Descriere soluţíe – secvpal

Prof. Nodea Eugen
Colegiul Naţional ”Tudor Vladimirescu”, Tg-Jiu

Determinarea secvenţei palindromice de lungime maximă pentru şirul `S`, `n = strlen(S)`:

Soluţia 1:

Abordarea recursivă `O(n^2)`

Notăm `lmax(i,j)` = lungimea maximă a secvenţei palindromice `S[i] … S[j]`

`lmax(i,i) = 1` pentru `i = 1, …, n`

$$lmax = \begin{cases} lmax(i+1, j-1) + 2, \ dacă\ S[i] = S[j] \\ \max\left(lmax(i+1, j), lmax(i, j-1)\right), altfel \end{cases}$$

Rezultatul se găseşte în `lmax(1,n)`.

O astfel de implementare este în schimb păguboasă datorită limitelor de memorie impusă.

Soluţia 2: `O(n^2)`

Ideea are la bază alegerea **`centrului`** secvenţei palindromice şi extinderea secvenţei spre stânga sau dreapta cât este posibil.

```
centru = i; // i=1,2..,n
st = dr = centru;
while(s[st] == s[dr] && st>=0 && dr<n)
{
      k += 2; // lungimea secventei
      --st; ++dr;
}
```

Nu trebuie neglijat cazul în care secvenţa are un număr par de caractere. `st = centru; dr = centru + 1;`

În realitate complexitatea algoritmului tinde spre : `O(nlogn)`

O implementare ce foloseşte această idee obţine `100 p`

Soluţia 3: `O(n)`

Există şi 0 soluţie care combină ideea din soluţia anterioară cu prelucrări derivate din suffix arrays (**algoritmul Manacher** - http://en.wikipedia.org/wiki/Longest_palindromic_substring),

Dar pentru a obţine `100 p` nu este necesară o astfel de abordare.

Determinarea numărului de secvenţe în timp liniar presupune o abordare de tip coadă.

Soluţia 4:

O soluţie **`brute-force`** obţine `20p`