Asmt 5: Frequent Items

Turn in through Canvas by 2:45pm:
Wednesday, Feb 26
100 points

Overview

In this assignment you will explore finding frequent items in data sets, with emphasis on streaming techniques designed to work at enormous scale. For simplicity you will work on more manageably sized data sets, and simulate the stream by just processing with a for loop.

You will use two data sets for this assignment:

- `http://www.cs.utah.edu/~jeffp/teaching/cs5140/A5/S1.txt`
- `http://www.cs.utah.edu/~jeffp/teaching/cs5140/A5/S2.txt`

The first data set S1 has a set of $m = 3{,}000{,}000$ characters, and the second one S2 has $m = 4{,}000{,}000$ characters. The order of the file represents the order of the stream.

As usual, it is highly recommended that you use LaTeX for this assignment. If you do not, you may lose points if your assignment is difficult to read or hard to follow. Find a sample form in this directory: `http://www.cs.utah.edu/~jeffp/teaching/latex/`

1 Streaming Algorithms

A (40 points): Run the Misra-Gries Algorithm (see **L11.3.1**) with $(k-1) = 9$ counters on streams S1 and S2. Report the output of the counters at the end of the stream. In addition to each counter report the estimated ratio of each label using the estimated count$/m$.

In each stream, use just the counters to report which characters *might* occur at least 20% of the time, and which must occur at least 20% of the time.

For S1 : character and its corresponding counts are

Object	Count	count/m
'a'	736664	0.24555466666666667
'b'	436662	0.145554
'c'	197649	0.065883
'k'	0	0.0
'd'	1	3.3333333333333335e-07
'p'	1	3.3333333333333335e-07
'v'	1	3.3333333333333335e-07
't'	1	3.3333333333333335e-07
'n'	1	3.3333333333333335e-07

For S2 :

Object	Count	count/m
'a'	1885833	0.47145825
'b'	685133	0.17128325
'c'	286358	0.0715895
'f'	1	2.5e-07
'h'	1	2.5e-07
'j'	1	2.5e-07
'o'	1	2.5e-07
'i'	1	2.5e-07
'v'	1	2.5e-07

The values obtained here are for $\hat{f}_j$. Actual frequency of the items is represented by f_j . We have the relation $f_j - m/k \leq \hat{f}_j \leq f_j$. To find how many objects must occur more than 20% of the time, we use the above equation to obtain $f_j \geq \hat{f}_j$. Now, we need elements that must occur more than 20% of the time for S1 i.e.,$(0.2)(3000000) = 600000$ and S2 : $(0.2)(4000000) = 800000$ times. So, the actual counts f_j must be greater than the estimated counts $\hat{f}_j$.

Must occur more than 20% of the time :
S1 : 'a'
S2 : 'a'

Now, the elements that might occur more than 20% of the time is given by the relation,
$f_j - m/k \leq \hat{f}_j$
$f_j \leq \hat{f}_j + m/k$

For S1 : m = 3000000 and k = 10
$f_j \leq \hat{f}_j + 300000$
Might occur more than 20% of the time = b : 436662 (I am not counting 'a' here, as it is already in must occur)
For S2 : m = 4000000 and k = 10
$f_j \leq \hat{f}_j + 400000$
Might occur more than 20% of the time = b : 685133 (I am not counting 'a' here as well, as it is already in must occur)

B (40 points): Build a Count-Min Sketch (see **L12.1.1**) with $k = 10$ counters using $t = 5$ hash functions. Run it on streams S1 and S2.

For both streams, report the estimated counts for characters a, b, and c. In addition to each counter report the estimated ratio of each of these labels using the estimated count/m. Just from the output of the sketch, with probably $1 - \delta = 31/32$ (that is assuming the randomness in the algorithm does not do something bad), which objects *might* occur at least 20% of the time, and which objects *must* occur at least 20% of the time.

	S1	count/m	S2	count/m
a	1019566	0.3398533333333334	2039810	0.5099525
b	599564	0.19985466666666668	799400	0.19985
c	360551	0.12018366666666666	440270	0.1100675

The PAC bound for the Count-Min Sketch for any 'q' is given by, $f_q \leq \hat{f}_q \leq f_q + \epsilon F_1$ *where* $\hat{f}_q \leq f_q + \epsilon F_1$ holds with probability at least $1 - \delta$. $\delta is \frac{1}{2^t}$ *and* $t = 5$ and hence $\delta = \frac{1}{32}$ and $1 - \delta = \frac{31}{32}$. We need to find the probability with 31/32 , which objects occurred more than 20 percent of the time.

For S1: 20% is 600000 and for S2: 20% is 800000.

We know that $k = 2/\epsilon$from this we can obtain$\epsilon = 2/k$.
Here, we have k=10. Thus, $\epsilon = 2/10 = 0.2$

Now, using $\hat{f}_q \leq f_q + \epsilon F_1$
For S1, $F_1 = 300000$
$\hat{f}_q \leq f_q$+600000

For S1, objects which might occur at least 20 percent of the time are 'a','b','c'. and must occur at least 20 percent of the time is 'a'.

For S2, objects which might occur at least 20 percent of the time are 'a','b','c'. and must occur at least 20 percent of the time is 'a'.

C (10 points): How would your implementation of these algorithms need to change (to answer the same questions) if each object of the stream was a "word" seen on Twitter, and the stream contained all tweets concatenated together? Solution:
If each object in the stream was a word seen on Twitter then there would be a few changes in the implementation of the algorithms.

1. To process the word, we need a buffer having the size of the word. The buffer should be able to hold the entire word so that different hash functions can be applied on it.

2. We need to have a limit on the number of characters in the word which is being processed.

3. Hash functions should be modified to hash words instead of characters
Misra Gries algorithm : Map the words instead of characters
Count min sketch algorithm : The word should be passed to the hash function rather than a character.

4. Logic needs to be added to handle cases like individual words - Identify the spaces and punctuation(comma, hyphen, ampersand, question mark etc) and separate words from these.

Input in this case needs certain pre-processing to make it accept only words and remove all the special characters. Then working of the algorithm is pretty much the same.

D (10 points): Describe one advantage of the Count-Min Sketch over the Misra-Gries Algorithm. Min Sketch Algorithm is better than Misra-Gries Algorithm in following cases:
1. Misra-Gries can only return information about limited objects in the stream. Because, it has limited number of labels and counters. Where as Count-Min Sketch stores information about all the objects in the stream. So, whenever queried it can return result of any object in the stream.
2. Count-Min Sketch works in the turnstile model as well. i.e., Instead of

always adding one to the hash location, a constant can be subtracted. This could be used to remove objects. This is helpful when we have to remove objects which we seen already. Misra-Gries, on the other hand, doesn't have these kind of options.

2 BONUS

The exact heavy-hitter problem is as follows: return *all* objects that occur more than 10% of the time. It cannot return any false positives or any false negatives. In the streaming setting, this requires $\Omega(\min\{m, n\})$ space if there are n objects that can occur and the stream is of length m.

A: (1 point) A 2-Pass streaming algorithm is one that is able to read all of the data in-order exactly twice, but still only has limited memory. Describe a small space algorithm to solve the exact heavy hitter problem, i.e., with $\epsilon = 0$, (say for $\phi = 10\%$ threshold).
Solution :
Let us recall the majority element problem where we have a single counter to find if there is an element that occurred more than 50% of the time. So, to determine if an element occurred more than n/2 times where n is the total length of stream, we need one counter. Formally, we will need k 1 counter to find the elements that occurred n/k times. So, here we are asked for an algorithm to find elements that occur more than 10%of the time, which is equivalent to the elements that occur more than n/10 times where n is the total length of the stream. From the above deduction, k = 10 and so we need 9 counters. Now, run the Misra-Gries algorithm with 9 counters. Whenever there is an element that occurs more than n/2 times, the majority element algorithm will always return this element as output. Similarly, when we run the misra-gries algorithm with 9 counters all those elements that will appear at least 10% of the time will appear in the output counters. But, there is a possibility that the objects returned by the Misra Gries could also have elements which not appeared 10% of time. Hence, we need the second pass. In the second pass, keep the count of all the objects that appeared in the 9 counters returned by Misra-Gries algorithm. This gives how many times each object returned by Misra-Gries appears in the stream. Finally, see if the count of these elements is greater than 10% of the total stream size. Return those elements that passed the 10% check.
To conclude, Misra-gries promise that all the objects that appeared more than 10% are captured in its 9 counters. And the second pass ensures that

there are no false-positives in the returned results. Thus we guaranteed that with two steps, we could accurately find the heavy hitters that is $\epsilon = 0$.