Lab 8: Bloom Filter

Zifeng Jiang

4 Dec 2019

Introduction

This lab is designed to give you practice working with Bloom filter. This is an individual assignment; you may not share code with other students.

Specification

Over view

Your task is to implement Bloom filter. You need to determine the number of hash functions **k** (we use one hash function with different seeds to simulate different hash functions) and the size of bit array **m**, which make the probability of false positives less than **0.0005**. $\mathbf{2}^{20}$ random strings will be added to your Bloom filter during the testing.

Preliminary

Download the zipped project archive — Lab 8.zip, and import it into your preferred IDE as an existing project. Open the `bloomfilter` package and run the `TestBloomFilter` class (For Eclipse users, right click on the file in the package explorer view, select `Run As` > `JUnit Test`. If Select Preferred Launcher dialog shows up, check "Use configuration specific settings" and select "Eclipse JUnit Launcher").

Note that the `testBloomFilter` should fail — you will fix this later.

Your Job

You need to complete the following methods in the `BloomFilter` class: `add` method: to add an element to the Bloomfilter. `query` method: to query whether the Bloomfilter contains the element.

`hash` method: to hash a string to a integer.

Testing the False Positives of Your Bloom Filter

The `TestBloomFilter` class tests your `BloomFilter` class. If you have successfully implemented the `BloomFilter` class and chosen appropriate **k** and **m**, this class will run without errors.

Submission

Raise your hand to call for the teaching assistant and demonstrate your results.

Create a zip file named YourStudentID-Name.zip that contains your code project and upload your zip file to the FTP server.

The text book and some web pages on the Internet may contain Java code for Bloom Filter implementation. Please write your own code.

Deadline

8 Dec 2019 23:59 GMT+08:00

Appendix

Adding JUnit 4 Library with Eclipse

After importing the zipped project into Eclipse, JUnit 4 library should be automatically configured for your project. If not, right click on your project in project explorer view, select `Properties` > `Java Build Path` > `Libraries` > `Add Library` > `JUnit`, and choose JUnit 4.

Adding JUnit 4 Library with IntelliJ IDEA

After importing the project into IntelliJ IDEA, JUnit 4 library should be automatically configured for your project. If not, try the following steps:

1. In the editor, place the cursor within the line containing the class declaration.
2. Press `Alt + Enter` to view the available intention actions. Select `Create Test`.
3. In the `Create Test` dialog, to the right of the text informing you that the JUnit 4 library is not found, click `Fix`.