22BIO201 Intelligence of Biological Systems 1

Lab Sheet 3

Contents

Problem 1: Generate a Random DNA Sequence

Description: Create a random DNA string with letters from the whole alphabet A, C, G, and T. First make a list of random letters and then join all those letters to a string. Also write another function to count the number of bases in the random sequence and measure the CPU time for large such DNA strings. (Hint : use import random, import time)

Reference

Illustrating Python via Bioinformatics Examples, Hans Petter Langtangen, Geir Kjetil Sandve, https://hplgit.github.io/bioinf-py/doc/pub/html/main_bioinf.html

Problem 2: Compute the Hamming Distance Between Two Strings

We say that position i in k-mers $p_1 \ldots p_k$ and $q_1 \ldots q_k$ is a mismatch if $p_i \neq q_i$. For example, CGAAT and CGGAC have two mismatches. The number of mismatches between strings p and q is called the Hamming distance between these strings and is denoted *HammingDistance*(p, q).

Hamming Distance Problem

Compute the Hamming distance between two DNA strings.

Given: Two DNA strings.

Return: An integer value representing the Hamming distance.

Sample Dataset

GGGCCGTTGGT

GGACCGTTGAC

Sample Output

3

Visit http://rosalind.info/problems/ba1G/ . Solve the problem. Use the sample dataset given in the site.

Problem 3: Find Patterns Forming Clumps in a String

Given integers *L* and *t*, a string *Pattern* forms an (*L*, *t*)-clump inside a (larger) string *Genome* if there is an interval of *Genome* of length *L* in which *Pattern* appears at least *t* times.

For example, TGCA forms a (25,3)-clump in the following *Genome*: gatcagcataagggtcccTGCAATGCATGACAAGCCTGCAgttgttttac

Clump Finding Problem

Find patterns forming clumps in a string.

Given: A string *Genome*, and integers *k*, *L*, and *t*.

Return: All distinct *k*-mers forming (*L*, *t*)-clumps in *Genome*.

Pseudocode:

```
ClumpFinding(Genome,k,L,t)
for i←0 to|Genome|-L
count←0 for all kmers in Genome(i,L)
  for j←0 to L- k
     kmer=Genome(i+j,L)
    count(kmer)=count(kmer)+1
  for all kmers in count
   if count(kmer)>=t and kmer has not been outputted
      output kmer
```

Sample Dataset

CGGACT**CGACA**GATGTGAAGAAATGTGAAGACTGAGTGAAGAGAAGAGGAA
ACA**CGACACGACA**TTG**CGACA**TAATGTACGAATGTAATGTGCCTATGGC

5 75 4

Sample Output

CGACA GAAGA AATGT

Visit http://rosalind.info/problems/ba1E/ . Solve the problem. Use the sample dataset given in the site.

Problem 4: Find a Position in a Genome Minimizing the Skew

Define the **skew** of a DNA string *Genome*, denoted *Skew*(*Genome*), as the difference between the total number of occurrences of 'G' and 'C' in *Genome*. Let $Prefix_i$ (*Genome*) denote the **prefix** (i.e., initial substring) of *Genome* of length *i*. For example, the values of *Skew*($Prefix_i$ ("CATGGGCATCGGCCATACGCCCATGGGCATCGGCCATACGCC")) are:

0 -1 -1 -1 0 1 2 1 1 1 0 1 2 1 0 0 0 0 -1 0 -1 -2

Minimum Skew Problem

Find a position in a genome minimizing the skew.

Given: A DNA string *Genome*.

Return: All integer(s) *i* minimizing *Skew*($Prefix_i$ (*Text*)) over all values of *i* (from 0 to |*Genome*|).

Sample Dataset

**CCTATCGGTGGATTAGCATGTCCCTGTACGTTTCGCCGCGAACTAGTTCA
CACGGCTTGATGGCAAATGGTTTTTCCGGCGACCGTAATCGTCCACCGA
G**

Sample Output

53 97