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 subsequence present in both of them. A subsequence is a sequence that appears in 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 diff (a file comparison program 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.

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
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”))

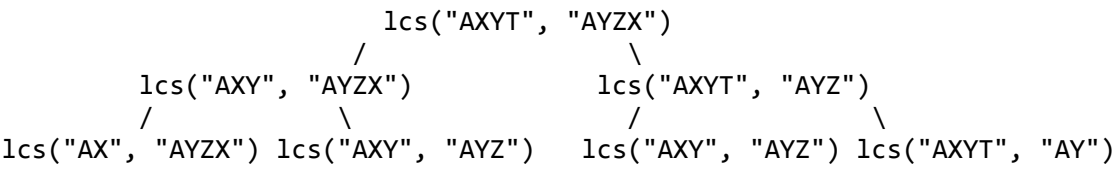
So the LCS problem has optimal substructure property as the main problem can be solved using solutions to subproblems.

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

```
/* A Naive recursive implementation of LCS problem */
#include<stdio.h>
#include<stdlib.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 )
{
   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 ) );

  getchar();
  return 0;
}
```

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



In the above partial recursion tree, lcs("AXY", "AYZ") is being solved twice. If we draw the complete recursion tree, then we can see that there are many subproblems 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.

```
/* Dynamic Programming implementation of LCS problem */
#include<stdio.h>
#include<stdlib.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 )
{
   int L[m+1][n+1];
   int i, j;

   /* Following steps build L[m+1][n+1] in bottom up fashion. Note
      that L[i][j] contains length of LCS of X[0..i-1] and Y[0..j-1] */
   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[0..m-1] */
    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 ) );

  getchar();
  return 0;
}
```

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

Related Topics:

- Linearity of Expectation
- Iterative Tower of Hanoi
- Count possible ways to construct buildings
- Build Lowest Number by Removing n digits from a given number
- Set Cover Problem | Set 1 (Greedy Approximate Algorithm)
- Find number of days between two given dates
- How to print maximum number of A's using given four keys
- Write an iterative O(Log y) function for pow(x, y)

Tags: Dynamic Programming

Writing code in comment? Please use ideone.com and share the link here.

Shubham Najardhane • 2 days ago
How to print the path?
means the sub sequence string.
if AXYT and AYZX the result should be AY
• Reply • Share ›

Nishant Garg • 10 days ago
O(N) solution
• Reply • Share ›

Nishant Garg • 10 days ago

```
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <algorithm>
using namespace std;

#define REP0(i, n) for (int i = 0; i < n; i++)
#define REP1(i, n) for (int i = 1; i <= n; i++)
#define REP(i, l, r) for (int i = l; i <= r; i++)
#define RP(i, r, l) for (int i = r; i >= l; i--)
```

```
#define FORE(i, x) for (int i = ti[x]; i != -1; i = e[i].n)

#define MAX_N 600000
#define MAX_M 260000

typedef struct node* ntp;
struct node { ntp c[26], f; int lv; } tr[MAX_N];
ntp head, tail;
char a[MAX_M], b[MAX_M];
```

see more

• Reply • Share ›

sk • a month ago

O(n^3) time complexity and O(1) space complexity solution:

http://ideone.com/hyf02X

• Reply • Share ›

Guest • a month ago

Pls help in dp approach
L[0][0] = 0;
then i=0,j=1; then how X[0-1] or X[-1] is compared with Y[0]?
Thanks in advance :)

• Reply • Share ›

KshitijAwadhiya ➚ Guest • 7 days ago

check this line

```
if (i == 0 || j == 0)
L[i][j] = 0;
```

• Reply • Share ›

Guest • a month ago

Why we are checking if (m == 1 || n == 1) instead of if (m == 0 || n == 0) ?

• Reply • Share ›

Shyam choudhary • a month ago

here is my code of the same

http://shyamcodersphere.blogsp...

• Reply • Share ›

prashant jha • 2 months ago

http://ideone.com/VoAVCR

• Reply • Share ›

mukesh thakur • 2 months ago

what are the information stored in lcs[x-1][y-1],lcs[i-1][j] and lcs[i][j-1]??

• Reply • Share ›

coder12489 • 3 months ago

Posting in solution in ruby, correct me if wrong :)

```
class Node
attr_accessor :val, :move
def initialize(val, move)
@val = val
@move = move
end

def val
@val
end

def move
@move
end
end

class LongestCommonSubsequence
```

see more

• Reply • Share ›

Aditya Goel • 5 months ago

We can solve this problem by using using O(max(m,n)) space complexity.

3 • Reply • Share ›

Hello_world • 6 months ago

strlen returns length of string including null. Do we need L[M+1][N+1] array ..
this will give wrong output.

• Reply • Share ›

Hello_world • 6 months ago

overlapping subproblem solution will give always "correct_result+1"

eg: for input "abc" and "abc"

longest_common_subsequence (0,0) =0

longest_common_subsequence (1,1) = 1

longest_common_subsequence (1,1) = 1

longest_common_subsequence (2,2) = 2

longest_common_subsequence (3,3) = 3

longest_common_subsequence (4,4) = 4

3 • Reply • Share ›

liger • 7 months ago

How the complexity is O(mn)? please explain.....if i am not wrong for two nested loops it should be O(n^2).

• Reply • Share ›

zinga liger • 3 months ago

if m==n, then only u r correct. Its n^2. But for m!=n, its mn

• Reply • Share ›

L liger • 6 months ago

It's nm because the two strings we are working with could be of different length. Your right it is two nested loops but we cannot always think that a nested loop will be n^2, it depends on the variable length we are looping. The outer loop runs for m times and the inner loop runs n times giving us O(mn).

1 • Reply • Share ›

RK- An Unproven Theorem L • 6 months ago

The outer loop runs for m times and the inner loop runs n times giving us O(mn).

• Reply • Share ›

huzefa biyawarwala • 7 months ago

guys , is there any thing in the LCS(longest common subsequence
) algorithm even if a single line , that can be rewritten in a much more efficient way ? if yes please help me to find it , i am searching for how to improve LCS algorithm ....

1 • Reply • Share ›

Anton Zuykov huzefa biyawarwala • 5 months ago

" if yes please help me to find it , i am searching for how to improve LCS algorithm "

Are you trying to find out yourself or are you trying to convince people to do it for you? ;)

• Reply • Share ›

huzefa Anton Zuykov • 5 months ago

@Anton: if there's anyone who has some idea , i can discuss with him abt that and then that can be implemented , besides this i am doing my research work in this area ,so its better fr me to take as many as ideas as possble ..... i hope u

get it now....

Reply • Share ›

Anton Zuykov ➚ huzefa • 5 months ago

In short - you don't ask such questions on websites. If a person is smart enough to have something to offer as an answer, he should be smart enough not to give that answer, at least, until he publishes a paper.

1 Reply • Share ›

Anton Zuykov ➚ huzefa • 5 months ago

I perfectly got what you said from the first reading of your post. The problem is that you didn't get what I was heading to.

If there is anyone who has some idea on how to improve things, I would suggest him to write that idea as a paper and publish it. That way he at least can get a deserved credit for its idea or/and work done.

1 Reply • Share ›

Guest • 7 months ago

If X[0] == Y[0] then L[i][j] =1 not 0. This is missed in the solution provided

Reply • Share ›

Hello_world ➚ Guest • 6 months ago

for i=1 and j=1 below line will include X[0] and Y[0] case...L[0][0]=0;

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

Reply • Share ›

rohit_90 • 7 months ago

Here is the space optimized code which uses only one array and two extra variables.

link:

http://ideone.com/XMFpXJ

4 Reply • Share ›

Jon Snow • 7 months ago

Below is link to space optimized code which uses O(n) space if length of both string are same. We need only current and previous row of matrix to compute next value. So after each iteration we just swapped pointers of both array. so current row become previous and previous row become current.
http://ideone.com/BWaUDL

1 Reply • Share ›

Guest • 8 months ago

static String GetLCS2(String s1, String s2)

{

String result = String.Empty;

if (String.IsNullOrEmpty(s1) ||

String.IsNullOrEmpty(s2))

return String.Empty;

for (int i = 0; i < s1.Length; i++)

{

for (int j = 0; j < s2.Length; j++)

{

count++;

see more

• Reply • Share ›

nfynt Guest • 5 months ago

it wont work for case like this:
s1="abcdcds"
s2="sss"

• Reply • Share ›

BATMAN Guest • 8 months ago

You think it works for all the cases ? ;)

• Reply • Share ›

sukanya • 8 months ago

for new programmer
to understand the flow of LCs
http://ideone.com/EOvLLf

• Reply • Share ›

DON • 8 months ago

/* Dynamic Programming implementation of LCS problem */

#include<stdio.h>

```
#include<stdlib.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 )

{

int L[m+1][n+1];

int i, j,count=1;

/* Following steps build L[m+1][n+1] in bottom up fashion. Note

that L[i][j] contains length of LCS of X[0..i-1] and Y[0..j-1] */
```

see more

• Reply • Share ›

sarthak • 9 months ago

what is programming of this seq.
* *
** **
*** ***
**** ****
****

• Reply • Share ›

liger sarthak • 7 months ago

public class StarPrint{
public static void main(String[] args)
{String a = *;
int n = 4; // n specifies the number of lines to printed

for(int i=1; i<n; i++)="" this="" loop="" will="" print="" the="" star="" upto="" the="" n-1="" times="" {="" system.out.println(a="" +"="" "="" +="" a);="" a="a" +="" "*";="" }="" to="" print="" the="" last="" line="" system.out.println(a+="" "*"="" +="" a);="" }="" }="">

• Reply • Share ›

Santanu Naskar • 9 months ago

A simple Java code that prints the length of LCS and characters also:

public class LCS{
public static void main(String[] args) {
try{

```
String A = "ABCDGH";
String B = "AEDFHR";
String ans = LCS1(A,B);
System.out.println(ans);
}
catch(Exception ee){System.out.println("Problem in Computation");}
}
static String LCS1(String A,String B){
int Alen = A.length(),Blen=B.length();
int[][] LCSArr = new int[Alen+1][Blen+1];
StringBuilder sb = new StringBuilder();
for(int i=1;i<=Alen;i++)
{
```

see more

• Reply • Share ›

portgas Santanu Naskar • 6 months ago

Does not print correct, try with "abcdef" "acdgea"

• Reply • Share ›

rahul giri • 9 months ago

lcs in easy way compare to DP..... lol

http://ideone.com/U1y5DX

• Reply • Share ›

Santanu Naskar rahul giri • 9 months ago

Will fail.

http://ideone.com/XrAw7A

aaaabbbbbccghhghiiikkk

12345gghi098

1 • Reply • Share ›

Vãîbhåv Joshî • 10 months ago

LCS DP java code

http://ideone.com/kr1E61

2 • Reply • Share ›

Karshit Jaiswal • 10 months ago

To print the LCS using DP

http://ideone.com/djbGIA

• Reply • Share ›

navneet1075 • 10 months ago

```
package com.exampke;

import java.util.ArrayList;

import java.util.Collections;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class StringAlgos2 {

public static void main(String[] args) {

String longestCommonSubsequence = StringAlgos

.getLongestCommonSubSequence("AAAFRTSRS", "AAFRSRS");

System.out.println("longest common subsequence is ========"
```

see more

• Reply • Share ›

Karshit Jaiswal • 10 months ago

Space Efficient LCS
Space Complexity : O(min(m,n))
http://www.ics.uci.edu/~eppste...
My Code of the above logic: http://ideone.com/R3lp88

2 • Reply • Share ›

saiyam agarwal ➦ Karshit Jaiswal • 10 months ago

I think there is error in your logic ... Line 20 should be
curr[j] = max(prev[j], curr[j-1]);

• Reply • Share ›

Karshit Jaiswal ➦ saiyam agarwal • 10 months ago

please provide a test case where it does not work.
it shud be prev[j-1]
Please refer to the research link given by me.

• Reply • Share ›

Karshit Jaiswal • 10 months ago

@GeeksforGeeks Guys please post the space optimized version of LCS which is based on Hirschberg's algorithm

Hirschberg's algorithm.

Also please update the article with some applications of LCS.
This will help the article to be complete.

jc • a year ago

/* Returns length of LCS for X[0..m-1], Y[0..n-1] */
int lcs(char *X, char *Y, int m, int n)
{
.
.
.
/* L[m][n] contains length of LCS for X[0..n-1] and Y[0..m-1] */
return L[m][n];
}
source of confusion between comments. Initially it says X[0..m-1] in the late comment it says X[0..n-1]

prashant jha • a year ago

the complexity of naive recursive fun is exponential but in dp there are m*n subproblems so its complexity is 0(mn)
here is my implementation
http://ideone.com/XD4UTd

prashant jha • a year ago

```
#include<iostream>
#include<string.h>
#define m 20
int arr[m][m];
using namespace std;
int fun(char st1[],char st2[],int low1,int low2,int high1,int high2)
{
if(arr[low1][low2]!=-1)
return arr[low1][low2];
if((low1>high1)||(low2>high2))
return 0;
if(st1[low1]==st2[low2])
{
arr[low1][low2]=1+fun(st1,st2,low1+1,low2+1,high1,high2);
return arr[low1][low2];
}
```

else
{

see more

1 • Reply • Share ›

Ankit Jain • a year ago

int lcs (char str1[],char str2[],int len1,int len2)
{
int M[len1+1][len2+1],i,j,flag[len1+1][len2+1],a=0;
for(i=0;i<len1+1;i++) {="" m[i][0]="0;" flag[i][0]="2;" }="" for(j="0;j<len2+1;j++)" {="" m[0][j]="0;" flag[0][j]="2;" }="" for(i="1;i<len1+1;i++)" {="" for(j="1;j<len2+1;j++)" {="" if(str1[i-1]="=str2[j-1])" {="" m[i][j]="M[i-1][j-1]+1;" flag[i][j]="3;" }="" else="" if(m[i-1][j]="">=M[i][j-1])
{
M[i][j]=M[i-1][j];
flag[i][j]=1;
}
else
{
M[i][j]=M[i][j-1];
flag[i][j]=-1;
}
}
}

see more

• Reply • Share ›

Joel • a year ago

Here's an implementation although not memoized that returns the sequence itself:

struct pair {

int start;

int end;

};

struct char_list {

char c;

struct char_list *next_char;

};

```
int find_len(struct char_list *list) {

int n = 0;
```

see more

• Reply • Share ›

Load more comments

- All permutations of a given string
- Memory Layout of C Programs
- Understanding “extern” keyword in C
- Median of two sorted arrays
- Tree traversal without recursion and without stack!
- Structure Member Alignment, Padding and Data Packing
- Intersection point of two Linked Lists
- Lowest Common Ancestor in a BST.
- Check if a binary tree is BST or not
- Sorted Linked List to Balanced BST

Recent Comments