Pairwise Sequence Alignment

Henry Zhang

March 2019

1 Part 1

$B(m,n) = 1$,if $m = 0 \vee n = 0$.
$B(m,n) = B(m-1,n), B(m,n-1), B(m-1,n-1)$ if $m \geq 1, n \geq 1$
Base case: When one of m and n have zero length, there would be only one alignment. When both of them has zero length, it would be trivial to prove.

Inductive step: When a dash is added to the end of the string of length m, the number of alignments is $B(m-1,n)$. When a dash is added to the end of the string of length n, the number of alignments is $B(m,n-1)$. When a dash is not added at the end of either string, the number of alignments is $B(m-1,n-1)$. $B(m,n)$ equals the sum of these three.

2 Part 2

$$S[i,j] = \begin{cases} S[i,j-1] + M_{-,Y_j}, \text{ for } i = 0 \vee j > 0 \\ S[i-1,j] + M_{X_i,-} \text{ for } i > 0 \vee j = 0 \\ max(S[i-1,j-1] + M_{X_i,Y_j}, S[i,j-1] + M_{-,Y_j}), S[i-1,j] + M_{X_i,-}) \text{ for } i > 0 \vee j > 0 \end{cases}$$

When i is at the boundary of the matrix, it means that the length of the first string is 0, then the value at [i,j] equals the value at [0,j-1] plus M_{-,X_j}.

When When j is at the boundary of the matrix, it means that the length of the first string is 0, then the value at [i,j] equals the value at [i-1,0] plus $M_{X_i,j}$.

When neither i or j is at the boundary, the value at [i,j] equals the maximum of the following:
1) value at [i-1, j-1], which is the optimal value for string lengths of i-1 and j-1, plus M_{X_i,Y_j}.

2) value at [i-1, j-1] plus M_{-,Y_j} 3) value at [i-1, j-1] plus M_{X_i,Y_j}

3 Part 3

Algorithm 1: ComputeGlobalAlignmentScores

Input: Sequences X and Y, and scoring matrix M
Output: A optimal score matrix S

$S \leftarrow$ an empty matrix of dimension $(|X| + 1) * (|Y| + 1)$;
$S[0, 0] \leftarrow 0$; `// Initialize the top left value`
for $i \leftarrow$ *0 to* $|X|$ **do**
    for $j \leftarrow 0$ *to* $|Y|$ **do**
        if $i = 0 \wedge j \neq 0$ **then**
            $S[i, j] = S[i, j - 1] + M_{-,Y_j}$
        else if $i \neq 0 \wedge j = 0$ **then**
            $S[i, j] = S[i - 1, j] + M_{X_i,-}$
        else if $i \neq 0 \wedge j \neq 0$ **then**
            $S[i, j] = max(S[i - 1, j - 1] + M_{X_i,Y_j}, S[i, j - 1] + M_{-,Y_j}, S[i - 1, j] + M_{X_i,-})$
return S

4 Part 4

Algorithm 2: ComputeAlignment

Input: Sequences X,Y, scoring matrix M, and an optimal matrix S
Output: A sequence string representing the optimal alignment

$X' \leftarrow$an empty sequence;
$Y' \leftarrow$an empty sequence;
$i \leftarrow |X|$;
$j \leftarrow |Y|$;
while $i \geq 0$ *and* $j \geq 0$ **do**
    if $S[i, j] = S[i-1, j-1] + M_{X_i, Y_j}$ **then**
        add X_i to the front of X';
        add Y_j to the front of Y';
        $i \leftarrow i - 1$;
        $j \leftarrow j - 1$;
    else if $S[i, j-1] + M_{-, Y_j}$ **then**
        add $-$ to the front of X';
        add Y_j to the front of Y';
        $j \leftarrow j - 1$;
    else if $S[i-1, j] + M_{X_i, -}$ **then**
        add X_i to the front of X';
        add $-$ to the front of Y';
        $i \leftarrow i - 1$;
if $i = 0$ *and* $j > 0$ **then**
    add $-$ to the front of X';
    add Y_j to the front of Y';
else if $i = 0$ *and* $j > 0$ **then**
    add X_i to the front of X';
    add $-$ to the front of Y';
return (X',Y')

5 Part 5

We can assume that sequences X and Y each have input sizes of m and n.

Running time of ComputeGlobalAlignmentScores:

The first two lines each take 1 operation. Within the innermost loop, each line takes 1 operation. So the two loops form $O(m * n)$ operation.

Running time of ComputeAlignment:

The first four lines each take 1 operation. The while loop takes O(min(m,n)) operations because each line within takes constant operations, and the rest takes constant operations.

In total, the complexity of GlobalAlignment is $O(mn)$.