2020/2021

Advanced Algorithms

Most Frequent Items Count

Hypothesis A1 - Lowercase strings

Diogo Andrade 89265

5 February 2021

Index

Problem

It is intended to determine the most frequent items in a data set, exploring methods that allow processing large data sets.

The goal of this work is to analyze the behavior of Count-min Sketch when any of its parameters are changed and answer the next question: “What is the influence of these changes on the results of computational tests?”.

With Count-min Sketch, taking into account the script it is supposed to use a fixed number of hashing functions, so only the size of the hash tables will vary.

To solve this problem, I developed two classes, one to generate data streams and another to counter, using Count-min Sketch strategy.

Execution explanation

Usage:

main.py -s SIZE -sh SIZE_HASH [-nh NUMBER_HASH]

optional arguments:

-s	SIZE	Size of data stream
-sh	SIZE_HASH	Size of hash tables
-nh	NUMBER_HASH	Number of hash tables (default=5)

Execution:

python3 main.py -s 100 -sh 60

Result: (using the file r_datastream_100.txt)

```
letter: w -     9 -    14 - 9.27%
letter: u -     4 -    14 - 9.27%
letter: z -     1 -    14 - 9.27%
letter: v -     5 -     9 - 5.96%
letter: p -     4 -     9 - 5.96%
letter: t -     6 -     8 - 5.30%
letter: l -     2 -     8 - 5.30%
letter: k -     8 -     8 - 5.30%
letter: a -     6 -     6 - 3.97%
letter: x -     5 -     6 - 3.97%
letter: e -     6 -     6 - 3.97%
...
Expected:   100
Counted:   151
```

Generator Class

This class will write a simulated data stream to a file, this data will be lowercase letters, separated by space. To generate this data, a fixed string with letters of different probabilities is used, and these letters will be chosen randomly.

The fixed string has size 86 and this letter count:

- ➔ s, t, u, v, w, x - 5
- ➔ g, h, i, j, k, l - 4
- ➔ a, b, c, d, e, f - 3
- ➔ m, n, o, p, q, r - 2
- ➔ y, z - 1

The execution of this class will produce an output text file, containing the prefix 'datastream_' and the size of the data stream in the file name.

Count-min Sketch Class

This class uses the Count-min Sketch strategy to count the frequency of the letters in the data stream.

This class have two methods:

- update: called whenever a new item is added;
- estimate: who will try to find the guest to the value count of a item;

As previously mentioned, for testing purposes only the size of the hash tables was varied and the number of hash tables being kept constant.

Test Results

- **Data stream size:** 100
  - **Size of hash tables with correct count:** 73, 109, 125, 128, 131, 146, 148, 152, 157, 161, 163, 172, 189, 193, 199
- **Data stream size:** 1000, 10000, 100000, 1000000, 10000000
  - **Size of hash tables with correct count:** 125, 148, 152, 157, 161, 189, 199

Correct count

- **Size of hash table:** 125
  - **Data stream size:** 100
    - **Letters that occur more than 5%:** w, k, a, t, e, c, j
  - **Data stream size:** 1000
    - **Letters that occur more than 5%:** s, x, u, v, l, t, g
  - **Data stream size:** 10000, 100000, 1000000, 10000000
    - **Letters that occur more than 5%:** x, t, u, v, s, w

Incorrect count

- **Size of hash table:** 86
  - **Data stream size:** 100
    - **Letters that occur more than 5%:** k, a, t, e, c, j
    - **Letters that occur more than 10%:** w, d
  - **Data stream size:** 1000
    - **Letters that occur more than 5%:** u, m, d, w, s, x
  - **Data stream size:** 10000
    - **Letters that occur more than 5%:** d, w, u, m, t
  - **Data stream size:** 100000, 1000000, 10000000
    - **Letters that occur more than 5%:** w, d, u, m

Note:

These tests were done with the data stream files sent with this report.

Conclusion

In the tests that were done, the value of the size of the hash table was varied, thus allowing to see for a view for 5 hash tables the size that is necessary to obtain a correct count. In the tests, hash table sizes between 1 and 200 were tested.

From the results of the tests presented above, we can see that starting with a data stream with size 1000, the size of the hash tables for a correct count are always the same.

Using sizes of hash tables that will produce a correct count of the items or that they will not, you will get more or less similar and expected results.

As exemplified in the tests above, only with a data stream size of 100 and a hash table size that will produce an incorrect value in the counts, letters are found that occur more than 10% of the time. As explained above the probabilities of the generator letters, none reaches a value close to 10%, that is, with a reduced size data stream and a wrong value for the size of the hash table, a significant error in the counting will occur, however having considering that the goal is to find the letters more often this will not be a problem.

In conclusion, answering to the influence of changing the size of the hash table on the results of computational tests, a small variation of this value is enough for this count-min sketch strategy not to count the items with the greatest accuracy.