Dynamic Programming | Set 4 (Longest Common Subsequence)

We have discussed Overlapping Subproblems and Optimal Substructure properties in Set 1 and Set 2 respectively. We also discussed one example problem in Set 3. Let us discuss Longest Common Subsequence (LCS) problem as one more example problem that can be solved using Dynamic Programming.

LCS Problem Statement: Given two sequences, find the length of longest

the same relative order, but not necessarily contiguous. For example, “abc”, “abg”, “bdf”, “aeg”, ‘”acefg”, .. etc are subsequences of “abcdefg”. So a string of length n has 2^n different possible subsequences.

It is a classic computer science problem, the basis of
that outputs the differences between two files), and has applications in
bioinformatics.

Examples:
LCS for input Sequences “ABCDGH” and “AEDFHR” is “ADH” of length 3.
LCS for input Sequences “AGGTAB” and “GXTXAYB” is “GTAB” of length 4.

We strongly recommend that you click here and practice it, before moving on to the solution.

The naive solution for this problem is to generate all subsequences of both given sequences and find the longest matching subsequence. This solution is exponential in term of time complexity. Let us see how this problem possesses both important properties of a Dynamic Programming (DP) Problem.

1) Optimal Substructure:
Let the input sequences be X[0..m-1] and Y[0..n-1] of lengths m and n respectively. And let L(X[0..m-1], Y[0..n-1]) be the length of LCS of the two sequences X and Y. Following is the recursive definition of L(X[0..m-1], Y[0..n-1]).

If last characters of both sequences match (or X[m-1] == Y[n-1]) then
L(X[0..m-1], Y[0..n-1]) = 1 + L(X[0..m-2], Y[0..n-2])

If last characters of both sequences do not match (or X[m-1] != Y[n-1]) then

Popular Posts
- Top 10 Algorithms and Data Structures for Competitive Programming
- Top 10 algorithms in Interview Questions

L(X[0..m-1], Y[0..n-1]) = MAX (L(X[0..m-2], Y[0..n-1]), L(X[0..m-1], Y[0..n-2])

Examples:
1) Consider the input strings “AGGTAB” and “GXTXAYB”. Last characters match for the strings. So length of LCS can be written as:
L(“AGGTAB”, “GXTXAYB”) = 1 + L(“AGGTA”, “GXTXAY”)

2) Consider the input strings “ABCDGH” and “AEDFHR. Last characters do not match for the strings. So length of LCS can be written as:
L(“ABCDGH”, “AEDFHR”) = MAX (L(“ABCDG”, “AEDFH **R**”), L(“ABCDG **H**”, “AEDFH”))

So the LCS problem has optimal substructure property as the main problem can be solved using solutions to subproblems.

2) Overlapping Subproblems:
Following is simple recursive implementation of the LCS problem. The implementation simply follows the recursive structure mentioned above.

C/C++ Python

```
/* A Naive recursive implementation of LCS problem */
#include<bits/stdc++.h>

int max(int a, int b);

/* Returns length of LCS for X[0..m-1], Y[0..n-1] */
int lcs( char *X, char *Y, int m, int n )
{
```


- How to begin with Competitive Programming?
- Step by Step Guide for Placement Preparation
- Reflection in Java
- Memory Layout of C Programs
- Heavy Light Decomposition
- Sorted Linked List to Balanced BST
- Generics in Java
- Aho-Corasick Algorithm for Pattern Searching
- Insertion Sort , Binary Search , QuickSort , MergeSort , HeapSort

```
    if (m == 0 || n == 0)
      return 0;
    if (X[m-1] == Y[n-1])
      return 1 + lcs(X, Y, m-1, n-1);
    else
      return max(lcs(X, Y, m, n-1), lcs(X, Y, m-1, n));
}

/* Utility function to get max of 2 integers */
int max(int a, int b)
{
    return (a > b)? a : b;
}

/* Driver program to test above function */
int main()
{
  char X[] = "AGGTAB";
  char Y[] = "GXTXAYB";

  int m = strlen(X);
  int n = strlen(Y);

  printf("Length of LCS is %d\n", lcs( X, Y, m, n ) );

  return 0;
}
```

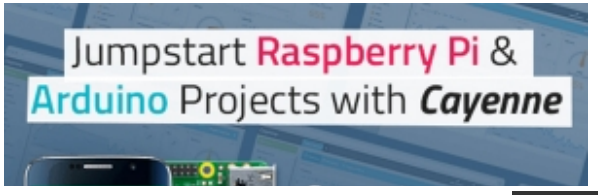
Run on IDE

Output:

```
Length of LCS is 4
```

Time complexity of the above naive recursive approach is O(2^n) in worst case and

- Common Interview Puzzles
- Interview Experiences
- Advanced Data Structures
- Design Patterns
- Dynamic Programming
- Greedy Algorithms
- Backtracking
- Pattern Searching
- Divide & Conquer
- Geometric Algorithms
- Searching
- Sorting
- Hashing
- Analysis of Algorithms
- Mathematical Algorithms
- Randomized Algorithms
- Recursion



worst case happens when all characters of X and Y mismatch i.e., length of LCS is 0.

Considering the above implementation, following is a partial recursion tree for input strings “AXYT” and “AYZX”

```
                        lcs("AXYT", "AYZX")
                       /                   \
              lcs("AXY", "AYZX")             lcs("AXYT", "AYZ")
              /              \                /                 \
lcs("AX", "AYZX") lcs("AXY", "AYZ")   lcs("AXY", "AYZ") lcs("AXYT
", "AY")
```

In the above partial recursion tree, lcs(“AXY”, “AYZ”) is being solved twice. If we

which are solved again and again. So this problem has Overlapping Substructure property and recomputation of same subproblems can be avoided by either using Memoization or Tabulation. Following is a tabulated implementation for the LCS problem.

C/C++ Python

```
/* Dynamic Programming C/C++ implementation of LCS proble
#include<bits/stdc++.h>

int max(int a, int b);

/* Returns length of LCS for X[0..m-1], Y[0..n-1] */
```


Tags

Accolite Adobe Advance Data Structures Advanced Data Structures Amazon array D-E-Shaw Directi Divide and Conquer Dynamic Programming Flipkart GATE GATE-CS-DS-&-Algo GATE-CS-Older GFacts Goldman Sachs Google Graph Greedy Algorithm Hashing Interview Experience Java MakeMyTrip MAQ Software MathematicalAlgo Matrix Microsoft Morgan Stanley Operating systems Oracle Pattern Searching Recursion Samsung SAP Labs SnapDeal stack STL Zoho

```
int lcs( char *X, char *Y, int m, int n )
{
   int L[m+1][n+1];
   int i, j;

   /* Following steps build L[m+1][n+1] in bottom up fash
      that L[i][j] contains length of LCS of X[0..i-1] ar
   for (i=0; i<=m; i++)
   {
     for (j=0; j<=n; j++)
     {
       if (i == 0 || j == 0)
         L[i][j] = 0;

       else if (X[i-1] == Y[j-1])
         L[i][j] = L[i-1][j-1] + 1;

       else
         L[i][j] = max(L[i-1][j], L[i][j-1]);
     }
   }

   /* L[m][n] contains length of LCS for X[0..n-1] and Y[
   return L[m][n];
}

/* Utility function to get max of 2 integers */
int max(int a, int b)
{
    return (a > b)? a : b;
}

/* Driver program to test above function */
int main()
{
  char X[] = "AGGTAB";
  char Y[] = "GXTXAYB";

  int m = strlen(X);
  int n = strlen(Y);
```

```
    printf("Length of LCS is %d\n", lcs( X, Y, m, n ) );

    return 0;
}
```

Run on IDE

Time Complexity of the above implementation is O(mn) which is much better than the worst case time complexity of Naive Recursive implementation.

The above algorithm/code returns only length of LCS. Please see the following post for printing the LCS.
Printing Longest Common Subsequence

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

References:

http://www.youtube.com/watch?v=V5hZoJ6uK-s
http://www.algorithmist.com/index.php/Longest_Common_Subsequence
http://www.ics.uci.edu/~eppstein/161/960229.html
http://en.wikipedia.org/wiki/Longest_common_subsequence_problem

Company Wise Coding Practice Topic Wise Coding Practice

191 Comments Category: Dynamic Programming Tags: Dynamic Programming

Related Posts:

- All ways to add parenthesis for evaluation
- Count all increasing subsequences
- Count distinct occurrences as a subsequence
- Longest repeating and non-overlapping substring

- Find minimum sum such that one of every three consecutive elements is taken
- Count Distinct Subsequences
- Non-crossing lines to connect points in a circle
- Count digit groupings of a number with given constraints

Previous post in category Next post in category

(Login to Rate and Mark)

2.8 Average Difficulty : **2.8/5.0**
Based on **155** vote(s)

Add to TODO List

Mark as DONE

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.

191 Comments GeeksforGeeks

Recommend 1 Share

Join the discussion…

```
#Longest common subsequence
public void longestCommonSubseq(String str1, String str2) {
if (str1 == null || str2 == null) return;
int a[][] = new int[str1.length()][str2.length()];
int i=0, j=0;
for (i=0; i<str1.length(); i++)="" {="" for="" (j="0;" j<str2.length();="'
str2.charat(j))="" {="" if="" (i="">0 && j>0)
a[i][j] = a[i-1][j-1]+1;
else
a[i][j] = 1;
}
else {
if (i>0 && j>0) {
a[i][j] = Math.max(a[i-1][j], a[i][j-1]);
}
}
}
}
```

see more

▲ | ▼ • Reply • Share ›

Manikandan Ramalingam • 2 months ago

Another way in java to print the common Sub sequence:-

```
public void longestCommonSubseq(String str1, String str2) {
if (str1 == null || str2 == null) return;
```

```
int a[][] = new int[str1.length()][str2.length()];
int i=0, j=0;
for (i=0; i<str1.length(); i++)="" {="" for="" (j="0;" j<str2.length();="'
str2.charat(j))="" {="" if="" (i="">0 && j>0)
a[i][j] = a[i-1][j-1]+1;
else
a[i][j] = 1;
}
else {
if (i>0 && j>0) {
a[i][j] = Math.max(a[i-1][j], a[i][j-1]);
}
}
}
```

see more

• Reply • Share ›

Bhanu Pant • 2 months ago

Pythonic implementation of recursive way:-

```
def lcs(str1, str2):
"""""""""
if len(str1) == 0 or len(str2) == 0:
return 0

if str1[-1] == str2[-1]:
return 1+lcs(str1[:-1], str2[:-1])
```

else:
return max(lcs(str1[:-1], str2), lcs(str1, str2[:-1]))

• Reply • Share ›

thavan • 2 months ago

how about this in python: set(X).intersection(Y)

• Reply • Share ›

Anand Kumar thavan • 2 months ago

How about this!! :P

a = [1,2,2,5,6]
b=[1,2,8,2]
above code will fail?

• Reply • Share ›

Ranga • 3 months ago

This was a famous problem. But I forgot the solution. When I thou
works for me. I could do memoization on below one as well. This

```
public class LongestCommonSubsequence {

public static void main(final String[] args) {
final String a = "BDCABA";
final String b = "ABCBDAB";

System.out.println("lcs :" + lcs(a.toCharArray(), b.toCharArray(), 0
```

```
}

public static int lcs(char[] a, char[] b, int a_low, int b_low, int a_hig
int b_high, int count_here) {
if (a_low > a_high || b_low > b_high) {
return count_here;
}
int max = 0;
for (int i = a_low; i <= a_high; i++) {
```

see more

Reply • Share ›

Narendra L • 3 months ago

Didn't see anyone doing in pythonic way

```
a = 'ABCDGH'
b = 'AEDFHR'

def oslen(x, y):
    if not( x and y):
        return 0

    if x[-1] == y[-1]:
        return 1 + oslen(x[0: -1], y[0:-1])
    else:
        return max(oslen(x, y[0:-1]), oslen(x[0:-1], y))

print oslen(a, b)
```

Reply • Share ›

Narendra L Narendra L • 3 months ago

If you ever wonder how spaces are not removed in my coc
Its ASCII space character its here " "
http://www.fileformat.info/inf...

Reply • Share ›

Karan Kapoor • 4 months ago

Top Down : http://ideone.com/uTX7nD

1 Reply • Share ›

Lehar Karan Kapoor • 4 months ago

Nice! :)

Reply • Share ›

Karan Kapoor Lehar • 4 months ago

Thanks ;) :)

1 Reply • Share ›

Yt R • 4 months ago

Using std::string solution:
http://code.geeksforgeeks.org/...

Reply • Share ›

Vardaan Sangar • 4 months ago

http://code.geeksforgeeks.org/...
Time:
O(n^2)
space:
DP soln O(n^2)

• Reply • Share ›

Surbhi • 4 months ago

In naive approach, what if we check first characters and move for
on)
Something like..

```
lcs( X, Y, 0, 0 );

int lcs( char *X, char *Y, int m, int n )
{
if (m == X.length() || n == Y.length())
return 0;
if (X[m] == Y[n])
return 1 + lcs(X, Y, m+1, n+1);
else
return max(lcs(X, Y, m, n+1), lcs(X, Y, m+1, n));
}
```

Is this approach correct? I tested for above two examples, and it c

• Reply • Share ›

no_limit ➔ Surbhi • 4 months ago

Yes it is the same thing.

• Reply • Share ›

Vikram singh • 4 months ago

easy solution :

```
/* problem statement:Longest common subsequence
Tag:Dynamic programming
i/p: we have two string s1 and s2
o/p :Lenth of longest common subsequence and that subsequenc
*/
#include <bits stdc++.h="">
using namespace std;
#define size 100
#define max(a,b) ((a)>(b)?(a):(b))
#define f first
#define s second
int main()
{
string s1="acbcf";
string s2="abcdaf";
int n=s1.length(),m=s2.length(),i,j;
int table[n+1][m+1];
```

see more

• Reply • Share ›

gizmogaurav • 5 months ago

Application of LCS :
1 .Molecular phylogenetics
2. wiki engine
3. Version control

ameya • 5 months ago

the code to get the sequence

```
#include <iostream>

int max(int a, int b)
{
return (a < b) ? b : a;
}

void lcs(string str1, string str2, int len1, int len2)
{
int arr[len2 + 1][len1 + 1];
int num;
string final_str = "";
string str;

for(int i = 0; i <= len2; i++)
{
for(int j = 0; j <= len1; j++)
```

• Reply • Share ›

Devabathula Vamsi Krishna • 5 months ago

Any Algorithm in O(m+n) complexity??

• Reply • Share ›

Shivam Maharshi • 6 months ago

Python code:

https://github.com/shivam-maha...

• Reply • Share ›

Sidharth Samant • 6 months ago

I used this Python code to solve this HackerRank challenege - htt
test cases terminate due to timeout. Can someone please help m

• Reply • Share ›

.NetGeek • 7 months ago

C# Implementation for both the Approach:

http://ideone.com/XVlF3R

Nice explaination by Tushar Roy: https://www.youtube.com/watch

1 • Reply • Share ›

avinash kumar • 7 months ago

Here is an improved one which uses O(n) space instead of O(mn

http://ideone.com/plvlezA8

Reply • Share ›

Rafid Bin Mostofa avinash kumar • 7 months ago

This code is not successful when those two strings have r
= "Better". While LCS should be 0, it shows 1 !!

Here (http://ideone.com/BOnvwD), I used the same algo
This one checks out fine.

And the bug in your code is probably in line no 13. When i=
== Y[j-1], I don't know that.

1 Reply • Share ›

avinash kumar Rafid Bin Mostofa • 7 months ago

thanks for pointing this out. And at line 13, i think ar
memory location, i.e., X[-1], X[-2], X[-3]..... all of the
That is why, the compiler does not report this as ar

Reply • Share ›

Anonymous avinash kumar • 5 months ago

good bottom up process

Reply • Share ›

David Sreid • 7 months ago

The dynamic programming method won't work for strings with the
one string separated from each other and appearing multiple time

E.g.
1
5 4
aaaba aaaa

returns
Length of LCS is 4

• Reply • Share ›

Vardaan Sangar David Sreid • 4 months ago
http://code.geeksforgeeks.org/...
Its working by some correction

• Reply • Share ›

Paul David Sreid • 7 months ago
That's the correct answer.

• Reply • Share ›

kalyana kannan • 8 months ago
Have come up with JAVA implementation, can someone review fo

```
public static void main(String[] args) {

String sequence1="ABCDGH";
String sequence2="KAEDFH";
String longSequence=getLongSequence(sequence1.toCharArray(
System.out.format("Long Sequence %s :: Sequence Length %d",l
```

}

private static String getLongSequence(char[] sequence1, char[] s
int flag=0;
String finalSeq = "";
for(int i=0;i<sequence1.length;i++){ for(int="" j="flag;j<sequence
finalseq="finalSeq.concat(""+sequence1[i]);" flag++;="" break;="" }

Reply • Share ›

Rashmi Mishra • 9 months ago

if we want to print the common chars too then while filling the dp a
print it .
http://code.geeksforgeeks.org/...

Reply • Share ›

linjapyc Rashmi Mishra • 7 months ago

You are absolutely wrong! For string "AGAGTAB" and "AG
”， which is obviously wrong.

Reply • Share ›

Rashmi Mishra linjapyc • 7 months ago

i had already mentioned here
http://www.geeksforgeeks.org/p... which is indeed
that above method is wrong

For your satisfaction, Yes i m wrong and hope that

Reply • Share ›

Rahul Singal • 10 months ago

top down approach of the above program:
http://shadowhackit.blogspot.i...

Reply • Share ›

anuj bajpai ➚ Rahul Singal • 10 months ago

according to the example you gave the answer should be '

1 Reply • Share ›

Rahul Singal ➚ anuj bajpai • 10 months ago

Thanks for making the correction

Reply • Share ›

Prashanth • 10 months ago

Sort both the strings and just go for one to one mapping..Its pretty

Reply • Share ›

Rohit Balodi ➚ Prashanth • 10 months ago

that is inaccurate as subsequence here means that matcl
you sort both strings then there is huge possibility of mess

2 Reply • Share ›

Vedant Tikku • a year ago

What will be LCS for baab and abb with this code?

Reply • Share ›

Rahul Singal ➚ Vedant Tikku • 10 months ago

It will be 3 - "aab"

▲ | ▼ • Reply • Share ›

Tilak Raj ➚ Vedant Tikku • a year ago

2 i.e. 'ab' .

▲ | ▼ • Reply • Share ›

raj ➚ Tilak Raj • 8 months ago

i think it will be bb

▲ | ▼ • Reply • Share ›

Vijay Kumar meena • a year ago

what if i want to know LCS of multiple (more then three) strings??

1 ▲ | ▼ • Reply • Share ›

Heis Issaqua • a year ago

// my java verson

```
public class LCSEfficient {
public static void main(String[] args) {
char[] c1 = {'A', 'G', 'G', 'T', 'A', 'B'};
char[] c2 = {'G','X' ,'T', 'X', 'A', 'Y', 'B'};
int m = c1.length;
int n = c2.length;
int opt[][] = new int[m+1][n+1];
```

```
System.out.println(opt[m][n] + "---" );
int res = FindLCS(c1, c2, c1.length, c2.length, opt);
System.out.println("the answer is "+ res);
}
public static int FindLCS(char[] c1, char[] c2,
int m, int n, int[][]opt) {
for(int i = m - 1; i >= 0 ; i--){
for(int j = n - 1; j >= 0; j--) {
if(c1[i] == c2[j]) {
```

see more

Reply • Share ›

Avi Kr ➚ Heis Issaqua • 2 months ago

Thanks :)

Reply • Share ›

Marin Veršić • a year ago

I firmly believe that python solution with dynamic programming fro
because of shallow list copy at line 9:
L = [[None]*(n+1)]*(m+1)

whereas it should be:
L = [[None]*(n+1) for i in xrange(m+1)]

Please make a correction!

Reply • Share ›

Nikhil Kumar Singh ➚ Marin Veršić • a year ago

Of course @Marin we will make this change as suggested declaring it as "incorrect" just for a shallow copy..I hope yo

^ | v • Reply • Share ›

Marin Veršić ➚ Nikhil Kumar Singh • a year ago

Well, when you write lis = [1]*5, a list is created by 5 times like: [1,1,1,1,1].

Just the same, when you write lis = [[1]]*5, a list is mutable) 5 times. But copying mutable objects doe reference to the same object is repeated 5 times ir copy. Therefore, doing something like lis[0][0] = 2 r

I hope now you see why your code gives wrong so

Check out this http://www.python-course.eu/de... fo

^ | v • Reply • Share ›

Nikhil Kumar Singh ➚ Marin Veršić • a year ago

Thanks for the explanation

^ | v • Reply • Share ›

GeeksforGeeks Mod ➚ Marin Veršić • a year ago

Thanks for writing to us. We shall look into this and update

^ | v • Reply • Share ›

Load more comments

Subscribe Add Disqus to your site Privacy