Inferring the history of growing random trees

Joint work with Christophe Giraud, Gábor Lugosi & Déborah Sulem

Simon Briend

Université Paris-Saclay
Pompeu Fabra University

This work is part of the project PID2022-138268NB-I00,
funded by MCIN/AEI/10.13039/501100011033 / and FEDER, UE.

January 22, 2024

Growing trees

We are interested in models of randomly growing trees,

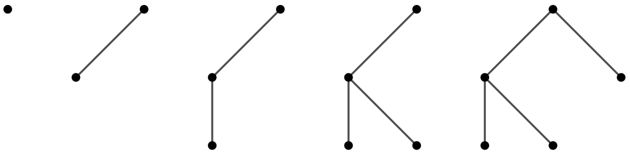


Figure: A growing tree

For example

- Uniform random recursive trees (URRT)
- Preferential attachment model (PA)

Often-time we only observe the unlabeled, undirected structure of the graph.

Real world examples

- Malware spreading between computers
- Political beliefs spreading in a community
- Rumours and fake news spreading online
- Online social group growing

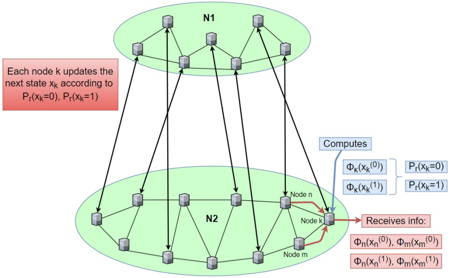


Figure: Taken from "Markov-Based Malware Propagation Modeling and Analysis in Multi-Layer Networks"

Our goal

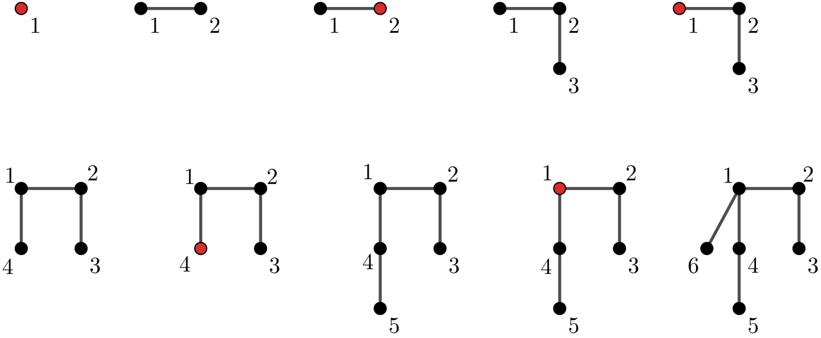
Infer the history from a snapshot of the present state of the tree. It answers real life question such as

- How did the malware that infected your company propagated in your systems?
- How did the fake news spread?
- How did Covid spread?

In mathematical term, we want to find an ordering procedure $\hat{\sigma}$ that is label invariant.

The URRT

A tree is grown recursively as follows



The URRT

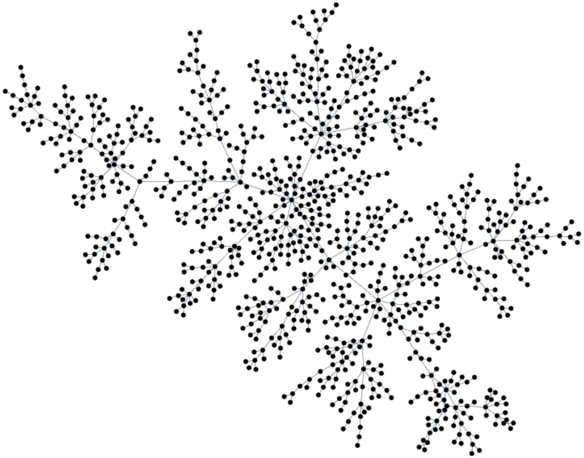


Figure: URRT of size 1000

Link to Seriation

- Inferring the position of vertices in a random geometric graph
- Or in a graphon
- In seriation problems, the points all have the same properties

- In our problem vertex 1 and n have very different properties
- This changes everything, for example what is a good error measure

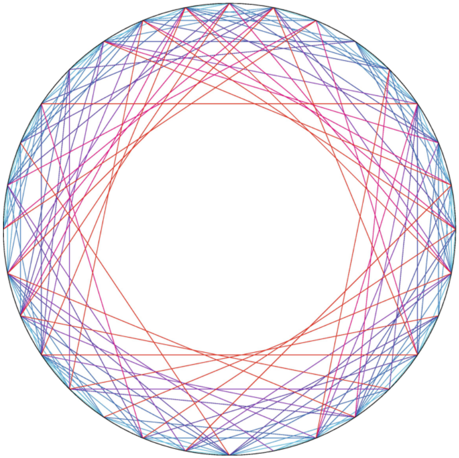


Figure: Taken from "Geometric Random Graphs on Circles"

A measure of error

Using something like $\max_i |\hat{\sigma}(i) - \sigma(i)|$ is not informative. Indeed, worst case scenario is n, best is $n/2$. We use

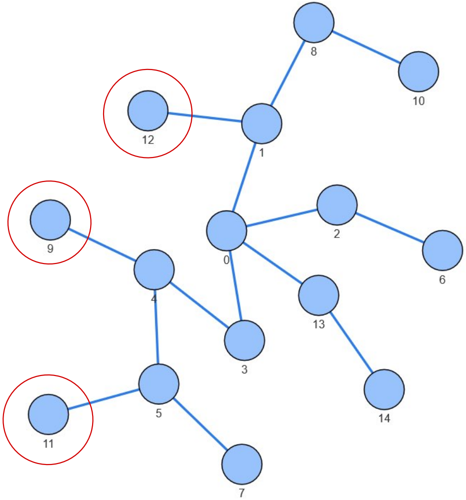
$$R_\alpha(\hat{\sigma}) = \sum_{i=1}^{n} \frac{|\hat{\sigma}(i) - \sigma(i)|}{\sigma(i)^\alpha} .$$

- It takes into account the inhomogeneity in the graph
- This is the right scaling for $\alpha \geq 1$

A lower bound

- In the URRT model, the probability of a tree depends only on its shape
- It means that all permutation of the vertices producing a recursive ordering have the same probability

- We can identify vertices that no ordering method can order better than random
- For example, any vertex arrived after time $n/2$, connected to $[n/2]$ and still a leaf at time n



A lower bound

Using these *exchangeable* vertices we prove that

For any $\alpha \geq 0$

$$R^*_\alpha \geq \frac{n^{2-\alpha}}{65} \, ,$$

where R^*_α is the minimum error over all label invariant ordering procedures.

Remark A simple argument gives $\mathbb{R}^*_\alpha \geq 1/2$.
For $0 \leq \alpha < 1$, any ordering procedure has $R(\hat{\sigma}) \lessapprox n^{2-\alpha}$

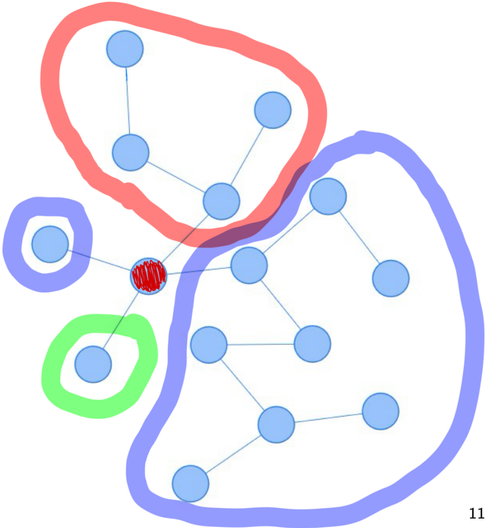
The Jordan centrality ordering

- In a rooted tree, we can define hanging subtrees
- We denote by $(T, u)_v$ the subtree hanging from v in the tree rooted in u

- The Jordan centrality of u is defined by

$$\Phi(u) = \max_{v \sim u} |(T, u)_v| \ .$$

- We order vertices by increasing value of their Jordan centrality
- This is a label invariant procedure



Another formulation

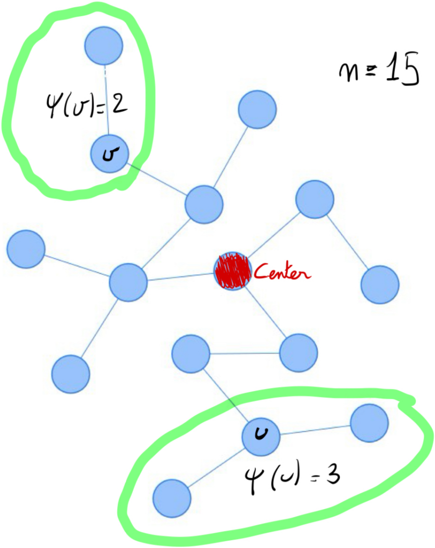
This method is the same as:

- Estimating the position of vertex 1 by the Jordan centroid
- Ordering vertices by the size of their hanging subtree rooted at the Jordan centroid

So a general class of algorithm could be:

- Estimate the position of vertex 1 and root the tree there (for example using Rumour centroid)
- Order vertices by the size of their subtrees in the rooted tree

Another formulation



Performance guarantees

Step 1: prove that $\hat{\sigma}_J$ (ordering by Jordan centrality) and $\hat{\sigma}'$ (odering vertices by number of descendants) have similar risks

- For all but vertices on the path $\{1 \to \text{center}\}$, $\phi(u)$ is equal to $n - 1 - de(u)$, where $de(u)$ is the number of descendants of u.
- It is well known that the arrival time of the centroid is dominated by an exponential RV (and hence the distance between vertex 1 and the center).

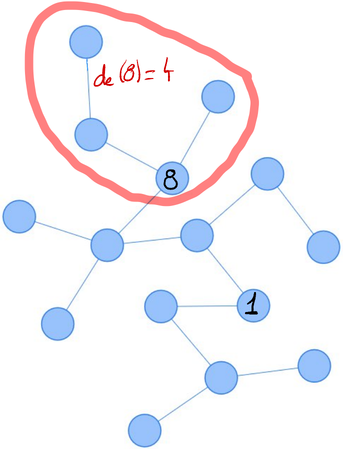
$$R_\alpha(\hat{\sigma}_J) - R_\alpha(\hat{\sigma}') \leq K \log^4(n) \ .$$

Performance guarantees

- Even if we can not compute it in practice, we can analyse the risk of $\hat{\sigma}'$
- $de(u)$ is exactly a Polya urn! So the descendent ordering is easy to analyse.

For $\alpha \in [1, 2)$

$$R_\alpha(\hat{\sigma}_J) \leq C_\alpha R^*_\alpha \ .$$



Remark on the limitation to $\alpha \leq 2$

A simple argument to prove Jordan can not do better:

- With probability $1/n$ vertex 1 is a leaf, thus ordered among the last vertices.
- So $\mathbb{E}[\hat{\sigma}_J(1)] \geq \log(n)$.

What happens when α grows:

- More emphasize is put on low index vertex.
- So the step "estimating position of vertex 1" gets more important
- Estimating the position of vertex 1 by the Jordan center is not good enough.
- A better method is to use Rumour centrality to estimate position of vertex 1.
- PROBLEM: We still miss some steps in the analysis.

Numerical illustration

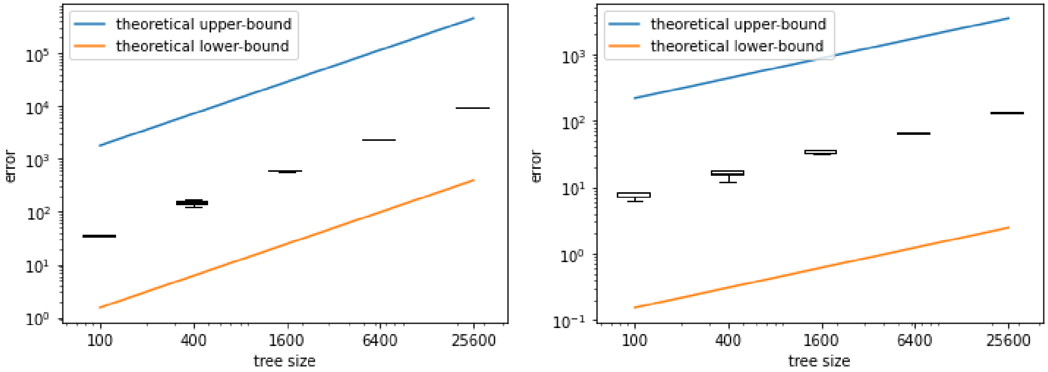


Figure: Risk of the Jordan ordering versus the tree size in logarithmic scales, for $\alpha = 1$ (left panel) and for $\alpha = 1.5$ (right panel), and for trees simulated from the URRT model. Here, we sample 20 trees for each size, and report a boxplot with the median, first, and last quartiles, for each tree size - whiskers extend from the box to display the full range of the data set.

Numerical illustration

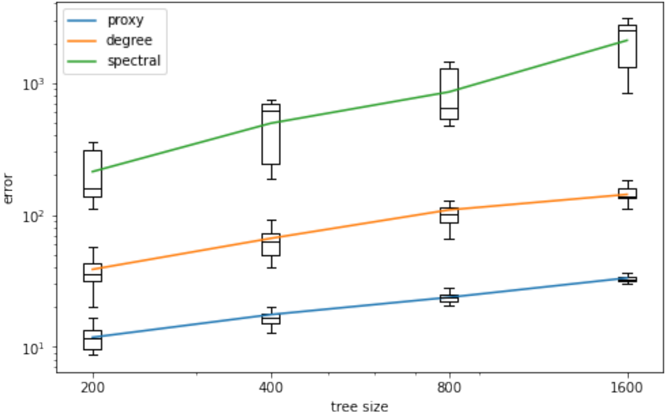


Figure: Risk versus the tree size n in logarithmic scales, for $\alpha = 1$, and for trees simulated from the URRT model. Here, we sample 20 trees for each size. We compare the Jordan (blue), degree (orange), and spectral methods (green), and report a boxplot with the median, first, and last quartiles, for each tree size - whiskers extend from the box to display the full range of the data set.

Numerical illustration

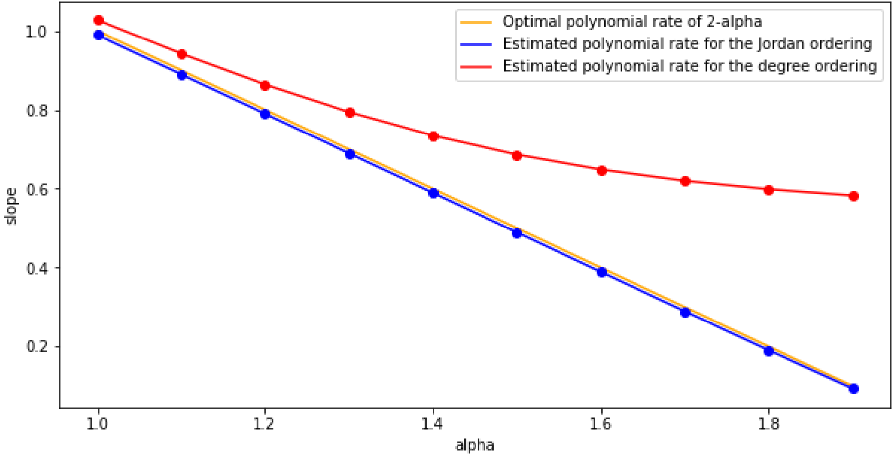


Figure: Estimated polynomial rate of growth of the risk for the Jordan and degree ordering for different value of α.