MT16121

Ankit Sharma

Problem 2 (part 3):

As per the question, if we use the simplest strategy in which we linearize the circular strings in all possible manners, then we would have a time complexity of $O(m^2n^2)$,

where 'm' is the length of the first circular DNA and 'n' is the length of the second circular DNA.

In order to squeeze the bounds of the worst case time complexity, we can either target to reduce the total number of pairs to be aligned (mxn for the simplest case) or we can target to reduce the time complexity of global sequence alignment algorithm i.e. Needleman Wunsch Alignment algorithm. The later one seems non-trivial so to present a solution, we shall aim at reducing the total number of pairs to be aligned.

Case 1:

$m > n$

if $m > n$, then we can begin with linearizing the sequence with length m randomly at any point and then we can align this linearized sequence with all possible linearized combinations of the string with length n.

This implies,

Worst case time complexity of $O(mn^2)$

Case 2:

$m < n$

if $m < n$, then we can begin with linearizing the sequence with length n randomly at any point and then we can align this linearized sequence with all possible linearized combinations of the string with length m.

This implies,

Worst case time complexity of $O(nm^2)$

Case 3:

$m = n$

This case would yield a time complexity of $O(m^2n^2)$ as we will have to linearize both the sequences for all possibilities.

Algorithm:

Step 1 – Compare m and n

Step 2 – If m > n,

str = linearize(stringm)

solution = needleman_wunsch(str, permute(stringn))

else if m < n,

str = linearize(stringn)

solution = needleman_wunsch(str, permute(stringm))

else

solution = needleman_wunsch(permute(stringm), permute(stringn))

Step 3 – return solution

where permute is a function to return all linearized permutations of the string which is passed as the argument.