Finding Correlated Pairs

Sam McCauley

October 24, 2017

1 Input

The input file gives a list of 256-bit vectors, one per line. Each vector is divided into 4 64-bit parts; each part is stores as a signed 64-bit integer. For example, the line of the file

```
2948679682100370091 -6730236453359443949 9154238340659291137 2505149300205180166
```

gives the bit vector (here I am breaking into 4 lines for readability; the bits give a single total vector):

```
0010100011101011110100001000110001010000000000010001101010101011
1010001010011001011001011001101010011111001000000010000000010011
0111111100001010011000110100010100000100010000100010010000000001
0010001011000100000101000000010000000100000011000101010100000110
```

The executable is given three command line arguments. One example would be

```
100vectors.txt 4 100
```

The first argument is the file being read. The second argument is the number of signed longs in a given bit vector. (We need 4 sets of 64 bits to make a 256-bit vector.) The final argument is the number of vectors given in the file.

The input will always have this format; only the constant 100 will change. Your program is not required to handle all of these as variables—for example, you may hardcode that each bit vector consists of 4 longs, or you may use the name of the file to determine the number of vectors.

1.1 How Vectors Were Generated

Each bit of every vector is a `1` with probability 1/3, and a `0` with probability 2/3. These choices are made independently.

The correlated vector (we'll call it v_c for simplicity) is *Pearson-correlated* with another vector v_1. For each digit in v_c, with probability 7/8 we set it equal to the corresponding digit in v_1. With probability 1/8, the digit is set randomly—`1` with probability 1/3, and `0` with probability 2/3. The Pearson-correlated vectors are much more likely to have a large number of `1`s in common than two random vectors.

2 Correlation

We define the *similarity* of two vectors to be the number of `1`s they have in common. We also call them *correlated*.

We guarantee that in each input instance, there is a single pair with similarity at least 65.

As an example, the following two vectors have similarity 93. If we find these two inputs in a file, we can safely output them as the correct answer.

```
-4545410128706124432 423485083363186329 28290711320299080 -1687296419800652167
-4256053852647568032 428569225130022555 28290711253190152 -1687296419734591911
```

3 Expected Output

The expected output on CodeJudge is the two indices of the correlated vectors, with the lower index first.