Question 6.8 from CTCI: Egg Drop

September 27, 2017

Question

There is a building of 100 floors. If an egg drops from the Nth floor or above, it will break. If it is dropped from any floor below, it will not break. You're given 2 eggs. Find N, while minimizing the number of drops for the worst case.

Explanation and Algorithm

We can do this in two ways:

For the brute force solution, let's drop one of the 2 eggs first. We will only use the second egg after the first has been broken. We will use the first egg to determine the subset of floors that we will drop the second egg from to find N.

For example, let us drop the first egg from every 10th floor. If the egg breaks on the 10th floor, we know that N must be less than or equal to 10 since an egg dropped from the Nth floor or above will break. We can incrementally drop the second egg from the 1st floor, 2nd floor, 3rd floor, etc. to find N. If the second egg does not break when dropped from floors 1 - 9, then N = 10. However, if the first egg does not break on the 10th floor, then we know that N must be greater than 10 since the eggs don't break when dropped from floors below N. We can then drop the first egg from the 20th floor. If it breaks when dropped from that floor, we can then incrementally drop the second egg from the 11th floor, 12th floor, etc. to find N. We can continue this process of dropping the first egg every 10 floors until the first egg breaks and then incrementally checking the 9 floors below, to find N.

In the worst case scenario, we must incrementally check every 10th floor until we reach the 100th floor. If the first egg breaks on the 100th floor, then we must check where the second egg drops from floors 91 - 99. At this point, we have checked the 10th, 20th, 30th,..., 100th floor as well as the 9 floors between the 90th and 100th floor. Thus, in the worst case, we have to check 19 floors in total.

For the optimal solution, however, we want to implement our strategy in such a way that the most drops required is consistent no matter which drop our

first egg breaks on. So Drops of Egg1 + Drops of Egg2 should be the same. So for each floor that we drop the first egg from, we should reduce the number of floors that Egg 2 must be dropped from by 1. For example, when we drop the first egg from floor 20 and then from floor 30, Egg 2 must be dropped from the 9 floors between floor 20 and floor 30. When we next drop the first egg, we want to make sure that the second egg has to check 8 floors. So we must drop the first egg from floor 39 instead of 40. Egg 2 would now have to check the 8 floors between floor 30 and 39. So Egg 1 must be dropped first from floor X, then floor $X-1$, then $X-2$, etc. We obtain the following equation: $X + (X-1) + (X-2) + (X-3) + ... + 1 = 100$. Solving for X, we obtain $X(X+1)/2 = 100$. We then get $X = 14$. So we should initially drop the first egg from floor 14. It will take 14 steps to determine the floor N in the worst case.

Hints

1. Consider using only a single egg first. How might you strategize dropping the egg to minimize the number of times you have to drop the second egg?
2. Let's say you drop the first egg from every 10th floor. If the egg breaks on one of these floors, what does that reveal about the value of N? How could you use the second egg to then determine N?
3. How might we optimize the number of drops in the worst case? (Think how to make the total number of drops consistent).

Sources

Question, answer and other material taken from Cracking the Coding Interview 6th edition by Gayle Laakmann McDowell.