COMPSCI 753

Algorithms for Massive Data

Semester 2, 2020

Tutorial 2: Data stream algorithms

Ninh Pham

1 Uniformly sampling

Suppose we have a stream of tuples with the schema

$$Score(university, courseID, studentID, score)$$

Assume that universities are unique, but courseID is unique only within a university and likewise, studentID is only unique within a university. For example,

UOA, CS752, sID-001, 10
UOA, CS753, sID-001, 9
UOA, CS753, sID-002, 8
AUT, CS752, sID-001, 9
AUT, CS752, sID-002, 7
AUT, CS753, sID-002, 5

Suppose we want to answer certain queries approximately from a 1/20 samples of the data. For each query below, indicate how you would construct the sample, i.e. tell what the key attribute should be and the method for sampling.

1. For each course in a university, estimate the average number of students.
2. Estimate the fraction of students who have a GPA of 7 or more.
3. Estimate the fraction of courses where at least half the students got score above 7.

Solution:

1. We choose key as $\{university, courseID\}$, sampling with probability 1/20. Hence for each university and courseID, we can have 1/20-fraction number of students.
2. We choose key as $\{studentID\}$, using hash function to sample with probability 1/20. Hence 1/20-fraction of students are in our sample set and we can compute their GPA to answer the question.

3. We choose key as $\{courseID\}$, using hash function to sample with probability $1/20$. Hence $1/20$-fraction of courses are in our sample set and we can identify which course has at least half the students got above 7.

2 Bloom filter

Consider the same situation from our lecture with 8 billion bits and 1 billion members of the set S, calculate the false positive rate if we use numbers of hash functions as $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$?

Solution:

k	1	2	3	4	5	6	7	8	9	10
FPR	0.1175	0.0489	0.0306	0.0240	0.0217	**0.0216**	0.0229	0.0255	0.0292	0.0342

3 Bloom filter

Suppose we have n bits of memory available and set S has m members. Instead of using k hash functions each mapping an element to a bit in the main memory, we could divide the n bits into k subarrays (assume n is divisible by k), and then use the i-th hash function, $i \in [1..k]$, to the i-th subarray.

1. As a function of n, m, and k, what is the probability of a false positive?
2. How does it compare with using k hash functions into a single array?

Solution:

We divide n bits of memory into k arrays, n/k is the size of subarrays. Call p as the probability of a false positive of the subarray (i.e. the fraction of 1s in this array). We have $p = 1 - (1 - k/n)^m \approx 1 - e^{-km/n}$.

Since we have to check all k hash function from k subarrays, the probability of false positive for the new solution is $p_1 = \left(1 - e^{-km/n}\right)^k$. This value is identical to the solution of using k hash functions into a single array, i.e. $p_2 = (1 - e^{-km/n})^k$.

4 Misra-Gries algorithm

Run the Misra-Gries algorithm with $k = 3$ for the stream below:

$$\{32, 12, 14, 32, 7, 12, 32, 7, 6, 12, 4\}$$

Solution: The result is: $\{32, 12, 4\}$.

5 CountMin sketch

Applying CountMin sketch to estimate the frequency of each element in the stream below:

$$\{1, 1, 1, 2, 4, 4, 3, 2, 3, 2, 3\}$$

Our CountMin sketch uses $d = 3$ hash functions:

$$h_1(x) = x + 1 \mod 3,$$

$$h_2(x) = 3x + 1 \mod 3,$$

$$h_3(x) = 5x + 2 \mod 3.$$

Solution:

First we compute the positions for each element in stream:

x	1	2	3	4
$h_1(x) = x + 1 \mod 3$	2	0	1	2
$h_2(x) = 3x + 1 \mod 3$	1	1	1	1
$h_3(x) = 5x + 2 \mod 3$	1	0	2	1

Construct the CountMin Sketch:

	0	1	2
h_1	3	3	5
h_2	0	11	0
h_3	3	5	3

Estimate the frequency of each element:

	1	2	3	4
Frequency	5	3	3	5