Université Jean Monnet – Università degli Studi di Padova
Linear counting lab: analysing the log file of a private network.

Christophe Gravier

Lab objectives:

- Implementation of a probabilistic data structure,
- Focusing given to linear counting,
- Application in the field of computer networks.

1 Functional Objectives

Note: You do not have to answer specific questions in this section but to carefully read the details provded to you.

We want to analyze the log file of the IP traffic on the network 192.168.1.0/24. The log file of the IP traffic on this network is provided online[1]. Its format is as follows:

```
source-ip - dest-ip - timestamp - content (fixed-size random bytes)
```

We are interested in a specific analysis of the log file that deals with cardinality estimations. The aim is to estimate the cardinality of the unique source IP addresses for each destination address, for each time slot given in hour.

For example, given the IP address 192.168.1.32 and the timestamp 1397834022 (which corresponds to 16 April 2014 15:13:42 GMT), we want to be able to estimate the number of unique source adresses that, within the time frame of one hour the timestamp 1397834022 belongs to (16 April 2014 15:00:00 GMT to 16 April 2014 15:59:59 GMT), have hit the destination address 192.168.1.32.

We will consider all IPs in the 192.168.1.0/24 subnet (network address excluded), and all time frames of 1 hours between 25th april 2014 to 1st may 2014 (this makes $(24 hours \times 7 days = 168$ time slots of 1 hour in this period). We therefore need one cardinality estimator for each pair of (dest-address, time-slot) that will provide the estimation of associated unique source address. There are 168 *times slots* $\times$ 255 IPs *addresses*, which means 42,840 required estimators.

We choose in this lab to have each estimator to be a linear counter. We will store the estimators in a two-dimensional array called `estimators`, which is made of 255 lines and 169 columns (this makes 42,840 cells). Each line denotes a destination IP address starting from 192.168.1.1 and up to 192.168.1.255. Each column denotes a time slot, starting from time slot April 25th 00:00:00 / April 25th 00:59:59 and ending with May 1st 23:00:00 / May 1st 23:59:59.

Under this design choice, `estimators[0][0]` for example is the estimator for the number of unique source IP addresses that were hiting the adress 192.168.0.1 on April 25th, between 00:00:00 and 00:59:59.

1. http://datasets-satin.telecom-st-etienne.fr/cgravier/lectures/hpc/labs/linearcounting/

Other example :

– `estimators[13][100]` denotes the estimator for the number of unique source IP addresses that had hit the address 192.168.0.14 on April 29th, between 03:00:00 and 03:59:59 (the 100th time slot starting at April 25th 00:00:00, if I am correct).

In the following, we will build such as system, step by step.

2 Determine the linear counter size

We will focus in this section on divising the required number of bits for each linear counters involved in the program.

Given the following assumptions :

– The traffic is uniformily distributed among all nodes,

– The biais of the estimators we want to build that is set to 0.01 for our application.

Answer the following questions :

1. Are all linear counters of the same size?
2. What is the size (required number of bits) of each linear counter?
3. What is the size in KiloBytes required for all linear counters?

I provide the following hints :

– *The input file to analyze which consists of 1,000,000 lines,*

– *In the lecture, we mention that a load factor of 10 means memory consumption is about 0.1% bit per unique values.*

– *You can use Fig 1 that was provided in the lecture slides.*

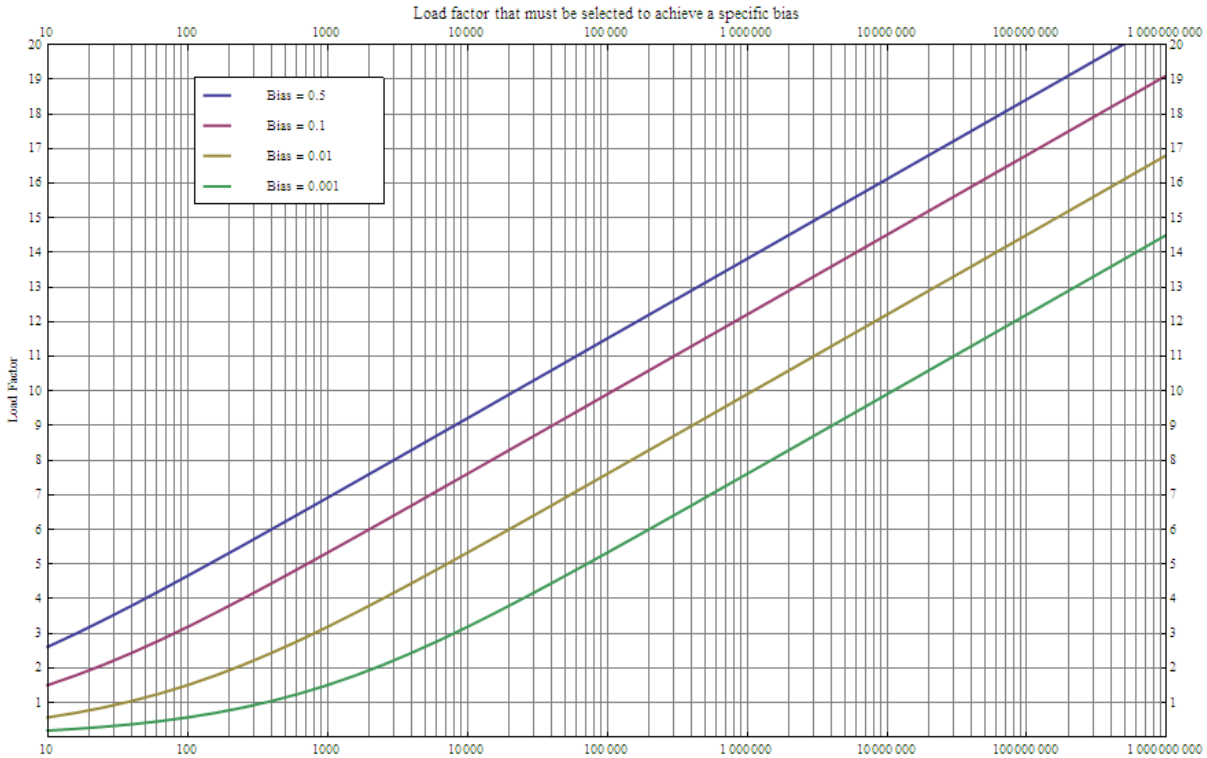

FIG. 1 – *Load factor for different bias values.*

3 Understand the code that is provided

Note: I have already started a Java project with the required code that provides several features we want to develop. In Section 4, you will be asked to find out the missing pieces, but in the current section you will only have to answer questions that are made to make you understand the code that I am delivering to you.

Most of the software is already developped and available at : https://github.com/cgravier/lab-linearcounting. You can use `git` if you are familiar with it, or in the other case you can download the entire project as a `.zip` archive by visiting https://github.com/cgravier/lab-linearcounting/archive/master.zip.

You can safely avoid looking at classes that are into the package : `fr.ujm.tse.datastructures.proba.linearcouting.generator`.

This is the code I used to generate the traffic file that I provide to you. I leave it there in case you are curious on its construction, but you do not need to understand the classes in this package to achieve total success in the lab.

Once you have performed a `git clone` or more straightforwardly downloaded the project archive, you are asked to perform and answer the following :

1. Open the project in your favourite Java IDE. I used Eclipse for this project but any you are familair with will do.
2. What is the Class `Packet` ? What are the difference between the methods `hashPacket()` and `hashCode()` in this class ?
3. What is the class `LinearCounter` ? What does the method `public void add(Packet packet)` do ?
4. What is the class `Estimators` ? What does the method `public void reccordPacket(Packet packet)` do ? Do you have now a more precise idea on what the methods `public int getIpRange()` and `public int getTimestampIndex()` do in the class `Packet` ?

4 Get your hands dirty!

Now is your turn to code and compelte this program !

You are asked to complete the program so that you can answer some basic cardinality estimation queries. For this, I advise you to follow these steps :

1. Complete the `public void readLogFile()` method in the `Analyzer` class (this is the main work) :
   (a) Initialize an empty estimators table (instanciate `Estimators`)
   (b) Read the log file line per line
   (c) For each line, parse the line and create a new `Packet` instance
   (d) Add the newly Packet its associated linear counter using the `public void reccordPacket (Packet packet)` from the `Analyzer` class.
2. Add a method `int getCardinality()` in the class `LinearCounter` that we will provide the actuall estimation of the LinearCounter (remember: the estimator is **not** the weight of the linear counter !). The formula is in the lecture slides ☺)
3. Use the 2D array of linear counters created at 1.a in order to answer the following queries :
   (a) How many unique source addresses have hit the address 192.168.1.1 in the first hour of April 25th 2014? (you can count the real number in the log file, but you will have to be ingenious ☺).

(b) What were the unique source addresses for the entire day of 26th of April that were hitting 192.168.0.2? That would be the sum of the estimators at the second line of the 2D array, for columns between columns index 23 and 46 (both included).

(c) What were the unique source addresses that were hitting 192.168.0.42 for the first hour of each day in the analyzed perdiod? That would be the sum of the estimators on line 41, for columns index 0, 23, 46, ... (actually all index columns j so that $j \ mod \ 23 = 0$).