Seminar 4
Algoritmi probabiliști. Complexitatea medie. Continuare

Ștefan Ciobâcă, Dorel Lucanu
Universitatea "Alexandru Ioan Cuza", Iași

March 15, 2022

Hint: multe formule utile (e.g. pentru serii, integrale, combinări, aranjamente, limite, probabilități) în informatica teoretică sunt disponibile aici: `https://www.tug.org/texshowcase/cheat.pdf`.

1. Fie următorul algoritm probabilist, care primește la intrare un număr natural n și un vector v de dimensiune n care conține o permutare aleatorie a numerelor de la 0 la $n-1$:

   ```
   for (i = 0; i < n; ++i) {
     if (v[i] % 2 == 0) {
       break;
     }
   }
   return 0;
   ```

   De notat că natura probabilistă a algoritmului este dată de intrarea aleatorie. Presupunând că fiecare permutare este echiprobabilă (este dată ca intrare cu aceeași probabilitate), care este timpul mediu de execuție al algoritmului pentru date de intrare de dimensiune n?

2. Fie următorul algoritm probabilist, care primește la intrare un număr natural n și un vector v de dimensiune n care conține o permutare aleatorie a numerelor de la 1 la n:

   ```
   max = v[0];
   sum = 0;
   for (i = 0; i < n; ++i) {
     if (v[i] > max) {
       max = v[i];
       for (j = 0; j < n; ++j) { // bucla for interioara
         sum = sum + 1;
       }
     }
   }
   return sum;
   ```

   Presupunând că fiecare permutare este echiprobabilă, care este timpul mediu de execuție al algoritmului pentru date de intrare de dimensiune n? Hint: calculați numărul mediu de execuții ale buclei `for` interioare.

3. Fie următorul algoritm probabilist, care primește la intrare un număr natural n și un vector v de dimensiune n care conține o permutare aleatorie a numerelor de la 1 la n:

```
sum = 0;
for (i = 0; i < n - 1; ++i) {
  for (j = i + 1; j < n; ++j) {
    if (v[i] > v[j]) {
      for (k = 0; k < n; ++k) { // bucla for interioara
        sum = sum + 1;
      }
    }
  }
}
return sum;
```

Presupunând că fiecare permutare este echiprobabilă, care este timpul mediu de execuție al algoritmului pentru date de intrare de dimensiune n? Hint: calculați numărul mediu de execuții ale buclei `for` interioare.

4. Rezolvați cele 3 probleme precedente pentru cazul în care vectorul v conține pe poziția i un număr natural între 0 și $n-1$, fiecare număr cu aceeași probabilitate, iar fiecare astfel de vector este echiprobabil (deci v nu mai conține neapărat o permutare).

5. În acest exercițiu vom proiecta un algoritm probabilist care primește la intrare un număr n și produce la ieșire o permutare de ordin n, fiecare permutare fiind echiprobabilă.

   (a) Arătați că algoritmul următor nu este corect:

   ```
   for (i = 0; i < n; ++i) {
       p[i] = i;
   }

   for (i = 0; i < n; ++i) {
       uniform j from [0..n-1];
       temp = p[i];
       p[i] = p[j];
       p[j] = temp;
   }
   ```

   în sensul în care nu generează fiecare permutare cu aceeași probabilitate. Hint: considerați toate rezultatele posibile pentru $n = 3$ și calculați probabilitatea fiecăruia. Se recomanda să se testeze fiecare dintre algoritmii prezentați.

   (b) Fie următorul algoritm (Fisher-Yates):

   ```
   fisher_yates(n) {
       used = [ 0 | i from [1..n]];
       notused = n;
       for (i = 0; i < n; ++i) {
           k = uniformNat(notused);
           j = 0;
           while (j < n) {
               if (used[j] == 0) {
                   if (k == 0) {
                       break;
                   }
                   k = k - 1;
               }
   ```

```
            j = j + 1;
        }
        p[i] = j;
        used[j] = 1;
        notused = notused - 1;
    }
    return p;
}
```

Arătați că algoritmul întoarce fiecare permutare de ordin n cu aceeași probabilitate. Care este complexitatea algoritmului?

(c) Fie următorul algoritm (Fisher-Yates optimizat):

```
fisher_yates(n) {
    for (i = 0; i < n; ++i) {
            p[i] = i;
    }
    for (i = 0; i < n; ++i) {
        //j = random(i + 1);
        uniform j from [0..i];
        temp = p[i];
        p[i] = p[j];
        p[j] = temp;
    }
    return p;
}
```

Arătați că, după a j-a iterație a algoritmului (pentru $j \in \{1, \ldots, n\}$), vectorul p conține pe primele j poziții (adică pe pozțiile $0, \ldots, j-1$) fiecare permutare a numerelor $0, \ldots, j-1$ cu aceeași probabilitate. Care este complexitatea algoritmului?

6. Fie următorul algoritm probabilist pentru alegerea minimului dintr-un tablou:

```
p = fisher_yates(n);
min = a[p[0]];
sum = 0;
for (i = 1; i < n; ++i) {
  if (a[p[i]] < min) {
    min = a[p[i]];
    for (j = 0; j < n; ++j) { // bucla for interioara
      sum = sum + 1;
    }
  }
}
print(sum);
```

(a) Executați de câteva ori algoritmul.

(b) Care este timpul mediu de execuție al algoritmului de mai sus, când tabloul a de la intrare conține n elemente distincte? Hint: de câte ori se execută bucla `for` interioară?