Birla Institute of Technology & Science, Pilani
2nd Semester 2016-17 - CS F211 – Data Structures and Algorithms

Lab 8 – 18th March
Topics – Bit Vectors and Bloom Filters

Problem 3

Design and implement a bloom filter using a bit vector as follows:

- A Bloom filter B is a bit vector of size *N*, and *d* hash functions, each computing a value mod *N*.
- The bit vector is initialized to all 0s.
- To insert an element e into B: compute addresses h_1(e.key), h_2(e.key) ... h_d(e.key) and set each of those locations (in the bit vector) to 1.
- To find an element e in B: compute addresses h_1(e.key), h_2(e.key) ... h_d(e.key) and get the bit values in those locations. And if any of those values is 0, then e is not found; else e is present. The latter may be incorrect i.e. false positives may occur because bits were set by other insertions (i.e. collision).
- A bloom filter is associated with a false positive rate (FP). FP is computed by the following formula:

$$FP = \left(1 - e^{-d*\frac{m}{N}}\right)^{d}$$

Here *N* is the size of the bit vector / hash table. "*m*" stands for total number of elements inserted into the bloom filter until now. *"d"* stands for number of hash functions used. The value of e is approximately 2.718.

You can use the following hash functions for implementing your bloom filter. Every hash function is of the form: $|A * e.key + B| \ mod \ N$. The values of A and B will vary for various hash functions as given by the table below:

Hash function 1:	A = 7, B = 11
Hash function 2:	A = 11, B = 13
Hash function 3:	A = 13, B = 17
Hash function 4:	A = 17, B = 19
Hash function 5:	A = 19, B = 23

Implement a bloom filter using a bit vector and implement the respective insert and find operations. You should implement the bloom filter using both variants of bit vectors we have seen before. You can refer to the following table for designing your functions.

Key	Function	Input Format	Description
0	CreateBloomFilter	0 X N	If X = 1, then create an empty bloom filter B along with a bit vector S in the form of integer word (N = 32 in this case); If X = 2, then create an empty bloom filter B along a bit vector S in the form of integer array of size N. You may have additional fields in your bloom filter structure which you may have to initialize.
1	Insert	1 V	Inserts a given value V into the bloom filter as explained above.
2	Find	2 V	Finds whether a given value V is present in the bloom filter or not. You should print 1 if V is present, 0 otherwise.
3	Clear	3	Clear the bloom filter and restore it to the default values. You shall have to clear all the variables related to the bloom filter and also the bit vector.
4	Experiment	4 K L 1 a_1 2 a_2 1 a_3 … 2 a_K	Next K lines, will contain calls to mix of insert and find functions (one function call in each line). After processing each line, print the value of false positive rate. Value of d indicates the number of hash functions you must be using. If L=2, then you must use the first two hash functions only to perform insert and find operations. L will be in the range: 1 to 5. You can refer to Sample input and output – 3 for clarity.

Sample input and output - 1

Sample Input	Sample Output

0 1 32
1 2
1 6
1 9
2 6
2 9
2 31
-1
	1
	1
	0

Sample input and output - 2

Sample Input	Sample Output

0 2 100
1 23
1 62
1 91
2 62
2 91
2 34
-1
	1
	1
	0

Sample input and output - 3

Sample Input	Sample Output

0 2 100
4 6 3
1 23
1 62
1 91
2 62
2 91
2 34
3
0 2 50
4 4 5
1 49
2 23
1 23
2 23
-1
	<FP1>
	<FP2>
	<FP3>
	1 <FP4>
	1 <FP5>
	0 <FP6>
	<FP7>
	0 <FP8>
	<FP9>
	1 <FP10>