Proiectarea Algoritmilor

Curs 3b – Programare dinamică

Alt exemplu: Arbori optimi la căutare (AOC)

- Def 2.1: Fie K o mulțime de chei. Un arbore binar cu cheile K este un graf orientat și aciclic A = (V,E) a.î.:
  - Fiecare nod $u \in V$ conține o singură cheie cheie(u) $\in$ K iar cheile din noduri sunt distincte.
  - Există un nod unic $r \in V$ a.î. i-grad(r) = 0 și $\forall u \neq r$, i-grad(u) = 1.
  - $\forall u \in V$, e-grad(u) ≤ 2; S(u) / D(u) = succesorul stânga / dreapta.

- Def 2.2: Fie K o mulțime de chei peste care există o relație de ordine $\prec$. Un arbore binar de căutare satisface:
  - $\forall u,v,w \in V$ avem ($v \in S(u)$ => cheie(v) $\prec$ cheie(u)) $\wedge$ ($w \in D(u)$ => cheie(u) $\prec$ cheie(w))

Căutare într-un arbore de căutare

- Caută(elem, Arb)
  - **Dacă** Arb = null
    - **Întoarce** null
  - **Dacă** elem = Arb.val // valoarea din nodul crt.
    - **Întoarce** Arb
  - **Dacă** elem $\prec$ Arb.val
    - **Întoarce** Caută(elem, Arb.st)
  - **Întoarce** Caută(elem, Arb.dr)

Complexitate: $\Theta(\log n)$

Inserţie într-un arbore de căutare

- Inserare(elem, Arb)
  - **Dacă** Arb = vid // adaug cheia în arbore
    - nod_nou(elem, null, null)
  - **Dacă** elem = Arb.val // valoarea există deja
    - **Întoarce** Arb
  - **Dacă** elem $\prec$ Arb.val
    - **Întoarce** Inserare(elem, Arb.st) // adaugă în stânga
    - **Întoarce** Inserare(elem, Arb.dr) // sau în dreapta

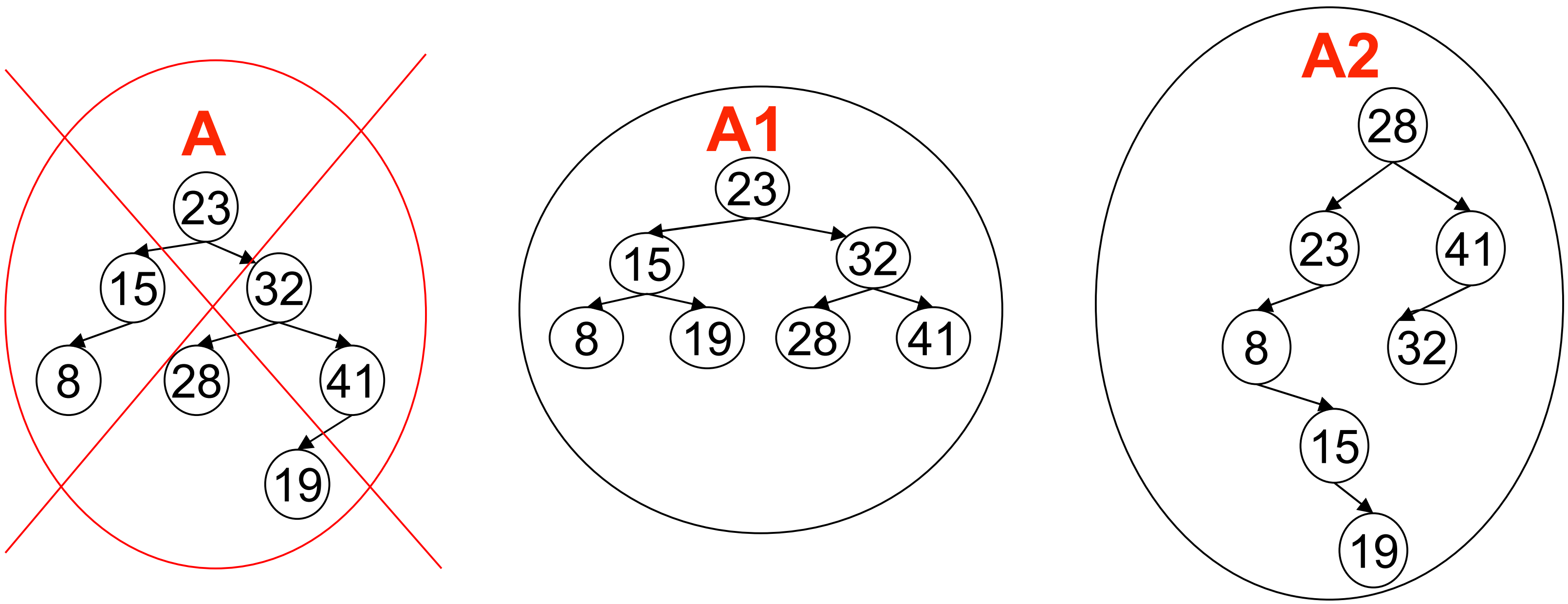
nod Stânga

nod Dreapta

Complexitate: Θ(logn)

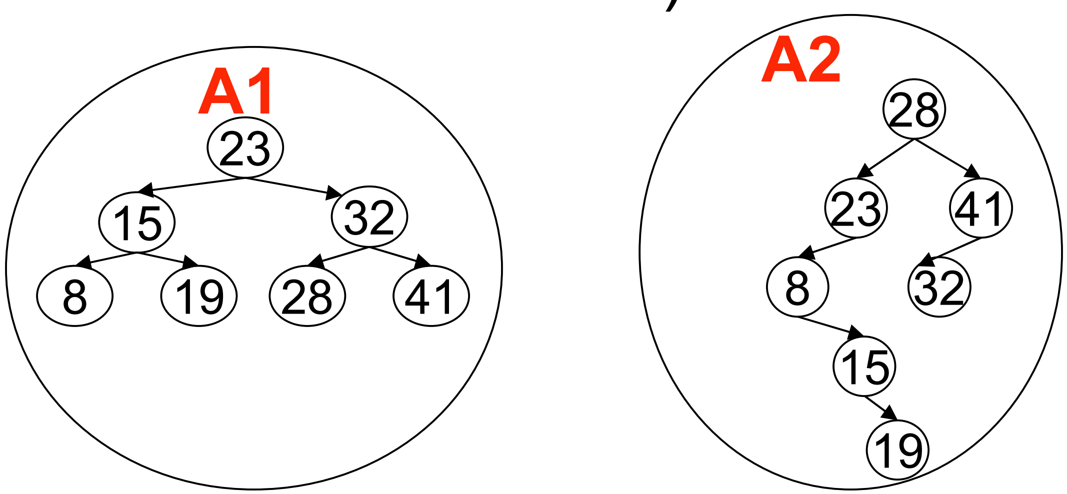
Exemplu de arbori de căutare

- Cu aceleaşi chei se pot construi arbori distincţi



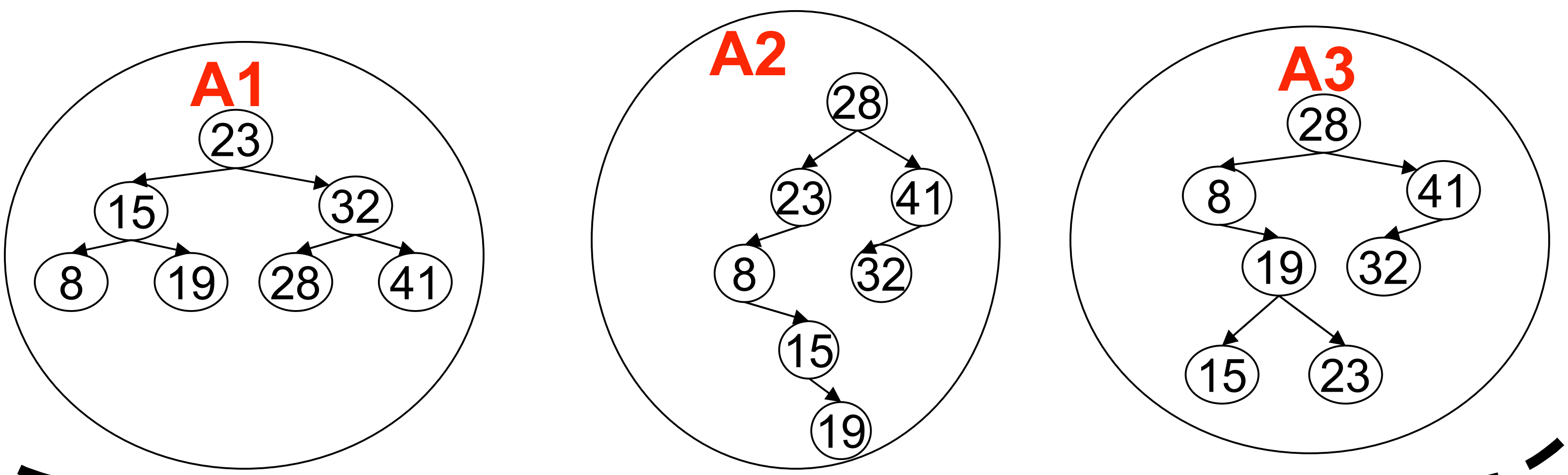
Exemplu (I)

- Presupunem că elementele din A1 şi A2 au probabilităţi de căutare egale:
  - Numărul mediu de comparaţii pentru A1 va fi: (1 + 2 + 2 + 3 + 3 + 3 + 3) / 7 = 2.42
  - Numărul mediu de comparaţii pentru A2 va fi: (1 + 2 + 2 + 3 + 3 + 4 + 5) / 7 = 2.85



Exemplu (II)

- Presupunem că elementele au următoarele probabilităţi:
  - 8: 0.2; 15: 0.01; 19: 0.1; 23: 0.02; 28: 0.25; 32: 0.2; 41: 0.22;
  - Numărul mediu de comparaţii pentru A1:
    - 0.02*1+0.01*2+0.2*2+0.2*3+0.1*3+0.25*3+0.22*3=2.85
  - Numărul mediu de comparaţii pentru A2:
    - 0.25*1+0.02*2+0.22*2+0.2*3+0.2*3+0.01*4+0.1*5=2.47



Probleme

- Costul căutării depinde de frecvenţa cu care este căutat fiecare termen.
- → Ne dorim ca termenii cei mai des căutaţi să fie cât mai aproape de vârful arborelui pentru a micşora numărul de apeluri recursive.
- Dacă arborele nu este construit prin sosirea aleatorie a cheilor putem ajunge la o simplă listă cu n elemente.

Definiţie AOC

- Definiție: Fie A un arbore binar de căutare cu chei într-o mulțime K; fie $\{x_1, x_2, \dots x_n\}$ cheile conținute în A, iar $\{y_0, y_1, \dots y_n\}$ chei reprezentante ale cheilor din K ce nu sunt în A astfel încât: $y_{i-1} \prec x_i \prec y_i, i = \overline{1,n}$. Fie p_i, i = 1,n probabilitatea de a căuta cheia x_i și q_j, j = 0,n probabilitatea de a căuta o cheie reprezentată de y_j. Vom avea relația: $\sum_{i=1}^{n} p_i + \sum_{j=0}^{n} q_j = 1$. Se numește arbore de căutare probabilistică, un arbore cu costul:

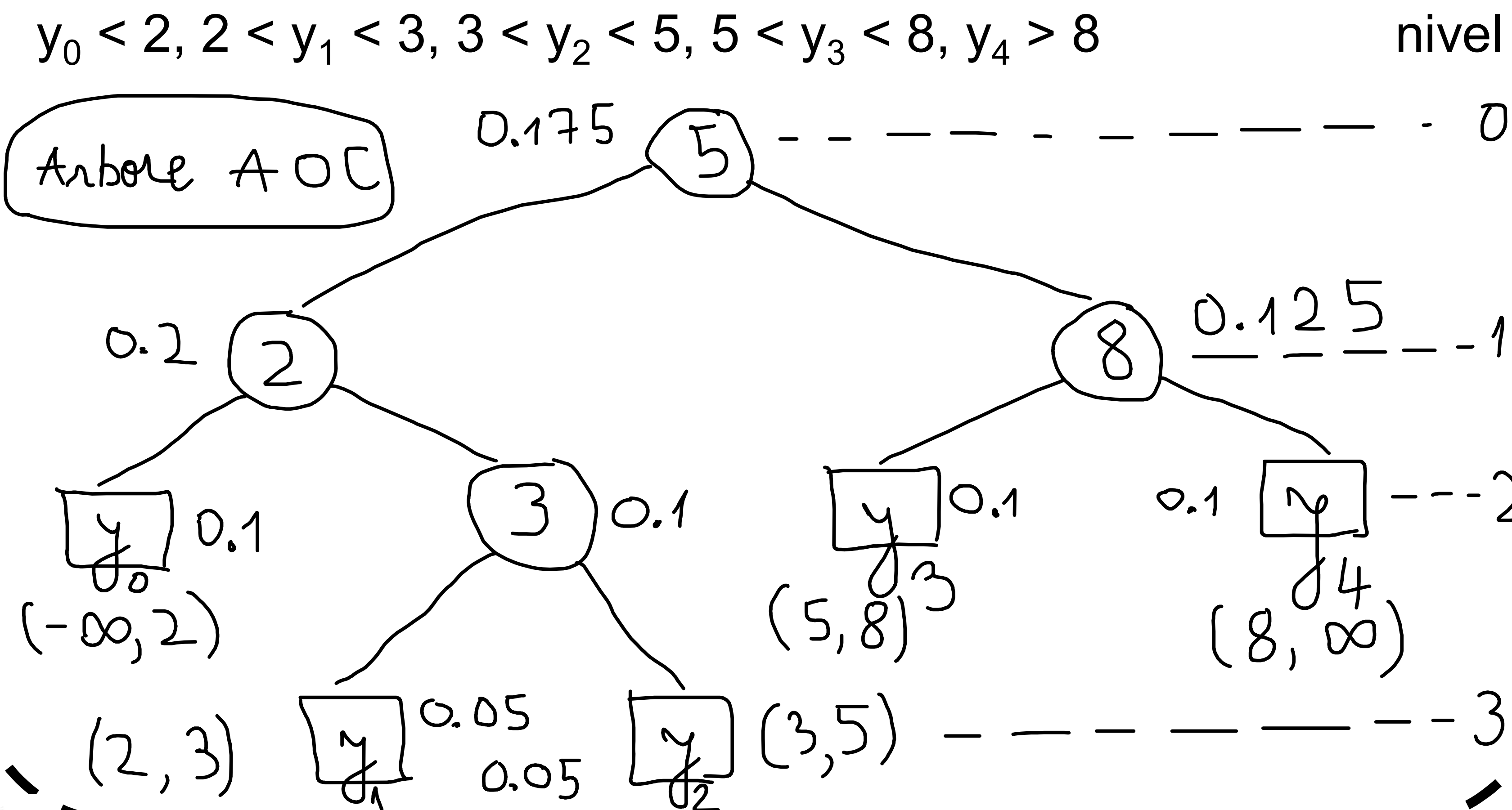
$$Cost(A) = \sum_{i=1}^{n}(nivel(x_i, A) + 1) * p_i + \sum_{j=0}^{n} nivel(y_j, A) * q_j$$

- Definiție: Un arbore de căutare probabilistică având cost minim este un arbore optim la căutare (AOC).

Exemplu AOC

$x_1 = 2, x_2 = 3, x_3 = 5, x_4 = 8$

$y_0 < 2, 2 < y_1 < 3, 3 < y_2 < 5, 5 < y_3 < 8, y_4 > 8$



Algoritm AOC naiv

- Generarea permutărilor $x_1, \ldots x_n$.
- Construcţia arborilor de căutare corespunzători.
- Calcularea costului pentru fiecare arbore.
- Alegerea arborelui de cost minim.
- Complexitate: $\Omega(n!)$ (deoarece sunt n! permutări).
- → căutăm altă variantă!!!

Construcţia AOC – Notaţii

- $A_{i,j}$ desemnează un arbore binar de căutare cu cheile $\{x_{i+1}, x_{i+2}, \ldots x_j\}$ în noduri și cu cheile $\{y_i, y_{i+1}, \ldots y_j\}$ în frunzele fictive.
- $C_{i,j}$ = Cost $(A_{i,j})$. $Cost(A_{ij}) = \sum_{k=i+1}^{j}(nivel(x_k, A_{ij})+1)*p_k + \sum_{k=i}^{j} nivel(y_k, A_{ij})*q_k$
- $R_{i,j}$ este indicele α al cheii x_α din rădăcina arborelui $A_{i,j}$.

$$w_{i,j} = \sum_{k=i+1}^{j} p_k + \sum_{k=i}^{j} q_k \quad w_{i,j} = \sum_{k=i+1}^{j-1} p_k + p_j + \sum_{k=i}^{j-1} q_k + q_j = w_{i,j-1} + p_j + q_j$$

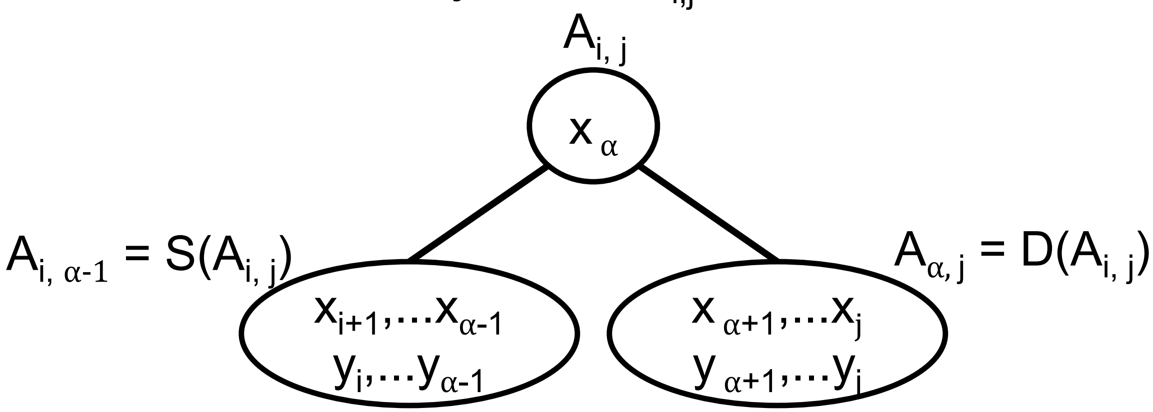
- Observație: $A_{0,n}$ este chiar arborele A, $C_{0,n}$ = Cost (A) iar $w_{0,n} = 1$.

Construcţia AOC - Demonstraţie

- Lemă: Fie i,j, $0 \le i < j \le n$ astfel încât $C_{i,j} = \min_{i<\alpha\le j}\{C_{i,\alpha-1} + C_{\alpha,j}\} + w_{i,j}$

şi $A_{i,\alpha-1}$, $A_{\alpha,j}$ sunt AOC, pentru orice α, $i < \alpha \le j$. Atunci $A_{i,j}$ este AOC.

- Demonstraţie:



$$Cost(A_{ij}) = \sum_{k=i+1}^{j}(nivel(x_k, A_{ij}) + 1) * p_k + \sum_{k=i}^{j} nivel(y_k, A_{ij}) * q_k$$

→ $$C_{i,j} = C_{i,\alpha-1} + C_{\alpha,j} + w_{i,j}$$

- $C_{i,j}$ depinde de indicele α al nodului rădăcină.
- dacă $C_{i,\alpha-1} + C_{\alpha,j}$ este minimă → $C_{i,j}$ este minim → $A_{i,j}$ AOC.

Construcţia AOC

- 1. În etapa d, d = 1, 2, … n se calculează costurile și indicele cheilor din rădăcina arborilor AOC $A_{i, i+d}$, i = 0, n-d cu d noduri și d + 1 frunze fictive.
- Arborele $A_{i, i+d}$ conține în noduri cheile $\{x_{i+1}, x_{i+2}, \dots x_{i+d}\}$, iar în frunzele fictive sunt cheile $\{y_i, y_{i+1}, \dots y_{i+d}\}$. Calculul este efectuat pe baza rezultatelor obținute în etapele anterioare.
- Conform lemei avem $C_{i,i+d} = \min_{i<\alpha\le i+d}\{C_{i,\alpha-1} + C_{\alpha,i+d}\} + w_{i,i+d}$
- Rădăcina $A_{i, i+d}$ are indicele $R_{i, j} = \alpha$ care minimizează $C_{i, i+d}$.
- 2. Pentru d = n, $C_{0, n}$ corespunde arborelui AOC $A_{0, n}$ cu cheile $\{x_1, x_2, \dots x_n\}$ în noduri si cheile $\{y_0, y_1, \dots y_n\}$ în frunzele fictive.

Algoritm AOC

```
AOC(x, p, q, n)
    Pentru i de la 0 la n
        {C[i,i] = 0, R[i,i] = 0, w[i,i] = q[i]}   // inițializare costuri AOC vid A[i,i]
    Pentru d de la 1 la n
        Pentru i de la 0 la n-d // calcul indice rădăcină și cost pentru A[i,i+d]
            j = i + d,  C[i,j] = ∞,  w[i,j] = w[i,j-1] + p[j] + q[j]
            Pentru α de la i + 1 la j // ciclul critic – operații intensive
                Dacă (C[i,α-1] + C[α,j] < C[i,j])  // cost mai mic?
                    { C[i,j] = C[i,α-1] + C[α,j] ; R[i,j] = α} // update
            C[i,j] = C[i,j] + w[i,j]  // update
    Întoarce gen_AOC(C, R, x, 0, n)  // construcție efectivă arbore A[0,n]
                                     // cunoscând indicii
```

Complexitate???

AOC – Complexitate temporală

- Număr total de subprobleme (Ns):
  - $n * (n - d + 1) = O(n^2)$
- Număr total de alegeri la fiecare pas (Na):
  - $j - i = d = O(n)$
- Informal, complexitatea temporală = Ns * Na
  - Complexitate temporală AOC: $O(n^2) * O(n) = O(n^3)$
- Optimizare – Knuth: $O(n^2)$

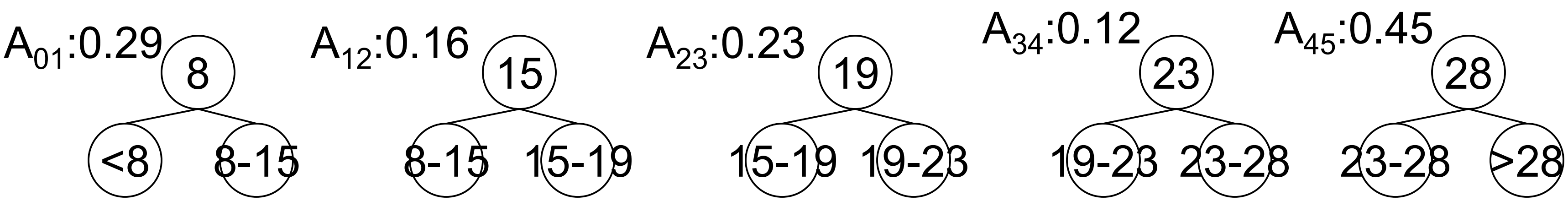
Exemplu construcţie AOC (I)

- 8: 0.2; 15: 0.01; 19: 0.1; 23: 0.02; 28: 0.25; (58%)
- [0:8): 0.02; (8:15): 0.07; (15:19): 0.08; (19:23): 0.05; (23:28): 0.05; (28,∞): 0.15 (42%)
- $C_{01}=p_1+q_0+q_1=0.2+0.02+0.07=0.29$
- $C_{12}=p_2+q_1+q_2=0.01+0.07+0.08=0.16$
- $C_{23}=p_3+q_2+q_3=0.1+0.08+0.05=0.23$
- $C_{34}=p_4+q_3+q_4=0.02+0.05+0.05=0.12$
- $C_{45}=p_5+q_4+q_5=0.25+0.05+0.15=0.45$

$$w_{i,j} = \sum_{k=i+1}^{j} p_k + \sum_{k=i}^{j} q_k$$

$$w_{i,j} = w_{i,j-1} + p_j + q_j \qquad C_{i,i+d} = \min_{i<\alpha\leq i+d}\{C_{i,\alpha-1} + C_{\alpha,i+d}\} + w_{i,i+d}$$

Exemplu construcţie AOC (II)

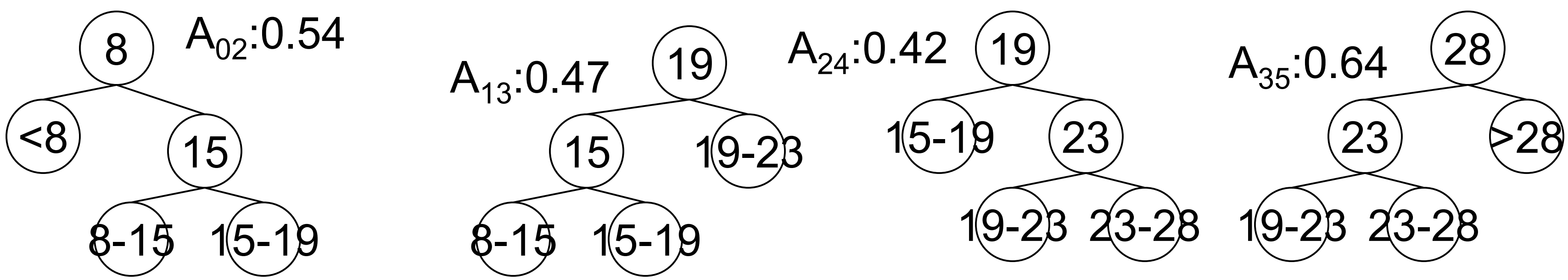


C_{02}= min {(C_{00}+C_{12}),(C_{01}+C_{22})} + w_{02} = min(0.16,0.29) + 0.38 = 0.54 R_{02}= 1 (α=1)

C_{13} = min{C_{11}+C_{23},C_{12}+C_{33}} + w_{13} = min(0.23,0.16) + 0.31 = 0.47 R_{13}= 3 (α=3)

C_{24} = min{C_{34},C_{23}} + w_{24} = min(0.12,0.23) + 0.3 = 0.42 R_{24}= 3

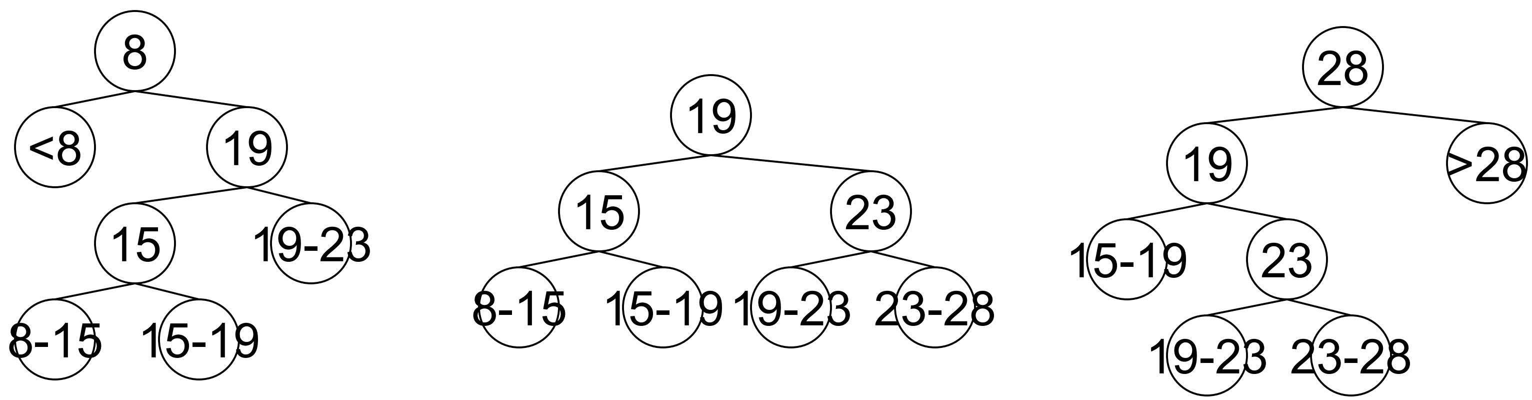
C_{35} = min{C_{45},C_{34}} + w_{35} = min(0.45,0.12) + 0.52 = 0.64 R_{35}= 5



$$w_{i,j} = w_{i,j-1} + p_j + q_j \qquad C_{i,i+d} = \min_{i<\alpha\le i+d} \{C_{i,\alpha-1} + C_{\alpha,i+d}\} + w_{i,i+d}$$

Exemplu construcţie AOC (III)

- $C_{03} = \min(C_{00}+C_{13}, C_{01}+C_{23}, C_{02}+C_{33}) + w_{03} = \min(0.47, 0.52, 0.54) + w_{03} \Rightarrow R_{03} = 1$
- $C_{14} = \min(C_{11}+C_{24}, C_{12}+C_{34}, C_{13}+C_{44}) + w_{14} = \min(0.42, 0.28, 0.47) + w_{14} \Rightarrow R_{14} = 3$
- $C_{25} = \min(C_{22}+C_{35}, C_{23}+C_{45}, C_{24}+C_{55}) + w_{25} = \min(0.64, 0.67, 0.42) + w_{25} \Rightarrow R_{25} = 5$



$$w_{i,j} = \sum_{k=i+1}^{j} p_k + \sum_{k=i}^{j} q_k \qquad C_{i,i+d} = \min_{i<\alpha\le i+d}\{C_{i,\alpha-1} + C_{\alpha,i+d}\} + w_{i,i+d}$$

AOC – Corectitudine (I)

- Teoremă: Algoritmul AOC construiește un arbore AOC A cu cheile $x = \{x_1, x_2, \dots x_n\}$ conform probabilităților de căutare p_i, $i = 1,n$ și q_j, $j = 0,n$.

- Demonstrație: prin inducție după etapa de calcul a costurilor arborilor cu d noduri.

- Caz de bază: $d = 0$. Costurile $C_{i,\,i}$ ale arborilor vizi $A_{i,\,i}$, $i = 0, n$ sunt 0, așa cum sunt inițializate de algoritm.

- Pas de inducție: $d \geq 1$. Ip. ind.: pentru orice $d' < d$, algoritmul AOC calculează costurile $C_{i,\,i+d'}$ și indicii $R_{i,\,i+d'}$, ai rădăcinilor unor AOC $A_{i,\,i+d'}$ cu $i = 0, n-d'$ cu cheile $\{x_{i+1}, x_{i+2}, \dots x_{i+d'}\}$. Trebuie să arătam că valorile $C_{i,\,i+d}$ și $R_{i,\,i+d}$ corespund unor AOC $A_{i,\,i+d}$ cu $i = 0, n-d$ cu cheile $\{x_{i+1}, x_{i+2}, \dots x_{i+d}\}$.

AOC – Corectitudine (II)

- Pentru d și i fixate, algoritmul calculează:

$$C_{i,i+d} = \min_{i<\alpha\leq i+d}\{C_{i,\alpha-1} + C_{\alpha,i+d}\} + w_{i,i+d}$$

- unde costurile $C_{i, \alpha-1}$ și $C_{\alpha, i+d}$ corespund unor arbori cu un număr de noduri d’ = α – 1 – i în cazul $C_{i, \alpha-1}$ și d’ = i + d – α în cazul $C_{\alpha, i+d}$.
- 0 ≤ d’ ≤ d – 1 → aceste valori au fost deja calculate în etapele d’ < d și conform ipotezei inductive → sunt costuri și indici ai rădăcinilor unor AOC.
- Conform Lemei anterioare, $C_{i, i+d}$ este costul unui AOC. Conform algoritmului → rădăcina acestui arbore are indicele r = $R_{i, j}$, iar cheile sunt $\{x_{i+1}, x_{i+2}, \dots x_{r-1}\} \cup \{x_r\} \cup \{x_{r+1}, x_{r+2}, \dots x_j\} = \{x_{i+1}, x_{i+2}, \dots x_j\}$
- Pentru d = n, costul $C_{0, n}$ corespunde unui AOC $A_{0, n}$ cu cheile x și cu rădăcina de indice $R_{0, n}$.

ÎNTREBĂRI?