Exercises 19 : "average case"

Questions

1. Here are a few questions about amortized analysis.

   a) In Java, the ArrayList class is an implementation of a list class that uses an underlying array. One nice property of a Java ArrayList is that the user doesn't need to take care of resizing the array when it gets full. For example, when a user calls the ArrayList.add() method, the method checks if the array is full and, if it is, it creates a new larger array (say, twice the size) and all the elements are copied from the old to new array.
   Suppose a Java ArrayList is initialized such that the underlying array has size 1 and suppose n elements are added to the list where n might be a very large number. For simplicity, suppose you add to the end of the list so that each add is O(1) *when there is room in the array*. What is the average (i.e. amortized) cost per element that is added to the list? Is it O(1) as you would hope? Or is it slower than that, because of the array resizing that must be done[1].

   b) Suppose you add n entries i.e. (key, value) pairs to a hashtable. Suppose you use the rule that you increase the size of the underlying array whenever the number of elements in the table is equal to the number of buckets (slots) in the table. As in the ArrayList example, this resizing is extra work. What is the average (amortized) cost per entry in the table, when you add n entries?

   c) Why is the O(n) performance of the fast buildheap method an example of amortization?

2. Recall the birthday problem from the lecture on hashing. There are 365 days in the year which we number {1, 2, ... , 365}. We ignore leap years. Assume the probabilities of each birthday are equal. Suppose we consider the birthdays of n people.

   a) What is the sample space ?

   b) Given n people, what is the expected number of pairs of people that have the same birthday? Assume that for any pair, the probability that they have the same birthday is 1/365 i.e. no twins present!

3. Suppose you have the numbers 1 to n and you consider randomly chosen ordering of these numbers.

   a) What is the sample space?

1

b) Consider each pair of numbers (i,j) where i < j. For any ordering, these two numbers might or might not be in the correct order, that is, i might come before j or after j in the ordering. If j comes before i, then we say that pair (i,j) is *inverted*. What is the expected value of the number of pairs that is inverted in a (uniformly) randomly chosen ordering ? Use the linearity of expectation.

4. What is the expected value of the number of times that a 5 appears when you roll one die 4 times?

5. What is the expected number of times that we need to roll a di until we roll a 1? (You will need a result from lecture 21 for this.)

Answers

1.

a) Suppose 1 operation is required for adding an element to the array when there is room, and 2^k operations are required when we resize the array for the kth time, namely when we resize we need to copy all the elements from the array of size 2^k to the array of size 2^{k+1}. The total work done therefore has two components: (a) n for adding each elements for the first time, and (2) the copies from smaller array to larger array. The second component has work:

sum_{ k = 0 to log n - 1} 2^k
= (2^{log n + 1} – 1)/(2 – 1)
= n – 1

So, the total amount of work is n + (n – 1) which is still O(n). Thus, ArrayList resizing has no effect in the O() sense. It is extra work and sometimes a lot of extra work but this is compenstated by the fact that it happens only rarely.

b) It's the same argument as in a). Every time you double the number of entries, you double the size of the table and remap (re-hash) all the entries to the new table.

c) For elements near the front of the array (which holds the initial list), you need to downHeap a distance of approximately log n. However, there are only a small number of these elements. When you add up the contribution of all elements and take the average, these bad cases can be ignored. The algorithm is O(n), or O(1) on average per element.

2.

a) The sample space is the set of n-tuples, where each element is a number from 1 to 365. There are 365^n possible outcomes in this sample space.

b) Let X_ij =1 if persons i and j have the same birthday where i < j, and X_ij = 0 if they don't have the same birthday. Using linearity of expectation, the expected value of the number of pairs of people that have the same birthday is the sum of the expected value of X_ij over all pairs (i,j) where i < j. The expected value of X_ij is 1/365 for each i,j, that is, E X_ij = 1*1/365 + 0*364/365. Summing over all n*(n-1)/2 people gives n*(n-1)/2/365.

3. [modified April 8.]
   a) There are n! possible orderings. That is the sample space.
   b) The informal answer is that "half the pairs are inverted on average". Such informal answers are fine for many questions. Here is a more formal answer. Let X_ij be a random variable that is 1 if the pair (i,j) is inverted and 0 otherwise. The number of inverted pairs in any randomly chosen ordering is Y = sum_{ i < j } X_ij. For any (i,j,) p(X_ij = 1) = 0.5 since the pair is equally likely to be inverted as not. There are n(n-1)/2 pairs (i,j) where i < j. The expected number of inverted pairs is the sum of E{ X_ij } over all i, j where i < j.
   Thus, E(X) = sum_{ (i,j) i < j } E X_ij.
   Now, E X_ij = sum_{ (i,j) i < j } (1 * p(X_ij = 1) + 0 * p(X_ij = 0)) = 1*½ + 0* ½ = ½.
   So, E(X) = sum_{ (i,j) i < j } E X_ij = n(n-1)/2 / 2.

4. Use linearity of expectation. For each throw, the probability is 1/6 that a 5 appears. Letting Xi =1 when a 5 appears on toss i, and Xi = 0 when it doesn't appear. The number times that a 5 appears in 4 tosses is the sum of Xi for i in 1 to 4. The expected value of Xi is 1/6. Applying linearity of expectation, the sum of expected values of Xi for i=1 to 4 is 4/6. So the answer is 2/3.

5. Suppose we roll it (i-1) times without a 1 coming up and then on the ith time a 1 comes up, that is, X = i. The probability of this event is (5/6)^(i-1) * 1/6. Note that this is just a Bernoulli trial where p0 = 5/6. As shown in lecture 21, the expected value of X will be 1/(1-p0) which is 6 in the case p0 = 5/6.