Frequency Moments

David Woodruff
IBM Almaden
dpwood@alum.mit.edu

SYNONYMS

L_p Norms, L_p Distances

DEFINITION

Consider a stream (i.e., an ordered list) $\mathcal{S} = a_1, a_2, \ldots, a_n$ of elements $a_i \in [m] \stackrel{\text{def}}{=} \{1, 2, \ldots, m\}$. For $i \in [m]$, its frequency f_i is the number of times it occurs in $\mathcal{S}$. The k-th frequency moment F_k of $\mathcal{S}$, for real $k > 0$, is defined to be $F_k(\mathcal{S}) = \sum_{i \in [m]} f_i^k$. Interpreting 0^0 as 0, we also define F_0 this way, so that it equals the number of distinct elements in $\mathcal{S}$. Observe that $F_1 = n$ is the length of $\mathcal{S}$. In the database community, F_2 is known as the repeat rate or Gini's index of homogeneity. It is also natural to define $F_\infty = \max_{1 \le i \le m} f_i$.

It is usually assumed that n is very large and that algorithms which compute the frequency moments do not have enough storage to keep the entire stream in memory. It is also common to assume that they are only given a constant (usually one) number of passes over the data. It is also assumed that the stream is presented in an arbitrary, possibly worst-case order. This necessitates the use of extremely efficient randomized approximation algorithms. An algorithm A (ϵ, δ)-approximates the kth frequency moment F_k if for any input stream $\mathcal{S}$, $\Pr[|A(\mathcal{S}) - F_k(\mathcal{S})| \le \epsilon F_k(\mathcal{S})] \ge 1 - \delta$, where the probability is over the coin tosses of A. Here, by $A(\mathcal{S})$ we mean that A is presented items in $\mathcal{S}$ one-by-one. Efficiency is measured in terms of the amount of memory and update time of the algorithm.

MAIN TEXT

The frequency moments were introduced by Alon, Matias, and Szegedy [1] in their seminal paper in 1996, and are important statistics for massive databases. Indeed, efficient algorithms for estimating F_0 can be used by query optimizers for finding the number of unique values of an attribute without having to perform an expensive sort on the entire column. Efficient algorithms for F_2 are useful for determining the output size of self-joins and for error estimation in the context of query result sizes and access plan costs. Moreover, F_k for $k \ge 2$ can indicate the amount of skew of a data stream, and this can determine which algorithms to use for data partitioning. These values can also be used to detect denial-of-service attacks. In general, F_k for large k can be used to approximate the most frequent value, potentially more efficiently than computing this value directly.

There is a large body of work on bounding the memory required of F_k-approximation algorithms. In their original work, Alon, Matias, and Szegedy surprisingly showed that F_2 can be (ϵ, δ)-approximated using only $O(\frac{\lg 1/\delta}{\epsilon^2}(\lg n + \lg m))$ bits of memory. It is now known that F_k for $k \le 2$ can be (ϵ, δ)-approximated in 1-pass using $O(\frac{\lg 1/\delta}{\epsilon^2})$ bits of space, up to a polylog nm factor, and there is an almost matching $\Omega(1/\epsilon^2)$ lower bound for 1-pass algorithms. For $k > 2$, a sequence of work showed that F_k can be approximated in $O(m^{1-2/k} \log 1/\delta)$ space, up to a $\text{poly}(\log nm, 1/\epsilon)$ factor, and there is an almost matching $\Omega(m^{1-2/k})$ bound. The memory required for approximating F_∞ is $\Theta(m)$. Note that for $k > 2$ the memory required depends polynomially on m, whereas for $k \le 2$ the dependence is logarithmic. The known algorithms use a clever combination of hashing (with limited independence), sketching, and bucketing ideas. The corresponding lower bounds come from reductions from problems in communication complexity, and draw from tools in combinatorics and information theory.

There are several natural questions about the computational complexity of frequency moments which remain unanswered. For instance, for constant δ, it is unknown if F_k for $0 \le k \le 2$ can be approximated more efficiently if more than one pass (but still a constant number) is allowed. Also, it is unknown how efficiently F_k can be approximated if the stream elements arrive in a random order.

RECOMMENDED READING

[1] N. Alon, Y. Matias, and M. Szegedy, *The Space Complexity of Approximating the Frequency Moments*, Journal of Computer and System Sciences, 58:137-147, 1999.

[2] Z. Bar-Yossef, T. Jayram, R. Kumar, and D. Sivakumar, *An Information Statistics Approach to Data Stream and Communication Complexity*, FOCS, p. 209-218, 2002.

[3] P. Indyk and D. Woodruff, *Optimal Approximations of the Frequency Moments of Data Streams*, STOC, p. 202-208, 2005.