GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

- Home
- Algorithms
- DS
- GATE
- Interview Corner
- Q&A
- C
- C++
- Java
- Books
- Contribute
- Ask a Q
- About

Array
Bit Magic
C/C++
Articles
GFacts
Linked List
MCQ
Misc
Output
String
Tree
Graph

Dynamic Programming | Set 11 (Egg Dropping Puzzle)

The following is a description of the instance of this famous puzzle involving n=2 eggs and a building with k=36 floors.

Suppose that we wish to know which stories in a 36-story building are safe to drop eggs from, and which will cause the eggs to break on landing. We make a few assumptions:

…..An egg that survives a fall can be used again.
…..A broken egg must be discarded.
…..The effect of a fall is the same for all eggs.
…..If an egg breaks when dropped, then it would break if dropped from a higher floor.
…..If an egg survives a fall then it would survive a shorter fall.
…..It is not ruled out that the first-floor windows break eggs, nor is it ruled out that the 36th-floor do not

cause an egg to break.

If only one egg is available and we wish to be sure of obtaining the right result, the experiment can be carried out in only one way. Drop the egg from the first-floor window; if it survives, drop it from the second floor window. Continue upward until it breaks. In the worst case, this method may require 36 droppings. Suppose 2 eggs are available. What is the least number of egg-droppings that is guaranteed to work in all cases?
The problem is not actually to find the critical floor, but merely to decide floors from which eggs should be dropped so that total number of trials are minimized.

Source: Wiki for Dynamic Programming

In this post, we will discuss solution to a general problem with n eggs and k floors. The solution is to try dropping an egg from every floor (from 1 to k) and recursively calculate the minimum number of droppings needed in worst case. The floor which gives the minimum value in worst case is going to be part of the solution.
In the following solutions, we return the minimum number of trails in worst case; these solutions can be easily modified to print floor numbers of every trials also.

1) Optimal Substructure:
When we drop an egg from a floor x, there can be two cases (1) The egg breaks (2) The egg doesn't break.

1) If the egg breaks after dropping from xth floor, then we only need to check for floors lower than x with remaining eggs; so the problem reduces to x-1 floors and n-1 eggs
2) If the egg doesn't break after dropping from the xth floor, then we only need to check for floors higher than x; so the problem reduces to k-x floors and n eggs.

Since we need to minimize the number of trials in *worst* case, we take the maximum of two cases. We consider the max of above two cases for every floor and choose the floor which yields minimum number of trials.

```
  k ==> Number of floors
  n ==> Number of Eggs
  eggDrop(n, k) ==> Minimum number of trails needed to find the critical
                    floor in worst case.
  eggDrop(n, k) = 1 + min{max(eggDrop(n - 1, x - 1), eggDrop(n, k - x)):
                 x in {1, 2, ..., k}}
```

2) Overlapping Subproblems
Following is recursive implementation that simply follows the recursive structure mentioned above.

```
# include <stdio.h>
# include <limits.h>

// A utility function to get maximum of two integers
int max(int a, int b) { return (a > b)? a: b; }

/* Function to get minimum number of trails needed in worst
  case with n eggs and k floors */
int eggDrop(int n, int k)
{
    // If there are no floors, then no trials needed. OR if there is
    // one floor, one trial needed.
    if (k == 1 || k == 0)
        return k;

    // We need k trials for one egg and k floors
    if (n == 1)
        return k;

    int min = INT_MAX, x, res;

    // Consider all droppings from 1st floor to kth floor and
    // return the minimum of these values plus 1.
    for (x = 1; x <= k; x++)
    {
        res = max(eggDrop(n-1, x-1), eggDrop(n, k-x));
        if (res < min)
            min = res;
    }

    return min + 1;
}

/* Driver program to test to pront printDups*/
int main()
{
    int n = 2, k = 10;
    printf ("\nMinimum number of trials in worst case with %d eggs and "
             "%d floors is %d \n", n, k, eggDrop(n, k));
    return 0;
}
```

```
Output:
Minimum number of trials in worst case with 2 eggs and 10 floors is 4
```

It should be noted that the above function computes the same subproblems again and again. See the following partial recursion tree, E(2, 2) is being evaluated twice. There will many repeated subproblems when you draw the complete recursion tree even for small values of n and k.

```
                         E(2,4)
                           |
          -------------------------------------
          |             |           |         |
          |             |           |         |
      x=1/\          x=2/\      x=3/ \    x=4/ \
        /  \           /  \       ....      ....
       /    \         /    \
 E(1,0)  E(2,3)     E(1,1)  E(2,2)
          /\  /\...         /  \
      x=1/  \               .....
        /    \
     E(1,0)  E(2,2)
            /   \
            ......

Partial recursion tree for 2 eggs and 4 floors.
```

Since same suproblems are called again, this problem has Overlapping Subprolems property. So Egg Dropping Puzzle has both properties (see this and this) of a dynamic programming problem. Like other typical Dynamic Programming(DP) problems, recomputations of same subproblems can be avoided by constructing a temporary array eggFloor[][] in bottom up manner.

Dynamic Programming Solution

Following is C/C++ implementation for Egg Dropping problem using Dynamic Programming.

```
# include <stdio.h>
# include <limits.h>

// A utility function to get maximum of two integers
int max(int a, int b) { return (a > b)? a: b; }

/* Function to get minimum number of trails needed in worst
  case with n eggs and k floors */
int eggDrop(int n, int k)
{
    /* A 2D table where entery eggFloor[i][j] will represent minimum
       number of trials needed for i eggs and j floors. */
    int eggFloor[n+1][k+1];
    int res;
    int i, j, x;

    // We need one trial for one floor and0 trials for 0 floors
    for (i = 1; i <= n; i++)
    {
        eggFloor[i][1] = 1;
        eggFloor[i][0] = 0;
    }

    // We always need j trials for one egg and j floors.
    for (j = 1; j <= k; j++)
        eggFloor[1][j] = j;

    // Fill rest of the entries in table using optimal substructure
    // property
    for (i = 2; i <= n; i++)
    {
        for (j = 2; j <= k; j++)
        {
            eggFloor[i][j] = INT_MAX;
            for (x = 1; x <= j; x++)
            {
                res = 1 + max(eggFloor[i-1][x-1], eggFloor[i][j-x]);
                if (res < eggFloor[i][j])
                    eggFloor[i][j] = res;
            }
        }
    }

    // eggFloor[n][k] holds the result
    return eggFloor[n][k];
}

/* Driver program to test to pront printDups*/
int main()
{
    int n = 2, k = 36;
    printf ("\nMinimum number of trials in worst case with %d eggs and "
             "%d floors is %d \n", n, k, eggDrop(n, k));
    return 0;
}
```

```
Output:
Minimum number of trials in worst case with 2 eggs and 36 floors is 8
```

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

As an exercise, you may try modifying the above DP solution to print all intermediate floors (The floors used for minimum trail solution).

References:
http://archive.ite.journal.informs.org/Vol4No1/Sniedovich/index.php

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

Tweet 1 g+1 3

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

64 Comments GeeksforGeeks 1 Login

Recommend Share Sort by Newest

Join the discussion…

Mission Peace • a month ago

https://www.youtube.com/watch?... Check out my video on same problem

2 ^ | ˅ • Reply • Share ›

lucy ➦ Mission Peace • a month ago

nice....... :) :) very helpful......

^ | ˅ • Reply • Share ›

lucy • a month ago

in dynamic solution res = 1 + max(eggFloor[i-1][x-1], eggFloor[i][j-x]); why max??????????????

^ | ˅ • Reply • Share ›

Siya ➦ lucy • a month ago

Because we want to know minimum number of trials in the worst case that is why we are taking maximum. It means what ever the case the Minimum number of trials in worst case with 2 eggs and 36 floors is 8.It means trials can be less than 8 but it can never be greater than 8.

1 ^ | ˅ • Reply • Share ›

lucy ➦ Siya • a month ago

thanks Siya but i have one more doubt that why "for (x = 1; x <= j; x++)" this loop for every floor......

^ | ˅ • Reply • Share ›

Siya ➦ lucy • a month ago

Its because of optimal substructure. Read this article for more clear view of it.

http://www.geeksforgeeks.org/d...

1 ^ | ˅ • Reply • Share ›

lucy • a month ago

why in recursive is taking max

^ | ˅ • Reply • Share ›

Guest • 3 months ago

i think we can solve this problem using binary search...

^ | ˅ • Reply • Share ›

Siya ➦ Guest • a month ago

We could solve if we have infinite amount of eggs but in this case eggs are limited to 2.

^ | ˅ • Reply • Share ›

Bandi Sumanth • 4 months ago

Bandi Sumanth • 4 months ago

"Minimum number of trials in worst case with 2 eggs and 10 floors is 4"
I am not able to understand how with only 8 trails in the worst case, one can determine the critical floor ? Can somebody pls explain with an example ? It would be of great help

• Reply • Share ›

Siya ➦ Bandi Sumanth • a month ago

Read this they have explained very well.

http://www.datagenetics.com/bl...

• Reply • Share ›

lucy ➦ Siya • a month ago

in dynamic solution res = 1 + max(eggFloor[i-1][x-1], eggFloor[i][j-x]); why max??????????????

• Reply • Share ›

Bandi Sumanth ➦ Siya • a month ago

Thank you... that was very helpful :) :)

• Reply • Share ›

kyleyu • 6 months ago

I can understand the meaning of DP formula. But still has a problem: How can we decide min(max(E(n-1,x-1),E(n,k-x))? For example, if at some stage x==8 is optimal, but when we calculate max(E(n-1,5-1),E(n,k-5)), the egg has already been broken at x=5, that means, even x==8 is optimal, but we cannot know because the egg already breaks at x=5. So, the problem here is not the DP formula, but how to do it given the conditions of this problem.

• Reply • Share ›

kyleyu ➦ kyleyu • 6 months ago

ok, I understand it. Because we only consider the worst case

• Reply • Share ›

CanHelp • 7 months ago

Detailed generic solution : http://www.writeulearn.com/3-e...

• Reply • Share ›

prasahnth • 7 months ago

could someone clarify on this... consider E(2,4) , 2 eggs and 4 floors, in the second iteration i.e x = 2, number of trials would be 1+(max (E(1,1), E(2,2)) .. the second term E(2,2) means the egg did not break and trail has to go upward i.e should check for floor 3 first and then with floor 4. but i understand E(2,2) evaluates the case of 2 eggs with 2 floors... plz help if my understanding is wrong... thanks,

• Reply • Share ›

Ajay Sreeram • 8 months ago

for (j = 1; j <= k; j++)
eggFloor[1][j] = j;

this have to be replaced by

for (j = 2; j <= k; j++)
eggFloor[1][j] = j;

because eggFloor[1][1] is assigned value 1 in twice

1 ⌃ | ⌄ • Reply • Share ›

Jun • 10 months ago

I think the recursive approach has a flaw.....it doesn't give correct answer, or answers for that matter, in many cases.example n=2 and k=100

⌃ | ⌄ • Reply • Share ›

vipinkaushal ➦ Jun • 8 months ago

it does not have any flaw , it's iterative version so time complexity is expo and it gives time limit exceeded and that's why we required dp solution

1 ⌃ | ⌄ • Reply • Share ›

anonymous • 10 months ago

Why do we do +1? why are we adding 1? I am not able to see it.

⌃ | ⌄ • Reply • Share ›

Guest ➦ anonymous • 10 months ago

+1 is for the drop at xth floor.

1 ⌃ | ⌄ • Reply • Share ›

guest ➦ Guest • 7 months ago

so what exactly is min representing?

⌃ | ⌄ • Reply • Share ›

shubhangi ➦ guest • 6 months ago

why max is used i am not able to see it please help

⌃ | ⌄ • Reply • Share ›

AlienOnEarth • a year ago

The recursive solution does not work for n=2, k =100

⌃ | ⌄ • Reply • Share ›

vipinkaushal ➦ AlienOnEarth • 8 months ago it does not work because it's iterative version so time complexity is expo and it gives time limit exceeded and that's why we required dp solution

• Reply • Share ›

typing.. • a year ago

for n=2 and k=7, how can be the answer is 3.. i don't understand. I think it would be 4. plz help me out....

• Reply • Share ›

typing.. typing.. • a year ago

sorry,, m asking for n='3' and k=7, how it is 3?

• Reply • Share ›

Akash Agrawal • a year ago

Here is a generalized solution for any number of eggs:

http://tech-queries.blogspot.i...

• Reply • Share ›

Venu Gopal Akash Agrawal • a year ago

TCS ke bahut maje le rahe ho :P

• Reply • Share ›

Akash Agrawal Venu Gopal • a year ago

???

• Reply • Share ›

Venu Gopal Akash Agrawal • a year ago

Test ka result aaya ki nahi abhi tak

• Reply • Share ›

Tera baap Venu Gopal • 3 months ago

@venu gopal : aap chutiye rahenge

• Reply • Share ›

ameya • 2 years ago

This problem can be solved in O(1) using just a simple formula:
seal(sqrt(2*n)-0.5) // n = number of floors in building

```
#include "stdio.h"
#include "math.h"

int main()
{
```

```
int n,ans;
float x;

Printf("Enter Number of floors in building: ");
scanf("%d",&n);
x=sqrt(2*n)-0.5;
ans=x;
if(x>ans)
++ans;
printf("%d\n",ans);

return 0;
}
```

1 ⌃ | ⌄ • Reply • Share ›

Code_Addict ➦ ameya • a year ago

Thanks for sharing above formula for cross-checking the answer, but can you post some link for its prove that minimum number of trials in worst case is independent of number of eggs(except for case when number of eggs is 1).

⌃ | ⌄ • Reply • Share ›

ameya ➦ Code_Addict • a year ago

http://www.datagenetics.com/bl...

4 ⌃ | ⌄ • Reply • Share ›

Code_Addict ➦ ameya • a year ago

Thanks for link!

⌃ | ⌄ • Reply • Share ›

Code_Addict ➦ Code_Addict • a year ago

Formula above gives wrong result for case:
no. of eggs=3 and no. of floors = 92.
By using DP: answer is 8
By using formula: answer is 14

⌃ | ⌄ • Reply • Share ›

Mihir ➦ ameya • a year ago

What you call a 'Simple Formula' has a derivation behind it. If this is an interview question indeed, you'll either have to mathematically prove this formula first or do the dynamic programming solution.

⌃ | ⌄ • Reply • Share ›

anju ➦ Mihir • a year ago

r u trying to say something similar to this post::

http://bit.ly/1biTdXo

⌃ | ⌄ • Reply • Share ›

Karshit Jaiswal ↪ anju • a year ago

fake link... Spammer..!!

⌃ | ⌄ • Reply • Share ›

prakash • 2 years ago

plz clarify my doubt DP version of this problem solution,
why we need 3 for loops?(.ie,x=1;x<=j;x++)

since this is overlapping sub problem,
res = 1 + max(eggFloor[i-1][x-1], eggFloor[i][j-x]);
itself will give min &#039res&#039 for eggfloor[i][j]
why are we doing it in for loop?

⌃ | ⌄ • Reply • Share ›

AMIT • 2 years ago

also wikipedia gives various other methods to modify time complexity upto nlogk

http://en.wikipedia.org/wiki/D...

⌃ | ⌄ • Reply • Share ›

rajx • 2 years ago

```
int rj_eggDrop(int eggs, int floors)
{
if(eggs==1)return floors;
int i, step=2;
int arr1[eggs], arr2[eggs];
for(i=0;i<eggs;i++)arr1[i]=1;
int *v1=arr1, *v2=arr2, *tmp;
while(1)
{
v2[0] = step;
for(i=1;i= floors)
return step;
}
step++;
tmp = v1;v1=v2;v2=tmp;
}
}
```

Time Complexity: O(floors)
Auxiliary Space: O(eggs)

1 ^ | ˅ • Reply • Share ›

brahma ➦ rajx • 10 months ago

could you elaborate this solution... ?

^ | ˅ • Reply • Share ›

kavita • 2 years ago

@venki how u arrived at this solution:General solution - n * k^(1\n), where n is # eggs, k is # of floors.

I am not getting results. please explain.

^ | ˅ • Reply • Share ›

ronny • 2 years ago

Can anyone explain this problem for k eggs.

^ | ˅ • Reply • Share ›

Niks • 2 years ago

What is the time complexity of recursive solution??

```
/* Paste your code here (You may delete these lines if not writing code) */
```

^ | ˅ • Reply • Share ›

AG • 2 years ago

x is out of scope in the line :
int min = INT_MAX, x, res;
of the recursive solution.

^ | ˅ • Reply • Share ›

leet • 3 years ago

How to do the exercise that is to print the floors? Please give some guidance.

```
/* Paste your code here (You may delete these lines if not writing code) */
```

^ | ˅ • Reply • Share ›

Load more comments

✉ Subscribe Ⓓ Add Disqus to your site ▷ Privacy

- Interview Experiences
- Advanced Data Structures
- Dynamic Programming
- Greedy Algorithms
- Backtracking
- Pattern Searching
- Divide & Conquer
- Mathematical Algorithms
- Recursion
- Geometric Algorithms

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

-