Today: pointer-manipulation / tree D.S.

- Euler Tour Technique
- BW- Decoding

Tree Contraction (next Tuesday)

LCA's, RMQ (Th)

List Ranking - lot of in 1980s

Input: collection of nodes in potentially multiple linked lists (input = pointers to their head)

Output: compute the position of each node in the list it belongs to.
Very easy to do in \(O(n)\) time sequentially.

Wyllie's (deterministic) Algorithm:

- "pointer jumping" technique

\[
\begin{array}{c}
\text{\(O(n)\) depth,}\quad \text{\(O(n^2)\) work}
\end{array}
\]

\[
\begin{array}{c}
\text{\(D[i] = D[i] + D[\text{succ}(i)]\)}
\end{array}
\]
\[ D[i] = 1 \text{ if } \text{succ}(i) \neq \bot, \ 0 \text{ o.w.} \]
\[ \text{succ, succ'} \]

for \( k = 1 + \log(n) \):

parfor \( i \) in \([1:n]\):

\[ D[i] = D[i] + D(\text{succ}[i]) \]

\[ \text{succ}'[i] = \text{succ}(\text{succ}[i]) \]

swap (succ, succ')

After the \( k \)-th iteration:

1. If \( \text{succ}(i) \neq \bot \), \( D[i] = 2^k \) and there are \( 2^k \) edges to \( \text{succ}(i) \) in the list.

2. If \( \text{succ}(i) = \bot \), then \( D(i) = \text{rank}(i) \)

\[ \Rightarrow \text{total \# rounds is } \log(n) \]

\[ \text{Work} = O(n \log(n)) \]

\[ \text{Depth} = O(\log^2 n) \text{ (in BF model)} \]
$O(n)$ work overall

$O(n \log(n))$ work

\[ \frac{n}{\log(n)} = O(n) \text{ work overall.} \]

Random sampling with rate \( \frac{1}{\log(n)} \Rightarrow \frac{n}{\log(n)} \) samples in expectation.

\[ P[\text{Bad Event}] \leq \frac{1}{n^c} \Rightarrow P[\text{Good Event}] \geq 1 - \frac{1}{n^c} \]

We say that an event (or cost bound) occurs (holds) \textbf{whp} with high probability if

\[ g(n) \in O(f(n)) \text{ whp if } g(n) \in O(c \cdot f(n)) \text{ with probability } \geq 1 - \frac{1}{n^c} \]

\[ e_i \quad E[c_i] \quad E[\text{Alg}] = \sum_i E[c_i] \]

\[ c_2 \quad E(c_2) \quad c_k \quad E(c_k) \]
Let $X_1, \ldots, X_n$ be independent R.V.'s where $X_i \in [0, 1]$

Define $X = \sum_{i=1}^{\infty} X_i$, $\mu = E[X]$

A $\delta > 0$:

$$P_r \left[ X \geq (1+\delta) \mu \right] \leq \exp \left(-\frac{\delta^2 \mu}{2(1+\delta)}\right)$$

- Hoeffding's inequality, Bernstein's inequality
(1) $O\left(\frac{n}{\log(n)}\right)$ samples w.h.p.

(2) Want to bound the search length $A$ sample

\begin{enumerate}
\item $X_i = \text{I.R.V.} = 1$ iff $i$th mode is sampled
\item $X = \sum_{i=1}^{n} X_i$, \quad $P\{X_i = 1\} = \frac{1}{\log(n)}$
\item $E(X) = \frac{n}{\log(n)} \in O(\log(n))$
\end{enumerate}

Apply Chebyshev:

$$P\left(X \geq (1+b)E(X)\right) \leq \exp\left(-\frac{\delta^2 E(X)}{3}\right)$$

$$\leq \exp\left(-\frac{\delta^2 \log(n)}{3}\right)$$

$$= \frac{1}{n^{2/3}} \approx \frac{1}{n^c}$$

$$= O\left(\frac{n}{\log(n)}\right) \text{ samples w.h.p.}$$

(2) Argue that traversal length for an arbitrary sample is short: $O(\log^2 n)$ w.h.p.
\( Z = \# \text{ vertices traversed by our sample} \)
\[
P(Z \geq x) = \left(1 - \frac{1}{\log(n)}\right)^x
\]
\[
x = c\log^2(n)
\]
\[
P(Z \geq c\log^2(n)) = \left(1 - \frac{1}{\log(n)}\right)^{c\log(n)\log(n)}
\]
\[
1 + x \leq \left(1 + \frac{x}{n}\right)^n \leq e^x
\]
\[
-c\log(n) \quad n \geq 1, \quad |x| \leq n
\]
\[
\leq e
\]
\[
= \frac{1}{n^c}
\]
\[
\Rightarrow \text{high probability bound for one sample.}
\]

**TODO:** is this tight?

What about all samples? Boyle's inequality

Union bound

\[
P(E_1 \cup E_2 \cup \ldots \cup E_n) \leq P(E_1) + P(E_2) + \ldots + P(E_n)
\]
$1) \frac{n}{\log(n)}$ sample whp.

\[ P(\text{any sample has traversal length } \geq c \log^2(n)) \leq n \left( \frac{1}{e^c} \right) = \frac{1}{n^c} \]

$2) \text{Each one traverses } O(\log^2(n)) \text{ nodes whp.}$

**Randomized List Rank ($A$)**

1. Samples = sample w.p. $\frac{1}{\log(n)}$
2. Splice out all non-sampled nodes from the list
3. Deterministic List Rank ($\text{Samples}$)

**Whylic**

1. $O(n)$ work, $O(\log(n))$ depth
2. $O(n)$ work, $O(\log^2(n))$ depth whp
3. $O(n)$ work whp, and $O(\log^2(n))$ depth (det).

$O(n)$ work whp, $O(\log^2(n))$ depth whp.
Historical Notes:

- Cole-Vishkin: $O(n)$ work

"Ruling Sets"

May discuss in a future lecture

$O(\log \log^* n)$ time algorithm.

$O(\log^2 n \log^d n)$ depth alg.