INP7079233 - BIG DATA COMPUTING (A-B) 2024-2025

Assignment of Homework 3 (deadline: June 15, 23.59)

In this homework, you will use the Spark Streaming API to devise a program which processes a stream of items and compares the effectiveness of the count-min and count sketches to estimate the individual frequencies of heavy hitters.

Spark streaming setting that will be used for the homework

For the homework, we created a server which generates a continuous stream of **integer items**. The server has been already activated on the machine **algo.dei.unipd.it** and emits the items (viewed as strings) on specific **ports (from 8886 to 8889)**. Your program must first define a **Spark Streaming Context sc** that provides access to the stream through the method **socketTextStream** which transforms the input stream, coming from the specified machine and port number, into a *Discretized Stream* (**DStream**) of **batches of items**. A batch consists of the items arrived during a time interval whose duration is specified at the creation of the context sc. **Each batch is viewed as an RDD of strings**, and a set of RDD methods are available to process it. A method **foreachRDD** is then invoked to process the batches one after the other. Typically, the processing of a batch entails the update of some data structures stored in the driver's local space (i.e., its working memory) which are needed to perform the required analysis. The beginning/end of the stream processing will be set by invoking **start/stop** methods from the context sc. Typically, the stop command is invoked after the desired number of items is processed.

The ports from 8886 to 8889 of **algo.dei.unipd.it** generate four streams of 32-bit integers:

- 8887: it generates a stream where a few elements are very frequent, while all the remaining are randomly selected in the 32-bit integer domain.
- 8889: it generates a stream where a few elements are very frequent, some elements are moderately frequent, and all the remaining are randomly selected in the 32-bit integer domain.
- 8886: it is the "deterministic" version of the stream 8887, meaning that it generates the exact same stream every time you connect to this port. It should be used to test your algorithm.
- 8888: it is the "deterministic" version of the stream 8889, meaning that it generates the exact same stream every time you connect to this port. It should be used to test your algorithm.

To learn more about Spark Streaming you may refer to the official Spark site. Relevant links are:

- Spark Streaming Programming Guide (full documentation)
- Transformations on Streams (list of transformations applicable to the RDDs in a DStream)

Example program

In order to see a concrete application of the above setting you can download and run the following **example program** which takes as input the port number (*port*) and a target number of elements (*threshold*) to be processed, and computes the exact number of distinct elements among those included in the smallest prefix of received batches whose aggregate size is $\geq$ *threshold, co*ming from *port* (the port number given in input):

- (**Java version**) DistinctItemsExample.java
- (**Python version**) DistinctItemsExample.py

Run the programs on any of the above ports and try threshold values 200000, 500000, and 1000000. Ignore messages regarding the fact that Spark Streaming is deprecated (i.e., no longer developed). When executing your programs, if you receive an error message such as *ERROR ReceiverTracker: Deregistered receiver for stream 0: Stopped by driver* do not worry. This is triggered by the stop signal and is due to an unclean management of the signal. However, *it has no consequence on the correctness of the execution.*

We strongly encourage to use this program as a template for your homework. The program must be run in local mode on your PC, exactly as the one devised for Homework 1. Note that the master must be set to local[*].

Hash functions

The homework requires you to generate a number of hash functions. To this purpose, we ask you to use the following family of hash functions, which map arbitrary integers (i.e., the items) into integers in the range $[0, C-1]$, for some suitably-defined value C. In particular, one such function h maps an item x into the value $((ax+b) \bmod p) \bmod C$, where $p = 8191$, a is a random integer in $[1, p-1]$ and b is a random integer in $[0, p-1]$. **Important:** note that a **cannot be 0**. In order to define a hash function from this family, for a given C, you must simply generate the two random values $a \in [1, p-1]$ and $b \in [0, p-1]$.

Task for HW3.

You must write a program **GxxHW3.java** (for Java users) or **GxxHW3.py** (for Python users), where xx is your 2-digit group number (e.g., 04 or 45), which receives in input the following **5 command-line arguments (in the given order):**

- **An integer** *portExp*: the port number
- **An integer** T: the target number of items to process
- **An integer** D: the number of rows of each sketch
- **An integer** W: the number of columns of each sketch
- **An integer** K: the number of top frequent items of interest

Your program must read the stream generated from machine **algo.dei.unipd.it** at port *portExp* and process the smallest prefix of received batches whose aggregate size is $\geq T$. Let Σ denote the stream of items contained in these batches (hence $|\Sigma| \geq T$). Your program must compute the following information:

- The exact frequencies of all distinct items of Σ

- A $D \times W$ **count-min sketch CM** for Σ, which was seen in class. Recall that for this sketch you need D hash functions h_j, with $0 \leq j < D$, which you can still generate randomly from the family described above.
- A $D \times W$ **count sketch CS** for Σ, which was seen in class. Recall that for this sketch you need $2D$ hash functions, h_j and g_j, with $0 \leq j < D$, which you can still generate randomly from the family described above.
- The average relative error of the **frequency estimates provided by CM for the top-K heavy hitters.** The top-K heavy hitters are defined as the items of $u \in \Sigma$ whose true frequency is $f_u \geq \phi(K)$, where $\phi(K)$ is the K-th value in the list of frequencies of the items of Σ, sorted in non-increasing order. E.g., if the true frequencies are 11, 10, 10, 9, 9, 9, 8, 8, 6 6, and $K = 5$, then $\phi(K) = 9$. **Important:** note that the number of top-K heavy hitters might be larger than K, which is a typical scenario for top-K queries. Recall that if $\tilde{f}_u$ is the estimated frequency for u, the relative error of this estimate is $|f_u - \tilde{f}_u|/f_u$.
- The average relative error of the **frequency estimates provided by CS for the top-K heavy hitters**, were the top-K heavy hitters are defined exactly as in the previous point.

The program should print:

- The input parameters provided as command-line arguments
- The number of processed items, that is, $|\Sigma|$
- The number of distinct items in Σ
- The average relative error of the frequency estimates provided by CM for the top-K heavy hitters.
- The average relative error of the frequency estimates provided by CS for the top-K heavy hitters.
- (Only if $K \leq 10$) True and estimated frequencies of the top-K heavy hitters. For the estimated frequencies consider only those of CM.

THIS FILE (TO BE ADDED) shows how to format your output. Make sure that your program complies with this format.

The program that you submit should run without requiring additional files. Test your program on your local or virtual machine using various configurations of parameters, and **fill the table given in this word form (TO BE ADDED), reporting the required results.**

SUBMISSION INSTRUCTIONS. Each group must submit a zipped folder GxxHW3.zip, where xx is your group number. The folder must contain the program (GxxHW3.java or GxxHW3.py) and the word form GxxHW3table.docx (suitably renaming xx with your group number) with the aforementioned table. Only one student per group must do the submission using the link provided in the Homework 3 section. Make sure that your code is free from compiling/run-time errors and that you comply with the specification, otherwise your grade will be penalized.

If you have questions about the assignment, contact the teaching assistants (TAs) by email to bdc-course@dei.unipd.it . The subject of the email must be "HW3 - Group xx", where xx is your group number. If needed, a zoom meeting between the TAs and the group will be organized.

Ultime modifiche: venerdì, 30 maggio 2025, 17:18

◄ Output example for dataset artificial1M7D100K.txt (16 executors)

Vai a...

Submission form for HW3 ►