ADA mini homework 3

(1)

Construct a DP table with size of DP[2][Sa.len() + 1], and define each element DP[i % 2][j] in array stand for the minimum operation cost it needs to transform from Sb.substr(0, i) to Sa.substr(0, j).

Before we start the loop, we need to consider the base case. First, DP[0][j] = j \* d, as it adds j characters to become Sa if Sb is empty. Second, every time we enter the second loop, we set DP[i % 2][0] to i \* e, as it delete all its characters to become an empty string as Sa. For all other general cases, we consider two conditions. First, if Sb[i] == Sa[j], we do nothing and set the cost of DP[I % 2][j] to DP[(I - 1) % 2][j – 1]. Else, we compare three different steps we can do. Add, remove, and replace, which equals to the cost of DP[i % 2][j – 1] + d, DP[(i - 1) % 2][j] + e, and DP[(i - 1) % 2][j – 1] + f, respectively. We choose the smallest one and assign DP[i][j] to that cost.

We can see that construct the table of i == Sb.len() only required the data of i == Sb.len() – 1, therefore, two rows are sufficient for us to calculate the answer.

Example DP table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Replace | Remove |  |  |
|  |  |  | Add | (i % 2, j) |  |  |

(2)

Transition function:

DP[I % 2][j] = If j == 0, i \* e.

Else If i == 0, j \* d

Else if Sa[j - 1] == Sb[I - 1], DP[(I - 1) % 2][j – 1]

Else min({DP[i % 2][j – 1] + d, DP[(i - 1) % 2][j] + e, and DP[(i - 1) % 2][j – 1] + f })

The C++ implementation of this function

1. for (int i = 0; i <= Sb.len(); i++) {
2. for (int j = 0; j <= Sa.len(); j++) {
3. if(i == 0)
4. DP[0][j] = j \*d
5. else if (j == 0)
6. DP[i % 2][j] = i \* e;
7. else if (Sa[j - 1] == Sb[i - 1])
8. DP[i % 2][j] = DP[(i - 1) % 2][j - 1];
9. else
10. DP[i % 2][j] = min({DP[(i - 1) % 2][j] + e, DP[i % 2][j - 1] + d, DP[(i - 1) % 2][j - 1] + f});
11. }
12. }