HW 7

Yunhao Lin

November 13, 2018

1. Calling RANDY(l,u) would return (i, a_i) where i is a uniformly integer from l to u and a_i is the ID of corresponding file. Suppose we have a number for all of the file from 1 to n, 0 means it's not possible, 1 means it's visited and -1 means not yet determined. The initial value is -1 means not determined. According to the algorithm, if the file is visited, it would be compared to x to check if it's the desired file and then return. So what we need to check is only the possbility for some file in the list to be visited.
   Assume we have index j and $A[j] \leq x < A[j+1]$. Then our goal is to find j. After we get a_i and i from the RANDY() call, if $x < a_i$ we mark all the the files on the right of i to 0, else mark all the files on the left to be 0. So for each file the possiblity to be visited is $\frac{1}{i-j}$ when $i > j$ and $\frac{1}{j-i+1}$ when $i \leq j$

$$p(A[i] == 1) = \begin{cases} \frac{1}{j-i+1}, (i \leq j) \\ \frac{1}{i-j}, (i > j) \end{cases}$$

   So our possiblity to find j is then $\sum_{i=1}^{j} \frac{1}{j-i+1} + \sum_{i=j+1}^{n} \frac{1}{i-j}$. Which is smaller than $\sum_{i=1}^{n} \frac{1}{i} + \sum_{i=1}^{n} \frac{1}{i}$. By the harmonic series, $\sum_{i=1}^{n} \frac{1}{i}$ is eqaul to O(logn). So our algorithm would find the correct index for x in $2O(logn) = O(logn)$ accesses of files.

2. - In order to determine the expected length of any single arc. Since we are picking k points on the circle. Then at the end we will have k arcs and the sum of their expectation would be 1, since this is a circle of unit circumference. Suppose each arc is X_i, then we get the formula $\sum_{i=1}^{k} E[X_i] = 1$. From here we can simply see that the expected value for sum X_i is going to be $\frac{1}{k}$. So we get $E[X] = \frac{1}{k}$. The expected value of any single arc is $\frac{1}{k}$.
   - In order to find the file with ID x that makes at most $O(\sqrt{n})$ calls. We need to find the closest bound on both end of the file with ID x.

Algorithm 1 Find file x

```
1:  Initialize small = 0;                          ▷ The closest index on the left
2:  Initialize large = inf;                        ▷ The closest index on the right
3:  for i = 1 to √n do
4:      r = RANDY();
5:      if r == x then return x;
6:      if r > small and r < x then small = r;
7:      if r < large and r > x then large = r;
8:  for j = small to large do
9:      if NEXT(j) == x then
10:         return x;
```

   Here we first call $\sqrt{n}$ times of the RANDY(). And every time we can find a index either larger than x or smaller than x. Then we are updating the bound on the lower side and higher side in order to get the bound close to x. If we find x from Call RANDY(), we just return right away. If not we will still get the bound on lower side and higher side. And after we get the bound on both side, we then call NEXT() from the lower bound to the higher bound. Since the list itself is sorted, we are sure our File with index x is in this range. And we would be able to find it.

Time Complexity: From part(a) we know that the expected arc length of the any arc would be $\frac{1}{k}$, where k stands for the number of slices. In the first for loop, basically what we did is to choose random number in the circle of the file. And therefore the length between *small* and *large*, the range our File x is, is going to be $\frac{n}{\sqrt{n}} = \sqrt{n}$. And we call the first loop $\sqrt{n}$ times and in each iteration all we did is compare and RANDY(). The time complexity for the first loop would be $O(\sqrt{n})$. The second for loop would Call NEXT() on this range for $\sqrt{n}$ times and the time complexity would be $O(\sqrt{n})$. So in total, time complexity of the algorithm is $O(\sqrt{n}) + O(\sqrt{n}) = 2O(\sqrt{n}) = O(\sqrt{n})$.