HW2-Lanqiu Yao

March 13, 2019

1 Bioinformatics Algorithms Homework 2

Lanqiu Yao

1.1 Question 1

Input: A Genome $G \in \Sigma_\star$ = A string (sequence of characters) over an alphabet of characters $\Sigma = a, t, c, g$; A reverse-complement restriction pattern k-cutters of length k, $p_k \in \Sigma_\star$, such that Reverse(Complement(p_k)) = p_k.

Output: An index i such that $p_k = G[i..i + k - 1]$ or the special value NIL if v does not appear in G.

Write pseudocode for linear search, which scans through the sequence, looking for pk. How will you optimize it, if you need to search for multiple pk's?

Psedocode Input:

- a dna string G
- a length of a reverse-complement string pk

Output:

- the index of the pk

Function:

Reverse_Complement_Index(G,k): for i in [1, n - k + 1]: count = 0; %% count how many reverse strings it has for j in [0, k-1]: dna1 = G[i+j] dna2 = G[i+k-1-j] if dna2 == reverse(dna1): %% function to check whether string1 and string2 are reversed count = count + 1 if count == k: v = i break if i == (n - k + 1) & count != k: v = NIL

To check whether the pseudocode is correct or not, we can make it into real code

Real python code

```
In [47]: def ComplementDNA(dna1, dna2):
             """
             Function to check whether the two nucleobases are complement
             -------------
```

```
    Input: dna1 and dna2, which are two strings

    Output: if the two input strings are complement, output is 1;
    """
    status = 0
    if dna1 == 'A' and dna2 =='T':
        status = status + 1
    if dna1 == 'T' and dna2 =='A':
        status = status + 1
    if dna1 == 'C' and dna2 =='G':
        status = status + 1
    if dna1 == 'G' and dna2 =='C':
        status = status + 1
    if status > 0:
        return(1)
    if status == 0:
        return(0)
```

```
In [48]: def Reverse_Complement_Index(dna,k):
    """
    Function to check whether there is a length k string in the input dna,
    that is a reverse complement dna string
    -------------
    Input: dna, which is a string
    Input: k, which is the length of the reverse complement string.

    Output: the index of the string.
    """
    n = len(dna)
    for i in range(n - k +1):
        count = 0
        for j in range(k):
            p0 = dna[i+j]
            p1 = dna[i+k-1-j]
            count = count + ComplementDNA(p0,p1)
        if count == k:
            v = i+1
            break
    if count != k:
        v = 'NIL'
    return(v)
```

Example

```
In [49]: dna = 'AGCTAGCTACGTAAAATTTT'

In [52]: Reverse_Complement_Index(dna,9)

Out[52]: 'NIL'
```

```
In [53]: Reverse_Complement_Index(dna,4)

Out[53]: 1

In [ ]: AGCT
```

If we need to search for multiple pk's, to make it more efficient, we can break the length of n sequence into two pieces and search those two pieces separately.

1.2 Question 2

In the same setting as the one described above: Let |G| = n and |pk| = k, what is the probability of a pk occurring in G, assuming that G is a random string with all characters occurring equiprobably. Consider the distance between two consecutive occurrences of pk in G: What is its average value? Variance?

My answer Set p_n^k as the probablity of having length k reverse-complement in the length n genome, then we can write induction as following:

$$\begin{aligned} p_n^k &= p_{n-1}^k + (1 - p_{n-1}^k)\frac{1}{4}\frac{2^k}{4^{k-1}} \\ &= p_{n-1}^k + (1 - p_{n-1}^k)\frac{1}{2^k} \end{aligned}$$

Then functoin means that: If we would like to find the probablity of having length k reverse-complement sequence in length n genome, there are two probability:

- 1. the length k reverse-complement sequence appears in the first $n-1$ sequence of the length n genome.
- 2. the lenght k reverse-complement sequence appears in the last k bases in the length n genome.

I. Therefore, if it is the first scenario, the probability is p_{n-1}^k.

II. If it is in the second scenario, we need that, the first $n-1$ bases do not have the length k reverse-complement sequences, the probability is $(1 - p_{n-1}^k)$. Besides, for the index of $n-k+1, ...n-1$ positions, we need to make sure that, when add one more base at the end, it can form a reverse-complement sequence. For a length k reverse-complement sequence, if we know what the first $\frac{k}{2}$ bases are, then we know the rest of them. Therefore, the string has $4^{k/2}$ possible orders. For the nth position, the probability to choose the right base is $\frac{1}{4}$. Therfore, the whole probability is $\frac{1}{4}\frac{2^k}{4^{k-1}} = \frac{1}{2^k}$

Combine them together, the p_n^k will be:

$$\begin{aligned} p_n^k &= p_{n-1}^k + (1 - p_{n-1}^k)\frac{1}{4}\frac{2^k}{4^{k-1}} \\ &= p_{n-1}^k + (1 - p_{n-1}^k)\frac{1}{2^k} \end{aligned}$$

Then, next step, we can calculate the induction.

$$p_n^k = p_{n-1}^k + (1 - p_{n-1}^k)\frac{1}{2^k}$$

$$p_n^k - 1 = p_{n-1}^k - 1 + (1 - p_{n-1}^k)\frac{1}{2^k}$$

$$p_n^k - 1 = (p_{n-1}^k - 1)\,\texttt{[U+FF08]}1 - \frac{1}{2^k}\,\texttt{[U+FF09]}$$

That is:

$$p_n^k - 1 = (p_{n-1}^k - 1)(1 - \frac{1}{2^k}) = (p_{n-2}^k - 1)(1 - \frac{1}{2^k})(1 - \frac{1}{2^k}) = ... = (p_k^k - 1)[1 - \frac{1}{2^k}]^{n-k}$$

And $p_k^k = \frac{2^k}{4^k} = \frac{1}{2^k}$. Then

$$p_n^k - 1 = (\frac{1}{2^k} - 1)[1 - \frac{1}{2^k}]^{n-k}$$

$$p_n^k = 1 - [1 - \frac{1}{2^k}]^{n-k+1}$$

The final answer is, the probability of a pk occurring in G is

$$p_n^k = 1 - [1 - \frac{1}{2^k}]^{n-k+1}$$

Expectation and variance For the second part of this question, we can follow the same idea. Set the distance between two length k reverse-complement sequences is x, where $x \in [1, n - 2k]$. Let $p_{n,k}^x$ be the probability of having a length x gap between two length k reverse-complement sequences.

Then we can write the induction like:

$$\begin{aligned} p_{n,k}^x &= p_{n-1,k}^x + (1 - p_{n-1,k}^x)\frac{2^k}{4^k}[1 - \frac{2^x}{4^x}]\frac{1}{4}\frac{2^k}{4^{k-1}} \\ &= p_{n-1,k}^x + (1 - p_{n-1,k}^x)\frac{1}{2^{k-1}}(1 - \frac{1}{2^x}) \end{aligned}$$

This folumar is because:

- 1. if the gap happens in the first $n - 1$ position, the the probability is $p_{n-1,k}^x$
- 2. if not, then the length k reverse complement sequence + length x gap + length k reverse complement sequence, which has the total length $2k + x$, must happens at the location $n - 2k - x + 1$th to nth bases in the length n genome. If so, the first length k reverse complement sequence has a probability of $\frac{2^k}{4^k}$ to happen, the middle gap has a probability of $1 - \frac{2^x}{4^x}$ to happen, the last $k - 1$ sequences has a probability of $\frac{1}{4}\frac{2^k}{4^{k-1}}$ to be a reverse complement sequence after adding one base.

Therefore, combine them together, the probability is

$$\begin{aligned} p_{n,k}^x &= p_{n-1,k}^x + (1 - p_{n-1,k}^x)\frac{2^k}{4^k}[1 - \frac{2^x}{4^x}]\frac{1}{4}\frac{2^k}{4^{k-1}} \\ &= p_{n-1,k}^x + (1 - p_{n-1,k}^x)\frac{1}{2^{k-1}}(1 - \frac{1}{2^x}) \end{aligned}$$

$$
\begin{aligned}
p_{n,k}^x - 1 &= p_{n-1,k}^x - 1 + (1 - p_{n-1,k}^x)\frac{1}{2^{k-1}}(1 - \frac{1}{2^x}) \\
&= (p_{n-1,k}^x - 1)(1 - \frac{1}{2^{k-1}}(1 - \frac{1}{2^x})) \\
&= (p_{n-2,k}^x - 1)(1 - \frac{1}{2^{k-1}}(1 - \frac{1}{2^x}))^2 \\
&= ... \\
&= (p_{2k+x,k}^x - 1)(1 - \frac{1}{2^{k-1}}(1 - \frac{1}{2^x}))^{n-2k-x}
\end{aligned}
$$

where

$$
p_{2k+x,k}^x = \frac{2^k}{4^k}(1 - \frac{2^x}{4^x})\frac{2^k}{4^k} = (1 - \frac{1}{2^x})\frac{1}{2^{k-1}}
$$

Therefore,

$$
p_{n,k}^x = 1 - (1 - \frac{1}{2^{k-1}}(1 - \frac{1}{2^x}))^{n-2k-x}(1 - (1 - \frac{1}{2^x})\frac{1}{2^{k-1}}) = 1 - (1 - \frac{1}{2^{k-1}}(1 - \frac{1}{2^x}))^{n-2k-x+1}
$$

sorry I do not know how to calculate its variance and expectaion since the probability formular is very complicated