Semester : VII Subject : Big Data Analytics Academic Year: 2024 – 2025

MODULE 4: Mining Data Stream

DATA SAMPLING IN DATA STREAMS

A data stream can be any form of key-value pairs. You can choose sample by picking a random key set of a desired size and take all key value pairs whose key falls into the accepted set regardless of the associated value.

In our example, the search query itself was the key with no associated value. In general, we select our sample by hashing keys only, the associated value is not part of the argument of the hash function. For example

Salary Ranges

- Assume that a data stream elements are tuples with three components - ID for some employee, department that employee works for and the salary of that employee.

  StreamData=tuples(EmpID, Department, Salary)

Bloom filter

Bloom filters enable us to select only those items in the stream that are on some list or a set of items. In general, Bloom filter is used for cases where the number of items on the list or set is so large, that you cannot do a comparison of each stream element with element of the list or check the set membership

Need of Bloom Filter

Web crawler performs many crawling tasks and uses different processors to crawl pages.

The crawler maintains a list of all URLs in the database that it has already found. Its goal is to explore the web pages in each of these URLs to find the additional URLs that are linked to these web page

It assigns these URLs to any of a number of parallel tasks, these tasks stream back the URLs they find in the links they discover on a page.

Working of Bloom filter:

Bloom filter itself is a large array of bits, perhaps several times as many bits as there are possible elements in the stream

The array is manipulated through a collection of hash functions. The number of hash functions can be one, although several hash functions are better. In some situations, even a few dozen hash functions may be a good choice.

Each hash function maps a stream element to one of the positions in the array.

During initialization all the bits of the array are initialized to 0.

Now, when a stream element, say x, arrives, we compute the value of $h_i(x)$ for each hash function h_i that are to be used for Bloom filtering.

A hash function maps a stream element to an index value on Bloom filter array. In case, this index value is 0, then it is changed to 1.

Flajolet-Martin- Algorithm To Count Different Elements In Stream

In stream processing sometimes instead of exact solution, you can accept approximate solutions. One such algorithm, which is used to count different elements in a stream in a single pass was given by Flajolet-Martin

Steps of the algorithm:

- Pick a hash function h that maps each of the *n* elements of the data stream to at least log2(*n*) bits.
- For each stream element *a*, let r(*a*) be the number of trailing 0's in h(*a*).
- Record R = the maximum r(*a*) seen.
- Estimated number of different elements= 2^R.

Example:

Given a good uniform distribution of numbers as shown in Table 1. It has eight different elements in the stream.
Probability that the right-most set bit is at position 0 = ½ At position 1 = 1/2 * 1/2 = 1/4
At position 2 =1/2 * 1/2 *1/2 = 1/8

…

At position n = $1/2^n$

Table 1: An Example of Flajolet-Martin Algorithm

	Number	Binary Representation	Position of the rightmost set bit
Uniform Distribution	0	000	-
	1	001	0
	2	010	1
	3	011	0
	4	100	2
	5	101	0
	6	110	1
	7	111	0

(Assuming that the index value of least significant bit is 0)

It implies that the probability of the right-most set bit drops by a factor of 1/2 with every position from the LSB to the MSB as shown in Figure 4.

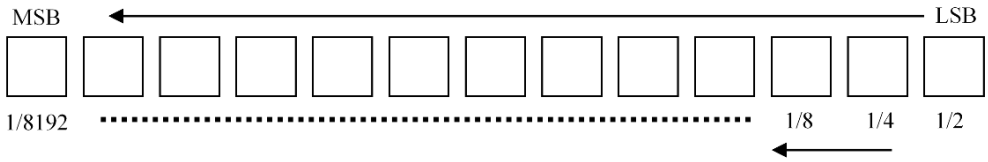

Figure 4: Probability in Flajolet-Martin Algorithm [4]

By keeping the record of these positions of the right-most set bit, say ρ, for each element in the stream. We will expect position of rightmost set bit = 0 to be 0.5, ρ = 1 to be 0.25, etc. Also consider that *m* is the number of distinct elements in the stream.

This probability will come to 0 when bit position *b* is greater than log *m*

- This probability will be non-zero, when *b* <= log *m*

Therefore, if we find the right-most unset bit position *b* such that the probability = 0, we can say that the number of unique elements will approximately be 2^b. This forms the core intuition behind the Flajolet Martin algorithm.