Institut for Matematik og Datalogi
Syddansk Universitet

Introduction to Computer Science
E14 – Study Group – Week 44

The purpose of this exercise is to consider the birthday paradox in connection with hash functions. It is a programming exercise, so bring your laptops. Form groups of around four persons, and divide up the work described below within your group, as a team of programmers. For instance, form two subgroups, with one working on task 1.b, while another starts on the rest.

Use your CPR numbers (or parts of your CPR numbers) as data which should be hashed into a table (array/list) of length m, indexed from 0 to $m-1$. (When you need more data, make some up.)

1. Using Python, write two functions hashing your data into the table using the following two methods (or more if you feel like it):

   (a) The method described in lecture and in the textbook (page 425), where the data value modulo m is the hash function value. (Modulo is % in Python.)

   (b) SHA-1, which is a standard hash function used for cryptographic purposes and believed to be very good. (SHA-1 is described on Wikipedia.) There is a hash library in Python. See

   https://docs.python.org/2/library/hashlib.html

   The first example provided there uses MD5, but you can switch that to SHA-1 by changing "md5" to "sha1". The second example is closer to what you will want to use. Note that after you have the hexdigest of your hash result, you have something in hexadecimal. You can change a hexadecimal value, for example "FF", to an integer value, using `int(0xff,16)`. SHA-1 has a fixed length output, so to get the values in the range zero to $m-1$, you have to take the output modulo m.

2. Test your hash functions. Create an array of length m which will count how many random large integers are hashed to each of the distinct

values between zero and $m - 1$. Look at the results for some relatively small values of m (for example, $m = 4, 8, 12, 20, 50, 100$).

3. With at least four different values of m, check how many values you need to hash before you get a collision. Do both functions behave as one would predict with the Birthday Paradox? For prediction, use the method from the slides and the textbook (page 427). Is there any difference between the two hash functions?

4. With at least four different values of m, check how even your distribution is. What is the difference between the largest number of strings hashing to any particular value and the smallest number? How does this seem to depend on the number of strings you hash? Is there any difference between the two hash functions?