Data Streams: Algorithms and Applications
by S. Muthukrishnan

Presentation by Ramesh Sridharan and Matthew Johnson, Part 2

Formalism [Sec. 4]

We consider input streams, which represent underlying shorter signals. We will use $a_1, a_2, \ldots, a_t, \ldots$ to represent the input stream, where a_t arrives at time t. This stream describes some underlying signal, $A[i]$ for $i \in [1, N]$ for some dimensionality N, which we would like to query. There are three typical models used:

- **Time Series:** $a_t = A[t]$
- **Cash Register:** $a_t = (j, I_t)$, and $A_t[j] = A_{t-1}[j] + I_t$, where $I_t \geq 0$.
- **Turnstile:** As above, but no restriction on I_t. In the **strict turnstile** model, $A_t[j] \geq 0\ \forall j\ \forall t$.

Basic Mathematical Techniques [Sec. 5] (continued)

Random Projections

Moments estimation

Here, we want to estimate the kth moment of a stream: $F_k = \sum_i A[i]^k$. This is useful in many practical settings, as we will see over the next few weeks. In this section, we focus on F_2.

We consider the random vectors $\mathbf{X}_{ij}[i]$ of length N whose elements are ± 1 and fourwise independent. We also define $X_{ij} = \langle A, \mathbf{X}_{ij} \rangle = \sum_\ell A[\ell]\mathbf{X}_{ij}[\ell]$.

We can show $\mathbb{E}[X_{ij}^2] = F_2$ by considering the square of the sum above, and noting that in expectation, the cross terms between $\mathbf{X}_{ij}$ are 0. We can also show that $\text{var}(X_{ij}^2) \leq 2F_2^2$ using a similar approach for X_{ij}^4, the second moment of the random variable X_{ij}^2.

To obtain an approximation that lies within $(1 \pm \epsilon)F_2$ with probability greater than $(1 - \delta)$, we consider i in the range $\{1, \ldots, \frac{16}{\epsilon^2}\}$, and j in the range $\{1, \ldots, 2\log\frac{1}{\delta}\}$, and look at the average across j, called Y_i. By the Chebyshev inequality, this is bounded by a constant. We then take the median of the Y_is. Unless more than half of the Y_is deviate from F_2 by ϵF_2, the median will be within the desired range. The probability of this error event occurring is given by the Chernoff bound as δ, so with probability $1 - \delta$ we have the desired bounds on our estimate.

Count-min sketch

We often want to keep track of $A[i]$ for all i, but this violates our space constraints. So, instead of maintaining $A[i]$ for all i, we instead maintain a 2-dimensional $d \times w$ array called count, where $w = \lceil \frac{e}{\epsilon} \rceil$ and $d = \lceil \ln\frac{1}{\delta} \rceil$. Associated with the array are d hash functions $h_1, \ldots, h_d : \{1, \ldots, N\} \to \{1, \ldots, w\}$. When we receive an update $a_i = (j, I_i)$, for each hash function h_k, we update $\text{count}[k, h_k(j)]$ to be $\text{count}[k, h_k(j)] + I_i$; that is, each cell maintains the cumulative sum of all updates whose index hashes to that value.

This allows us to efficiently solve the point-estimation problem, i.e. find $A[i]$ for an arbitrary i. Our estimate is

$$\hat{A}[i] = \min_j \text{count}[j, h_j(i)]$$

This is (certainly) bounded from below by $A[i]$ and (with probability at least $1 - \delta$) from above by $A[i] + \epsilon\,||A||_1$.

Note that $\text{count}[j, h_j(i)]$ has not only the I_ks corresponding to index i, but also the I_ks corresponding to any other index that hashes to the same value. So, $\hat{A}[i]$ is bounded from below because of these "extra values." The upper bound comes from applying the Markov inequality to the probability $\mathbb{P}(\hat{A}[i] \geq A[i] + \epsilon\,||A||_1)$. This is equivalent to $\mathbb{P}(\text{count}[j, h_j(i)] \geq A[i] + \epsilon\,||A||_1\ \forall j)$. This is equivalent to the probability that the sum of the "extra values" is less than $\epsilon ||A||_1$. The expectation of this "extra weight" is $||A||_1 / w$, and since they are pairwise independent, we can obtain a bound by multiplying their probabilities. Using the Markov inequality then gives the desired result.

Note that many of the problems expressed in earlier sections can be solved using this technique.

Sampling

Estimating Number of Distinct Elements

The problem is to estimate $D = |\{i|A[i] \neq 0\}|$. If $A[i]$ is the number of occurrences of i in the stream, D is the number of distinct items. More generally, D is the size of the support of $A[i]$.

One way of estimating D in the cash register model keeps a bit vector c of length $\log_2 N$ and uses a hash function $f : [1, N] \rightarrow \{1, 2, \ldots, \log_2 N\}$ such that $\mathbb{P}[f(i) = j] = 2^{-j}$ and any update j to item i sets $c[f(i)]$ to 1. An unbiased estimate of the number of distinct items is given by $2^{k(c)}$, where $k(c)$ is the lowest index j such that $c[j] = 0$. Intuitively, if the probability that any item is mapped into the counter at index j is 2^{-j}, then if there are D distinct items, we expect $D/2$ of them to be mapped to $c[1]$, $D/4$ to be mapped to $c[2]$, etc. However, that relies on the existence of a fully random hash function, and so it has been extended to allow a hash function that can be stored in $O((\frac{1}{\epsilon^2} \log \log m + \log m \log(1/\epsilon)) \log(1/\delta))$. For the turnstile model, the methods for estimating D uses L_p-sum estimation for small p.

Basic Algorithmic Techniques [Sec. 6]

The Algorithmic Techniques section is differentiated from the Mathematical Techniques section in that it focuses on more deterministic settings in which the main innovations are in careful data structure planning.

Estimating wavelet coefficients

In the the time series model, consider the problem of approximating the signal by using the B largest Haar wavelet coefficients (see Figure 1 for a depiction of the Haar wavelets). Because of the time-localization of the Haar wavelets, we can essentially walk along the signal while keeping two data structures: a heap of the B largest coefficients so far, and a list of $\log N$ *straddling* coefficients, i.e. the "in-progress" coefficients. The meaning of the straddling coefficients and the relationship of those structures is best visualized by drawing the Haar wavelets on a binary tree sitting on top of the signal.

Using the above method, we can compute the best B-term approximation to the signal in the Haar wavelet domain in $O(B + \log N)$ space.

Deterministic heavy hitter with sparsity

Although we cannot deterministically solve the heavy hitters problem in the general case, we can if we impose a sparsity constraint over A: we assume that no more than k indices have nonzero values in A, and we want to find those k indices and/or their corresponding values in A.

Take the x consecutive primes $p_1, \ldots, p_x$ larger than k, where $x = (k-1)\log_k N + 1$. For each prime p_j, construct a table T_j of size p_j. In each table, each index i is mapped to $i \mod p_j$. Our update rule for an update (i, I_i) is

$$T_j[i \mod p_j] := T_j[i \mod p_j] + I_i$$

We then claim that each index will have at least one table where it is the only index in its entry. Two indices can share the same entry in at most $\log_k N$ tables. Otherwise, their difference would be divisible by $\log_k N$ primes. However, this would imply that the difference is larger than N (since it is the product of more than $\log_k N$ numbers greater than k). We can repeat this argument for every pairing, requiring $(k-1)\log_k N + 1$ tables. We could estimate $\hat{A}[i]$ by

$$\hat{A}[i] = \frac{1}{x} \sum_{j=1}^{x} T_j[i \mod p_j]$$

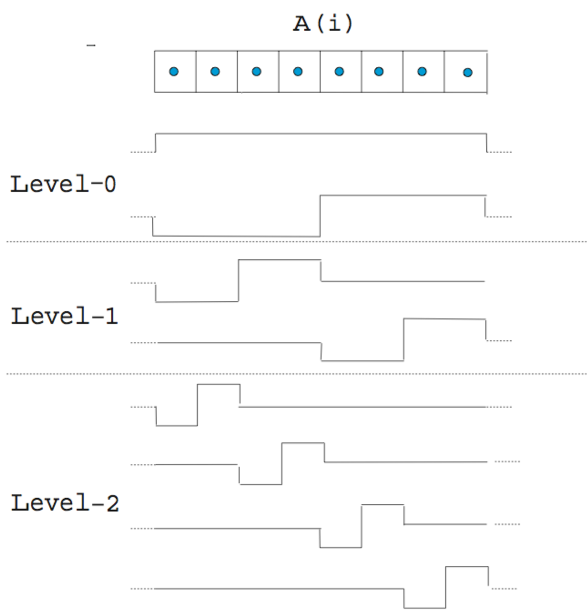

Figure 1: Set of Haar wavelets for a signal of length 8.