CMPSCI 711: More Advanced Algorithms

Section 6-1: Sliding Windows and Smooth Histogram

Guest Lecturer: Michael Crouch

Last Compiled: April 29, 2012

Sliding Windows Overview

- Variation of streaming model with **window length** n (known).
- Only care about most recent n things.
- For input $a_1 a_2 a_3 \ldots a_t$, we care about $a_{t-n+1} \ldots a_{t-1} a_t$.
- Resources sublinear in n.

Example Problem: BITCOUNTING

- Input $\mathbf{a}$, $a_i \in \{0, 1\}$.
- Window W of length n.
- Output $\|\mathrm{W}\| \equiv \sum_{i=t-n+1}^{t} a_i$.
- [Illustrative whiteboard example.]
- Can't get exact solution:
  - When we add one item to the right, one item slides off of the left.
  - We'd have to remember **every** element to keep an exact sum.
- We'll need to approximate! Ideas?

First Idea:

- Note that one sum is $O(\log n)$ bits. We **can** keep sums in **some** ranges.
- Maintain sum of all elements in various suffixes of the stream ("buckets").
- If there's a bucket that's "about the same size" as the window, it'll have "about the same" set of elements, so it'll have "about the same" sum.

Suffix Buckets

- Bucket: **start time** and (exact) sum of the elements since that time.
- Maintain buckets $B_1, B_2, \ldots, B_k$ of lengths

$$B_1 \supsetneq B_2 \supsetneq B_3 \supsetneq \cdots \supsetneq B_k$$

  [helpful whiteboard picture]
- (k not fixed, but we'll prove it bounded)
- When we read a new element,
  - Add the new element to every existing bucket.
  - Create a new bucket B_{k+1} containing only that element (and shift indices).
  - Delete enough "unnecessary" buckets to keep space usage down.
  - Keep enough buckets to have a good estimator for $\|W\|$.

Unnecessary Buckets

- Delete as many buckets as possible while maintaining:
  - $B_1 \supseteq W \supsetneq B_2$
    - We need one bucket that includes the entire window.
  - $\text{len}(B_i) \leq (1+\epsilon)\,\text{len}(B_{i+1})$
    - We need one bucket at each power of $(1+\epsilon)$.
- "As many as possible"? We maintain [equivalent via whiteboard]
  - $\text{len}(B_i) \leq (1+\epsilon)\,\text{len}(B_{i+1})$
  - $\text{len}(B_i) > (1+\epsilon)\,\text{len}(B_{i+2})$
- First property will show we get small error.
- Second property will show we use small space.

Estimation

- We've maintained

$$\mathsf{len}(B_2) < \mathsf{len}(W) \leq \mathsf{len}(B_1) \leq (1+\epsilon)\,\mathsf{len}(B_2)$$

- This gives us an estimate:

$$\begin{aligned} \|B_2\| \leq \|W\| &\leq \|W\| + \|B_1 \setminus W\| \\ &\leq \|W\| + \epsilon\,\mathsf{len}(B_2) \\ &\leq \|W\| + \epsilon n \end{aligned}$$

- Space usage:
  - Length of buckets increases by at least $(1+\epsilon)$ for every 2 buckets.
  - $O(\log_{1+\epsilon} n) = O(\epsilon^{-1} \log n)$ buckets.
  - Each bucket requires $O(\log n)$ bits.

$$O(\epsilon^{-1} \log^2 n) \text{ bits.}$$

- Could do slightly better by using $\frac{1}{2}(\|B_1\| + \|B_2\|)$.

Was This a Good Estimator?

- We obtained $\left\|\widehat{\mathrm{W}}\right\| \in \|\mathrm{W}\| \pm \epsilon n$.
- If $\|\mathrm{W}\| \ll n$, this is a terrible estimator for $\|\mathrm{W}\|$.
- [Insightful whiteboard picture shows the bad case: the only 1s in the stream occur in $\mathrm{B}_1 \setminus \mathrm{W}$.]
- How can we do better?

Contents-Based Histograms

- In the last algorithm, we split windows based on **size**.
- This gave us errors proportional to size.
- We want error proportional to **number of** 1**s**.

Contents-Based Buckets

- We previously maintained:
  - $B_1 \supseteq W \supsetneq B_2$
  - $\text{len}(B_i) \leq (1+\epsilon)\,\text{len}(B_{i+1})$
- Now, maintain:
  - $B_1 \supseteq W \supsetneq B_2$
  - $\|B_i\| \leq (1+\epsilon)\,\|B_{i+1}\|$
- Error:

$$\|B_2\| \leq \|W\| \leq (1+\epsilon)\,\|B_2\|$$

- Number of buckets $O(\log_{1+\epsilon} \|W\|) = O(\epsilon^{-1} \log n)$.
- Total size $O(\epsilon^{-1} \log^2 n)$ bits.

How Can We Extend This?

- Let's look at which properties of $\|\cdot\|$ we used.
- $\|\cdot\|$ has a very strong property: For any buckets A, Z, the concatenation AZ satisfies $\|\mathrm{AZ}\| = \|\mathrm{A}\| + \|\mathrm{Z}\|$.
  - Probably too strong.
- We maintained one bucket at each power of $(1+\epsilon)$:

$$\|\mathrm{B}_{i+1}\| \leq \|\mathrm{B}_i\| \leq (1+\epsilon)\,\|\mathrm{B}_{i+1}\| \tag{1}$$

- No matter what we add later, these buckets will **always** satisfy (1).
- Formally, for any buckets A, B where B is a suffix of A, and for any bucket Z, we have the implication

$$\frac{\|\mathrm{A}\| \leq (1+\epsilon)\,\|\mathrm{B}\|}{\|\mathrm{AZ}\| \leq (1+\epsilon)\,\|\mathrm{BZ}\|}$$

- If "most" of the 1s in A are in its suffix B, then "most" of the 1s in AZ are in BZ.

What Other Functions Have This Property

- Consider the Distinct Elements problem, $\|\cdot\|_0$.
- When B is a suffix of A, do we have

$$\frac{\|A\|_0 \leq (1+\epsilon)\|B\|_0}{\|AZ\|_0 \leq (1+\epsilon)\|BZ\|_0} \quad ?$$

- Yes! Consider A, B, Z as sets:

$$\begin{aligned}\|AZ\|_0 &= \|A\|_0 + \|Z \setminus A\|_0 \\ &\leq (1+\epsilon)\|B\|_0 + \|Z \setminus A\|_0 \\ &\leq (1+\epsilon)\|B\|_0 + (1+\epsilon)\|Z \setminus B\|_0 \\ &\leq (1+\epsilon)\|B\|_0\end{aligned}$$

- So we can break up the input into buckets whose L_0 norms are at successive powers of $(1+\epsilon)$.
- How is this different from the L_1 case?

Keeping Sketches in Buckets

- We could keep track of the L_1 norm of a bucket using $O(\log n)$ bits.
- We can't exactly track the L_0 norm of each bucket.
- Inside each bucket, we'll use a sketch that we **already have** from the general streaming model!
- We've seen a general streaming sketch for L_0 (Lect. 2, pp. 11-12):
  - For probability $T = 1, (1+\epsilon), (1+\epsilon)^2, \ldots,$ (size of domain):
  - Sample random subsets of the domain; include each elt. w/ pr. $1/T$.
  - Choose the T where $\approx 1/e$ fraction of the subsets never occur.
- We'll put an ϵ-estimating sketch for L_0 in each bucket, and show that we can still $O(\epsilon)$-estimate $L_0(W)$.

Sketches with Approximate Buckets

- Buckets $B_1 \supsetneq B_2 \supsetneq \cdots \supsetneq B_k$, each with start time and linear L_0 sketch giving
$$D_i \in \|B_i\|_0(1 \pm \epsilon)$$
- Delete buckets, maintaining
  - $B_1 \supset W \supsetneq B_2$
  - $D_i \leq (1 + 2\epsilon)D_{i+1}$
  - $D_i > (1 + 2\epsilon)D_{i+2}$
- This guarantees
  - $\|B_i\|_0 \leq (1 + \epsilon)\|B_{i+1}\|_0$
  - $\|B_i\|_0 > (1 + \epsilon)\|B_{i+2}\|_0$
- If unbounded-stream algorithm requires space $s(n, \epsilon)$, we need space
$$O(\tfrac{1}{\epsilon} \log n(s(n, \epsilon) + \log n))$$

Sliding-window updates to underlying vector

- In the distinct-elements example, we were maintaining frequencies for some universe of elements.
- We can look at the sliding windows model as it **updates** an **underlying vector**.
  - Underlying vector $\mathbf{a} = a_1 \ldots a_m$
  - Each input symbol is a_i += z
  - We ask questions about $\mathbf{a} = \sum_{\text{updates in W}} a_i \hat{e}_i$.
    - (only last n updates "matter").
- We'll require all updates to be positive (otherwise, there are strong lower bounds on most problems).
- A natural starting question is vector norms.
- L_1 norm is the same as sum.

How can we generalize?

- We used the property

$$\frac{\|A\| \leq (1+\epsilon)\|B\|}{\|AZ\| \leq (1+\epsilon)\|BZ\|}$$

- This property isn't true for several interesting functions ($\|\cdot\|_p$, $p > 1$)
- [Some examples on the board of how L_2 norm changes]
- Can we find a weaker version of this property that helps us analyze L_2?
- Let's look at how we used this property:
  - We maintained buckets according to the top line. This is where our space bound came from.
  - If the top line is true **now**, then the bottom line will be true **forever**. This gives us our accuracy guarantee.
  - But did we need to use the same ϵ for both?

A more general property

- How can we guarantee $\|AZ\|_2 \leq (1+\epsilon)\|BZ\|_2$ for any Z?
- We **can** show (Lemma)

$$\frac{\|A\|_2 \leq (1+\frac{\epsilon^2}{2})\|B\|_2}{\|AZ\|_2 \leq (1+\epsilon)\|BZ\|_2}$$

  - L_1: if $A \setminus B$ is "unimportant", then no suffix can make it more important.
  - The L_2 norm satisfies a weaker property: if $A \setminus B$ is "unimportant", no suffix can make it **much** more important.
- We keep buckets with L_2 norms at successive powers of $(1+\frac{\epsilon^2}{2})$.
- We can be confident that no matter what happens later, we'll have an $O(1+\epsilon)$ approximation.
- $O(\log_{1+\epsilon^{-2}/2}(n)s(n))$ buckets. $s(n) = O(\epsilon^{-2}\,\text{polylog}(n))$ bits.
- $O(\epsilon^{-4}\,\text{polylog}(n))$ bits total.

Lemma

$$
\begin{aligned}
(1 - \tfrac{\epsilon^2}{2})\|A\|_2 &\leq \|B\|_2 \\
(1 - \epsilon^2)\|A\|_2^2 &\leq \|B\|_2^2 \\
\|A\|_2^2 - \|B\|_2^2 &\leq \epsilon^2\|A\|_2^2 \\
&\dots \\
\|A \setminus B\|_2^2 &\leq \|A\|_2^2 - \|B\|_2^2 \\
\|A \setminus B\|_2 &\leq \epsilon\|A\|_2 \\
&\dots \\
\|AZ\|_2 &\leq \|BZ\|_2 + \|A \setminus B\|_2 \\
\|AZ\|_2 &\leq \|BZ\|_2 + \epsilon\|A\|_2 \\
\|AZ\|_2 &\leq \|BZ\|_2 + \epsilon\|AZ\|_2 \\
(1 - \epsilon)\|AZ\|_2 &\leq \|BZ\|_2
\end{aligned}
$$

Smooth Histograms

- We'll call a function $\beta(\epsilon)$-smooth if it is
  - Positive ($f(A) > 0$).
  - Non-decreasing ($f(AZ \geq f(A) + f(Z)$).
  - Polynomially bounded $f(A) \leq \text{poly}(\text{len}(A))$.
  - For any A, any B which is a suffix of A, and any adjacent Z, and for any $\epsilon > 0$ we have

$$\frac{(1-\beta(\epsilon))\|A\|_2 \leq \|B\|_2}{(1-\epsilon)\|AZ\|_2 \leq \|BZ\|_2}$$

Theorem

Given a $\beta(\epsilon)$-smooth function f which has an unbounded-stream approximation algorithm requiring space $s(n, \epsilon)$, we can use smooth histograms to ϵ-approximate f in the sliding window model with space

$$O\left(\beta^{-1} \log n(s(n,\epsilon) + \log n)\right)$$