- The MSA has widespread applications in disease outbreaks, approximating the traveling salesman problem, wireless networks, natural language processing, genetics etc.

Motivation

- The MSA has widespread applications in disease outbreaks, approximating the traveling salesman problem, wireless networks, natural language processing, genetics etc.
- Since the 60s, a lot of effort has been put into finding a good algorithm to sample this object due to Edmonds, Chu-Liu, Bock, Tarjan, Gabow, Galil, Spencer...

Motivation

- The unoriented version of this model (Minimum spanning trees or MST for short) has been studied by probabilists and computer scientists (Kruskal, Prim, Schramm, Lyons, Peres, Addario–Berry, Goldschmidt....). MSA has not received that much attention from probabilists. It is important to rectify this.

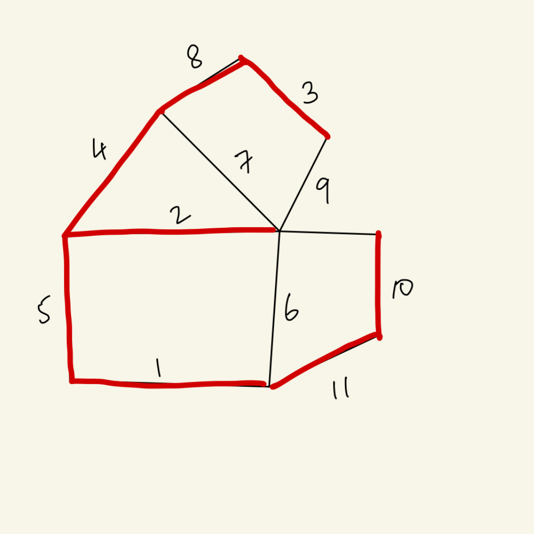
Motivation

- The unoriented version of this model (Minimum spanning trees or MST for short) has been studied by probabilists and computer scientists (Kruskal, Prim, Schramm, Lyons, Peres, Addario–Berry, Goldschmidt....). MSA has not received that much attention from probabilists. It is important to rectify this.
- Another viewpoint: this can be seen as a 'ground state' or a state with minimal energy in a statistical mechanics model with disorder. Existence of a unique ground state is an important open question, for example in spin glass models.

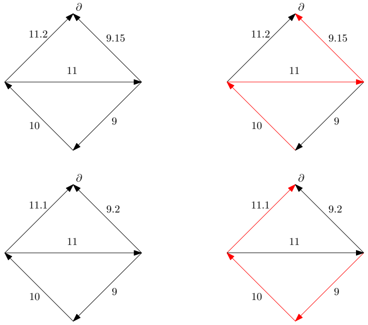
Motivation

- The unoriented version of this model (Minimum spanning trees or MST for short) has been studied by probabilists and computer scientists (Kruskal, Prim, Schramm, Lyons, Peres, Addario–Berry, Goldschmidt....). MSA has not received that much attention from probabilists. It is important to rectify this.
- Another viewpoint: this can be seen as a 'ground state' or a state with minimal energy in a statistical mechanics model with disorder. Existence of a unique ground state is an important open question, for example in spin glass models.
- My personal reason: it leads to cool mathematics.

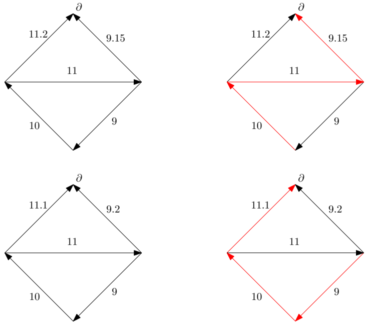
Algorithms for MSA: Kruskal and Prim's/ invasion percolation

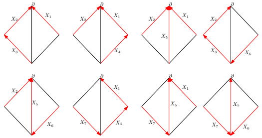


MSA depends on weights



MSA depends on weights





The CLEB algorithm (Chu, Liu, Edmonds and Bock)

We have $G = (V, \partial)$ with weights $(U_{0,\vec{e}})_{\vec{e} \in \vec{E}}$

The CLEB algorithm (Chu, Liu, Edmonds and Bock)

We have $G = (V, \partial)$ with weights $(U_{0,\vec{e}})_{\vec{e} \in \vec{E}}$

Phase 1: contraction phase.

- From every vertex subtract the minimum outgoing weight from the weight of every outgoing edge.

The CLEB algorithm (Chu, Liu, Edmonds and Bock)

We have $G = (V, \partial)$ with weights $(U_{0,\vec{e}})_{\vec{e} \in \vec{E}}$

Phase 1: contraction phase.

- From every vertex subtract the minimum outgoing weight from the weight of every outgoing edge.
- Let Y_1 be the zero weight outgoing edges.

The CLEB algorithm (Chu, Liu, Edmonds and Bock)

We have $G = (V, \partial)$ with weights $(U_{0,\vec{e}})_{\vec{e} \in \vec{E}}$

Phase 1: contraction phase.

- From every vertex subtract the minimum outgoing weight from the weight of every outgoing edge.
- Let Y_1 be the zero weight outgoing edges.
- 'contract' any cycle.
- We get a new weight collection $(V_1, \vec{E}_1, (U_{1,\vec{e}})_{\vec{e} \in \vec{E}_1})$.

The CLEB algorithm (Chu, Liu, Edmonds and Bock)

We have $G = (V, \partial)$ with weights $(U_{0,\vec{e}})_{\vec{e}\in\vec{E}}$

Phase 1: contraction phase.

- From every vertex subtract the minimum outgoing weight from the weight of every outgoing edge.
- Let Y_1 be the zero weight outgoing edges.
- ‘contract’ any cycle.
- We get a new weight collection $(V_1, \vec{E}_1, (U_{1,\vec{e}})_{\vec{e}\in\vec{E}_1})$.
- Iterate.

The CLEB algorithm (Chu, Liu, Edmonds and Bock)

We have $G = (V, \partial)$ with weights $(U_{0,\vec{e}})_{\vec{e}\in\vec{E}}$

Phase 1: contraction phase.

- From every vertex subtract the minimum outgoing weight from the weight of every outgoing edge.
- Let Y_1 be the zero weight outgoing edges.
- 'contract' any cycle.
- We get a new weight collection $(V_1, \vec{E}_1, (U_{1,\vec{e}})_{\vec{e}\in\vec{E}_1})$.
- Iterate.
- Stop when there is no cycle.

The CLEB algorithm (Chu, Liu, Edmonds and Bock)

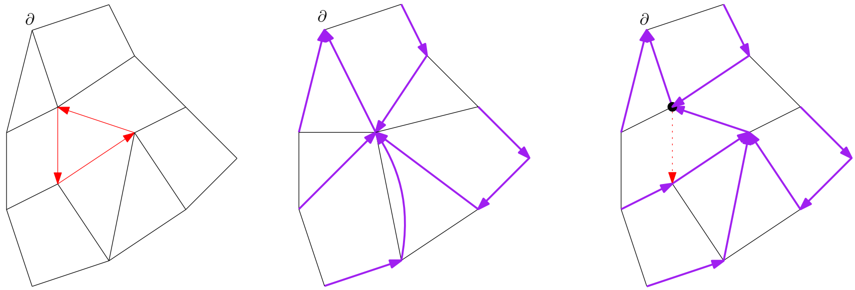
Phase 2: the uncontraction phase.

- Order the loops contracted in reverse order.

The CLEB algorithm (Chu, Liu, Edmonds and Bock)

Phase 2: the uncontraction phase.

- Order the loops contracted in reverse order.
- ‘uncontract’ the loops to get an MSA of $(V_i, \vec{E}_i)$ in every step.



Extensions of CLEB

- We can expose edges in any order and contract loops as they come. (Due to Tarjan).

Extensions of CLEB

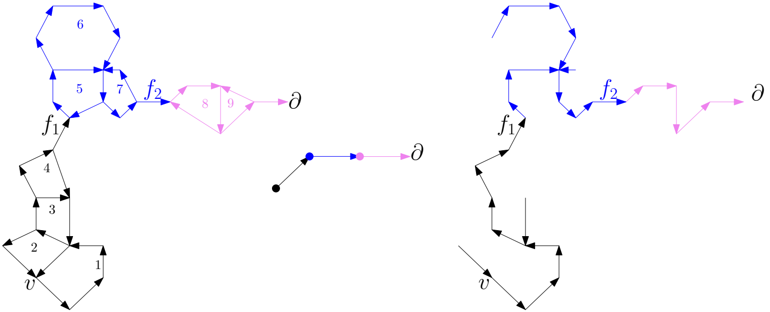
- We can expose edges in any order and contract loops as they come. (Due to Tarjan).
  Creates a collection of exposed paths in the contracted graph.

Extensions of CLEB

- We can expose edges in any order and contract loops as they come. (Due to Tarjan).
  Creates a collection of exposed paths in the contracted graph.
- A useful choice: Always expose edges from the the tip of a contracted path. (Gabow et al.) We call this CLEB walk algorithm.

Extensions of CLEB

- We can expose edges in any order and contract loops as they come. (Due to Tarjan).
  Creates a collection of exposed paths in the contracted graph.
- A useful choice: Always expose edges from the the tip of a contracted path. (Gabow et al.) We call this CLEB walk algorithm.



When the weights are i.i.d. Exponential, the CLEB process becomes particularly pleasant, when we randomize over the weights.

When the weights are i.i.d. Exponential, the CLEB process becomes particularly pleasant, when we randomize over the weights.

- Memoryless property of Exponential: If $X_1, X_2 \sim$i.i.d. Exponential (1) then conditioned on $X_2 = \min\{X_1, X_2\}$ and the value of X_2, $X_1 - X_2 \sim$Exponential (1)

When the weights are i.i.d. Exponential, the CLEB process becomes particularly pleasant, when we randomize over the weights.

- Memoryless property of Exponential: If X_1, X_2 ∼i.i.d. Exponential (1) then conditioned on $X_2 = \min\{X_1, X_2\}$ and the value of X_2, $X_1 - X_2$ ∼Exponential (1)
- The CLEB process now becomes the same (in law) as a 'Loop contracting random walk'.

When the weights are i.i.d. Exponential, the CLEB process becomes particularly pleasant, when we randomize over the weights.

- Memoryless property of Exponential: If $X_1, X_2 \sim$i.i.d. Exponential (1) then conditioned on $X_2 = \min\{X_1, X_2\}$ and the value of X_2, $X_1 - X_2 \sim$Exponential (1)
- The CLEB process now becomes the same (in law) as a 'Loop contracting random walk'.

Definition

The loop contracting random walk is transient if its 'trace' converges to an infinite path.

CLEB process/ Loop contracting random walk

It is sometimes useful to look at the projection of the edges exposed by the CLEB process in the base graph G.

Lemma

Edges exposed by loop contracting random walk on G converges almost surely.

CLEB process/ Loop contracting random walk

It is sometimes useful to look at the projection of the edges exposed by the CLEB process in the base graph G.

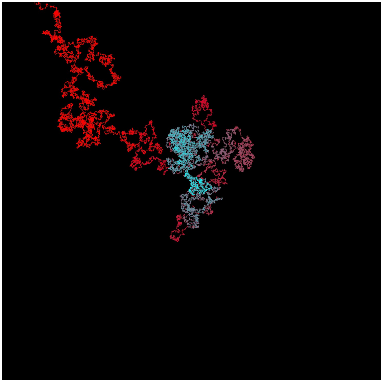
Lemma

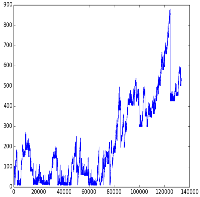
Edges exposed by loop contracting random walk on G converges almost surely.

Lemma

One can recover the MSA from the loop contracting random walk in the infinite graph if the loop contracting random walk is almost surely transient.

Simlulation





Loop contracting random walk

Theorem (R., Sen 24)

We prove the loop contracting random walk is transient on regular trees, their bounded subdivisions as well as (infinite) Galton Watson trees.

Loop contracting random walk

Theorem (R., Sen 24)

We prove the loop contracting random walk is transient on regular trees, their bounded subdivisions as well as (infinite) Galton Watson trees.
As a corollary, wired MSA limits exist almost surely in all these graphs.

Lemma (Comparison with simple random walk)

Let v be a vertex in a finite tree T. Glue all the leaves into a single vertex ∂. Let $v \neq \partial$. Let

$$\mathcal{H} := \text{ hit } \partial \text{ before returning to } v$$

$$\mathbb{P}(\mathcal{H} \text{ occurs for Simple random walk}) \geq \mathbb{P}(\mathcal{H} \text{ occurs for loop contracting random walk.})$$

Lemma (Comparison with simple random walk)

Let v be a vertex in a finite tree T. Glue all the leaves into a single vertex ∂. Let $v \neq \partial$. Let

$$\mathcal{H} := \text{ hit } \partial \text{ before returning to } v$$

$$\mathbb{P}(\mathcal{H} \text{ occurs for Simple random walk}) \\ \geq \mathbb{P}(\mathcal{H} \text{ occurs for loop contracting random walk.})$$

Unfortunately there is no 0-1 law for loop contracting random walk

Proof of one-endedness

Question

Under what kind of local perturbation is the MSA stable?

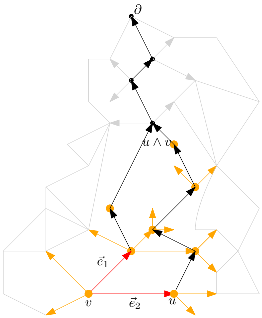
Proof of one-endedness

Question

Under what kind of local perturbation is the MSA stable?

Can we change weights locally so that

$$\text{new MSA} = \text{old MSA} \setminus \{\vec{e}_1\} \cup \{\vec{e}_2\}?$$



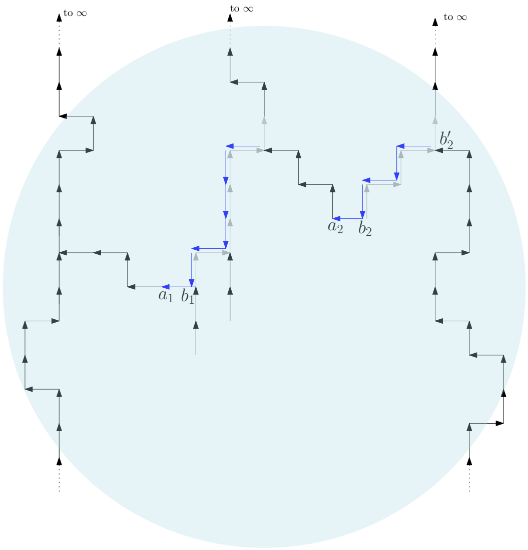
Proof of one endedness

- By some well established theory of unimodular graphs, we obtain that the number of components are
  - infinite,
  - each component either has one or two ends,
  - there are zero or infinitely many two ended infinite components.

Proof of one endedness

- By some well established theory of unimodular graphs, we obtain that the number of components are
  - infinite,
  - each component either has one or two ends,
  - there are zero or infinitely many two ended infinite components.
- In the latter case, we perform a surgery: we glue two of them together so that
  - there is a component with ≥ 3 ends,
  - we do it in an absolutely continuous way.

surgery



Thanks for listening!