CS229r: IT ∈ CS

4/16/2019

LECTURE 21

TODAY: DATA STRUCTURE LOWER BOUND
(★ DYNAMIC)

- MODEL
- RESULTS DESIRED
- PROOF STYLE
- SOME PROOFS

DATA STRUCTURE PROBLEM

- "Maintain data under updates while answering queries quickly".
- 3 considerations
  - Update time: Time per Update: t_u
  - Query time: Time to answer query: t_q
  - Space: S.
- Today: Just worry about t_u vs. t_q. (Will allow $S=\infty$)
  (for static, this is uninteresting/unuseful)

Examples:

"SET MEMBERSHIP":

- UPDATES $\in$ $\{\text{INSERT}, \text{DELETE}\} \times U \triangleq \{-2^{\ell}, \dots +2^{\ell}\}$
- QUERY $\in U$
- PROBLEM INSTANCE: $\sigma_1, \sigma_2 \dots \sigma_m$

  - $\sigma_i \in$ UPDATES $\cup$ QUERY

    $S_i \triangleq \text{\&}S(\sigma_1 \dots \sigma_i)$

  - Every sequence $\sigma_1 \dots \sigma_i$ represents set $S_i \subseteq U$ as follows

    - $S_0 = \emptyset$ (empty set)
    - $S_i = S_{i-1}$ if $\sigma_i \in$ QUERY
      $= S_{i-1} \cup \{j\}$ if $\sigma_i = (\text{INSERT}, j)$
      $= S_{i-1} \setminus \{j\}$ if $\sigma_i = (\text{DELETE}, j)$

  - QUERY $\sigma_i = j \Rightarrow$ responds with $\mathbb{1}_{j \in S_i}$

    $n \triangleq \max_{i \in [m]} |S_i|$

- Usual Solutions: Search trees: $t_u, t_q = O(\log n)$

  HASH tables: $t_u, t_q = O(1)$

Abstractly: Data Structure problem given by two functions

Update$(\sigma_1 \dots \sigma_m)$ & Query$(\sigma_1 \dots \sigma_m)$

Data Structure Solution Model: CELL PROBE (Yao)

Solution = 1. "Sequence of Cells" $\boxed{y_1}\boxed{y_2} \cdots \boxed{y_s} = \boxed{y}$

$y_i \in \{-2^w, \dots 2^w\}$

2. **Updates**: Given update $U \in$ UPDATE

$\rightarrow$ { Sequence of operations of the form READ(j) & WRITE(j, V)

(READ $r_1, r_2 \dots r_k$

WRITE $(r_{k+1}, V_{k+1}) \dots (r_t, V_t)$ $t = t_u$

r_i, V_i determined by U & previous READS (ie., ~~READS~~. $y_{r_1} \cdots y_{r_{i-1}}$)

---

3. Query: Only uses READS.; takes time $t = t_q$.

HASHING: if $W \geq 2\log m$, then can do ~~ooo~~ SET MEMBERSHIP in $O(1)$ time.

- Does any problem require ~~≥~~ $\omega(1)$ time?

OPEN
- Does any dynamic problem require ~~w.p.~~ $(\log m)^{\omega(1)}$ time? (with $w = (\log m)^c$)
- " (static) " $(\log m)^{\omega(1)}$ with $w = (\log m)^c$?

TODAY: WILL SEE PROBLEM FOR WHICH

$$t_U + t_q = \Omega(\log m).$$

PROBLEM: Very Basic Problem in Computational Geometry

- Array Maintenance with Prefix Queries: (≤ 1-d range counting)
- Maintain Array $A[0] \dots A[n-1]$ under $\Delta_i \in \{-2^w, \dots, 2^w\}$
  - UPDATE: $(i, \Delta_i) \longrightarrow$ Add Δ_i to $A[i]$
  - QUERY: $i \longrightarrow$ return $\sum_{j \le i} A[j]$.

Theorem: [FREDMAN + SAKS '89, PATRASCU + DEMAINE '04, LARSEN]

$$t_U + t_q \geq \Omega(\log n) \quad \text{provided} \quad w = (\log n)^{O(1)}.$$

Proof Overview: "Information Transfer Argument".

- Assume ∃ Data Structure Contradicting Theorem
- Pick "random" updates sequence + query $\overline{U} = U_b, U_c, U_a$ (before, during, after)
- Give $\overline{U}$ to Alice, (U_b, U_a) to Bob
- Use DS to build low comm. message from $A \to B$ that allows Bob to reconstruct U_c
- (contradiction)

More insight into "omitted U_c".

- Since it is a small # updates & processed quickly

quick proc → Alice can send short message to Bob.

- Since Bob can answer all queries after

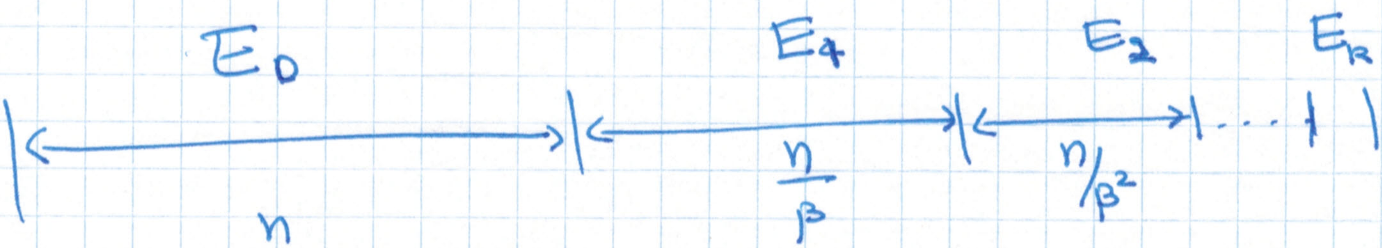
correct operation → (U_b, U_c, U_a) & many queries depend on U_c

he can reconstruct U_c

<u>Strategy 1</u>:

- Fix $\beta \approx (\log n, w, t_u)^c$

- Sequence of updates broken into epochs $\quad k = \log_\beta n$



$$E_i = \langle (j \cdot \beta^i, \Delta_{j,i}) \rangle_{j = 0,1,\dots \frac{n}{\beta^i} - 1}$$

$\Delta_{j,i} \in \{-2^\ell \dots 2^\ell\}$ ↑ uniform independent

- Finally 1 query $q \in_U [n]$.

~~Claim~~:

At time $m = n + \frac{n}{\beta} + \dots \frac{n}{\beta^k}$:

- for every cell y_j - ~~ask~~ assign color i if it was last updated in ~~cell~~ epoch E_i.

- $C_i \triangleq \{ j \mid y_j$ ~~updat~~ has color $i \}$
- $P(q) = \{ j \mid$ query q leads to $\text{READ}(j) \}$

Claim: $\mathbb{E}_{q, E_1 \dots E_k} \left[|P(q) \cap C_i| \right] = \Omega(1)$ provided $\beta \geq \dots$

(Defer Claim for now):

But assuming Claim to hold ~~for~~ queries $P(q)$

$$\mathbb{E}\left[|P(q)|\right] = \sum_{i=1}^{k} \mathbb{E}\left[|P(q) \cap C(i)|\right] \geq \Omega(k)$$
$$= \Omega(\log_\beta n)$$

$\boxtimes$

Proof of Claim:

Suppose ~~Data~~ Data Structure exists. Consider following CC Problem

Alice $\leftarrow E_1 \ldots E_R, i$ Bob $\leftarrow (E_1 \ldots E_R) \setminus E_i$

$$\xrightarrow{\quad m \quad}$$

$$\downarrow$$

$$\hat{E}_i$$

~~Bo~~

- $H(E_i \mid \{E_1 \ldots E_R\} \setminus E_i) = |E_i| \cdot \ell$ $[\Delta_{i,j}$ independent$]$

$$= \frac{n}{\beta^i} \cdot \ell$$

- Alice's protocol:

(i). Send to Bob, index + content of all cells in $C_{i+1} \ldots C_R$

length of message $\approx \frac{n}{\beta^{i+1}} \cdot w \cdot t_u$

(ii). Sends to Bob, $Q \subseteq [n]$ for which he will be wrong based only on (i) & how to correct them.

(will show this only requires $\epsilon \cdot \frac{n}{\beta^i} \cdot \ell$).

- Bob's recovery:

  - After receiving (i): Reconstructs Data Structure which is erroneous only on C_i; runs ~~quer~~ all queries & see
  - ~~By~~ what their "majority vote" is on $\{\Delta_{i,j}\}_j$
  - Alice's correction: corrects incorrect $\Delta_{i,j}^{F}$ (giving index j & correct

<u>Subclaim</u>: Since $\mathbb{E}\left[|C_q \cap P(q)|\right] \le \epsilon \Rightarrow$ Bob is incorrect on $O(\epsilon)$ fraction of j's.

$\boxtimes$

Needed $\beta > \frac{W}{\ell} \cdot t_u$

$\Rightarrow$ Get $t_q = \Omega\left(\frac{\log n}{\log\left(\frac{t_u \cdot W}{\ell}\right)}\right) \approx \Omega\left(\frac{\log n}{\log\log n}\right)$

if $W, t_u = \log n^{O(1)}$

Weak! Wanted to show $t_u + t_q = \Omega(\log n)$

Better query + update sequence:

- Alternate query & update: Update$(\overleftarrow{i}, \Delta_i)$; Query$(\overleftarrow{i})$
- $\overleftarrow{i}$ = integer rep'd by i in reverse. [location det. updates random $\Delta_i \in \{-2^\ell, \ldots 2^\ell\}$ unif.]
- New analysis + etc.