Fast Algorithms for Constructing Maximum Entropy Summary Trees (Howard Karloff)

This talk [1] revolved around a question. A simple question, really. *"How to draw a rooted tree?"* Not *any* tree, though: a big one. A huge, gigantic n-node tree T, with $n \gg \lfloor 2^{22.891112} \rfloor$ [2], one a tiny sheet of paper (or a screen, for that matter). Most of the time, the approach is either to

- draw the whole thing, get a binocular and hope something can be seen at all;
- go for interaction, and draw/zoom on parts of the graph on-the-fly.

Here, none of this: T is T, magnificent n-node weighted tree; and we are given a parameter $k \geq n$, to be though of as $k \ll n$; the task is to draw a k-node tree T', the "best summary" of T. (At this point – a demo! From the Mathematical Genealogy Project: the tree being made of the academic descendants of a obscure mathematician, someone named Gauss. That's a hell of a tree.)

The key of their approach is *not* to draw only k carefully chosen nodes of T, and dismiss the others; but instead to allow T' to combine subsets of T's nodes into a big, meta-node named "others" (†).

Sure. But what is a "good summary" anyway? Quite naturally, the authors went for entropy – a tree is *good* if, assigning weights to nodes in the straightforward fashion (for $v \in T'$, $\omega_v \propto$ the total weight of original nodes of T in the subtree of T' rooted at v), the entropy

$$H(T') = -\sum_{v \in T'} \omega_v \log \omega_v \leq \log k$$

is big. That is, if T' is as balanced as possible, or phrased differently remains very *informative*.

So, what's the problem? The problem is exactly (†), when it comes to computing such an optimal k-node summary tree. Indeed, if we allow an algorithm to combine an arbitrary subset of a node v into a metanode others_v (and we do!), there is the slight problem of having exponentially many possibilities – and that could turn out pretty bad. An intuitive idea [3] would be to sort v's children by non-decreasing weights, and combine together a *prefix* (the first, say, ℓ of them) – they shouldn't be too important anyway. Sadly, it does not work – there are counter examples, even for $n = 7$ and $k = 4$.

In a previous work [4], Kenneth Shirley and Howard Karloff managed to give a *pseudo*polynomial algorithm (polynomial in the total weight), and a polynomial-time (additive) approximation one. This works came for the kill, removing the "pseudo" to give

- a $O(k^2 n + n \log n)$-time algorithm for computing an optimal k-node summary tree;
- a (better) $O(n + \text{poly}(k, 1/\varepsilon))$-time approximation algorithm

as well as a faster (albeit non-provably optimal) greedy algorithm. The crux of the approach? Prefixes don't work, alright. But *extended prefixes* do! Instead of (after sorting a node v's children) merging the nodes $\{1, \ldots, \ell\}$, it is sufficient to merge the $\{1, \ldots, \ell, \ell'\}$ for some $\ell' \geq \ell + 1$ (yes, it is surprising to me too).
And *voilà*! Polynomial time.

Testing Equivalence of Polynomials under Shifts (Rafael Mendes de Oliveira)

And now, some testing [5]! Rafael has a device: a blackbox computing a n-variate polynomial $P \in \mathbb{F}[X_1, \ldots, X_n]$ (for some fixed field $\mathbb{F}$). And another one – computing $Q \in \mathbb{F}[X_1, \ldots, X_n]$. On inputs $\mathbf{x} \in \mathbb{F}^n$, these two devices allow him to compute $P(\mathbf{x})$, $Q(\mathbf{x})$ in constant time.
Rafael also has two coauthors, Zeev Dvir and Amir Shpilka, and a question:

Question 1 (Shift Equivalence Testing (SET))**.** Given a bound on their degree, decide if P and Q are *morally* the same polynomial. That is, if there is a shift vector $\mathbf{a}$ such that $P(\cdot + \mathbf{a}) = Q(\cdot)$ (and if so, please, find one.)

This is arguably a natural generalization of the infamous Polynomial Identity Testing (PIT) problem, where the goal is to decide whether P is the all-zero polynomial or not. PIT does not have any known efficient deterministic algorithm, but a very simple polynomial-time randomized one; and the task of improving this state of affairs has ramifications in many areas.

So SET must be difficult, since it not only generalizes the question, but also asks for an explicit witness – this vector $\mathbf{a}$. After giving some background on what is known about related questions (is P "shift-equivalent" to a polynomial that can be represented with very few non-zero coefficients, for instance), Rafael gave the main result of their paper:

Theorem 1. There is an efficient (randomized) algorithm for SET. Furthermore, the only randomized part comes from solving instances of PIT.

In other words, in spite of what it seems, SET is not much harder than PIT; and any improvement on the use of randomness for PIT directly translates to the corresponding improvement for SET!

Roughly speaking, the idea of the proof is to try to find $\mathbf{a}$ "one step at a time", by successively coming up with candidates $\mathbf{a}_d$, $\mathbf{a}_{d-1}$, $\ldots$, $\mathbf{a}_1$. How? Find $\mathbf{a}_d$ that matches the degree-d part of $P(\cdot + \mathbf{a})$ and $Q(\cdot)$ – this is just a linear system to solve, along with a call to a randomized subroutine for PIT. Then find $\mathbf{a}_{d-1}$ that matches both the degree-d and degree-$(d-1)$ parts of $P(\cdot + \mathbf{a})$ and $Q(\cdot)$ – this no longer linear, alas. But, and that's the trick, one can plug the known value of $\mathbf{a}_d$ in the quadratic system to make it collapse to a linear system, which has the same solutions as the quadratic one!

And so on, and so forth: after computing $\mathbf{a}_{k+1}$, one just has to solve a linear system to get $\mathbf{a}_k$, using the nice properties of $\mathbf{a}_{k+1}$ (simplifications "trickles down"), before calling PIT to make sure the result is correct so far. And at the end of the day, either $\mathbf{a}_1$ is a good shift-vector, or there is none – and a final call the PIT subroutine lets us know which of the two cases hold.

Hop.

[1] *Fast Algorithms for Constructing Maximum Entropy Summary Trees*, R. Cole and H. Karloff. ICALP, 2014.

[2] This is a prime number. Check it out.

[3] Rule of thumb: when I write "intuitive", it is a trap.

[4] *Maximum Entropy Summary Trees*, K. Shirley and H. Karloff, Computer Graphics Forum. 2013.

[5] *Testing Equivalence of Polynomials under Shifts*, Z. Dvir, R.M de Oliveira, and A. Shpilka. ICALP, 2014.