Streaming Algorithms

Meng-Tsung Tsai

04/10/2018

Reminder

Written Assignment #1 is due by tonight. You need to LaTex your solution and submit it on New E3 (https://e3new.nctu.edu.tw).

You are encouraged to discuss with your classmates, TA, or me. However, the writeup shall be your own.

Reference

- "Space-Efficient Online Computation of Quantile Summaries," Greenwald and Khanna (2001)

Quantile Summaries

Problem Definition

Input: a sequence of n (possibly repeat) values $a_1, a_2, \ldots, a_n$ where n is unknown before the end of input is reached, an integer $q \in [1, n]$, and a value $\boldsymbol{\varepsilon} \in [1/n^2, 1]$. Define $A = \{a_i : i \in [n]\}$ and rank

$$r(a_i) = |\{a_j \in A: a_j < a_i \text{ or } (a_j = a_i \text{ and } j \leq i)\}|.$$

Output:

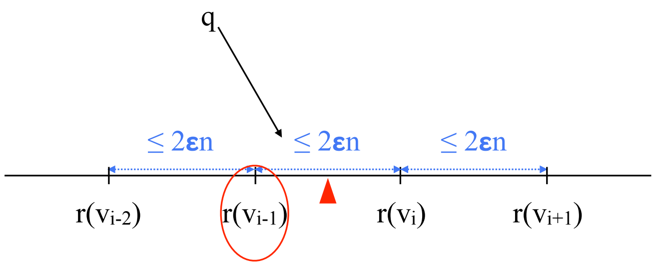
an a_i for some i so that $|r(a_i) - q| \leq \boldsymbol{\varepsilon} n$

Goal: use $O(\boldsymbol{\varepsilon}^{-1} \log \boldsymbol{\varepsilon} n)$ space (not bits).

Data Structure

A Subset V of A

Roughly speaking, we will maintain a subset $V = \{v_0, v_1, \ldots, v_{s-1}\}$ of A as A grows so that (1) $s = O(\boldsymbol{\varepsilon}^{-1} \log \boldsymbol{\varepsilon} n)$, (2) $r(v_0) = 1$, (3) $r(v_{s-1}) = n$, (4) $r(v_{i-1}) \leq r(v_i)$ for each $i \geq 1$, and (5) $r(v_i) - r(v_{i-1}) \leq 2\boldsymbol{\varepsilon} n$.



Reducing to the Bin-Ball Problem

We already know that if we sample $O(\boldsymbol{\varepsilon}^{-1} \log n)$ a_i's uniformly at random from A with replacement and keep track the min (resp. max) element in A, then we get a subset V that satisfies the desired property w.h.p., but we have no idea what $r(a_i)$ for $a_i \in V$ are.

In our programming assignment #0, we scan the input for one additional pass to figure what $r(a_i)$ for $a_i \in V$ are.

In today's lecture, we will see a deterministic algorithm that can answer this question in a single pass.

A Set S of Triples

Let $r_{min}(v_i)$ and $r_{max}(v_i)$ be under- and over-estimates of $r_n(v_i)$ among the values seen so far, after all values are read, we get $r_{min}(v_i) \leq r(v_i) \leq r_{max}(v_i)$.

We define a set $S = \{(v_i, g_i, \Delta_i) : v_i \in V\}$ so that

(1) $g_i = r_{min}(v_i) - r_{min}(v_{i-1})$ for each $i \geq 1$, // gap between two consecutive under-estimates

(2) $\Delta_i = r_{max}(v_i) - r_{min}(v_i)$ for each i, // difference between under- and over-estimates of an single v_i

(3) $(v_0, g_0, \Delta_0) = (v_0, 1, 0)$,

(4) $(v_{s-1}, g_{s-1}, \Delta_{s-1}) = (v_{s-1}, 1, 0)$.

Some Identities (as n increases)

(1) $r_{min}(v_0) = r_{max}(v_0) = 1$

(2) $r_{min}(v_{s-1}) = r_{max}(v_{s-1}) = n$

(3) $\sum_{0\leq i<s} g_i = (\sum_{1\leq i<s} r_{min}(v_i) - r_{min}(v_{i-1}))+1 = n-1+1 = n$

(4) $r_{min}(v_i) = \sum_{j\leq i} g_j$

(5) $r_{max}(v_i) = (\sum_{j\leq i} g_j) + \Delta_i$

(6) $|\{a_j : r(v_{i-1}) < r(a_j) < r(v_i)\}| \leq g_i + \Delta_i - 1$.

Bounded $g_i + \Delta_i$

Claim. Given S, if $\max_i g_i + \Delta_i \leq 2\boldsymbol{\varepsilon}n$, then for any q one can output some v so that $|r(v) - q| \leq \boldsymbol{\varepsilon}n \coloneqq e$.

Proof. Case 1. If $q > n-e$, setting $v = v_{s-1}$ suffices.

Case 2. Otherwise, picking the smallest t (such a t exists) so that

$$r_{max}(v_t) = (\sum_{j\leq t} g_j) + \Delta_t \geq q+e, \text{ i.e. } r_{max}(v_{t-1}) < q+e.$$

Then $r_{min}(v_{t-1}) = (\sum_{j<t} g_j) = r_{max}(v_t) - g_t - \Delta_t \geq q+e-2e = q-e$.

Hence, setting $v = v_{t-1}$ suffices.

Goal: Maintain a set of triples S whose $\max_i g_i+\Delta_i \leq 2e$.

Algorithm

Pseudocode

Generate-S(){

$S \leftarrow \varnothing$; $s \leftarrow 0$; $n \leftarrow 0$;

foreach incoming a_i{
Insert (S, a_i); // add a triple into S
++ n;
}

if($n \equiv 0 \bmod 1/(2\boldsymbol{\varepsilon})$)
Compress(S); // merge some triples in S
}

Pseudocode

Insert(S, a_i){

if(a_i has $r(a_i) < r(a_j)$ for all $j < i$){ // i.e. $a_i < a_j$ for all $j < i$
insert a triple (a_i, 1, 0) to S; // it requires to increase $r_{min}(v)$ and $r_{max}(v)$ by 1 for all v in V where $r(v) \neq r(a_i)$
}
if(a_i has $r(a_i) > r(a_j)$ for all $j > i$){ // i.e. $a_i \geq a_j$ for all $j < i$
insert a triple (a_i, 1, 0) to S; // no effect on $r_{min}(v)$ for other v's
}
if(a_i is any other value){
find v_{t-1}, v_t in V so that $v_{t-1} \leq a_i < v_t$;
insert a triple (a_i, 1, int($2\boldsymbol{\varepsilon}$n)-1) in-between ($v_{t-1}$, g_{t-1}, Δ_{t-1}) and (v_t, g_t, Δ_t); // all triples before a_i's triple have no changes
} // all triples after a_i's need to increase $r_{min}(v)$ and $r_{max}(v)$ by 1
} // $r_{min}(a_i) \geq r_{min}(v_{t-1}) + 1$ and $r_{max}(a_i) \leq r_{max}(v_t) = r_{min}(v_{t-1})+1+g_t+\Delta_t-1$

Pseudocode

Delete(S, v_i){ // a building block of Compress(S)

replace the two triples (v_i, g_i, Δ_i), (v_{i+1}, g_{i+1}, Δ_{i+1}) with (v_{i+1}, g_i+g_{i+1}, Δ_{i+1}); // Exercise: verifying this replacement doesn't change the $r_{min}(v)$ and $r_{max}(v)$ for all v in V.

$s \leftarrow s-1$;
}

Pseudocode

Compress(S){

for($i = s-2$; $i \geq 0$; i--){
if(band(Δ_i, int($2\boldsymbol{\varepsilon}$n)) $\leq$ band(Δ_{i+1}, int($2\boldsymbol{\varepsilon}$n)) and $g_i^* + g_{i+1} + \Delta_{i+1} \leq$ int($2\boldsymbol{\varepsilon}$n)){
Delete(S, v) for all triples (v, g(v), Δ(t)) in S whose ancestor is triple (v_i, g_i, Δ_i) (including v_i); // $g_i^* \coloneqq \sum_{v \text{ is a descendant of } v_i} g(v)$
Replace(v_{i+1}, g_{i+1}, Δ_{i+1}) with (v_{i+1}, $g_i^*+g_{i+1}$, Δ_{i+1});
}
}

... (v_{i-c}, g_{i-c}, Δ_{i-c}) $\underbrace{(v_{i-c+1}, g_{i-c+1}, \Delta_{i-c+1}) \quad \ldots \quad (v_i, g_i, \Delta_i)}$ (v_{i+1}, g_{i+1}, Δ_{i+1})

triple (v_i, g_i, Δ_i)'s descendants

Pseudocode

Compress(S){

for($i = s-2; i \geq 0;$ i--){

if(band(Δ_i, int($2\boldsymbol{\varepsilon}$n)) $\leq$ band(Δ_{i+1}, int($2\boldsymbol{\varepsilon}$n)) and $g_i^* + g_{i+1} + \Delta_{i+1} \leq$ int($2\boldsymbol{\varepsilon}$n)){

Delete(S, v) for all triples (v, g(v), Δ(t)) in S whose ancestor is triple (v_i, g_i, Δ_i) (including v_i); // $g_i^* := \sum_{v \text{ is a descandant of } vi} g(v)$

Replace(v_{i+1}, g_{i+1}, Δ_{i+1}) with (v_{i+1}, g_i^*+g_{i+1}, Δ_{i+1});

}

}

... (v_{i-c}, g_{i-c}, Δ_{i-c}) (v_{i+1}, g_i^*+g_{i+1}, Δ_{i+1})

Definition of Descendant

Map each triple (v_i, g_i, Δ_i) in S to a node X_i in tree T. Let R be a special node created as the root for T. Note that T has s+1 nodes.

We let V_j be the parent of V_i if j is the samllest index greater than i whose band(Δ_j, int($2\boldsymbol{\varepsilon}$n)) > band($\Delta_j$, int($2\boldsymbol{\varepsilon}$n)); if no such a j exists, let R be the parent of V_i.

Claim. For any V_i, its descendant nodes form a consecutive segment in S.

Definition of band(Δ, $2\boldsymbol{\varepsilon}$n)

band(Δ, int($2\boldsymbol{\varepsilon}$n)) := α so that

$$p - 2^{\alpha} - (p \bmod 2^{\alpha}) < \Delta \leq p - 2^{\alpha-1} - (p \bmod 2^{\alpha-1})$$

where p = int($2\boldsymbol{\varepsilon}$n).

int($2\boldsymbol{\varepsilon}$n) may increase as n increase, and setting band() function as the above makes the cutting boundaries stable.

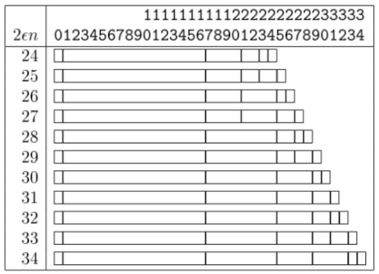


Figure 1: Band boundaries as $2\epsilon n$ progresses from 24 to 34. The rightmost band in each row is band 0.

Analysis

Result

$s = O(\varepsilon^{-1}\log \varepsilon n)$. It needs to get familar with the above notion to understand the analysis. We may lose our focus to go through the details in class. If interested, pick up Section 2.3 (< 2 pages) in the reference paper.

Applications

Heavy Hitter (Insertion-only)

If a value appears more than rn times, if we query the Greenwald and Kanna structure for every $q \in [2\varepsilon n, 4\varepsilon n, 6\varepsilon n, ..., n]$, we can output a set K so that every value $k \in K$ has frequency $\geq (r-2\varepsilon)n$ and every value k whose frequency $\geq rn$ is in K.

Convex Hull

One can use Greenswald and Kanna quantile structure to compute the convex hull of n given points in R^2 in $O(1/\delta)$ passes using $O(h\, n^{\delta} \log n)$ space.

You may find more applications from the papers that cite Greenswald and Kanna's paper.