Obscure results and open problems: some of my favorites.

David Aldous

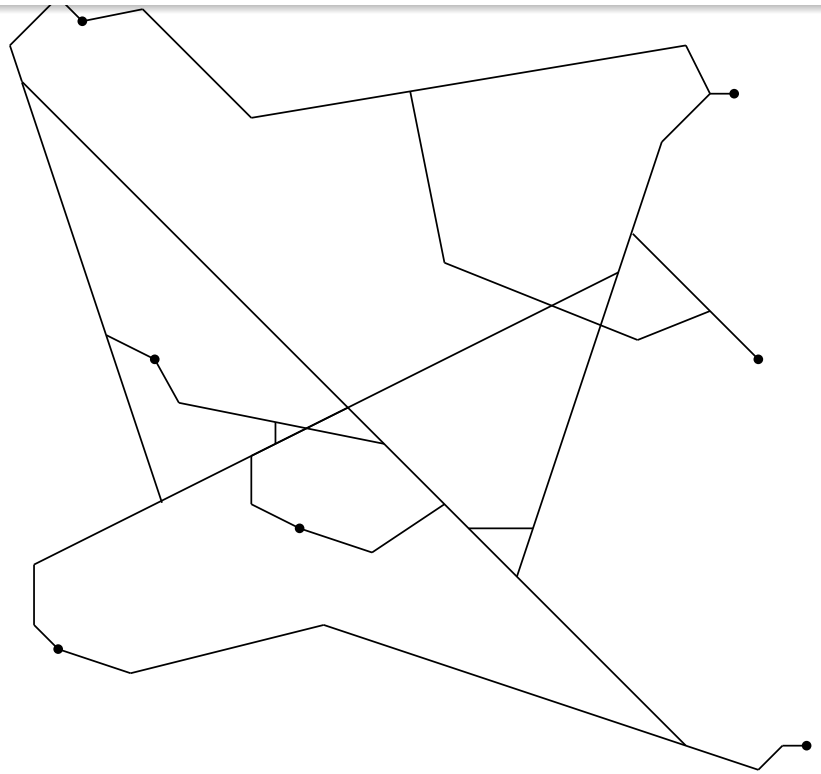
28 January 2025

Sub-title: *Things I would like some smart young person to work on.*

- Probability distributions on routed planar networks.
- Combinatorics of fringe trees in a phylogenetic model.
- Nearest neighbor of nearest neighbor, on the complete network. **Are you smarter than an AI?**
- The Nearest Unvisited Vertex walk.
- Find a qualitatively realistic model for a subway network.
- A layer model for random DAGs

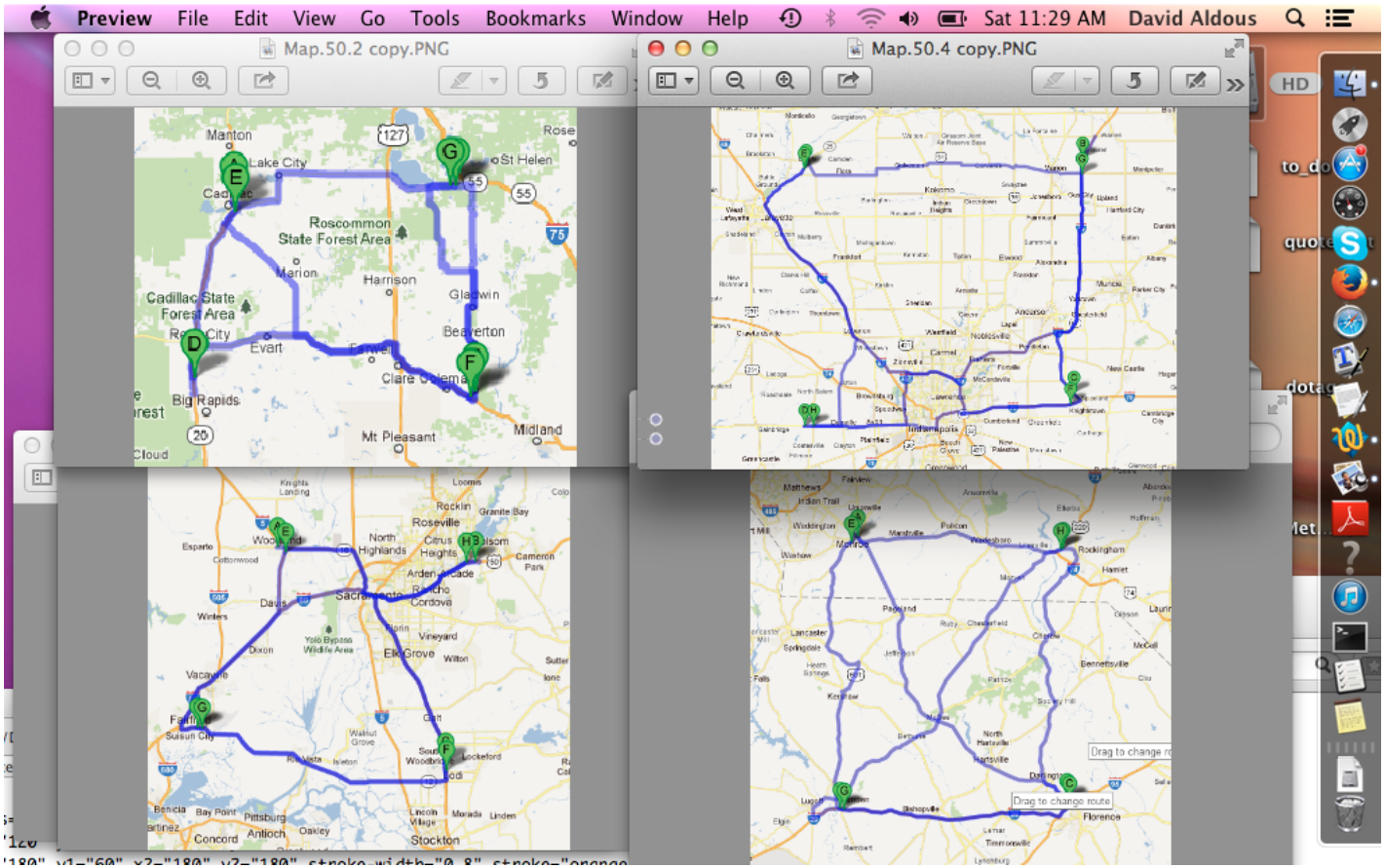
Topic 1: Probability distributions on routed planar networks.

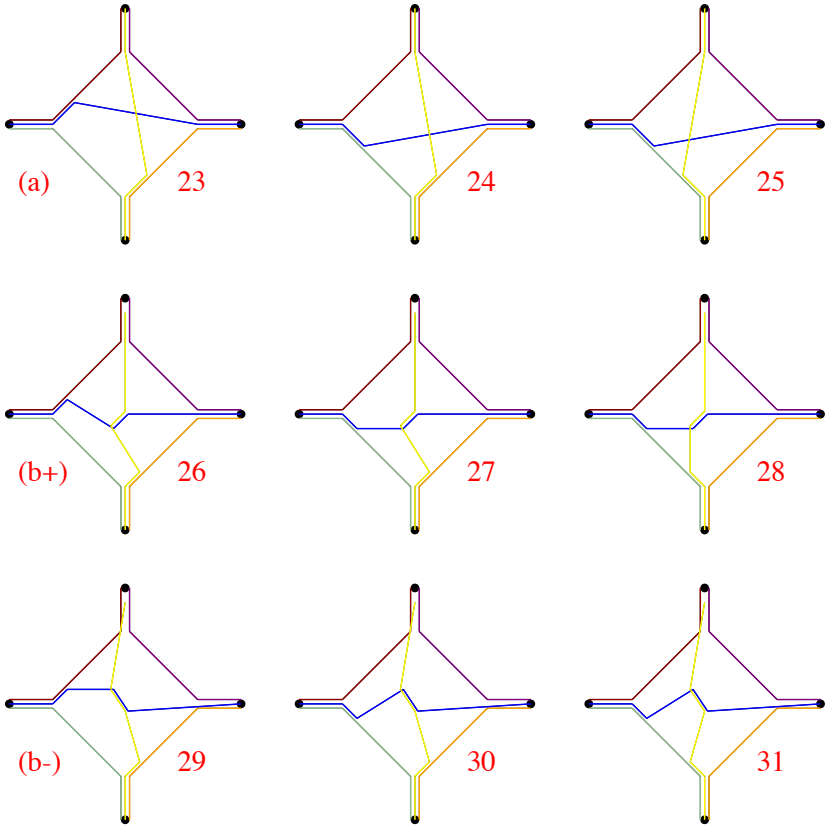
7 points in a window.



Scale-invariance means: doing this within a randomly positioned window, the statistics of the subnetwork observed don't depend on the scale, i.e. don't depend on whether the side length is 10 km or 100 km.

As undergraduate project we have looked at real-world subnetwork topologies (for $k = 4$ vertices, roughly at corners of a square).





https://www.stat.berkeley.edu/~aldous/Research/all-types.html 3/7

and listed all 71 topologies on 4 vertices – different conventions from usual planar graph theory. Could compare distributions over these topologies in real-world spatial networks and in models of spatial networks.

Open problems about "routed networks"

- Efficient naming (coding) of the different topologies?
- Enumeration? [cf. Catherine Greenhill talk Thursday]
- Is there a (mathematically) canonical one-parameter family of probability distributions on these topologies?
- (In our small data study) empirical frequencies of topologies are very non-uniform. Can one make a model where some of the real-world-rare topologies don't occur?
- We have 2 explicit models of scale-invariant networks; calculate/simulate the distribution for these models.

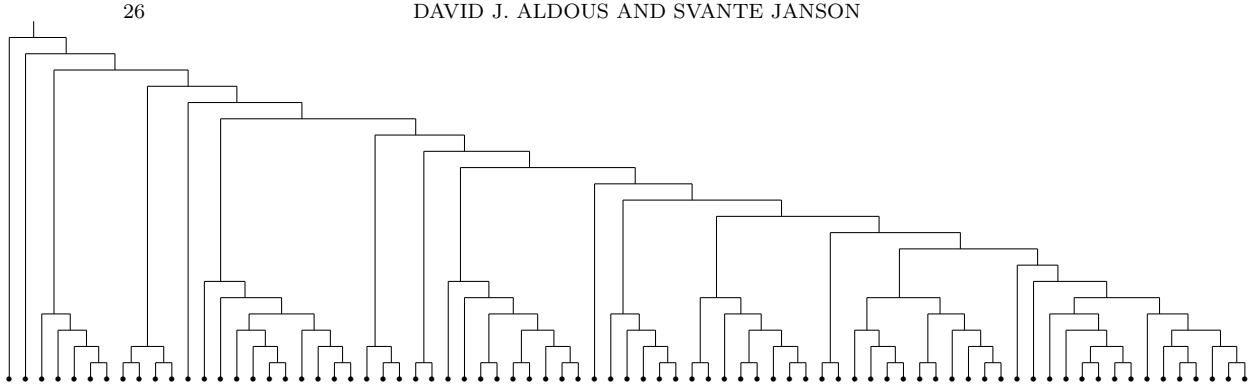
References:

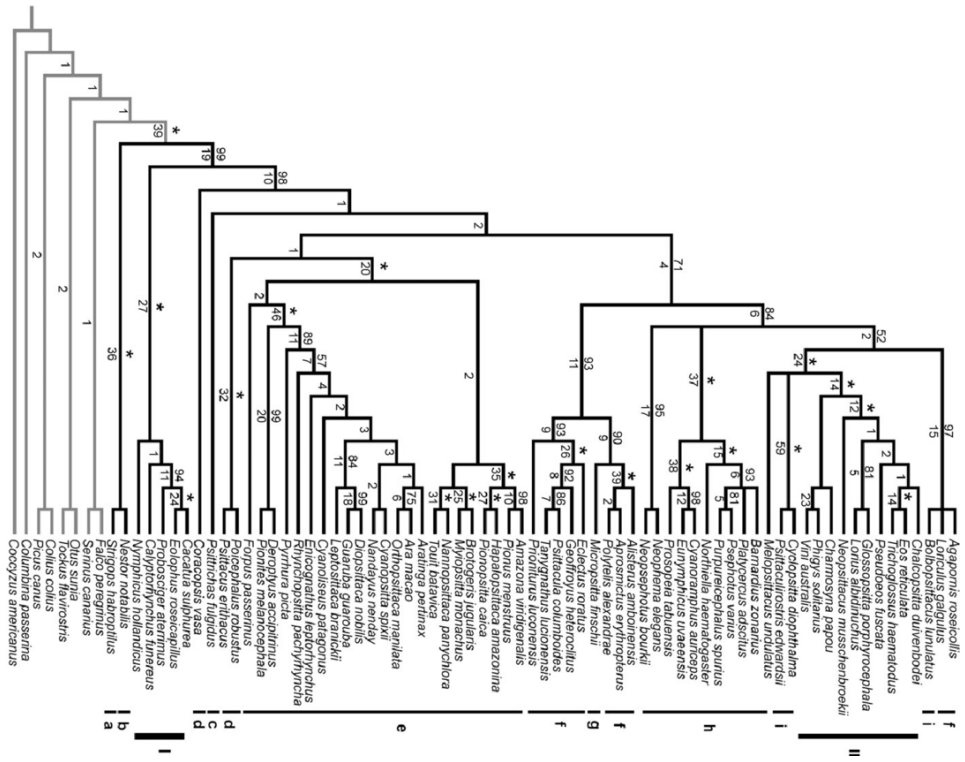
https://escholarship.org/content/qt1g44k4dk/qt1g44k4dk.pdf (the 71 topologies).

https://arxiv.org/abs/1204.0817 and https://arxiv.org/abs/2407.07887 (scale-invariant network models).

[All links are live on these slides posted on my home page]

Topic 2: Combinatorics of fringe trees in a phylogenetic model.





- Broad ongoing project with Svante Janson and Boris Pittel: average age ≈ 76.
- We have a model for a random phylogenetic tree on n leaves (Svante talk tomorrow)
- Has zero parameters, so gives explicit numerical predictions.
- Model designed to mimic the (very uneven) way that large splits occur in real phylogenetic trees.
- But does it happen to also mimic the fringe behavior?

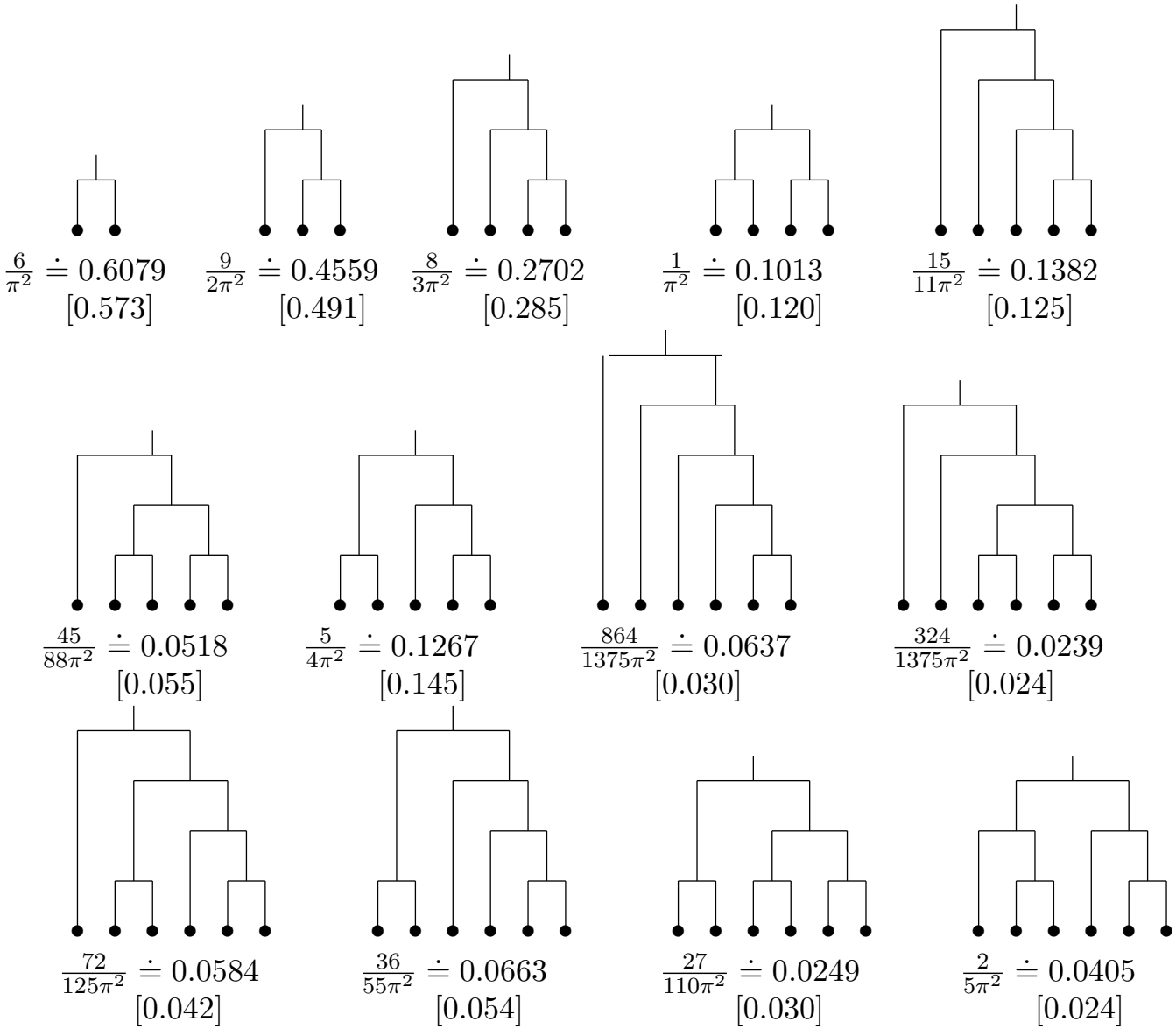


FIGURE 7. Proportions of leaves in clades of a given shape, for each shape with $2-6$ leaves in the fringe tree. The top number is from our model, the bottom number $[\cdots]$ from our small data set.

Here's one class of open problems about the model.
Write $p(\chi)$ = proportion of leaves in a given shape χ tree, in model (asymptotics of size tree).

- Can calculate numerically recursively.
- But we don't have a good description of the set of probabilities $\{p(\chi) : |\chi| = m\}$ over all m-leaf trees.
- Variety of combinatorial questions (like birthday or coupon collector) about the fringe trees within a large n-leaf tree: for instance
- How many different shapes occur?
- Largest shape that appears twice?
- Smallest shape that does not appear?

References:

https://arxiv.org/abs/2412.09655. (preprint)

Topic 3: Nearest neighbor of nearest neighbor, on the complete network.

Well known fact:

For the Poisson point process in the plane, the probability that a point is the nearest neighbor of its nearest neighbor equals a computable constant c.

One could surely prove by known methods (subadditivity)

For $2n$ uniform random points in a square, the probability that **every** *point is the nearest neighbor of its nearest neighbor is $\asymp \beta^n$ for some (not known to be computable) constant β.*

I will discuss a combinatorial version. Write RNN for the "reciprocal nearest neighbor" relation.

Consider the complete graph G_m on m vertices, put i.i.d. edge lengths (say $U[0,1]$) .

$$\mathbb{P}(\text{vertices 1 and 2 are RNN's}) = 1/(2m-3)$$

because this just says that edge (1,2) is the shortest of the $2m-3$ edges at 1 or 2. So

$$\mathbb{P}(\text{vertex 1 has a RNN}) = (m-1)/(2m-3)$$

That was easy! Continuing, if $m = 2n$ **it can be shown**

$$p(n) := \mathbb{P}(\text{every vertex has a RNN}) = \frac{(2n-1)!!\ (2n-3)!!}{(4n-3)!!}$$

with notation $(2n-1)!! := (2n-1)(2n-3)(2n-5)\cdots 3\cdot 1$.

So $p(2) = 1/5, p(3) = 1/21, p(4) = 5/429, \ldots\ldots$ and $p(n) \sim 2^{-2n+3/2}$ as $n \to \infty$.

Digression: [Extract from email received September 2024]

To keep track of how far the AI systems are from expert-level capabilities, we are developing **Humanity's Last Exam**, which aims to be the world's most difficult AI test.

We're assembling the largest, broadest coalition of experts in history to design questions that test how far AIs are from the human intelligence frontier. If there is a question (**any topic**) that would genuinely impress you if an AI could solve it, we'd like to hear it from you!

We have already received questions from researchers from MIT, UC Berkeley, Stanford, and more.

The top 50 questions will earn $5000 each.
The next top 500 questions will earn $500 each.

How hard can this be?

Some explicit or implicit side conditions for entry to competition:

- You know the undisputed answer.
- Answer cannot be found on internet.
- Not depend on private information; must be answer-able in principle by sufficiently intelligent human or AI.
- State question in words (or only undergraduate-level math concepts)
- No trick questions – seek questions that someone might actually ask.
- 5 AIs must all give incorrect answers.

Then amongst questions accepted for entry, human judges decide prize winners, presumably seeking a wide range of topics and interesting questions.

- -

So harder than it looks at first sight?

Some typical accepted questions here:
https://lastexam.ai/

I tried 10 math questions. 6 got accepted as entries. Amused myself in planning how to spend a hypothetical $5,000 prize . . .

. . . but only got a $500 prize.

And the question was . . . . . .

There is a general "substructures" problem: *given an event that is exponentially unlikely for a n-vertex model, how large is the largest vertex-subset such that the event holds for the induced structure?* This is very similar to a general "largest common substructure" for two independent realizations of a n-vertex model.

Familiar example are "largest clique" in Erdös-Rényi, and "longest increasing subsequence" of a random permutation. Other examples include leaf-labelled trees (Miklós Rácz talk Thursday).

Open problem: Within our model G_{2n}, what is the size V_n of the largest vertex-subset within which every vertex is RNN?

We have made the obvious first steps:

- Upper bound $c_2 n$ via first moment method
- Lower bound $c_1 n$ via greedy algorithm
- Some usual tricks can improve these constants (starter project for grad student?)

Of course we expect $\mathbb{E}[V_n] \sim cn$; but no conjecture for c and no abstract argument that some c exists. New idea needed! Would be really impressed if AI could solve this . . . . . .

References: Unpublished notes on request.

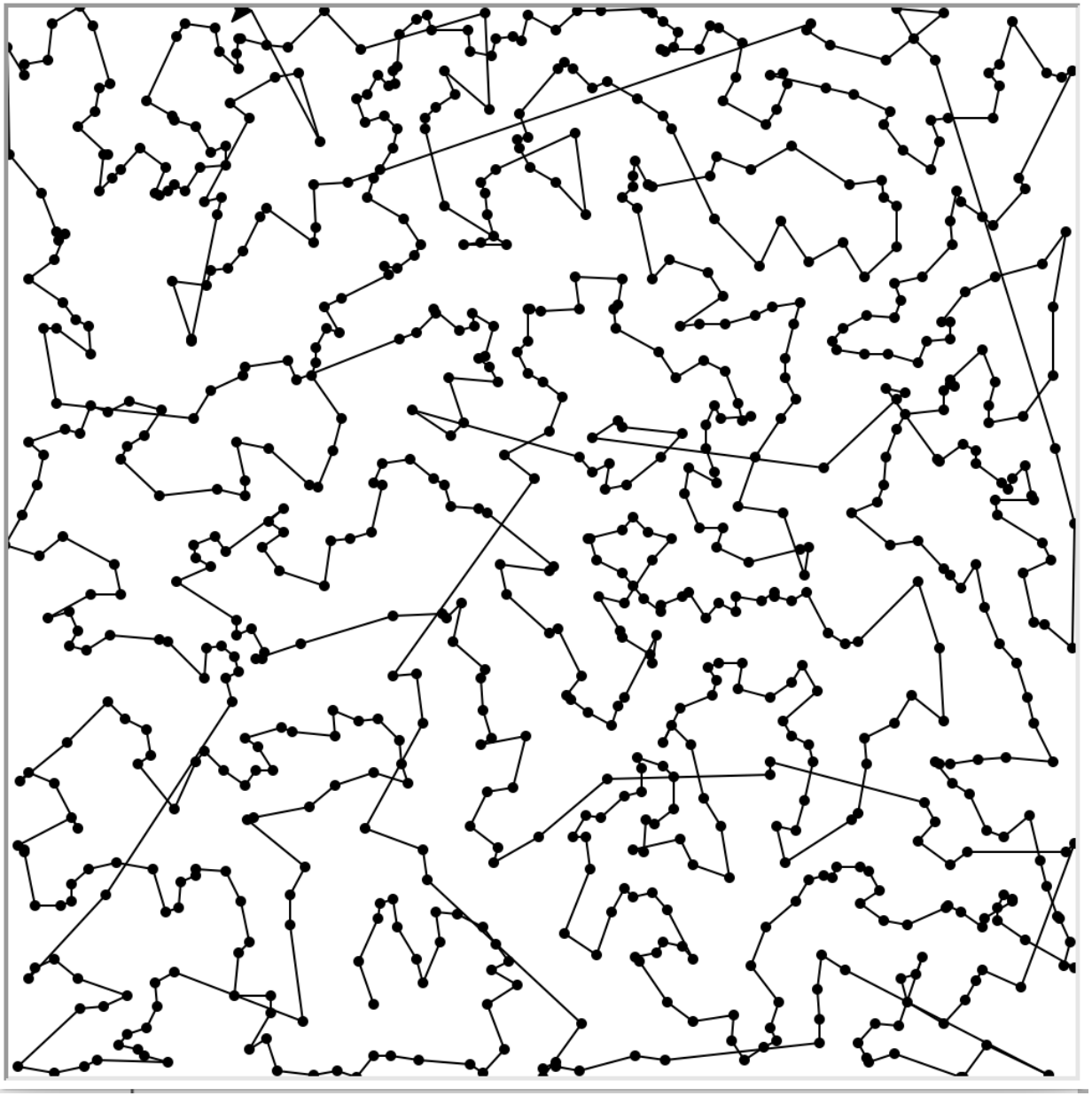
Topic 4: The Nearest Unvisited Vertex walk on Random Graphs

Consider a connected undirected graph G on n vertices, where the edges e have positive real lengths $\ell(e)$. Imagine a robot that can move at speed 1 along edges. We need a rule for how the robot chooses which edge to take after reaching a vertex. Most familiar is the "random walk" rule, choose edge e with probability proportional to $\ell(e)$ or $1/\ell(e)$. One well-studied aspect of the random walk is the *cover time*, the time until every vertex has been visited.

Instead of the usual random walk model, let us consider the **nearest unvisited vertex** (NUV) walk

> *after arriving at a vertex, next move at speed* 1 *along the path to the closest unvisited vertex*

and continue until every vertex has been visited. Note this is deterministic and has some length (= time) $L_{NUV}(G, v_0)$ where v_0 is the initial vertex.



In informal discussion we imagine lengths are scaled so that distance to closest neighbor is order 1, so L_{NUV} must be at least order n.

Natural first question: when is it $O(n)$ rather than larger order?

There is scattered old "algorithms" literature discussing the NUV walk as heuristics for TSP or as an algorithm for a robot exploring an unknown environment, but that literature quickly moved on to better algorithms.

There is a key starting math observation – implicit but rather obscured in the old literature. For now, we stay with non-random graphs.

Consider **ball-covering**: for $r > 0$ define $N(r) = N(G, r)$ to be the minimal size of a set S of vertices such that every vertex is within distance r from some element of S. In other words, such that the union over $s \in S$ of $\mathrm{Ball}(s, r)$ covers the entire graph.

Proposition

(i) $N(r) \le 1 + L_{NUV}/r, \ 0 < r < \infty$.
(ii) $L_{NUV} \le 2 \int_0^{\Delta/2} N(r)\, dr$ *where* $\Delta = \max_{v,w} d(v, w)$ *is the diameter of the graph.*

Note that for continuous spaces, *metric entropy* implies a notion of *dimension* via $N(r) \approx r^{-\dim}$ as $r \downarrow 0$. In our discrete context, if we have *dimension* in the sense

$$N(r) \approx n r^{-\dim}, \quad 1 \ll r \ll \Delta$$

then the Proposition has informal interpretation that L_{NUV} is always $O(n)$ when $\dim > 1$.

The ball-covering relation is not helpful from the algorithms viewpoint. But it is useful for some **random** graph models. In particular, in a model where we take a unweighted graph and then assign random edge-lengths, understanding "balls" is precisely the basic issue in **first passage percolation (FPP)**.

Consider the random graph G_m that is the $m \times m$ grid, that is the subgraph of the Euclidean lattice $\mathbb{Z}^2$, assigned i.i.d. edge-lengths $\ell(e) > 0$. with $\mathbb{E}\ell(e) < \infty$. Because the shortest edge-length at a given vertex is $\Omega(1)$, clearly L_{NUV} is $\Omega(m^2)$. Using the shape theorem for FPP on $\mathbb{Z}^2$ one can show

Corollary

For the 2-dimensional grid model G_m above, the sequence $(m^{-2} L_{NUV}(G_m),\ m \geq 2)$ is tight.

The same techniques would give $O(n)$ upper bounds in other simple models of n-vertex random graphs.

Open problems

- Are there general methods (subadditivity or local weak limits don't seem to work) to prove existence of a limit $c = \lim_n n^{-1} L_{NUV}(G_n)$ for simple models?
- Evaluate c?
- Order of magnitude of $\mathrm{var}(L_{NUV})$ not clear from our small-scale simulations – seems $n^{1\pm\varepsilon}$.

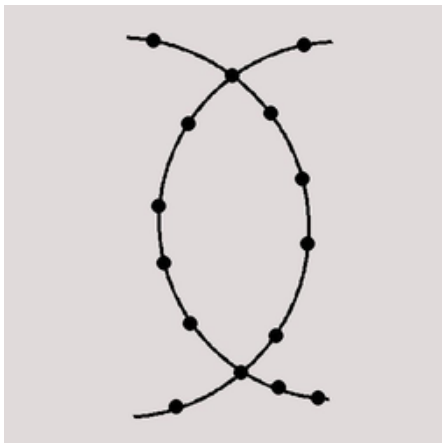
References:.

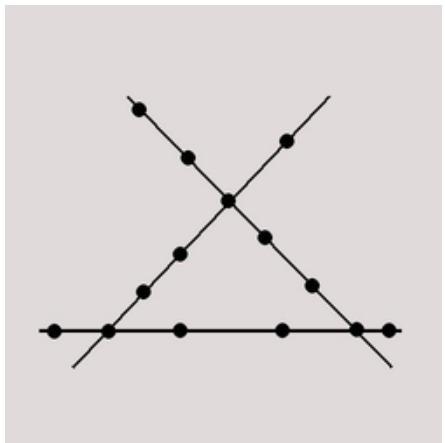
https://arxiv.org/abs/1912.13175. (Paper)
https://www.stat.berkeley.edu/users/aldous/stel-project.html
(Undergrad coding project to analyze a simple prototype of 4X computer games such as Stellaris)

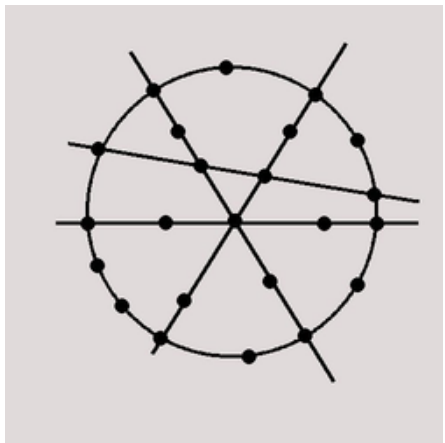
Topic 5: Find a qualitatively realistic model for a subway network.
Human population density varies tremendously, so any realistic model of (say) spatial networks in the human world should not assume positions as a grid or as a homogeneous Poisson process.

Easier said than done! The simplest setting (that I can imagine) is to imagine a large 20th century city with a subway network, where typical journeys are between less dense suburbs and more dense centers.

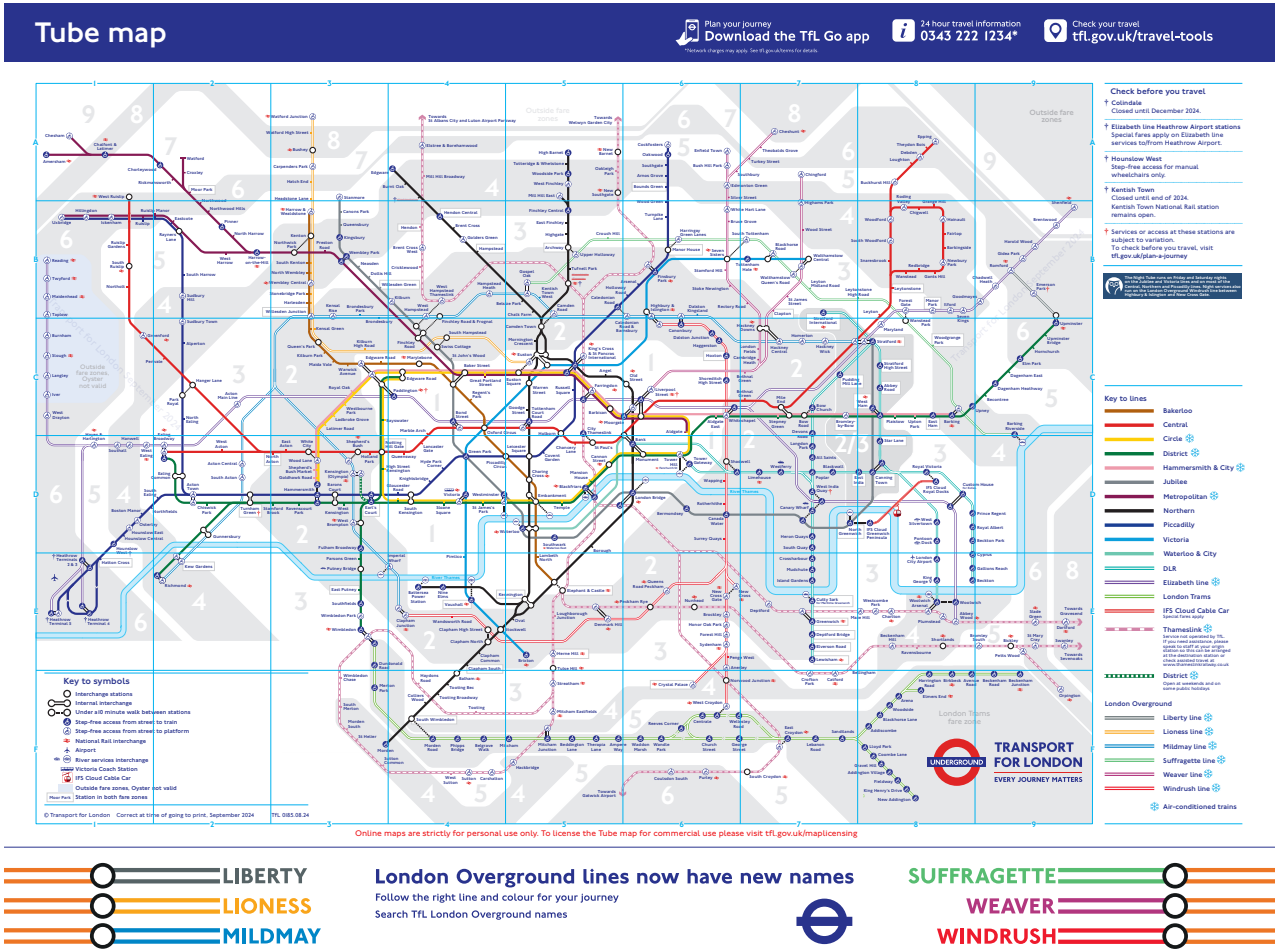
Wikipedia – rapid transit shows typical topologies (shapes) for small subway-type networks.







What do we expect to see in a large network?



We typically see a well-connected core, often delineated by a circular line, from which branching lines spread away.

Question: Can we reproduce these qualitative features as an optimal network within some toy model?

Our very toy model

- Gaussian or power-law density of source/destination.
- Travel fast on subway network, slow off network.
- Seek to minimize mean journey time.

One parameter S = fast/slow ratio.
What are the optimal networks for different total lengths L?

In next figure, dashed circle is 1 s.d. of standard bivariate Gaussian, which contains 40% of population. (Similar results for power-law density).

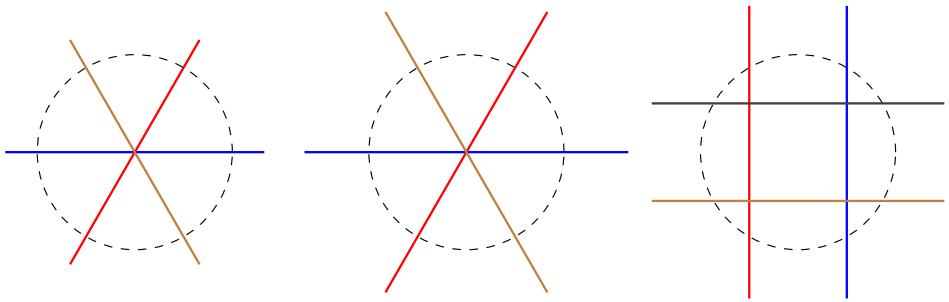


Figure 10: General model, S = 8 and W = 0.05, Gaussian population density; for L around 12 the **star** and **hashtag** shapes are almost equally good.

Figure 11: General model, S = 4 and W = 0: for L = 18 the star is optimal but for L = 21 the **3x3 grid** is optimal for the Gaussian density,

So our model doesn't seem to give the desired qualitative property. (Though we lack some efficient code to study large L for different networks).

Open problems:.

- Devise a better model!
- Efficient code for our model with larger L.

References: https://arxiv.org/abs/1902.08786
Unpublished notes on request.

Topic 6: A layer model for random DAGs

Probability models for DAGs arise (explicitly or implicitly) in many contexts, but (to my knowledge) there is no broad survey of such models.

Someone should write one!

I will use a quite natural "layer" model. This is reminiscent of neural network models, but one can study different types of question. I will consider a minimum-cost flow question.

Take M layers, each with N vertices. For each $1 \leq i \leq M-1$ create directed edges from some vertices in layer i to some vertices in layer $i+1$. The choice of edges is uniform random, subject to the constraint

each layer-i vertex has out-degree 2, and each layer-$(i+1)$ vertex has in-degree 2.

This defines a random graph with MN vertices w and with $(2M-1)N$ directed edges e.

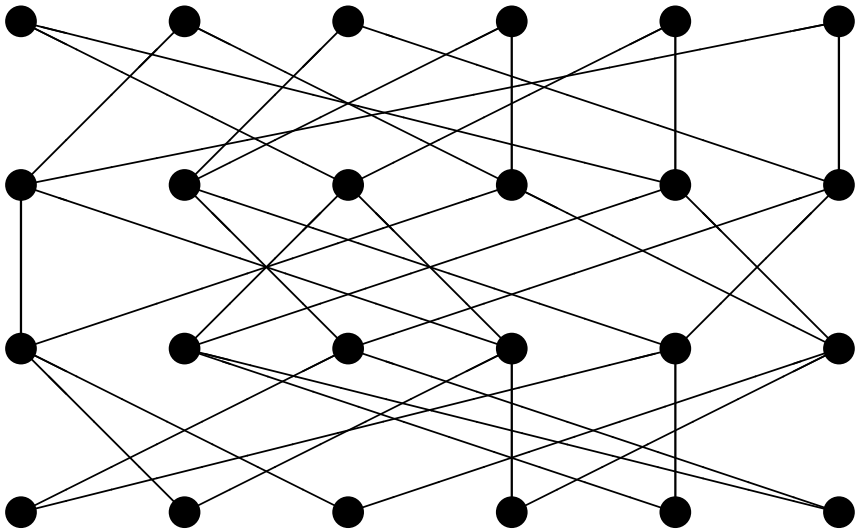


FIGURE 1. A realization of the random layer graph with $M = 4$, $N = 6$.

The **cost** of a flow with volume v across edge e is a function $\Phi(e, v) > 0$. For the model we specify a probability distribution over functions $\phi(v)$. So a realization of the random network is a realization of the random DAG, with a realization of i.i.d. functions $(\Phi(e, v),\ e \in \mathrm{Edges})$.

Within a realization, there is a minimum-cost flow of any given volume, from the top to the bottom. To consider $M, N \to \infty$ limits we work with normalized volume v^* and normalized optimal cost c^*, that is the volume of flow is Nv^* and the total cost is MNc^*. In the limit we will show how to determine c^* as a function of v^*. This will depend only on the distribution of the random function $\Phi(v)$.

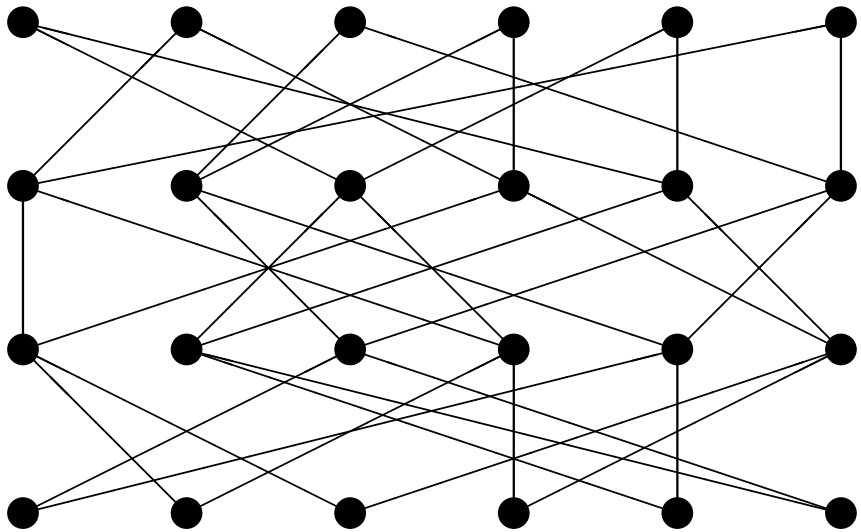


FIGURE 1. A realization of the random layer graph with $M = 4$, $N = 6$.

The local limit network is just the 4-regular tree, so we do calculations within the tree, via the **cavity method** of statistical physics. This method will give the answer in terms of a recursive distributional equation (particular type of fixed-point equation), which usually can only be solved numerically.

This is a non-rigorous method (would require work to make rigorous – maybe not worth doing): hard to explain what you're doing but easy to actually do (if you have seen a simpler example first).

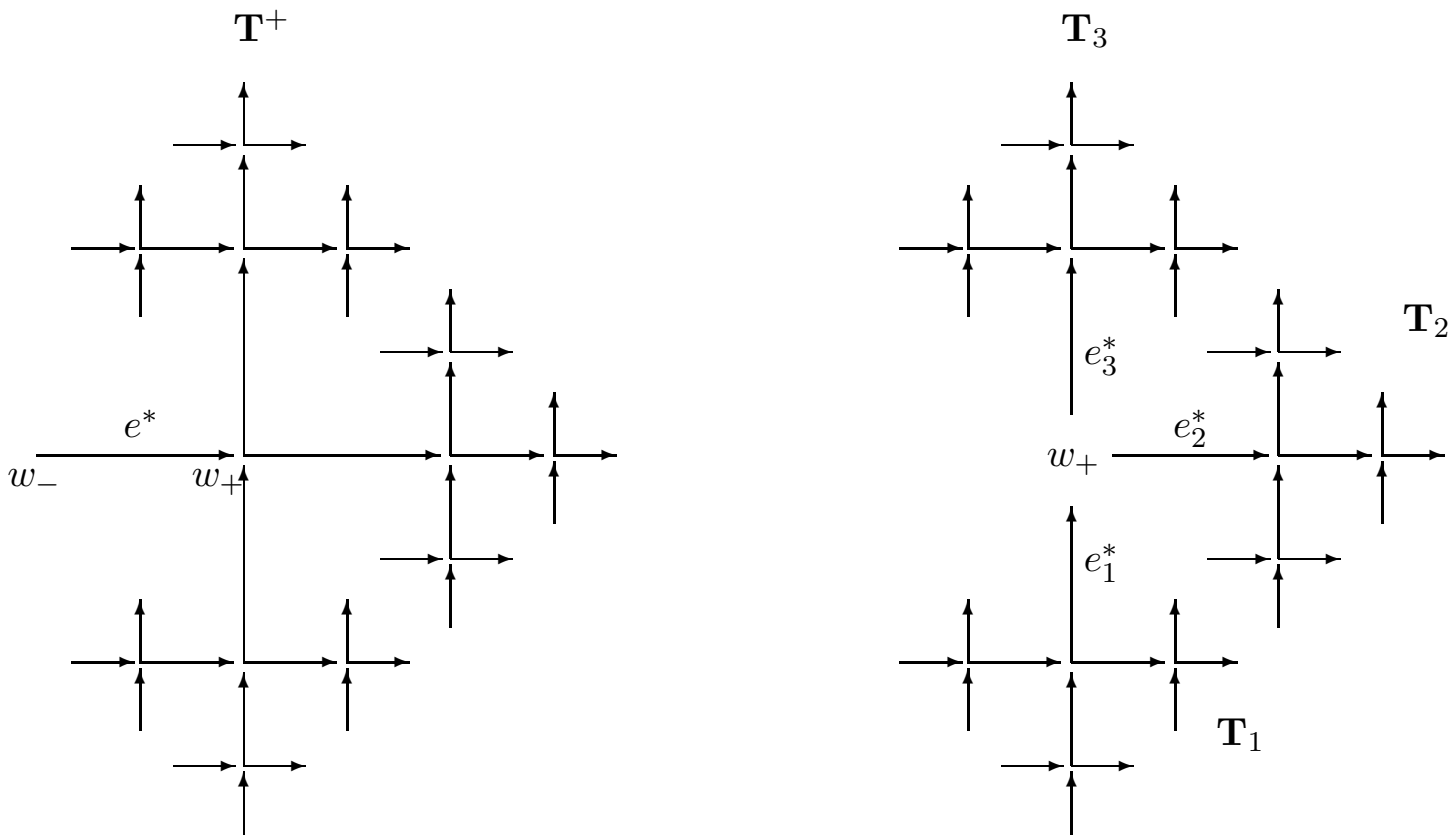


Figure 6. The tree $\mathbf{T}^+$ and its recursive decomposition into $\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3$.

Directed edge e^* in 4-regular tree defines 3 subtrees.

Imagine random process $\mathbf{X} = (X(v), v \geq 0)$ defined as
$X(v) :=$ relative cost of optimal network flow constrained
to have flow v across e^* (relative to flow 0).
Can relate optimal flow on tree to optimal flows on subtrees

$$\mathbf{X} =_d F_\lambda(\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \Phi_1, \Phi_2, \Phi_3)$$

$F_\lambda(x_1(\cdot), x_2(\cdot), x_3(\cdot), \phi_1(\cdot), \phi_2, (\cdot), \phi_3(\cdot))$ is the function

$$v \to \inf_{v+v_1=v_2+v_3} \sum_{i=1}^{3} (\phi_i(v_i) - \lambda v_i + x_i(v_i))$$

$$- \inf_{v_1=v_2+v_3} \sum_{i=1}^{3} (\phi_i(v_i) - \lambda v_i + x_i(v_i))$$

Because we are minimizing under constraint, we introduced a Lagrange multiplier λ and seek to minimize, over flows $(f(e))$ on the entire tree,

$$\sum_e (\Phi(e, f(e)) - \lambda f(e))$$

Minimizing the quantity above for a given value (v, say) of $f(e^*)$ gives

$$\Phi(e^*, v) - \lambda v + X^+(v) + X^-(v)$$

where the flow is decomposed into flows on T^+ and on T^- with the same value of $v = f(e^*)$. Thus the optimal flow is obtained by minimizing over all possible values v, and so the optimal flow across e^* is

$$f(e^*) = \arg\min_v \left(\Phi(e^*, v) - \lambda v + X^+(v) + X^-(v)\right).$$

This is all for one realization of the network.

Finally, this optimal flow $f = f_\lambda$ has normalized volume and cost

$$v(f_\lambda) = \mathbb{E}[f(e^*)]; \quad c(f_\lambda) = \mathbb{E}[\Phi(e^*, f(e^*))].$$

This gives the relationship between v^* and c^* via this parametrized-by-λ format.

Admittedly this is a rather artificial problem, with an unappealing solution (depends on the solution of the RDE for random process $(X(v), v \geq 0)$); and hard to make rigorous.

Deservedly obscure!

Open problem: Other aspects of layer model of DAGs from a "random graphs" perspective?

References:

https://arxiv.org/abs/cond-mat/0502346. (This model)
https://www.stat.berkeley.edu/users/aldous/Papers/me101.pdf (Cavity method)