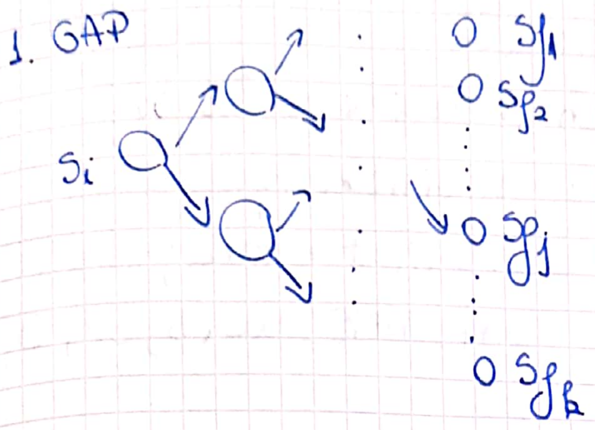
28.01.2020

Recapitulare

1. GAP



(S_i , S_{f_j} , S , A)

- S_i ↓ starea inițială
- S_{f_j} ↓ starea finală unde vreau să ajung
- S ↓ mulț. stărilor
- A ↓ tranziții între stări (mulțimea tuturor muchiilor)

Alg $\xrightarrow{?}$ (S_i, S_f, S, A) ↓ GAP

Alg → rezultat

GAP

```
GAP_TIME(Si, Sf, S, A) {
    for each s ∈ S
        s.vizitat = false;
    Si.vizitat = true;
    GAP_TIME_rec(Si, Sf, S, A);
    return Sf.vizitat;
}
```

```
GAP_TIME_rec(Si, Sf, S, A) {
    if (Si == Sf)
        break;
    for each s vecini al lui Si {
        if (!s.vizitat) {
            s.vizitat = true;
            GAP_TIME_rec(s, Sf, S, A);
        }
    }
}
```

DFS

Complexitate → temporală

spațială · SPACE(n log n)

$O(\text{noduri} + \text{arce}) = O(n^2)$

$\Rightarrow GAP \in TIME(n^2)$

```
GAP_SPACE(Si, Sf, S, A) {
    GAP_SPACE_D&i(Si, Sf, n, S, A);
}
```

divide & impera

```
GAP_SPACE_D&i(i, j, dist, S, A) {
    if (dist == 0)
        return (i == j);
    if (dist == 1)
        return (i, j) ∈ A;
    for each k ∈ S
        return GAP_SPACE_D&i(i, k, dist/2,
               GAP_SPACE_D&i(k, j, dist/2, S, A)
}
```

$T(n) = 2nT\left(\frac{n}{2}\right) + \overbrace{O(1)}^{=1} \cdot 2$

$nT\left(\frac{n}{2}\right) = \frac{n^2}{2} T\left(\frac{n}{4}\right) + n$

$n^2 T\left(\frac{n}{4}\right) = \frac{n^3}{2^3} T\left(\frac{n}{8}\right) + n^2$

$\vdots$

$\frac{n^k}{2^{k-2}} T\left(\frac{n}{2^k}\right) = \frac{n^k}{2^{k-2}}$

$k = \log n$

$\frac{n^{\log n}}{n}$

temporală → imensă $= TIME(n^{\log n - 1})$

Complexitate → spațială :

- adâncimea maximă a recursivității este log m
- la fiecare apel, folosesc un nr. cst. de variabile (i, j, dist, buffere de fișier)
  → 5 variabile, adică nr cst
- fiecare variabilă ia valori maxim m ⇒ folosesc log m memorie

⇒ O(log m) la fiecare apel
log m → adâncime max.
⇒ O(log² m) memorie

⇒ GAP ∈ SPACE(log² m)

```
NGAP (Si, Sf, m, S, A) {
    current = Si;
    for(i=0; i<m; i++) {
        if(current == Sf)
            SUCCESS;
        s = choice(S);
        if((current, s) ∉ A)
            FAIL;
        current = s;
    }
    FAIL;
}
```

NGAP ∈ NTIME(m)

NGAP ∈ NSPACE(log m)

fiecare variabilă are log m
nr. cst de variabile

S, A) &&

$LOGSPACE \subseteq NLOGSPACE \subseteq PTIME \subseteq NPTIME \subseteq$

$PSPACE = NPSPACE$

$TIME(f)$

$PTIME = \bigcup TIME(f)$, unde f - fcție polinomială

$NTIME \subseteq NSPACE$

TEOREMĂ:

$NSPACE(f) \subseteq TIME(c^f)$

Lemă: Fie un alg. nedeterminist cu complexitate spațială $O(f(n))$, $f(n) \geq \log n$.

Construcția spațiului stărilor se poate face cu complexitatea temporală $O(k^{f(n)})$

$Alg \xrightarrow{k^{f(n)}} (S_i, S_f, S, A) \xrightarrow{GAP \in TIME(n^2)}$ rezultat

$GAP \in TIME(\underbrace{n}_{\text{nr. stărilor = nr de noduri}}{}^2)$

$n = k^{f(n)}$

$\Rightarrow GAP \in O((k^{f(n)})^2) = O((k^2)^{f(n)}) = O(c^{f(n)})$

Corolar: $NLOGSPACE \subseteq PTIME$

$f = a \cdot \log n$

$\Rightarrow NLOGSPACE \subseteq TIME(c^{a \log n}) = TIME((2^{\log c})^{a \log n}) =$

$= TIME(n^{a \cdot \log c}) \subseteq PTIME$

Teoremă (Savitch): $NSPACE(f(n)) \subseteq SPACE(f^2(n))$

$Alg \xrightarrow[f(n)]{?} (S_i, S_f, S, A) \xrightarrow{GAP}$ rezultat

$GAP \in SPACE(\log^2 n)$, $n \to k^{f(n)}$

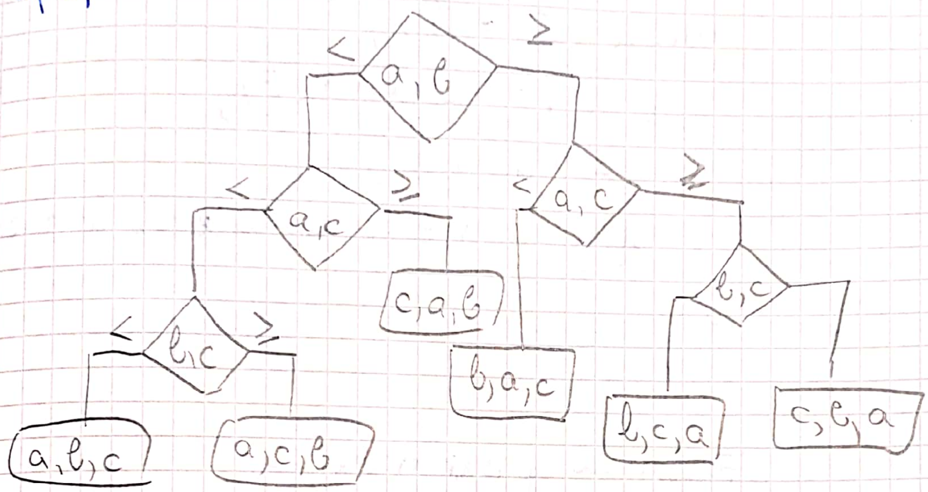
$GAP \in SPACE(f^2(n)) \Rightarrow NSPACE(f) \subseteq SPACE(f^2)$

Corolar: $NPSPACE \subseteq PSPACE$

6/CA 2020 $M = \{ m \mid P_m(m) \neq \perp \}$

7/CA 2020 Dem. că ∀ alg. de sortare prin comparații de chei are complexitatea $\Omega(n \log n)$

a, b, c



n! frunze

h niveluri ⇒ maxim 2^h noduri pe ultimul nivel

$$n! \leq 2^h$$

Stirling : $n! \geq \left(\frac{n}{e}\right)^n$ ⇒ $2^h \geq \left(\frac{n}{e}\right)^n$

$$h \geq n \log n - n \log e \in \Theta(n \log n)$$

$$\Rightarrow h \in \Omega(n \log n)$$

Complexitatea mea este chiar înălțimea arborelui = h

8/CA 2020 Scrieți un alg. de aproximare pt. acoperirea optimă cu noduri a unui graf. Calculați și factorul de aproximare al acestui algoritm.

```
cover (N, E) {
    M = { };
    while (E != ∅) {
        (u, v) = random (E);
        M = M ∪ {u, v};
        șterg din E muchiile cu un capăt u sau v;
    }
}
```

Factor de aproximare: 2

4. Th. Cook

SAT ∈ NPC

Pasul 1: SAT ∈ NP

Avem o formulă $F(x_1, \dots, x_k) = C_1 \wedge C_2 \wedge \dots \wedge C_q$ (clauze)

$C_i = e_1^i \vee e_2^i \vee \dots \vee e_m^i$

$e_j^i = x_s$ sau $\overline{x_s}$

```
NSAT (F) {
    for(i = 1; i <= k; i++) {
        xi = choice ({0, 1});
    }
    for (i = 1; i <= q; i++) {
        satisfacut = false;
        for (j = 1 ; j <= m; j++) {
```

```
        if ((e'j == "xs" && xs == 1) ||
            (e'j == "x̄s" && xs == 0)) {
              satisfacut = true;
              break;
        }
      }
    }
    if (!satisfacut)
        FAIL;
  }
  SUCCESS;
}
```

Pasul 2: Orice pereche (alg, date) se poate transforma într-o expresie F ai alg(date) = SUCCESS $\Leftrightarrow$ F este satisfiabilă

NP-hard

$\Rightarrow$ SAT $\in$ NPD
SAT $\in$ NP

$\Rightarrow$ SAT $\in$ NPC

Corolar: SAT $\in$ P $\Leftrightarrow$ P = NP

SAT $\in$ NPC $\Leftrightarrow \forall Q \in NP, Q \leq_p$ SAT $\in$ P $\Rightarrow$

$\Rightarrow \forall Q \in NP, Q \in P$

- $SAT \le_p 3\text{-}SAT$

$\downarrow F$ $\quad F'$

NPC

Pt. fiecare clauză C din F:

- Cazul 1: C are un literal l ⇒ adaug în F' următoarele clauze:

$(l \vee z_1 \vee z_2) \wedge (l \vee \overline{z_1} \vee z_2) \wedge (l \vee z_1 \vee \overline{z_2}) \wedge (l \vee \overline{z_1} \vee \overline{z_2}) = l$

- Cazul 2: C are 2 literali $l_1 \vee l_2$ ⇒ adaug în F' următoarele clauze:

$(l_1 \vee l_2 \vee z_1) \wedge (l_1 \vee l_2 \vee \overline{z_1})$

- Cazul 3: C are 3 literali ⇒ adaug în F' C-ul

- Cazul 4: C are $k > 3$ literali: $C = l_1 \vee l_2 \vee l_3 \vee \ldots \vee l_k$

Introduc $z_1, z_2, \ldots, z_{k-3}$ și clauzele

$(l_1 \vee l_2 \vee z_1) \wedge (l_3 \vee \overline{z_1} \vee z_2) \wedge (l_4 \vee \overline{z_2} \vee z_3) \wedge \ldots$

$\wedge (l_{k-2} \vee \overline{z_{k-4}} \vee z_{k-3}) \wedge (l_{k-1} \vee l_k \vee \overline{z_{k-3}})$

C satisfiabilă cu l_i true

Apare în combinația: $(l_i \vee \overline{z_{i-2}} \vee z_{i-1})$

$(l_{i-1} \vee \overline{z_{i-3}} \vee z_{i-2}) \wedge (l_i \vee \overline{z_{i-2}} \vee z_{i-1}) \wedge (l_{i+1} \vee \overline{z_{i-1}} \vee z_i)$

Aleg z_{i-2} = true, z_{i-1} = false, z_i = false, ... z_{k-3} = false

$z_1, z_2, \ldots, z_{i-2}$ = true

11. iii: inorder (t) => inorder (rotate (t)); inorder(L); inorder (R); maxT (L) ≤ x; minT(R) ≥ x

CB: t = empty: true => inorder (empty) = true

Pi: t = node (L, x, R)

)= l

MS: inorder (node (L, x, R)) = inorder (L) && inorder (R) && maxT(L) ≤ x && minT(R) ≥ x

membrul stâng

• MS fals: fals => orice

• MS true

MD = inorder (rotate (L, x, R)

Cazul 1: L = empty => MD = inorder (node (empty, x, R))
= inorder (empty) && inorder(R) && maxT(empty) ≤ x &&
A A
−∞ ≤ x => A

minT(R) ≥ x
A

Cazul 2: L = node (LL, y, LR)

MD = inorder (node (LL, y, node (LR, x, R))) =
inorder (LL) && inorder (node (LR, x, R)) &&
maxT (LL) ≤ y && minT(node(LR, x, R)) ≥ y

inorder (L) = inorder (LL) && inorder (LR) &&
maxT (LL) ≤ y && minT (LR) ≥ y

de aici

MD = inorder (LL) && inorder (LR) && inorder (R) &&
maxT (LR) ≤ x && minT (R) ≥ x &&
maxT (LL) ≤ y && minT (LR) ≥ y &&
x ≥ y && minT (R) ≥ y

$\max T(L) \leq x \Rightarrow \max T(\text{node}(LL, y, LR)) \leq x$

$\Rightarrow \max T(LL) \leq x$

$y \leq x$

$\max T(LR) \leq x$

$\min T(R) \geq x \geq y$

→ Dați un ex. de 2-acoperire cu 7 muchii și 5 vf.

2 vf. care acoperă tot graful