CS369G: Algorithmic Techniques for Big Data Spring 2015-2016

Lecture 8: CR-PRECIS and Count Sketch

Prof. Moses Charikar *Scribe: Carson Kent*

1 Overview

Previously, we introduced the Count-Min sketch for computing frequencies of elements in a stream. In this lecture, we will first discuss a derandomized variant on Count-Min and then introduce a related sketch: Count-Sketch. Analysis will show that Count-Sketch produces estimates which are less sensitive to heavy tailed distributions and we will conclude by making connections between Count-Sketch and Count-Min.

2 Derandomizing Count-Min

Recall that, for a given element i, we are interested in estimating it's frequency f_i up to an additive error of ϵK. Abstractly, K is a particular global statistic of the stream and accounts for the fact that the error in estimation ought to be relative to the size and/or the distribution of the elements in the stream. For Count-Min [1], $K = F_1$ and we have the one-sided bound

$$f_i \leq \widetilde{f_i} \leq f_i + \epsilon F_1 \ \ \forall i \in [n]$$

with high probability. The cost of accomplishing this an attractive $O((\log n)/\epsilon)$ space where the $\log n$ factor accounts for the fact that the bound must hold jointly for all i.

It is natural to wonder if, at the expense of slightly greater storage, a more absolute bound can be obtained by derandomizing the Count-Min sketch. Indeed, this is exactly the task accomplished by the CR-PRECIS sketch.

2.1 The CR-PRECIS sketch

The algorithm for constructing the CR-PRECIS [2] sketch only differs from Count-Min sketch by making a deterministic choice of hash functions instead of a stochastic one. Thus, the description of the algorithm is exactly the same up to the selection of hash functions and will therefore be omitted. Instead, we will focus on analysis of the method for choosing the hash functions and how this affects the space guarantees of the algorithm.

2.1.1 Analysis

Consider selecting the hash functions induced by the first t primes $q_1, \ldots, q_t$. That is, let

$$h_j(i) = i \mod q_j$$

for an element $i \in [n]$ of the stream. To show that this selection of hash functions provides a similar space guarantee as the random hash functions used in Count-Min we need to establish two conditions:

1. Collisions between the hashes of elements are infrequent enough that t does not have chosen to be too large.
2. The magnitude of the q_j is sufficiently small such that the range of the h_j does not require too much storage.

For the first condition, consider the following fact

Fact 1. *(Chinese Remainder) If $n_1, \ldots, n_k$ are coprime and $a_1, \ldots, a_2$ are any integers, then the set of equations*

$$\begin{aligned} x &\equiv a_1 \mod n_1 \\ x &\equiv a_2 \mod n_2 \\ &\vdots \\ x &\equiv a_k \mod n_k \end{aligned}$$

has a solution x, which is unique modulo $N = n_1 n_2 \cdots n_k$

This implies that, for $i \neq j \in [n]$, if $h_{j_1}(x), \ldots, h_{j_m}(x)$ hash i and j to the same bucket then

$$i \equiv j \mod q_{j_1} \cdots q_{j_m}$$

The statement cannot be true if $q_{j_1} \cdots q_{j_m} \geq n$. Thus, the trivial bound $q_j \geq 2$ shows that a collision between two elements i and j can occur at most $\log_2 n$ times.

The immediate consequence of this bound is that

$$\sum_{j=1}^{t} C(h_j(i)) \leq t f_i + \log(n) F_1$$

where $C(h_j(i))$ is the count of the jth bin that i is hashed to. Thus,

$$\widetilde{f_i} = \min_j \left(C(h_j(i)) \right) \leq f_i + \frac{\log(n) F_1}{t}$$

Moreover, setting

$$t = \frac{\log n}{\epsilon}$$

we see that we can obtain an ϵF_1 approximation.

For this choice of t, we must now bound how large q_t can be– as $O(q_t)$ buckets must be stored for each hash. It is a celebrated result in number theory that

Fact 2. *(Prime Number Theorem) The number of prime numbers less than n is $\Theta(n/\log n)$.*

This immediately gives the bound $q_t = O(t \log t)$. Multiplying this by the number of hashes, it quickly follows that the total space used by CR-PRECIS is

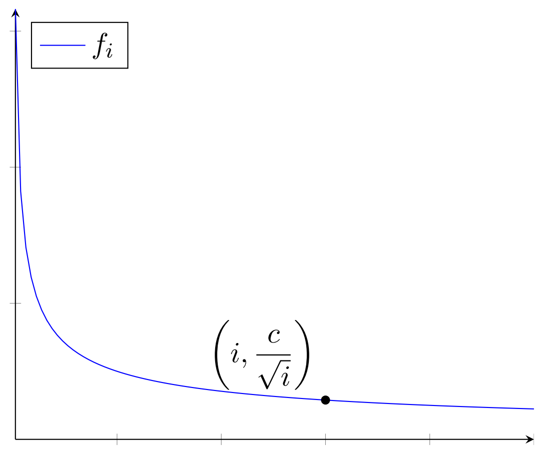
$$O\left(\frac{\log^2 n \log\log n}{\epsilon^2}\right)$$

Hence, at the expense of some polylog factors and an extra factor of ϵ CR-PRECIS avoids random variate generation altogether and provides an absolute guarantee on the error of the estimated frequencies.

Note, it is an open problem whether $o(1/\epsilon^2)$ is a lower bound for a deterministic algorithm.

3 The Count-Sketch Algorithm

For streams where the elements are drawn from a heavy-tailed distribution, the frequency of "heavy hitters" can be less well distiguished from frequencies of the other elements in the stream. Indeed, an example is the Zipfian distribution with $p = 1/2$ (shown below) where the frequency of the ith element is proportional to $1/\sqrt{i}$.



For Count-Min and CR-PRECIS, such a distribution forces one to choose ϵ to be very small in order to make ϵF_1 meaningful.

To help lessen this sensitivity, notice that, for a sequence of elements drawn from the Zipf distribution,

$$F_2 = O\left(\frac{F_1^2 \log n}{n}\right)$$

This suggests that a sketch which provides an error bound relative a suitable function of a higher frequency moment might be less sensitive to heavy-tailed streams. Indeed, the trivial bound that $\sqrt{F_2} \leq F_1$, implies that achieving $\epsilon\sqrt{F_2}$ error yields a sketch which is always less sentitive to heavy-tailed streams than Count-Min or CR-PRECIS.

3.1 Algorithm and Analysis

To construct the Count-Sketch [3], first notice that the error of the Count-Min sketch is fundamentally affected by the degree to which hash collisions occur between higher frequency elements. Thus, to reduce the effect of these collisions, consider maintaining a signed sum of the frequencies of the elements which hash to each bucket.

If the sign of each element in this sum is chosen uniformly from $\{\pm 1\}$, the intuition is that collisions with between high frequency elements will tend to cancel on average– reducing their overall effect on the frequency estimates.

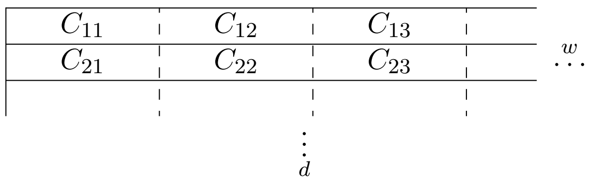
Formally, consider the following:

1. For constants w and d (to be defined later) independently choose $2d$ random hash functions $h_1, \ldots, h_d$ and $s_1, \ldots, s_d$ such that $h_i : [n] \to [w]$ and $s_i : [n] \to \{\pm 1\}$. It is necessary that the hash families from which h_i and s_i are chosen be pairwise independent.
2. For each item q_i in the stream, add $s_j(q_i)$ to the $h_j(q_i)$th bucket of the jth hash.

At the end of this process, one has wd sums (C_{ij}) where

$$C_{ij} = \sum_{h_i(k)=j} s_i(k) f_k$$

A convenient way to visualize these elements is in a table with d rows and w columns:



Now, it is not hard to see that $E(C_{ij}) = 0$ for all i, j– prohibiting use of the estimator employed by Count-Min. Thus, to recover the frequency of the ith element from $C_{h_j(i)j}$, notice that (by the 2-independence of s_j)

$$\mathbb{E}\left(s_j(i) C_{h_j(i)j}\right) = f_i$$

Further,

$$\mathrm{Var}\left(s_j(i) C_{h_j(i)j}\right) \le \mathbb{E}\left(C^2_{h_j(i)j}\right) \le \frac{F_2}{w}$$

by the 2-independence of h_j.

Hence, $\hat{f}_{ij} = s_j(i) C_{h_j(i)j}$ is an unbiased estimator for f_i and, by choosing $w = 3/\epsilon^2$, Chebyshev promises that

$$\mathbb{P}(|\hat{f}_{ij} - f_i| > \epsilon\sqrt{F_2}) \le \frac{1}{3}$$

Setting $d = O(\log n)$ we can use the standard median trick

$$\widetilde{f}_i = \mathrm{median}_j\left(\hat{f}_{ij}\right)$$

so that

$$|\widetilde{f_i} - f_i| \le \epsilon\sqrt{F_2}$$

holds with high probability.

Thus, Count-Sketch requires total space of order

$$wd = O\left(\frac{1}{\epsilon^2}\log n\right)$$

4 Concluding Notes

The table below contrasts Count-Min and Count-Sketch

	Space	Error
Count-Min	$O\left(\frac{1}{\epsilon}\log n\right)$	ϵF_1 (one-sided)
Count-Sketch	$O\left(\frac{1}{\epsilon^2}\log n\right)$	$\epsilon\sqrt{F_2}$ (two-sided)

Ignoring the distinction between one-sided and two-sided error, it is easy to see that Count-Sketch provides an error guarantee which is generally stronger that Count-Min, but at the expense of slightly greater space.

It is also worth noting that the degree to which a hash from a random family tends not to introduce collisions between important (i.e. highly frequent) elements has been exploited well beyond the field of streaming algorithms. Indeed, in machine learning, a very important method of dimensionality reduction called the "hashing trick" relies upon this fact to produce reduced datasets of features that are nearly identical to the sketches constructed by Count-Min.

In the next lecture, we will introduce the problem of sparse recovery and demonstrate an application of Count-Sketch to solving it.

References

[1] G. Cormode, S. Muthukrishnan, An improved data stream summary: the count-min sketch and its applications, *Journal of Algorithms*, 55(1), pp.58–75, 2005.

[2] S. Ganguly, A. Majumder, CR-PRECIS: a deterministic summary structure for update data streams, *ESCAPE'07*, pp.48–59, 2007.

[3] M. Charikar, K. Chen, M. Farach-Colton, Finding frequent items in data streams, *Theoretical Computer Science*, 312(1), pp.3–13, 2004.