Assignment 1

Bioinformatics (CIS 455)

Javier Arechalde

February 12, 2018

Problem 1-1 Jones & Pevzner, Problem 2.1

1.

We are going to write the pseudocode for an algorithm, that given a list of n numbers, returns the largest and smallest number in that list.

Algorithm 1 Finding largest and smallest numbers

```
1: Initialize max to L[0] and min to L[0]
2: for n in number list L do
3:     if n < min then
4:         min = n
5:     end if
6:     if n > max then
7:         max = n
8:     end if
9: end for
10: Return max and min
```

The time complexity of my algorithm will be $O(n) = n$, and the running time will be $2(n-1)+2$ because for every number in the list that starts from the second position after initialization, we need to check if the number is greater than the current maximum, or smaller than the current minimim.

2.

Now we will implement an algorithm that performs only $3n/2$ comparisons to find the smallest and largest numbers in the list. For this implementation we will use the tournament method described below (Divide & Conquer).

Algorithm 2 Divide & Conquer pseudocode

```
1: We have a list l with n integers l = (l_1, l_2, ...l_n)
2: function FINDMAXMIN(list,alow,ahigh)
3:     if lengthlist = 1 then
4:         Maximum and minimum are the same
           return maximum and minimum
5:     end if
6:     if lengthlist = 2 then
7:         Compare both elements in the list to find the maximum and minimum
           return maximum and minimum
8:     end if
9:     Divide the array in half, and store the two halves in arrL and arrR
10:    Call findmaxmin to find the maximum and minimum of arrL and arrR
11:    if Maximum of arrL or arrR is greater than the current max then
12:        Update global maximum
13:    end if
14:    if Minimum of arrL or arrR is smaller than the current min then
15:        Update global minimum
16:    end ifreturn the global maximum and the global minimum and the global minimum
17: end function
18: In the end, we have the maximum and minimum of the given list of integers.
```

In this case, we will divide the initial list with n numbers into two lists of $n/2$ each, for eacg of this lists, we will need to find the maximum and minimum of this lists, which can be done in $n/2 - 1$ comparisons. So in total we will need $3n/2 - 2$ comparisons to find the maximum and minimum of a list containing n numbers.

Problem 1-2 Jones & Pevzner, Problem 2.2

Now we will write the pseudocode for two algorithms that iterate over every index from $(0, 0, ..., 0)$ to $(n_1, n_2, ..., n_d)$, one of them will be recursive and the other one iterative.

Iterative pseudocode:

For the iterative pseudocode, we would have to implement d nested for loops to print all the possible combinations of indexes.

Algorithm 3 Iterative pseudocode

```
for i in range 0 → n_1 do
    for j in range 0 → n_2 do
        ...
        for k in range 0 → n_d do
            Print the results
        end for
        ...
    end for
end for
```

Recursive pseudocode:

In this case, we will model our problem as a tree, so in order to explore all the possible combinations, we will have to visit all the leafs in the tree. Our recursive algorithm will go one level lower if there are still nodes to visit, and once we visit all the leaves in downstream that node, we will go one level higher and visit all the children nodes of this next node.

Algorithm 4 Recursive pseudocode

```
function RECURSIVE(node)
    if We still didn't reach position d then
        for i in range 0 → n_i do
      RECURSIVE(next node downstream)
        end for
    end if
end function
```

Problem 1-3 Jones & Pevzner, Problem 2.3

- Yes, $\log n = O(n)$, because $O(n)$ is an upper bound for $\log n$, as it grows faster.
- No, $\log n = \Omega(n)$, because $\Omega(n)$ is not a lower bound for $\log n$, as $\log n$ grows significantly slower.
- No, $\log n = \Theta(n)$, because $\log n$ is not $O(n)$ and $\Omega(n)$ at the same time.

Problem 1-4 Jones & Pevzner, Problem 2.17

Will the viruses eventually kill all the bacteria?

In order to find if the viruses end up killing all the bacteria in the Petri dish, we only need to prove that the growth rate of the viruses is higher than the growth rate of the bacteria.

In the first minute, the virus kills one bacteria, and produces another copy of himself, and all the remaining bacteria reproduce, making 2 viruses and $2(n-1)$ bacteria.

The number of viruses continue to double at each step, also, the number of bacteria double at each step too, but before the bacteria double, the viruses kill as many bacteria as the number of viruses there are at that moment. Then, the growth rate of bacteria is slower than the growth rate of the viruses, because is slowed by the number of viruses at each step. Thus, at some point, the bacteria will be exterminated by the viruses.

Algorithm design

Algorithm 5 Algorithm for calculating the number of steps

1: At the beginning we have $n_v = 1$ virus and $n_b = n$ bacteria
2: At each step, the number of viruses double, the bacteria double too, but n_v are killed by the virus too before they double.
3: **while** $n_b > n_v$ **do**
4: $\quad n_b = 2 * (n_b - n_v)$
5: $\quad n_v = 2 * n_v$
6: $\quad steps++$
7: **end while**

Results

This are the results obtained by implementing and running over a sample test, our algorithm design.

```
javier@javier-ThinkPad-T440s:~/Work/BioInformatics/Assignment_1$ python Virus.py

Number of steps: 999
```

Figure 1: Output

In this graphic we compare the number of virus and the number of bacteria at each step.

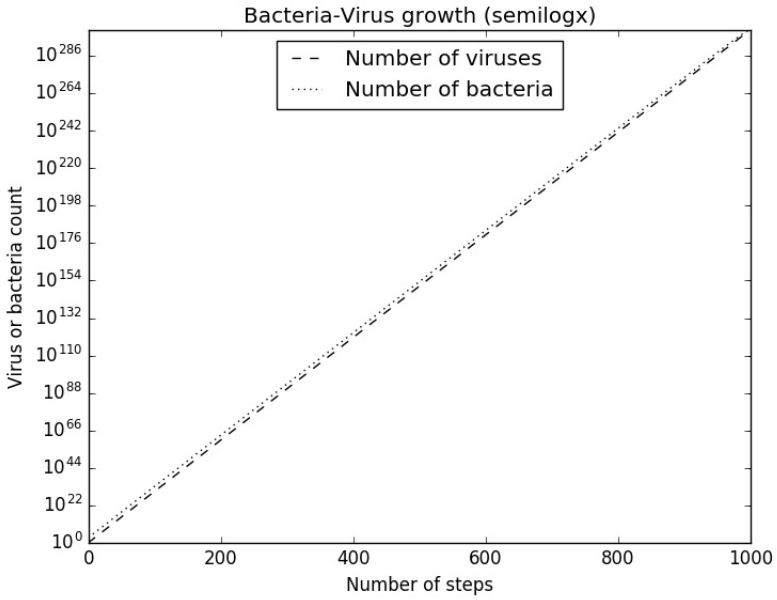

Figure 2: Evolution of Viruses vs. Bacteria

Problem 1-5

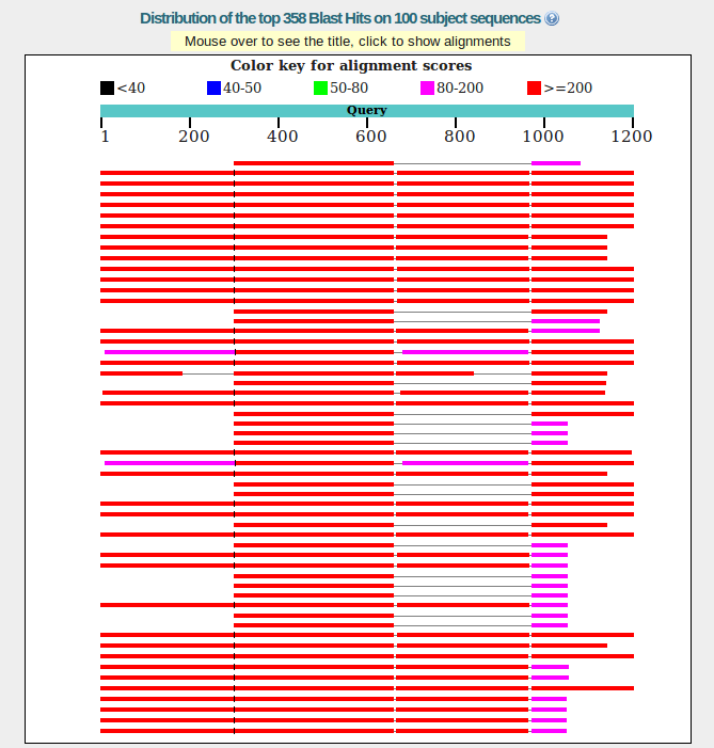

Figure 3: DNA Sequence

How does the running time depend on n?

The running time of my algorithm related to the number of bacteria n is $n-1$, which is the number of steps it will take the number of viruses to reach, and terminate the number of bacteria in the Petri dish.

5.A

The highest total score is 3599 with a max score of 435.

5.B

The description of the result with the highest score is: *Cloning vector pAxCALRL, complete sequence.*

5.C

The query coverage of this result is: 99%

5.D

The DNA sequence in Jurassic Park is **fictional**, because the tool we used couldn't find a match without gaps in the database. The lowest fraction of gaps found was 12%, that's why the maximum Ident value is 88%.

Problem 1-6 Rosalind

This problem was solved on **Rosalind**.