CS420+500: Advanced Algorithm Design and Analysis

Lectures: Jan 18 + Jan 20, 2017

Prof. Will Evans *Scribe: Tianri Gu*

In this lecture we:

- Finish up Linear Decision Tree;
- Chan's Algorithm;

1 Linear Decision Tree Continue

Lemma. *Any **Linear Decision Tree** that computes F has*

$$height \leqslant \left\lceil \log_3 \left(\sum_{outputs\ t} \#F_t \right) \right\rceil$$

Theorem. *Any **Linear Decision Tree** that computes **Element Uniqueness** has*

$$height \in \Omega(n \log n)$$

Proof. idea: Are [x = {1,2,3,4....,n}, y = {2,1,3,4...,n}] the same **connected component**?
NO! Let x be a vector of n unique numbers and $y \neq x$ be any permutation of x. There must be indices i and j s.t. $x_i < x_j$ and $y_i > y_j$. Any continuous path from x to y must contain point z with $z_i = z_j$(by **Intermediate value theorem**). z is a **NO-input**, so x and y are not in the same connected component. Thus none of the n! permutations of x are in the same **connected component** which implies $\#F_{yes} \geq$ n!.

(**Intermediate Value Theorem**: Let p be a path from points x to y where

$$p : [0, 1] \in \mathbb{R}^n, p[0] = x, p[1] = y$$

Let $q(t) = p(t)_j - p(t)i$ (the difference between i^{th} and j^{th} coordinate in point $p(t)$). At some point $q(t) = 0$ because $q(0) > 0, q(1) < 0$ so it had to have crossed from positive to negative.)

□

d^{th} order **Algebraic Decision Trees** are like **Linear Decision Trees**, but functions in the internal nodes are d^{th} order polynomials.
The number of **connected components** of input space that can reach the same leaf is $\leq d(2d - 1)^{n+h-1}$ in decision tree with height h. $\Rightarrow \Omega(n \log n)$ lower bound for **Element Uniqueness**.

2 Chan's Algorithm

We have Jarivs' March in $O(nh)$ where h is the size of Convex Hull, and Graham's Scan in $O(n \log n)$. In 1996, Timothy M. Chan combined the "best" part from both algorithms and introduced a new algorithm to find Convex Hull in $O(n \log h)$, named Chan's Algorithm.

Chan's Algorithm Detail

Given n points P and a guess h for the number of hull vertices.

1. Divide points into n/h groups of size h. ($O(n)$)
2. Use Graham's Scan to find CH of each group. ($O(h \log h) * \lceil n/h \rceil = O(n \log h)$)
3. Find lowest point P_0 in P. ($O(n)$)
4. Do Giftwrapping (Jarvis' March) for h steps.
   Find the right tangent from P_i to each group hull.
   P_{i+1} = rightmost of these tangent points.
   If $P_{i+1} = P_0$, output the hull. (Notice that the rightmost tangent of each hull can be done using circular binary search in $O(logh)$. Thus in total this step can be done in $O(\log h) * \lceil n/h \rceil = O(n/h \log h) \in O(n \log h)$)
5. Output "h h is too small".

Time Complexity Generate guesses for h^* (true number of hull vertices). Start from

$$h = 4, 16, 256...2^{2^t}$$

Then sum all the cost until the right guess h^*.

$$\sum_{h=2^{2^t}}^{h^*} O(n \lg h) = \sum_{t=1}^{\lceil \lg \lg h^* \rceil} O(n 2^t) = O(n \sum_{t=1}^{\lceil \lg \lg h^* \rceil} 2^t) = O(n \lg h^*)$$