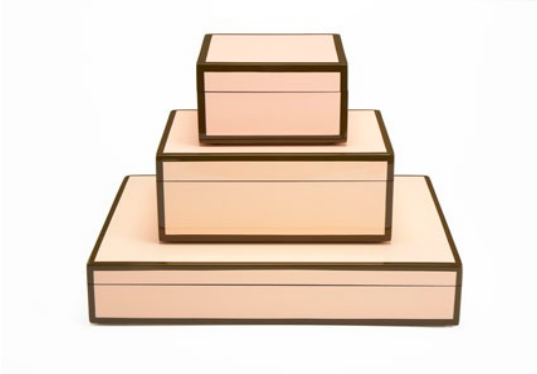
Dynamic Programming | Set 22 (Box Stacking Problem)

You are given a set of n types of rectangular 3-D boxes, where the i^th box has height h(i), width w(i) and depth d(i) (all real numbers). You want to create a stack of boxes which is as tall as possible, but you can only stack a box on top of another box if the dimensions of the 2-D base of the lower box are each strictly larger than those of the 2-D base of the higher box. Of course, you can rotate a box so that any side functions as its base. It is also allowable to use multiple instances of the same type of box.

Source: http://people.csail.mit.edu/bdean/6.046/dp/. The link also has video for explanation of solution.



The Box Stacking problem is a variation of LIS problem. We need to build a maximum height stack.

Following are the key points to note in the problem statement:
1) A box can be placed on top of another box only if both width and depth of the upper placed box are smaller than width and depth of the lower box respectively.
2) We can rotate boxes. For example, if there is a box with dimensions {1x2x3} where 1 is height, 2×3 is base, then there can be three possibilities, {1x2x3}, {2x1x3} and {3x1x2}.
3) We can use multiple instances of boxes. What it means is, we can have two different rotations of a box as part of our maximum height stack.

Following is the **solution** based on DP solution of LIS problem.

1) Generate all 3 rotations of all boxes. The size of rotation array becomes 3 times the size of original array. For simplicity, we consider depth as always smaller than or equal to width.

2) Sort the above generated 3n boxes in decreasing order of base area.

3) After sorting the boxes, the problem is same as LIS with following optimal substructure property.
MSH(i) = Maximum possible Stack Height with box i at top of stack
MSH(i) = { Max (MSH(j)) + height(i) } where j < i and width(j) > width(i) and depth(j) > depth(i).
If there is no such j then MSH(i) = height(i)

4) To get overall maximum height, we return max(MSH(i)) where 0 < i < n

Following is C++ implementation of the above solution.

```
/* Dynamic Programming implementation of Box Stacking problem */
#include<stdio.h>
#include<stdlib.h>

/* Representation of a box */
struct Box
{
  // h -> height, w -> width, d -> depth
  int h, w, d;  // for simplicity of solution, always keep w <= d
};

// A utility function to get minimum of two intgers
int min (int x, int y)
{ return (x < y)? x : y; }
```

```
// A utility function to get maximum of two intgers
int max (int x, int y)
{ return (x > y)? x : y; }

/* Following function is needed for library function qsort(). We
   use qsort() to sort boxes in decreasing order of base area.
   Refer following link for help of qsort() and compare()
   http://www.cplusplus.com/reference/clibrary/cstdlib/qsort/ */
int compare (const void *a, const void * b)
{
    return ( (*(Box *)b).d * (*(Box *)b).w ) -
           ( (*(Box *)a).d * (*(Box *)a).w );
}

/* Returns the height of the tallest stack that can be formed with give type
int maxStackHeight( Box arr[], int n )
{
   /* Create an array of all rotations of given boxes
      For example, for a box {1, 2, 3}, we consider three
      instances{{1, 2, 3}, {2, 1, 3}, {3, 1, 2}} */
   Box rot[3*n];
   int index = 0;
   for (int i = 0; i < n; i++)
   {
      // Copy the original box
      rot[index] = arr[i];
      index++;

      // First rotation of box
      rot[index].h = arr[i].w;
      rot[index].d = max(arr[i].h, arr[i].d);
      rot[index].w = min(arr[i].h, arr[i].d);
      index++;

      // Second rotation of box
      rot[index].h = arr[i].d;
      rot[index].d = max(arr[i].h, arr[i].w);
      rot[index].w = min(arr[i].h, arr[i].w);
      index++;
   }

   // Now the number of boxes is 3n
   n = 3*n;

   /* Sort the array 'rot[]' in decreasing order, using library
      function for quick sort */
   qsort (rot, n, sizeof(rot[0]), compare);

   // Uncomment following two lines to print all rotations
   // for (int i = 0; i < n; i++ )
   //    printf("%d x %d x %d\n", rot[i].h, rot[i].w, rot[i].d);

   /* Initialize msh values for all indexes
```

```
       msh[i] -> Maximum possible Stack Height with box i on top */
    int msh[n];
    for (int i = 0; i < n; i++ )
       msh[i] = rot[i].h;

    /* Compute optimized msh values in bottom up manner */
    for (int i = 1; i < n; i++ )
       for (int j = 0; j < i; j++ )
          if ( rot[i].w < rot[j].w &&
               rot[i].d < rot[j].d &&
               msh[i] < msh[j] + rot[i].h
             )
          {
               msh[i] = msh[j] + rot[i].h;
          }


    /* Pick maximum of all msh values */
    int max = -1;
    for ( int i = 0; i < n; i++ )
       if ( max < msh[i] )
          max = msh[i];

    return max;
}

/* Driver program to test above function */
int main()
{
  Box arr[] = { {4, 6, 7}, {1, 2, 3}, {4, 5, 6}, {10, 12, 32} };
  int n = sizeof(arr)/sizeof(arr[0]);

  printf("The maximum possible height of stack is %d\n",
         maxStackHeight (arr, n) );

  return 0;
}
```

Output:

```
The maximum possible height of stack is 60
```

In the above program, given input boxes are {4, 6, 7}, {1, 2, 3}, {4, 5, 6}, {10, 12, 32}. Following are all rotations of the boxes in decreasing order of base area.

```
10 x 12 x 32
12 x 10 x 32
32 x 10 x 12
4 x 6 x 7
4 x 5 x 6
6 x 4 x 7
5 x 4 x 6
7 x 4 x 6
6 x 4 x 5
```

```
1 x 2 x 3
2 x 1 x 3
3 x 1 x 2
```

The height 60 is obtained by boxes { {**3**, 1, 2}, {**1**, 2, 3}, {**6**, 4, 5}, {**4**, 5, 6}, {**4**, 6, 7}, {**32**, 10, 12}, {**10**, 12, 32}}

Time Complexity: O(n^2)
Auxiliary Space: O(n)

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

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.

Shashank Kumar • 22 days ago

Doesn't work correctly:
Box arr[] = {{1,1,1}, {1,2,3}};
Result: 4 (expected result: 3)

problem is that the code doesn't differentiate if it is looking at rotation of same box or different boxes.

In above example, the erroneous comparison is:
{3,1,2} vs {1,2,3} (both are rotation of same box)

Can someone suggest a good way of overcoming this problem?

Can someone suggest a good way of overcoming this problem?

• Reply • Share ›

Siddarth Shashank Kumar • 12 days ago

4 is perfectly correct. Note that we can use multiple instances of the same box.

• Reply • Share ›

shashank Siddarth • 6 days ago

yea.. you are right.. I missed the constraint.

Thansk

• Reply • Share ›

newbie • a month ago

Could anyone please tell me, why I am getting an o/p 45 instead of 60 for the input set given above.
below is my code. Thanks in advance.

//You are given a set of n types of rectangular 3-D boxes, where the i^th box has height h(i), width w(i) and depth d(i) (all real numbers).
//You want to create a stack of boxes which is as tall as possible, but you can only stack a box on top of another box if the dimensions of
//the 2-D base of the lower box are each strictly larger than those of the 2-D base of the higher box.

#include<cstdio>
#include<cstdlib>
#include<iostream>

#define INT_MIN -1

using namespace std;

---

see more

• Reply • Share ›

Jongi • 3 months ago

Why is sorting on basis of area needed?
The consideration is that both width and depth of the upper box should be less. So can't we sort in decreasing order by width first, and then find longest decreasing sub sequence on basis of depth?

• Reply • Share ›

helper • 7 months ago

@GeeksforGeeks , the terms used for dimensions look misleading.

Please use length, breadth, height.

• Reply • Share ›

TheRationale helper • 7 months ago

Perhaps breadth is used in the UK, but width is certainly more common in the US - no modern terminology uses breadth.

• Reply • Share ›

clish TheRationale • 6 months ago

Because the boxes rotate, order doesn't matter. That said, I've only ever seen 3d triplets as (length x width x height) -- height is never first. Maybe this is what helper meant ;o)

1 • Reply • Share ›

Abhishek Bind • 7 months ago

This is not the correct implementation as for the same cases we get two different answers http://ideone.com/Boy5ot

We have to consider all possible cases of three boxes (6 cases) instead of three .

• Reply • Share ›

aa1992 Abhishek Bind • 7 months ago

i think its because you have taken dimensions as {40,3,1} in one case and {40,1,3} in other case.As mentioned width<=depth but in 1st case its violating the statement as(3>1) therefore this combination is never generated in the 2nd case hence two different answers.

• Reply • Share ›

richa • 8 months ago

why do we need to calculate the max element of msh at end isnt it always the last element ie msh[n-1] ??

• Reply • Share ›

aa1992 richa • 7 months ago

no it is not sorted acc to heights,its sorted only acc to areas.But we need the maximum height hence we need to find max sub sequence of increasing height.

• Reply • Share ›

Ariel Demian • 8 months ago

Here's my implementation: https://github.com/ArielDemian...

• Reply • Share ›

Kartikeya singh • 9 months ago

Isnt' there still any solution for the unique box stacking problem!? I guess the above solution

can't be modified in order to take only a unique instance of each box.

6 ▲ | ▼ • Reply • Share ›

vikyrulz ➦ Kartikeya singh • 25 days ago

How about if we only do msh calculation if jth box is not same as ith box. in Box structre, we can also add box number, so if ith box number is same as jth box number, we skip msh calculation in the nested for loop. Will that work?

▲ | ▼ • Reply • Share ›

PeterRevel • 10 months ago

Anyone got an average compile time for this? Can't tell if my code is taking to0 long

▲ | ▼ • Reply • Share ›

toughtimes • 10 months ago

rather than sorting by area we can sort by width or length..suppose i sort by lenght...then i compare width,if width of j>width of i (j>i) than area will be greater by default!!

▲ | ▼ • Reply • Share ›

Karshit Jaiswal • 10 months ago

Very Beautifully Explained...!!
Thanks guys... :)

using STL sort : http://ideone.com/YQasp4

1 ▲ | ▼ • Reply • Share ›

nikhil12321 • a year ago

why we use LIS after sorting ??
we can determine the maximum height by just simply adding because we can place all sorted boxes one by one as base line of one is shorter than other ..

▲ | ▼ • Reply • Share ›

Vivek ➦ nikhil12321 • a year ago

consider two boxes A and B A= 10 x 1 x 1; B=5 x 3 x 1 even though B's base area is greater than that of B , it can't be placed before A because depth of B is less than that of A

▲ | ▼ • Reply • Share ›

Dante Fan • a year ago

if we only have one instance of every box, how to solve this problem?

▲ | ▼ • Reply • Share ›

Shashwat ➦ Dante Fan • 8 months ago

I would be same. Just remove the rotation part!

▲ | ▼ • Reply • Share ›

Karshit Jaiswal ➚ Dante Fan • a year ago

any one ??
I think den we can simply sort and apply LIS .. but I am not sure.. o.O

1 ^ | v • Reply • Share ›

a ➚ Karshit Jaiswal • 3 months ago

yeah

^ | v • Reply • Share ›

Anick Saha • a year ago

```
/* Compute optimized msh values in bottom up manner */
for (int i = 1; i < n; i++ )
for (int j = 0; j < i; j++ )
if ( rot[i].w < rot[j].w &&
rot[i].d < rot[j].d &&
msh[i] < msh[j] + rot[i].h
)
{
msh[i] = msh[j] + rot[i].h;
}
```

Why the above processs , and why not this.. :

```
/* Compute optimized msh values in bottom up manner */
for (int i = 1; i < n; i++ )
for (int j = 0; j < i; j++ )
if ( (rot[j].w * rot[j].d >
rot[i].w < rot[i].d) &&
msh[i] < msh[j] + rot[i].h
)
{
msh[i] = msh[j] + rot[i].h;
}
```

^ | v • Reply • Share ›

Karshit Jaiswal ➚ Anick Saha • 10 months ago

because we want every edge(dimension) and not just the base area to be bigger..
According to your thinking, why are you even applying LIS, you already have the array in decreasing order of base area.. just add up the height.. (but thats not what the problem wants)..!!
#peace

1 ^ | v • Reply • Share ›

Guest • a year ago

for input :
77 20 30

89 24 36

11 43 70

Expected Ouput : 177
Actual Output : 210

1 • Reply • Share ›

TheGust ➚ Guest • 2 months ago

77 20 30
89 24 36
20 30 77
24 36 89
making a total of 77+89+20+24 = 210
remember you can use a box more than once

• Reply • Share ›

prashant jha • a year ago

my c++ code for box stacking problem
http://ideone.com/CNE9Vk

• Reply • Share ›

Guest • a year ago

i sorted in increasing order .. and the answer is 64 now. why ?

• Reply • Share ›

Karshit Jaiswal ➚ Guest • 10 months ago

reverse the condition in LIS of above code.. dat myte help o.O

• Reply • Share ›

rahul • a year ago

those who are confused with 6 rotations and y 3 are taken.,.please read below.

H*W*D: in the sol they have taken this 3 parameters in decreasing sequence...thats y 3 rotations and theats y w is max and depth is min..

if we want to take increasing sequence then we can take 3 in opposite side..that is width is min and depth is max.

And if we are allowed to take mixture like

for 123
6 permutations are there

for incresing we have 123
2 13
3 12

same for decreasing...if we want to take mixture ,then we need to take all 3 perm.

• Reply • Share ›

alfasin • a year ago

Maybe I'm missing something cause I have an issue with the solution:

If we take the original list (4 boxes): {4, 6, 7}, {1, 2, 3}, {4, 5, 6}, {10, 12, 32} and take the maximum out of each box we'll get: 7+3+6+32=48 so any solution above 48 is not possible. Actually, the solution that we got contains 7 boxes - so we need to remove the "duplicate" boxes (such as {3, 1, 2}, {1, 2, 3} for example) in order to get the *real* solution

1 • Reply • Share ›

Vishnu alfasin • a year ago

You haven't read the problem properly..
"It said that you can use multiple instances of the same type of box"

2 • Reply • Share ›

Tanish Gupta • 2 years ago

can we do it in this way .Let there be 2 arrays of size 3n a and b.first we store 3n possible ways in array , then sorting according to breadth and applying LIS on length and updating array a according to height .similarly sorting according to length and applying LIS on breadth and updating array b according to height.finally finding max(max(a),max(b)).

• Reply • Share ›

geekguy • 2 years ago

@all Who are confused like me , why we need to sort ?

Here the problem is to find the maximum height stack.

Lets say we sort the boxes in the increasing(non decreasing) order of their base area. Now If we apply LIS then we will get the answer =1, because ever box's base area is larger/same than the boxes on its right.

So, To maximize the height of the stack we need to sort then in decreasing base area.

Hope this helps! :)

• Reply • Share ›

ssd geekguy • 2 years ago

i think we both misunderstood it after watching the mit video explanation. The thing is that both the width and depth have to be smaller than the previous one. So a base area j < i th base area doesn't necessarily mean that both the width and depth are smaller than i. It might be one of them smaller than i th but still got smaller base area. We want to sort them in decreasing order so that we can find as many as width and depth strictly smaller boxes during iteration.

• Reply • Share ›

geekguy ssd • 2 years ago

That's what I have written ! :)

• Reply • Share ›

ssd geekguy • 2 years ago

But after the sort, then no need to find LIS since from start to end already the longest....

• Reply • Share ›

geekguy ssd • 2 years ago

We find LIS of strictly decreasing width and depth and not the base area.

Sorting base area is for maximizing our chance !

1 • Reply • Share ›

Jorge Arévalo • 2 years ago

Whoru Zerozero I&#039ve just asked that. I think you&#039re solving the box.com challenge :-)

• Reply • Share ›

Jorge Arévalo • 2 years ago

What if you delete the constraint 3? If you just can use one instance of each box (the "optimum" to maximize the height)? I&#039m trying to solve that problem.

2 • Reply • Share ›

Roy L • 2 years ago

why do we need to sort the box in decreasing order of box area?

• Reply • Share ›

Roy L • 2 years ago

why do we need to sort the box in decreasing order of box area?

1 • Reply • Share ›

Zavatar Roy L • 2 years ago

I think this is a weak order to satisfy width(j) > width(i) and depth(j) > depth(i),which means it isn't possible that having a sequence where two elements have reverse order

in the decreasing order of box area

• Reply • Share ›

alfasin Zavatar • a year ago

base 3x3 is bigger than base 4x2 yet w(j) < w(i)

1 • Reply • Share ›

Zavatar alfasin • a year ago

But 3x3 couldn't be placed on the top of 4x2. I mean that 1. box(j) could be placed on box(i) implies area(j) must be smaller than area(i);
2. box(j) could be placed on box(i) equals width(j) > width(i) and depth(j) > depth(i);

So in the sorted array, if box(j) could be placed on box(i), index j must be bigger than i. In the following LIS algorithm, we aim to find the satisfying box(j), width(j) > width(i) and depth(j) > depth(i).

• Reply • Share ›

rishabh sharma • 2 years ago

Please tell why only 3 cases of rotation are considered because
in a base any of the sides can be used as the depth or the width,
or if box j is to be stacked above box i then any 1 dimension of box j base should be smaller than box i bases any one dimension,and the other dimension correspondingly.

• Reply • Share ›

akki • 2 years ago

how is the rotation of boxes is considered.
say we have 1 * 2 * 3
so if we rotate about any axis that will remain same.
if we rotate about depth axis
then we get 2 * 1 * 3 (depth remained same)
if we rotate about width axis then we must get 3*2*1
width must remain same how we get 3*1*2(all three changed at same time...how is it possible ?
on which axis we have rotated the box that all three get changed.?)

• Reply • Share ›

geekguy akki • 2 years ago

There are 6 possible combinations and we are considering only 3 of them where depth is always smaller than or equal to width.

• Reply • Share ›

Load more comments

- # Popular Posts

  - 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

- Follow @GeeksforGeeks

- # Recent Comments

  - lt_k

    i need help for coding this function in java...

    Java Programming Language · 1 hour ago

  - Piyush

    What is the purpose of else if (recStack[*i])...

    Detect Cycle in a Directed Graph · 1 hour ago

  - Andy Toh

    My compile-time solution, which agrees with the...

    Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago

  - lucy

    because we first fill zero in first col and...

    Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago

  - lucy

    @GeeksforGeeks i don't n know what is this long...

    Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 3 hours ago

  - manish

    Because TAN is not a subsequence of RANT. ANT...

    Given two strings, find if first string is a subsequence of second · 3 hours ago