

CURS 1

Examen:

- 2 părți (o parte gălă pentru cine vrea doar să treacă)
 - (o parte cu probleme)

↳ nu ne dă teorie
 - proiecte pentru puncte extra
-

calculabilitate = ce nu pot calcula chiar dacă am memorie/timp oricât de mare.

Complexitate = ce pot și nu pot calcula eficient

Funcția lui Ackermann (este recursivă, dar nu este primativ recursivă)

$$f: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$$

$$A(0, m) = m + 1$$

$$A(m+1, 0) = A(m, 1)$$

$$A(m+1, m+1) = A(m, A(m+1, m))$$

$$A(1, 0) = A(0, 1) = 2$$

$$A(1, 1) = A(0, A(1, 0)) = A(0, 2) = 3$$

$$\Rightarrow A(1, m+1) = A(0, A(1, m)) = A(1, m) + 1$$

$$A(1, m) = m + 2$$

FUNCȚIA LUI COLLATZ

def collatz(m):

 while (m > 1):

 if (m % 2 == 0)

 m = m / 2

 else

 m = 3m + 1

 return 0

m=2 → 1 → return 0

m=3 → 10 → 5 → 16 → 8 → 4 → 2 → 1 → return 0

m=6 → 3 ...

m=7 → 22 → 11 → 34 → 17 → 52 → 26 → 13 → 40 → ... → 1
→ return 0

Puteam să intuiam că funcția va returna mereu 0, dar nu a putut să demonstreze acest lucru.

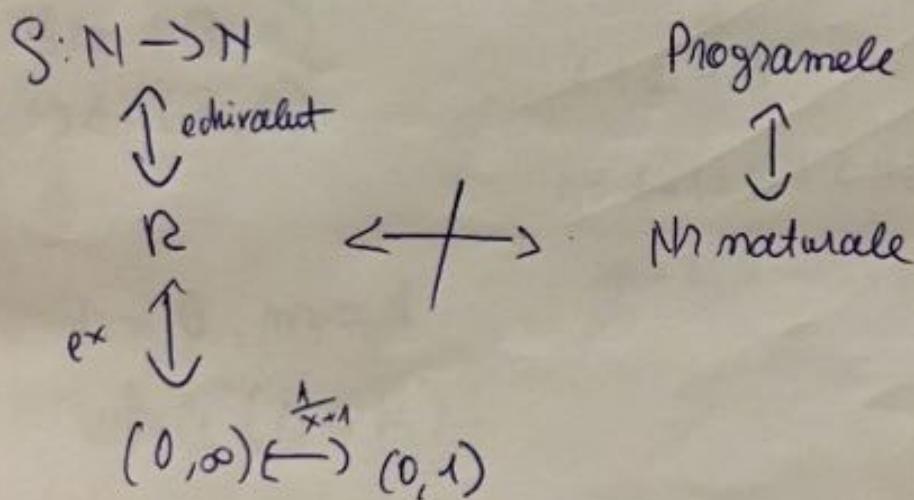
$f: \mathbb{N} \rightarrow \mathbb{N}$
 sunt mai "multe" decât programe
 infinit

Or multime A are aiasi nr de elem. cu B (\Rightarrow există o funcție bijectivă $f: A \rightarrow B$)

Ex: $f: \mathbb{N} \rightarrow 2\mathbb{N}$

$$f(x) = 2x \rightarrow \text{e bijectivă}$$

nr pare sunt mai puține decât nr naturale



T NU EXISTĂ O BIJECTIE ÎNTRE nr naturale și $(0, 1)$

Dem: pp există bijectie

$$x_0 = 0, a_1 a_2 \dots$$

$$x_1 = 0, b_1 b_2 \dots$$

⋮

Seminar 1

Cum pot lista toate punctele din xoy

$$0 \in (0,0)$$

$$1 \in (0,1) \dots$$

$$2 \in (1,0)$$

$$3 \in (0,2) \dots$$

$$4 \in (1,1)$$

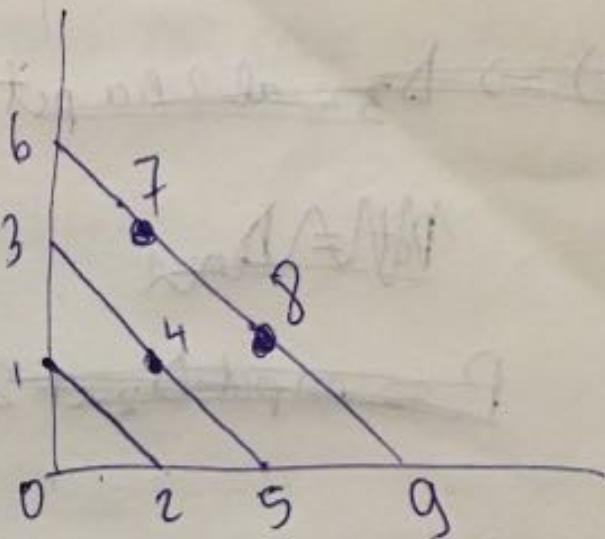
$$5 \in (2,0)$$

$$6 \in (0,3) \dots$$

$$7 \in (1,2)$$

$$8 \in (2,1)$$

$$9 \in (3,0).$$



$$(m, n) \rightarrow k$$

$$k \rightarrow (m, n)$$

$$1+2+3$$

$$n = 1+1=2+2+3$$

$$m = 2, 3$$

$$D = 0, m = 1$$

while ($D < k$) :

$$D = D + mn$$

$$mn = m + 1$$

$$D = D$$

$$D < 3$$

$$D = 1$$

$$m = 2$$

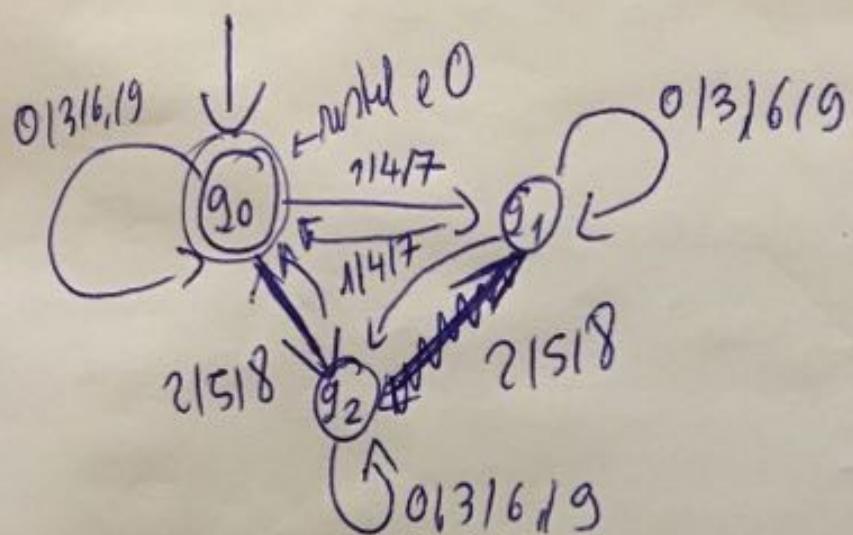
$$1 < 3$$

$$\boxed{D = 3}$$

$$m = 3$$

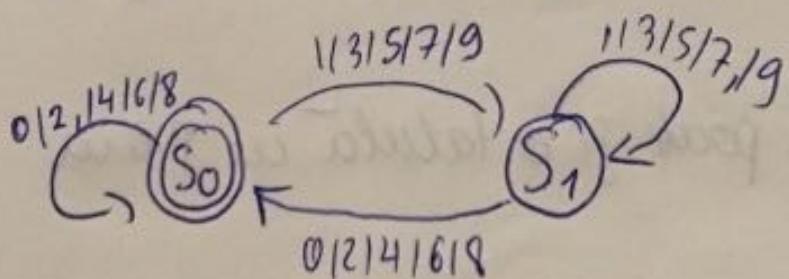
$$3 < 3$$

- Automat finit care acceptă m care se divid cu 3



Sunătă	Litură	Simbol care treacă
s_0	$0 3 6 9$	s_0

- Automat finit m par



Automat nondeterminist

$\{q_0\}$

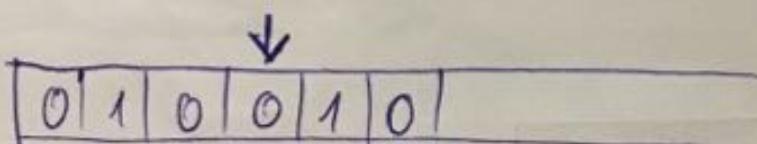
$\downarrow a$

$\{q_1, q_2\}$

$\downarrow a$

$\{q_4, q_5\}$

Automat cellular



- toate celulele execută în paralel o instrucțiune

- avem o funcție $f(0 \cdot 0 \cdot 1) = 1$ ~~că să adună celula vecină la mijloc~~

$\begin{matrix} \checkmark & \downarrow & \text{vecin} \\ \text{vecin} & \text{celula centrală} \end{matrix}$

$$f(1 \cdot 0 \cdot 1) = 0$$

$$1 \cdot 1 \cdot 1 = 0$$

- fiecare automat celular poate fi o tablă cu 8 linii

CURS 2

Program



\sum (afărat) : $\Sigma^* = \{y_1 \dots y_m \mid y_i \in \Sigma\}$

- Există funcții care nu sunt calculabile (există o funcție care nu corespunde niciunui program)

PROBLEMA DE DECISIE:

Ex: INPUT $x = x_1 \dots x_n$
OUTPUT Este x palindrom?

⑦ (Matiyasevich)

INPUT: polinom carecon p($x_1 \dots x_n$) $\in \mathbb{Z}\{x_1, \dots x_n\}$

DE DECIS: Pre ceeață $p(x_1 \dots x_n) = 0$
sau are reăzintegi?

~~o formă de date a formulei rezolvării~~

$$f: \Sigma^* \rightarrow \Sigma^*$$

(N) (N)

Functii primitiv recursive . partial recursive

$$f: \mathbb{N}^3 \rightarrow \mathbb{N}^2$$

↓ ↓
 dacă năspunsul e o perche de m
 peșt să convertește la un singur m

$$f': \mathbb{N} \rightarrow \mathbb{N}$$

conversie 3 în 2 $\Rightarrow \langle \langle x_1, x_2 \rangle, x_3 \rangle$ ie primele 2 și
rezultatul îl codifică cu al 3-lea

Functii de bază:

$$- f_0(x) = 0_{\text{numărul lui } x} \quad \text{fiecare } 0$$

$$f_1(x) = s(x) = x + 1 \quad \text{sucesor}$$

$$f_{i,m}(x_1, \dots, x_m) = x_i \quad \text{proiecție}$$

Din aceste funcții simple se fac operații pt a forma alte funcții

Operații:

1. COMPOZITIE

$$f: \mathbb{N}^m \rightarrow \mathbb{N}$$

$$g_1, g_2, \dots, g_m: \mathbb{N}^m \rightarrow \mathbb{N}$$

$$f(g_1, g_2, \dots, g_m)(x_1, \dots, x_m) = f(g_1(x_1, \dots, x_m), \dots, g_m(x_1, \dots, x_m))$$

Dacă 2 funcții sunt calculabile, compozitia lor e calculabilă

2. RECURSIE PRIMITIVĂ

$$f(x_1, \dots, x_m)$$

$$\underline{g(y_1, \dots, y_{m+1})}$$

$$g(x_1, \dots, x_m, 0) = f(x_1, x_2, \dots, x_m)$$

$$g(x_1, \dots, x_m, y+1) = h(x_1, x_2, \dots, x_m, y, g(x_1, \dots, x_m, y))$$

y
un număr natural

DEF $f: N \rightarrow N$ se numește primitiv recursivă dacă pot să se exprime folosind funcțiile de la urmări primări compozitie, rec. primitivă

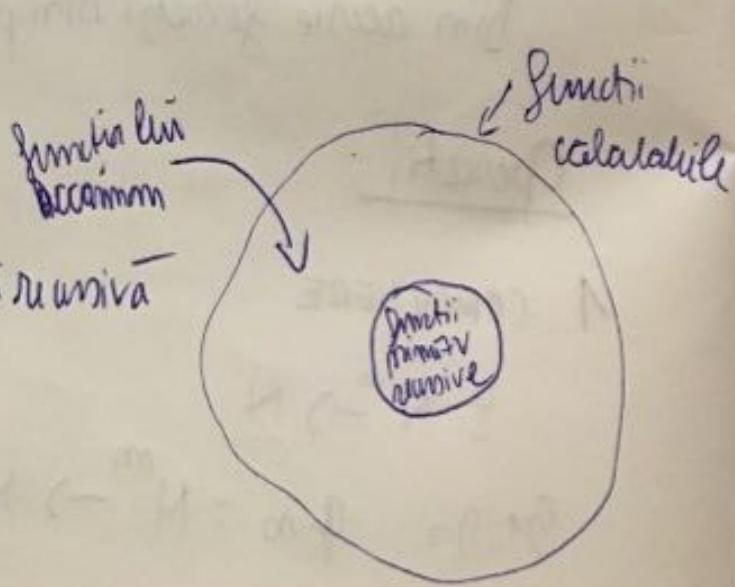
Ex: • $S(x) = 10$ (compozitie funcția succesor de 10 ori)

$$\underbrace{S \circ S \circ S \dots \circ S}_{10 \text{ ori}} = S_0$$

$$\bullet f(y, y) = x + y \text{ este primitivă și următoare}$$

$$f(x, 0) = S(x)$$

$$f(x, y+1) = S(f(x, y))$$



$\beta_0, \beta_1, \beta_2, \dots, \beta_m$ → enumerarea tuturor f m.m. recursive

$$\forall m \in \mathbb{N} \quad f(m) = \beta_m(m) + 1 \pm \beta_m(m)$$

Concluzie: Trebuie să încăpătez în model funcții care nu returnează valori

Notății: $f: N \xrightarrow{\circ} N$ funcție parțială (posibil să nu returneze valoare)

$f(x) \uparrow$ pe imputul x și nu returnează nimic

3. Mișcări

$$\text{am } f(x_1, \dots, x_{m+1})$$

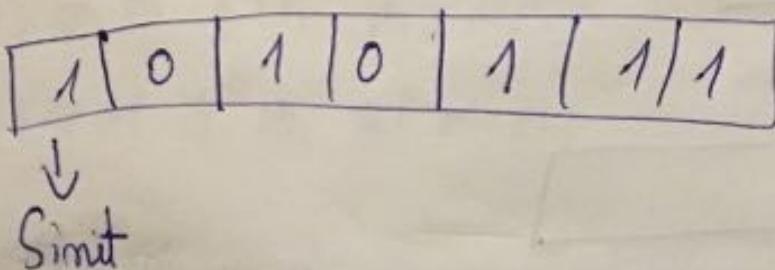
difinim $g(x_1, \dots, x_m) = \begin{cases} \text{al mai mic } y \in \mathbb{N} \text{ așa că } f(x_1, \dots, x_m, y) = 0 \\ \text{nedefinit dacă } y \text{ nu există} \end{cases}$

Df: O funcție f este parțial recursivă, dacă poate fi obținută din funcțiile de laată prin opere compunute, recursie primută și mișcare.

recursivă = parțial recursivă + totală

CURS 3

Maximă Turing (1936-1937)



Automatul să intre se deplasarea doar la dreapta

S multime stări (S_{init} , S_{halt} etc)

Σ alfabet \rightarrow litere (alfabetul e finit)

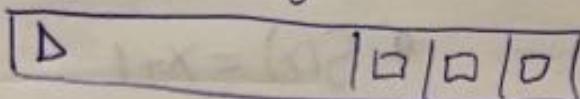
$\delta: S \times \Sigma \rightarrow S$ funcție de transiție

Maxima turing poate să scrie pe tot simbolul pe care l-a citit

$\delta: S \times \Sigma \rightarrow S \times \Sigma \times \{left, right, \text{stop}\}$

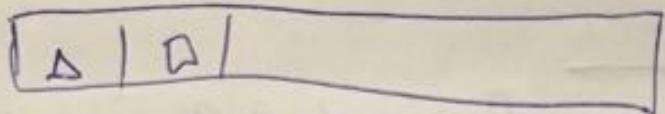
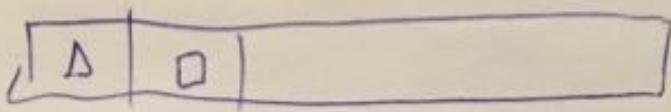
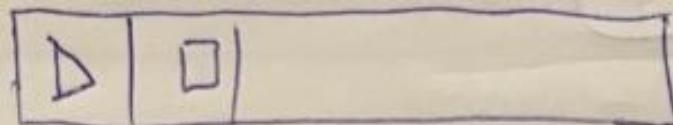
Maxima turing cu mai multe remși

→ nu a shge! c modanly
handă de întrebare



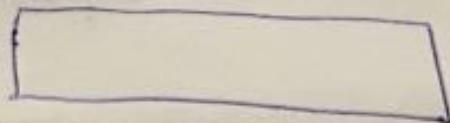
... Σ
infinită la dreapta

- $\Delta \rightarrow$ capătul remși stângă
- \square → literă
- $0/1$ etc



temp de lucru = m depoz.

Piromâie de ieșire



Funcție de
pentru

$$f: S \times \Sigma^k \rightarrow S \times \Sigma^{k-1} \times S \leftarrow, \rightarrow, \downarrow, 3^k$$

↓ stări ↓ ↓ stare menajare ↓ la scris pe
 următoarele menajare laturile de lucru

atârnă, aruncă atât pe loc

Stare $S_{halt} \rightarrow$ mașina se oprește

(T)

O funcție $f: N \xrightarrow{\text{definit}} N$ este parțial rec.

dacă f este calculabilă de o mașină Turing

Funcții de bază :

- $c(x_1, \dots, x_m) = 0$ (constantă)

- $s(x) = x + 1$ (succesori)

- $\pi_i^A(x_1, \dots, x_m) = x_i$

Operări:

- {comparare
- rec. primăritivă
- maximizare

Universalitate

Program $\rightarrow \Sigma^*$ ($\rightarrow \mathbb{N}$)
cu

Enumerare $M_1, M_2, \dots, M_n, \dots$ a tuturor M. Turing

- a) $\forall M \exists M_i \quad M \equiv M_i$;
- b) $\forall M \exists^\infty M_i \quad a_i \models M = M_i$;

(A) $\exists \text{MT universală } M(x) \text{ calculată } \langle \alpha, z \rangle \text{ a.i. } x = \langle \alpha, z \rangle$

$M(x)$ simulează $M_\alpha(z)$

Dacă $M_\alpha(z)$ acceptă în T pasi, atunci $M(x)$ acceptă în $c/\log T$ pasi unde $c > 0$ nu depinde de z .

(Def) Model de decizie ($\rightarrow L \subseteq \Sigma^*$)

$x \in L$	$x \notin L$
DA	
	NU

A.s.m recursiv enumerabilă de

$$g(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases} \text{ poate fi calculată de o MT (este part rec)}$$

A.s.m reuniță

$$g(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

CURS 4

- ① Vreau să simulez o maximă cu bandă infinită în ambele direcții ~~și~~ o maximă cu bandă infinită doar la dreapta

4	□	□	a	b	c	d
-3	-2	-1	1	2	3	4

A	a	ab	b	bc	c	cd
4	2	2	3	3	4	4
-1	-2	-3	-3	-4		

Reprezentăm în același ordine și perioada 1 și per. -1

O literă din alfabet este o secvență de litere una roșie și una albastră. Astăzi alfabet este produsul cartesian al alfabetului anterior.

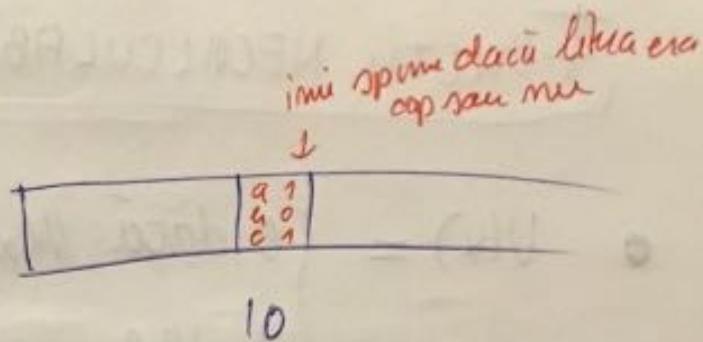
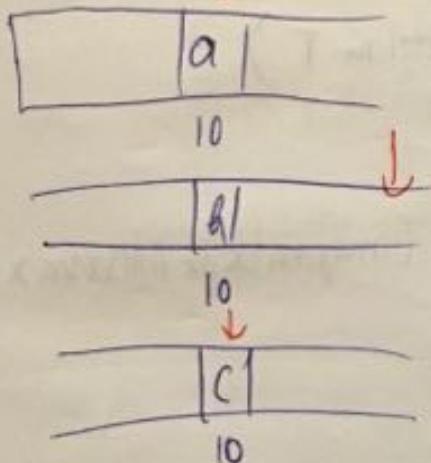
- ② Simulez o maximă Turing cu $|Σ| > 2$ cu OMT cu $Σ = 2$

idee: $\boxed{255} \Rightarrow \boxed{11111111}$

$\delta : \{0 \dots 255\} \times Σ \rightarrow \dots$

- ③ Simulez k lumi cu o bandă

copieacii



Concluziile: modelul M.T este relevant la modificări, adică calcularea același lucru indiferent de model

Mașină turing universală, =compiler / interpreter

- poate enumera toate mașinile T.

$M_1, M_2, \dots, M_n, \dots$



Codificare pt "programul" mașinii

- \exists M.T. M astfel încât $M(y), y = \langle i, x \rangle$
 \nwarrow codul mașinii \nwarrow inputul mașinii
 $M(y)$ simulată $M_i(x)$

- mt.univ = un program care poate simula toate progr. posibile
 (un fel de compilator care rulează orice program)

FUNCTII NECALCULABILE (de majinu T)

- $$U(x) = \begin{cases} 0 & \text{dacă } M_x(x) = 1 \quad (\text{majina } x \text{ pe intrare } x \text{ dă } 1) \\ 1 & \text{altfel} \end{cases}$$

$$U: \mathbb{N} \rightarrow \mathbb{N}$$

Când nulă și maximă x po împăt x să ar putea să rulate la infinit și nu o să returneze niciun rezultat

! Funcție parțial rec = poate fi calculată de MT

① Funcția U nu este parțial recursivă

Demonstratie: diagonalizare dim ursul 1

maximut	λ	0	1	00	01	10	11	N
M_1								
M_2								
:								
M_m	$U(\lambda)$	$U(0)$	$U(1)$...				N

Pp că U e calculată de M.T numită M_m (o maximă dim listă)

Calculz $U(n) = M_n(m)$

$\Rightarrow M_n$ calculază U

$U(n) = 0$ dacă $M_n(m) = 1$

1 altfel   

• Problema oprirei

(reprezentă)

$$\text{HALT}(\langle \alpha, x \rangle) = \begin{cases} 1 & \text{dacă } M_\alpha(x) \downarrow \\ 0 & \text{altfel} \end{cases}$$

Demonstratie:

Pp că există M_{HALT} care calculază funcția noastră

Afirmă că rămășimă maximă T , care calculază funcția U de încântare (care să întindă dacă nu e calalălită)

$$M_U \left\{ \begin{array}{l} \text{input } \alpha \\ \text{simulează } \text{HALT}(\alpha, \alpha) \\ \quad M_{\text{HALT}} \\ \text{dacă } \text{HALT}(\langle \alpha, \alpha \rangle) = 0 \rightarrow \text{return 1} \\ \quad \text{HALT}(\langle \infty, \infty \rangle) = 1 \rightarrow \text{return 0} \end{array} \right.$$

• "Wang Tiles" (Dominoeuri)

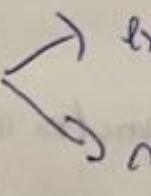
Wang

- am minte patrate 1×1 cu ~~laturi~~ coloanele oricun



- imput: K tipuri de piese colorate
- scop: să acoperă tot planul cu piesele pe care le-ai. Pot opera? Regula: dacă am 2 piese vecine, ultimul trebuie să fie același pt latura comună
- pot folosi o piesă de multe ori

Nu \exists nicio maximă T . ceea ce să calculeze în general răspunsul la întrebarea asta.

Bergen 

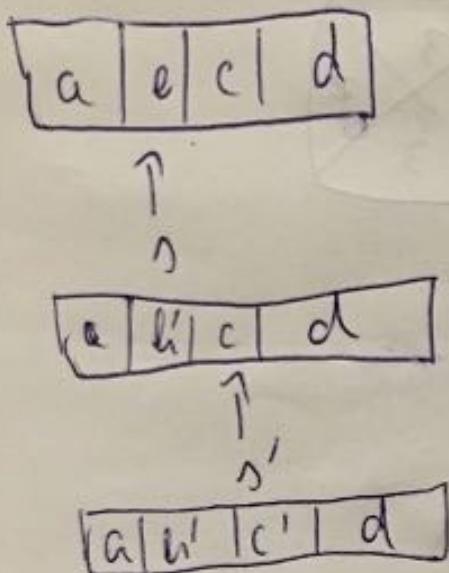
- există acoperiri care nu sunt periodice
- nu \exists un algoritm pt prob. acesta

Dacă ar exista un alg. atunci ar exista un alg și pt prob. opinii

Record: 11 piese cu 4 ușeri care acoperă planul neperiodic

CURS 5

"Dominantele Wong"



CAND (1902)

O maximă teuring care scrie cât mai mulți "1"

$S(m)$ nr de pasi maxim pe care ii poate face OMT cu m stări
înainte să se opreasă.

Vream OMT initial să înceapă cu m max de pasi înainte ca ea să se opreasă

O să arătăm că $S(n)$ nu este calculabilă de o MT

Dem:

$$S(1) = 1$$

$$S(2) = 6 \quad (\text{se vede asta, a fost demis} \text{ de către} \text{ Karatsuba})$$

$$S(3) = 21$$

$$S(4) = 107$$

$$\text{A fost demis că } S(5) > 47176870$$

$$S(6) > 8690.333.381\ldots$$

(T)

Functia care merge de la m la $S(m)$ nu poate fi calculata de nici o máximá Turing.

PP $M \rightarrow \frac{M}{S(m)}$ (PP că avem o MT care e calculabilă)

- arăt că dacă máxima asta ar există, atunci am putea avea și o MT care să calculeze problema oprișii, după care rățiu că nu există.

INPUT $y = \langle \alpha, x \rangle$

Fie m m de stări pt M_α
(calculat $S(m)$)

Simulez $M_\alpha(x)$ pt $S(m)$ pasi;

Se opreste \rightarrow returnez 1

Nu se opreste \rightarrow ret. 0

MULTIME RECURSIVĂ

RECURSIV ENUMERABILĂ

O mulțime $A \subset \Sigma^*$ este recursivă dacă $f_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$
poate fi calculată

$$K = \{ \langle \alpha, x \rangle \mid M_x(\alpha) \downarrow \text{ reprezintă}\}$$

\uparrow
nu e recursivă

$$g_K(\langle \alpha, x \rangle) = \begin{cases} 1 & M_x(\alpha) \downarrow \\ 0 & \cancel{\text{nu e}} \end{cases}$$

poate fi calc de o MT, dar MT nu e să se exprime

(DEF)

$A \subset \Sigma^*$ este recursiv enumerabilă

dacă $g_\alpha(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$ este calculată de MT

K = multimea predicilor $\langle \alpha, x \rangle$, nu e recursivă, e recursiv enumerabilă

$K \rightarrow$ cea mai grea problemă rec enum.



oprel A se reduce la oprel B ($\Leftrightarrow A \leq_m B$)

dacă $\exists f: \Sigma^* \rightarrow \Sigma^*$ calculabilă

astă $x \in A \Leftrightarrow f(x) \in B$

Recurivă = există un alg care îmi spune 1 sau 0

Rec enum = există un alg care are nevoie să nu spună nimic
când $x \in A$ e în f

(îmi dă răspunsul pe care l-a, și ciclează pe $\text{N}U$)

① K are propriețatea n. e. $A \leq_m K$ K completă pt re
ca

dacă îl stiu pe K (ea mai grea prol) pot să rezolv f

Bog pt K

INPUT $\langle \alpha, x \rangle$

Dacă $M_A(x) \in K$ return 1

Dacă $f(\alpha, x) = \langle \beta, y \rangle M_B(y) \downarrow$ return 0

⋮

K nec bo (contradicție)

Probleme rezolvabile = probleme recursive

Probleme parțial rezolvabile = probleme recursive cu program

Probleme nerezolvabile

Complexitatea Palmagorov

$$A = \overbrace{001} \overbrace{001} \overbrace{001}$$

$$B = 100111010 \rightarrow$$

Rezolvare e mai ușoară

A are 3 de 001

for $i \in 1, 3$

output "001"

$$K(x(g)) = \min \{ \exists \underline{\text{d}} \underline{\text{a}} \underline{\text{y}} \text{ s.t. } M_K(\underline{y}) = x \}$$

$$K(x) = K(x|\lambda)$$

(+) minimă a lui măsurăt programe generată X

$$K(x) \leq |x| + O(1)$$

"măsurăt x"

2^k programe de lungime $\ell \times$

2^k următoare pot genera cu programul

Cel mai eficient mod de a genera următorul e să il scriu direct (nu am o regulă) \Rightarrow următorul program

CURS 6)

- A recursivă ($\Leftrightarrow \beta_A(x) = \begin{cases} 1, & x \in A \\ 0, & x \notin A \end{cases}$)
 \downarrow
calc. de MT

O multime e recursivă dacă poate răspunde la întrebarea
"Apare în element multimi?"

- A recursiv enumerabilă $\Leftrightarrow \beta_A(x) = \begin{cases} 1, & x \in A \\ \text{calculată de MT}, & x \notin A \end{cases}$

Mă interesează să verific dacă $x \in A$, nu mă interesează dacă nu aparține.

maximă peste \times se oprește

Ex: $K = \{ \langle \alpha, x \rangle \mid M_\alpha(x) \downarrow \}$ este recursiv enumerabilă,
dar nu este recursivă

A rec. enum $\not\Rightarrow \bar{A}$ rec enum

A recursivă $\Rightarrow \bar{A}$ recursivă

T O multime A e recursivă dacă A este enumerabilă și
 \bar{A} este enumerabilă

Dem: $\Rightarrow M \quad \beta_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$

$M' \quad \beta_A(x) = \begin{cases} 1 & x \in A \\ \uparrow & x \notin A \end{cases}$

$\Leftrightarrow A$ rec enum

$M_1 \beta_A(x) = \begin{cases} 1 & x \in A \\ 1 & x \notin A \end{cases}$

\bar{A} rec enum

$M_2 \beta_A(x) = \begin{cases} 1 & x \in \bar{A} \\ 1 & x \notin \bar{A} \end{cases}$

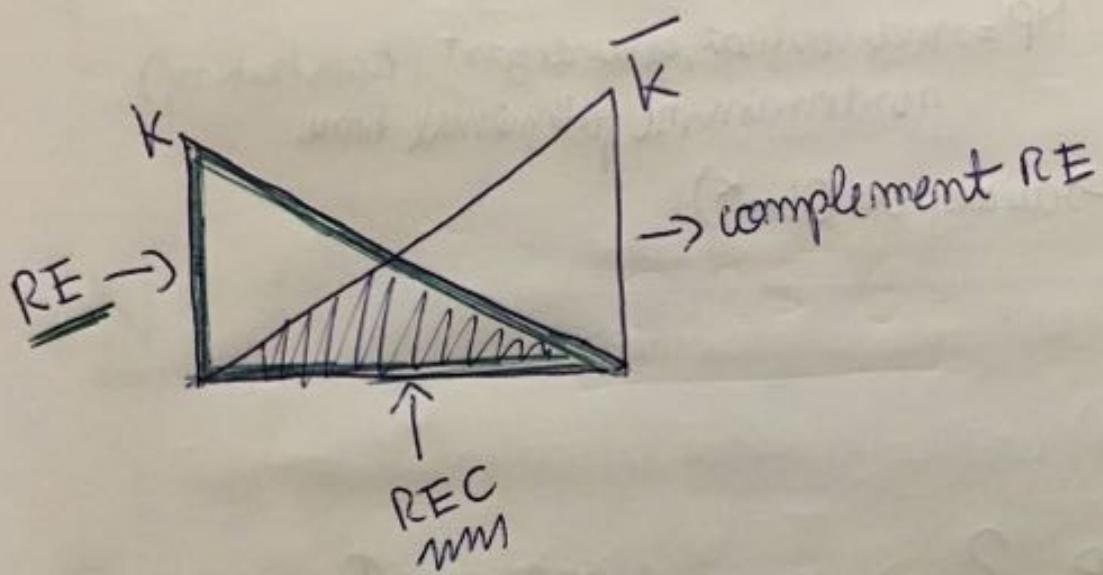
Uma dimbre maximă returnează 1, dacă M_1 returnează 1 atunci returnează 1, dacă M_2 returnează 1 atunci returnează 0

{ Simulează în paralel $M_1(x)$, $M_2(x)$

Dacă $M_1(x) = 1 \rightarrow$ return 1

Dacă $M_2(x) = 1 \rightarrow$ return 0

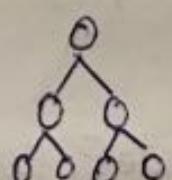
programul
care îmi arată
că A e recursivă



COMPLEXITATE COMPUTATIONALĂ

ALG EFICIENT \equiv complexitate $O(m^k)$ polynomial
 $k = 2, 3, \dots$

BACKTRACKING \equiv



$O(2^m)$ exponential și
nu este eficient

Complexitate polinomială vs exponentială

exponentială = să verific o soluție e ușor
să găsești o soluție e greu

NP = unde verificat, greu de găsit (backtracking)
monotonie polinomial time

(P) → analg. cu $O(m^k)$

P ≠ NP ? → premiu 1.000.000 \$ pentru rezolvare

$$M(x) \leq f(n) \text{ pasi}$$

$|x|=m$

M pe o intrare de 1000 de pasi să facă maxim $f(1000)$ pasi

OTIME (f) = {A | A poate fi rezolvată de o MT a î
 $M(x)$ face $f(|x|)$ pasi}

$P = \bigcup_{c \geq 1} \text{OTIME}(m^c) \rightarrow$ clasa problemelor cu complexitate m^c , unde c e constantă

Programare liniară : ex: alg. simplex

$$\begin{cases} \max(2x+3y) \\ x+y \leq 5 \\ 2x+y \leq 7 \\ x, y \geq 0 \end{cases}$$

NP (nondeterministic)

$L \subseteq \{0,1\}^*$ este în clasa NP (\Rightarrow există un polinom P

$$P: N \rightarrow N$$

$$\exists Q \text{ MT } a?$$

1) M(x,y) nulă în $O(|xy|)$ pasi

2) $x \in L \Leftrightarrow \exists u \in \{0,1\}^{P(|x|)}$ așa că $M(<x,y>) = 1$

witness
pt x

Dem: Îmlocuiesc variabilele din x cu valori specifice de m

Verifică $p(m) = \text{TRUE}$

Exemplu: Sudoku $m \times m$

Ex: (TSP) - comisul călător

- se dă matricea de distanțe
- de drivă: dacă există un circuit C pe graf
cu $\sum d(x_i, x_{i+1}) \leq c$
(trece prin toate varfurile și face suma să fie
în mult umăr dat)

x = codificare a imputului

u = codificare a circuitului

$M(x, u)$ verifică faptul că u este un circuit și că
lungimea lui u este $\leq k$ conform lui x

SEMINAR 3

INPUT: φ

DECIS: $\exists u \in \{0,1\}^*$ $\varphi(u) = \text{TRUE}$

$$L \subseteq \{x \mid \Delta A\}$$

O problemă de decizie = un limbaj pt care răspunsul e DA

Dacă L e regulat atunci am o funcție $\beta_L(x) = \begin{cases} 1 & x \in L \\ 0 & x \notin L \end{cases}$
ce poate fi calculată de un algoritm.

- simulez curîndul pe automatul x , dacă ajung în starea finală răsp e DA, altfel e NU

• O probă de decizie A se numește recursiv (\Rightarrow) dacă există (M)

$$M$$
 care calculează funcția $\beta_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$

• O probă recursiv oricărui lățu

$$\beta_A(x) = \begin{cases} 1 & x \in A \\ 1 & x \notin A \end{cases}$$

Ex: INPUT: $P(x_1 \dots x_m)$

DECIS: dacă $\exists y_1 \dots y_m$ a? $P(y_1 \dots y_m) = 0$

- complementul problemei cu patratele e nec enun
(adică să arăt că NU poate acoperi)

Dacă $A \sqsubset \bar{A}$ rec enun $\Leftrightarrow A$ rec

Nec enun, dacă nu rec : $k = \{x, x > |M_2(x)|\}$

Dacă ar fi rec am arăta și rez. pt prob. opriții.

Def: $A \leq_m B$ (\Rightarrow f calculabilă d o MT

aș $x \in A$ ($\Rightarrow x \in B$)

Dacă ar avea un alg pt B ar putea crea un alg pt

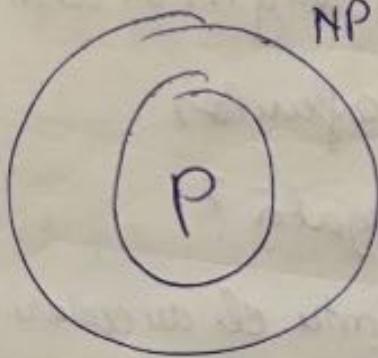
Ex: INPUT : (G, k)

DECIS : pe să nu să valorez G cu k ușori

Ex: $k=2$ ușori $\Delta 3 \cdot 2 = 6$ variabile

CURS 7

jyclixul. ? cod clasa NP



NP = clasa probleme de verificat rezolvare cu orice
datorită egru

$A \in NP \Leftrightarrow$ există un predicat $p(\dots)$ calculabil în
timp polynomial astfel încât $x \in A$ dacă și numai dacă $\exists y \in \{0,1\}^{p(x)}$

astfel încât $p(x, y) = 1$ mantenă $p(x)$

Ex: (1) Sudoku x = tablă parțial completată
 y = tablă completată

Sudoku ($n \times n$) este în clasa NP

(2) SAT x = formulă logică sau în variabile
 y = curând $y_1, \dots, y_n \in \{0,1\}$

Așa că în propoziție să văd dacă există o combinație de
valori care să o facă adevărată

3) TSP

4) Colorarea unui graf

$$x = (G, k)$$

$$y = \{1, \dots, k\}$$

cuv de lungime cu cel mai multă

Un marcat e o colorare a văzărilor

Cum verificăm că colorarea e legată?

↳ oricare 2vf cu muchie între ele au culori diferite

5) Compozits = $\{x \in \{0,1\}^n \mid \exists y, z \neq 1 \text{ a.s.t } x = y \cdot z\}$

Nim binar

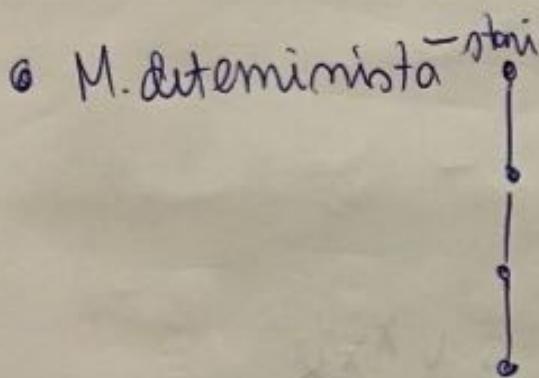
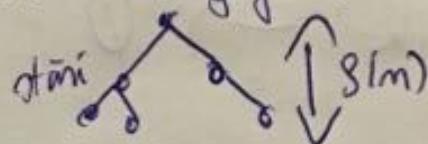
Um m_1, m_2 are lungime $\log(n)$
cu alte

NP = nondeterministic

Defin: Masina T nondeterministica = poate sa faca mai multe
chestii im raspuns cu aceiasi stare

Un automat finit nondeterminist poate fi similit de unul determinist, dar numărul de stări crește

- Maxima nedeterminanta = nu putem spune ce va face exact
Ex: un aparat de cafea care o dată dă cafea, altă dată ciorba sau

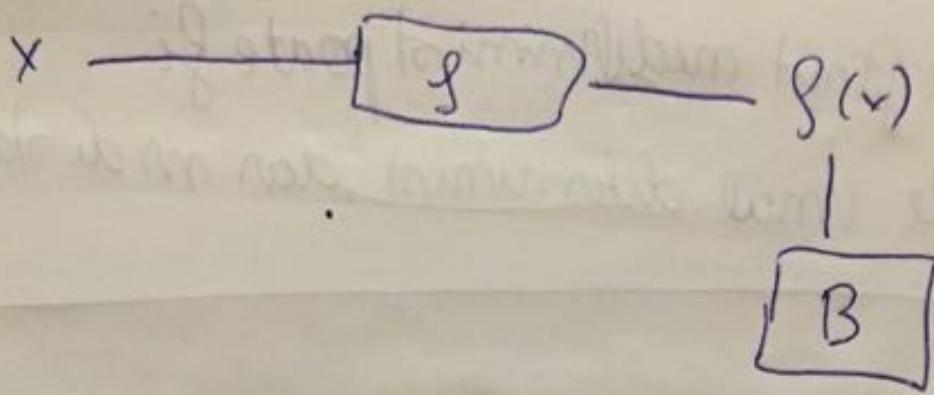


$A \in \text{NTIME}(f(n)) \Rightarrow$ M indă
M(x) face O(f(x)) pași a?

$$n = L(A) \quad (\text{număr de pași})$$

① $\text{NP} = \bigcup_{k \geq 1} \text{NTIME}(a^k)$

Def $A \leq_m^P B$ $\exists f: \Sigma^* \rightarrow \Sigma^*$
cu reducere
 $a)$ $\forall x \in \Sigma^* \quad x \in A \Leftrightarrow f(x) \in B$



Ex: colorarea grafulor se reduce la SAT

$$(G, k) \rightarrow \emptyset$$

$$x_{v,i} = \begin{cases} \text{TRUE } ((v)=i) \\ \text{FALSE } \text{ altfel} \end{cases}$$

θ

$$\begin{aligned} & x_{v,1} \vee x_{v,2} \vee \dots \vee x_{v,k} \\ & \bar{x}_{v,i} \vee \bar{x}_{x_j} \quad \forall x \in V \\ & \quad \forall 1 \leq i < j \leq k \end{aligned}$$

$$\overline{\overline{x}_{v,1} \vee \overline{x}_{x_j}} \quad , \quad \left. \right\}$$

Def A este NP-hard dacă A se reduce la B
 $\nexists B \in \text{NP} \quad B \leq_m^P A$

A este NP-completă dacă B se red la A

$\nexists A \in \text{NP} \quad \nexists B \in \text{NP} \quad B \leq_m^P A$

Orică problemă din NP se poate reduce la A

① SAT e NP completă

Consecință $P = NP \Leftrightarrow$ SAT are alg. polinomiali

TSP, colorare \vdash NP complete

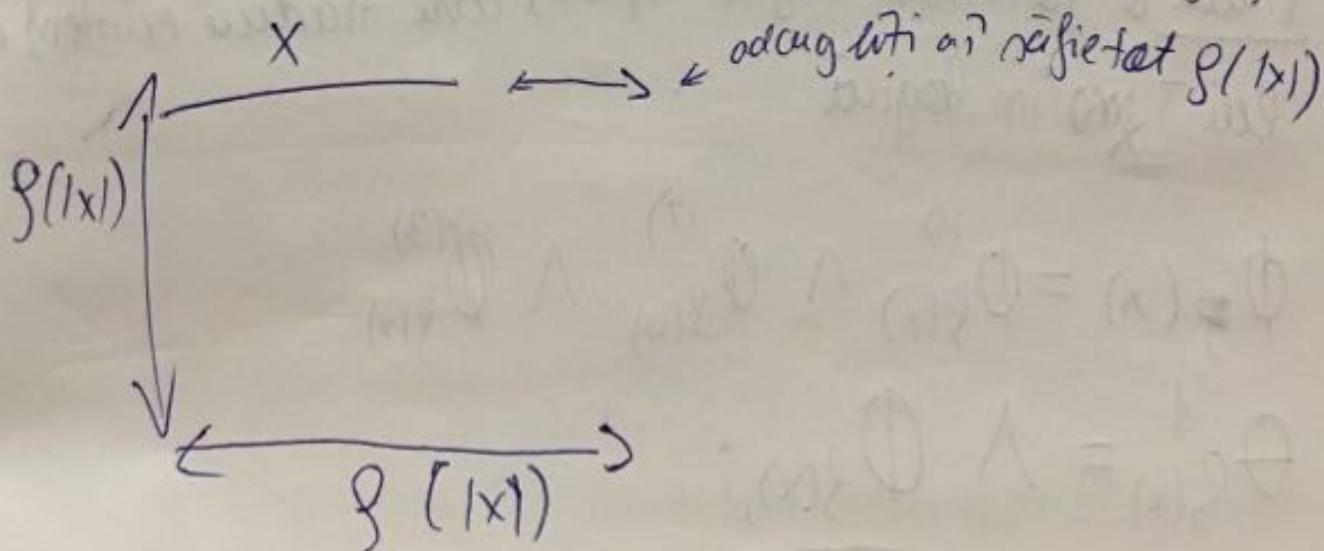
Sfîrșit de demonstrare:

Fie M o mașină T. nondeterministă care pe intrarea x face cel mult $p(|x|)$ pași.

Vreau să arătă $L(M) \leq_m^P \text{SAT}$

$x \in \Sigma^* \longrightarrow \Phi(x)$ formula logică

ar $M(x)$ acceptă ($\models \Phi_{p(x)}$) satisfacă



x_5	x_1	x_2	\dots	x_m	\square	\square	\square
-------	-------	-------	---------	-------	-----------	-----------	-----------

PP: M dă acceptă atunci capul ei e pe prima poziție

accept

$x \in I(M) \Rightarrow T \text{ (abilitate) } a?$

- 1) T pe prima linie codifică $M(x)$ în starea inițială
- 2) T pe ultima linie codifică $M(x)$ acceptat
- 3) T pe linii consecutive codifică o mișcare legală a lui M conform lui δ_M (funcția de transiție a mașinii)

Vinean o formulă logică $T_F(x)$ care traduce existența lui $T(x)$ în logică

$$\Phi_{\#}(x) = \Phi_{g(x)}^{(1)} \wedge \Phi_{g(x)}^{(2)} \wedge \Phi_{g(x)}^{(3)}$$

$$\Phi_{g(x)}^1 = \bigwedge_i \underbrace{\Phi_{g(x), i}}_{\text{căuta } i \text{ de pe prima linie}}$$

$$\boxed{\square} \leq 2^k \quad \theta_{f(x), 1}(\underline{y_1, y_2, \dots, y_k})$$

bitii codificații

Bitii imi codifică o configurație posibilă a omului adăpostit.

O să scriu o formulă de genul păfiecare cîștigă

LAB 4

$A \in \text{SAT}$

$$(G, 2) \rightarrow \emptyset_{G, 2}$$

$$(G, 3) \rightarrow \emptyset_{G, 3}$$

$$\Delta. (v, i) \quad x_{v,i} = \begin{cases} \text{TRUE} & c(v)=1 \\ \text{FALSE} & \text{altfel} \end{cases}$$

$\emptyset_{G, 2}:$

1) $x_{v,1} \vee x_{v,2} \vee \dots \vee x_{v,k}$ fiecare varf ia maximum celor k

2) $\bar{x}_{v,i} \vee \bar{x}_{v,j}$ varful v urmăriște 2 culori

3) (v, w)
 multie $\bar{x}_{v,i} \vee \bar{x}_{w,i}$ nu au aceeași culori
 $\bar{x}_{v,k} \vee \bar{x}_{w,k}$ alternează culori

from pysat.solvers import Minisat22 \rightarrow solver

From pysat.formula import CNF, CNFBool, IDPool

def main():

 vertices = [1, 2, 3] \rightarrow noduri
 edges = [(1, 2), (2, 3), (1, 3)] $\left\{ \begin{array}{l} \text{am codat graful cu} \\ \text{vectori nimatrice} \end{array} \right.$
 no_colors = 2 \rightarrow muchii

cnf = CNF()

\uparrow Variabila care initial e o formula goala, initializam formula

vpool = IDPool (start_id=1)

creare numere de variabile inceand cu 1 (x_1, x_2, \dots)

$$x_1 \vee x_2 = \{1, 2\}$$

colors = lambda i, j: vpool.id('x' + str(i) + 'v' + str(j))
\downarrow \uparrow $\overline{x}_1 \vee \overline{x}_2 \vee x_3 = \{-1, 2\}$
 $x_{i,j}$ reprezinta o valoare care poate fi 1 sau -1

for i in vertices: \rightarrow adaug clauza 1

cnf.append([(colors(i, j)) for j in range(no_colors)])

CURS 8

(T)

3-SAT este NP-completă

$$\left\{ \begin{array}{l} \text{Dc A este NPC} \\ \text{B este în NP} \\ A \leq_{P_m}^{\rho} B \end{array} \right| \Rightarrow B \text{ este NP-completă}$$

Asta înseamnă că SAT $\leq_{P_m}^{\rho}$ 3-SAT

ϕ_0

$$x \vee y \vee \bar{z} \vee t$$

$$x \vee \bar{y} \vee \bar{z} \vee \bar{t} \vee m$$

$$\left| \begin{array}{l} g(\phi_0) \\ x \vee y \vee \alpha \rightarrow \text{variabilă nouă} \\ \bar{z} \vee \bar{t} \vee t \\ \hline \begin{array}{l} x \vee \bar{y} \vee \bar{z} \vee \alpha \\ \bar{z} \vee \bar{t} \vee m \\ \hline x \vee \bar{y} \vee \beta \end{array} \\ \bar{\beta} \vee \bar{z} \vee \alpha \\ \bar{\alpha} \vee t \vee m \end{array} \right.$$

Obs: 2-SAT ∈ P

(T)

k -SAT $k \geq 3$ NP-completă

$$x \vee y \vee z$$

$$x \vee y \vee z \vee \alpha$$

$$x \vee y \vee z \vee \bar{\alpha}$$

Obs: Dc $P \neq NP \Rightarrow \exists A \in NP$

$A \notin P, A$ nu e NP-completă

① Variantele SAT sunt fără în P și în NPC

Ex 1-im-K SAT ($K=3$)

Se dă ~~clauze~~, $|C_i| = 3$ (variabile), $\Phi = C_1 \wedge C_2 \wedge \dots \wedge C_n$

Vrem: să assignăm A pt variabilele x_1, x_2, \dots, x_n care satisfac exact un literal în fiecare clauză.

② 1-im-3 SAT este NPC

Dem: 3-SAT \leq_m^P 1-im-3 SAT

Φ	$\{S(\emptyset)$	$R(x, y, z) = \text{"exact one true value among } x, y, z\text{"}$
$x \vee y \vee z$	$R(x, a, d)$ - variabilă	
	$R(y, b, d)$	
	$R(a, b, c)$	
	$R(c, d, g)$	
	$R(g, e, FASLE)$	

③ ILP este NP-completă (integer linear programming)

Se dă un program linear + restricții de integralitate

Decid $OPT(\text{integ}) \leq \alpha$

Dem: SAT \leq_m^P ILP

$$\begin{cases} \min (2x + 3y + 4z) \\ x - y + 7 \geq 5 \\ x + y + z \geq 10 \\ x, y, z \geq 0 \end{cases}$$

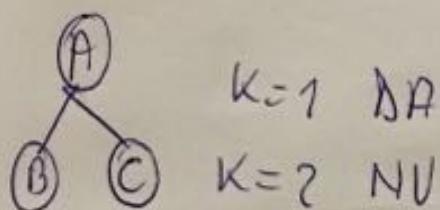
$$\begin{array}{|c|c|} \hline \emptyset & S(\emptyset) \\ \hline x \vee y \vee z & x + y + (1-z) \geq 1 \\ & x, y, z \in \{0, 1\} \\ \hline \end{array}$$

Independent ... (is)

Se dă $G = (V, E)$ graf

$$1 \leq k \leq n$$

definim $\exists w \subseteq V, |w|=k$ astfel încât $(a, b) \in E, a \notin w$ sau $b \notin w$



⑦ is este NPC

$IS \in NP(G, k)$ Multor $w = w_1, \dots, w_m \in \{0, 1\}^m$

$$\text{i)} \sum w_i = k$$

ii) w codifică un IS

$3-SAT \leq_m^P IS$

∅ m-clause

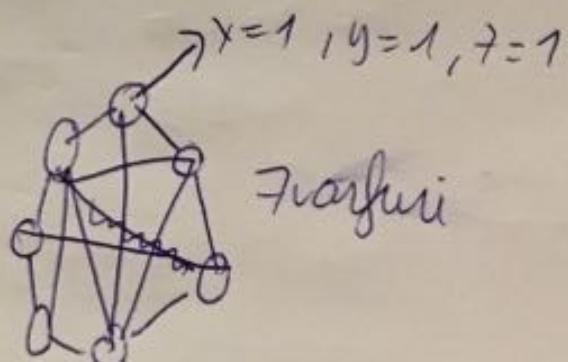
m = variabile

$C = x \vee y \vee z$

\exists combinații care satisfac C

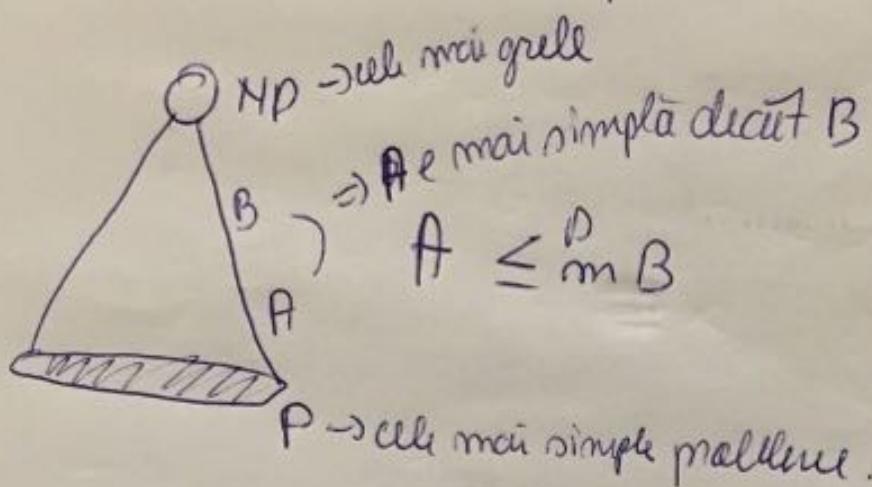
G_k

toate multibile posibile



CURS 9

Demonstrati de NP-complitudine



Problema dim P pot să o reduc la una dim NP

Dacă $A \in P$, $B \neq \emptyset$, Σ^* atunci $\forall B \in NP$, A se reduce la B

Dem: Fie M o mt polinomială care decide pt A

$$f: x \rightarrow f(x) \text{ a.i } x \in A \Rightarrow f(x) \in B$$

Fie $x_0 \in B$

$x_1 \notin B$

Plig pt funcția f :

$x \notin A$

INPUT x adică $x \in A$
 rulez $M(x)$ \leftarrow

Dacă $M(x) = 1 \Rightarrow$ return x_0

Pentru return x_1

Dacă $x \in A$ atunci $g(x) = x_0 \in B$

$x \notin A$ atunci $g(x) = x_1 \notin B$

Să asta e fix ce înseamnă să arăt că nu văd că problema are o reducere la NP ~~(timp polinomial)~~

NP - completă

oricare și B din NP ,
 B se reduce la A

A este NPC dacă $A \notin \text{NP}$ și $\forall B \in \text{NP} \quad B \leq_m^P A$

Lemă: $A, B \in \text{NPC} \Rightarrow A \leq_m^P B \wedge B \leq_m^P A$

Adică toate problemele NPC sunt la fel de grele

Def: $A = P_m^B \Leftrightarrow A \leq_m^P B \wedge B \leq_m^P A$

T) Fie A NPC, atunci $P = \text{NP} \Leftrightarrow A$ are algoritm stabil

Că să arăt $P = \text{NP}$ trebuie să dau un algoritm stabil pentru NP .

Noi creem totuși că $P \neq \text{NP}$, atunci un NP nu are algoritm stabil.

Atâtă timp că $P \neq \text{NP}$ nu există algoritmi n^2, n^3 etc pentru o problemă NPC.

⑦ SAT este NPC

⑦ Problema INDEPENDENT SAT este NPC

Să dă un graf și un $K \geq 1$

De decis: are G o multime S de varfuri

$$|S| = \underline{k}$$

cardinal

at ~~traversări~~ ~~nu~~ ~~nu~~ ~~v₁ ES sau v₂ ES?~~
orice ar fi o muchie $t(v_1, v_2)$ $v_1 \notin S$ sau $v_2 \notin S$

Alg complexitate $O(n^k)$

Înțelegem toate $S \in V$

$$|S| = k$$

k e oricără de mare deci problema nu e polinomială
pot să o rezolv pentru k mic

Dem $3\text{sat} \leq_m^P IS$

$\emptyset \rightarrow$ G_\emptyset, K_\emptyset

$x \vee y \vee z$

7 soluții ($x=1, y=1 \dots$ toate sunt OPO)

Claim

di困难度

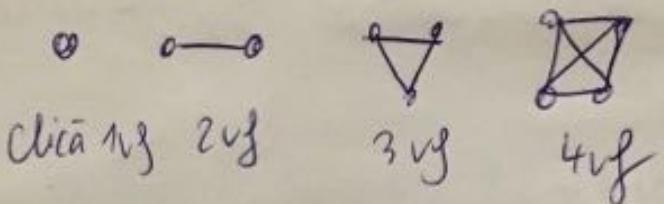
$= S$

CLIQUE

Sedă (G, k)

Vream $\exists S \subseteq V |S|=k$
ăndicăd at $\forall x, y \in S (x, y) \in E$

cliquer = vîf conecitate între ele



① Clique este NPC

Dm: i) clique $\in NP$

ii) $|S| \leq p_m$ clique

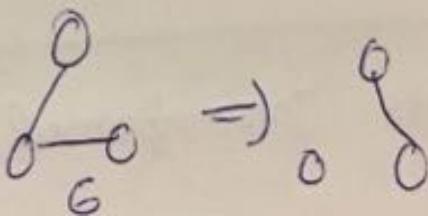
1) utmres $y_1 \dots y_n \in \{0, 1\}^M$

$$y_i = \begin{cases} 1 & \text{vî } i \in S \\ 0 & \text{altele} \end{cases}$$

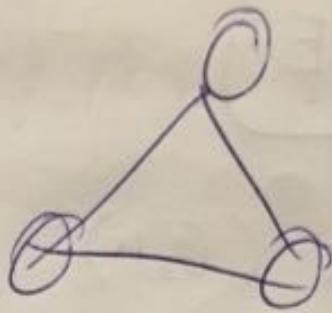
at $y_i=1 y_{j-1} \Rightarrow (v_i, v_j) \in E$

2) $(G, k) \rightarrow (G', k')$

$$k' = k$$



VORTEX COVER



Sẽ dã (G, k)

Vrau $S \subseteq V$

$$|S|=k$$

at $V(v_1, v_2)$ mudié \hat{d} n G

$v_1 \in S$ na $v_2 \in S$

$$|S|=2 \Rightarrow DA$$

$$|S|=1 \Rightarrow NV$$

①

VC este NPC

VC \in NP lafel

$$|S| \leq P_m^m |VC| \quad (G, k) \rightarrow (G', k')$$

$$G' = G$$

$S \subseteq V$ este $|S|$ im G

$$k' = m - k$$

②

V is este VC im $G' = G$

În practică: $A \in NP \rightarrow A \in P$ sau A este NPC

① $P \neq NP \Rightarrow \exists A \in NP$

- A nu este NPC
- $A \notin P$

PT problema satisfiabilității

PT (Schaeffer, 1977)

SAT(S) 3-SAT $S = \{x \vee y \vee z, x \vee y \vee \bar{z}, x \vee \bar{y} \vee z, \bar{x} \vee \bar{y} \vee \bar{z}\}$

$\left\{ \begin{array}{l} \text{am 3 variabile} \\ \text{am minima negată, una negată, 2 negate, 3 negate} \end{array} \right.$

Fișe S ⊂ multimea de constanțe

$$S = \{c_1, c_2, \dots, c_m\}$$

• c_1, c_2, \dots, c_m satisfăcute de 0...0

$$\Downarrow \\ SAT(S)_{sa} \Rightarrow SAT(S) \in P$$

• $c_1, c_2, \dots, c_m \quad c_i \equiv \emptyset; \in 2\text{-SAT}$

$$SAT(S) \in P$$

• $c_1, \dots, c_m \quad c_1 \equiv \emptyset, c_1 \in HORN$

$$SAT(S) \in P$$

$$\bar{x} \vee \bar{y} \vee \bar{z}$$

$$\bar{x} \vee \bar{y} \vee \bar{z} \vee t$$

- II - $c_i \equiv \phi_i \in \text{Neg-Horn} \leq 1$ van negativ
 $SAT(S) \in P$
- + II - $c_i \equiv \phi_i$ ϕ_i sistem ec liniar este Σ
 $SAT(S) \in P$
- ? În toate celelalte cazuri $SAT(S)$ este NPC

Văde sunt instanțele care
mai grele pt o plă NPC?

3-SAT: NPC \Rightarrow există formule de tip 3-SAT
care sunt „dificile”

$$\emptyset \Rightarrow \emptyset_1 = \emptyset \wedge c$$

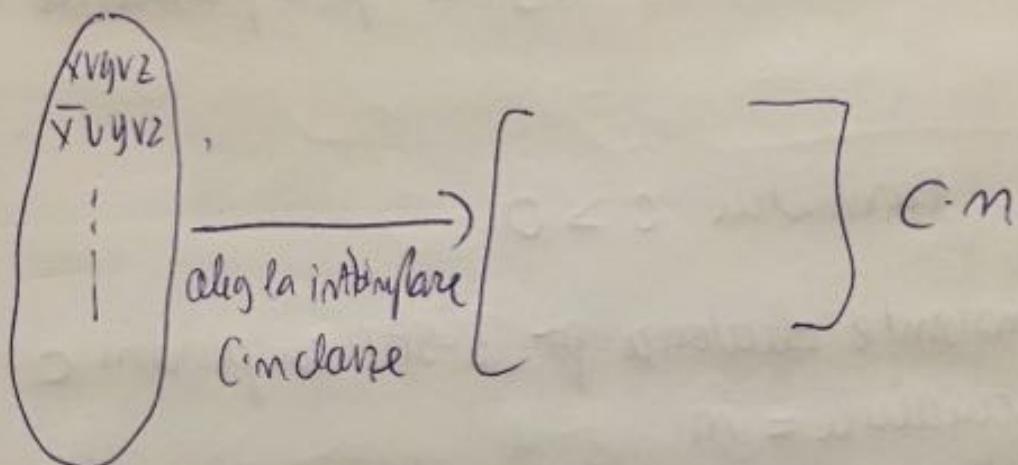
\emptyset_1 "mai puțin satisfacțională" decât \emptyset

$$\boxed{\emptyset_1 \in SAT \Leftrightarrow \emptyset \in SAT}$$

Generări formula:

$$c = \frac{\# \text{clase}}{\# \text{variabile}}$$

$(m, c) \Rightarrow \emptyset$ la întâmplare cu m variabile și $c \cdot m$ clase



Probabilitatea ca o formulă să aibă soluții?

Când $c \nearrow p_2(\emptyset \text{ ESAT}) \searrow$

Intuitie: Multe pă NPC au "hamșuri de foșă"

Cele mai multe grile carei ale căror pă NPC

sunt în jurul trădi foșă

Există în foșă foșă pă dificile

Ex: 1im3 SAT este NPC

dacă în medie 1im3 SAT este preliminară

CURS 11)

Vnde sunt instanțele "dificile" pt a problemă NP-completă

3-SAT Parametrul $c > 0$

Generez instanțe aleatoare pt 3-SAT cu param c
{\$ variabile = m
clause = c * m

Cu cât am mai multe clause într-o formulă cu atât e mai puțin probabilitate să fie satisfacabilă.

Multe condiții \Rightarrow greu de îndeplinit

C mare \Rightarrow formule cu soluții satisfacibile puține

îNVIȚIIE : Multe probleme NPC au proprietăți de faza "cel mai greu" care se întâlnesc la formuția de fază

DPLL - Davis-Putnam-Loveland-Loveland

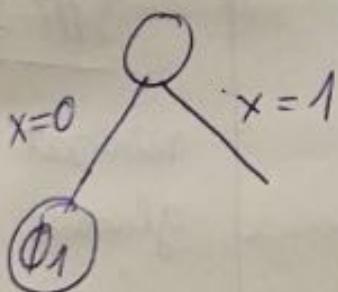
- ↳ algoritm pentru SAT (backtracking)
- ↳ complet (răspunsul DPLL sau HV e întotdeauna corect)
- ↳ reieșă pînă la fin în răspuns

SAT $\phi(x_1, \dots, x_n)$ formă normală conjunctivă (CNF)

$$\Rightarrow \theta = c_1 \wedge c_2 \wedge \dots \wedge c_m \leftarrow \text{clase}$$

unde o clasă este $\bar{x} \vee y \vee \bar{z}$

$$\phi \left[\begin{array}{l} x \vee \bar{y} \vee \bar{z} \\ \bar{x} \vee y \vee z \\ \bar{x} \vee \bar{y} \vee z \\ z \vee \bar{x} \vee \bar{z} \end{array} \right] \left\} \text{clase} \right.$$



$$\phi_1 = \begin{cases} \bar{y} \vee \bar{z} & \text{satisfăcută} \\ z \vee \bar{x} \vee \bar{z} & \text{satisfăcută} \end{cases}$$

① Literal pur

$x=0$ satisfăcă toate clasele în care apare x, \bar{x}

Def: x literal pur dacă x apare doar pozitiv sau doar negativ în ϕ

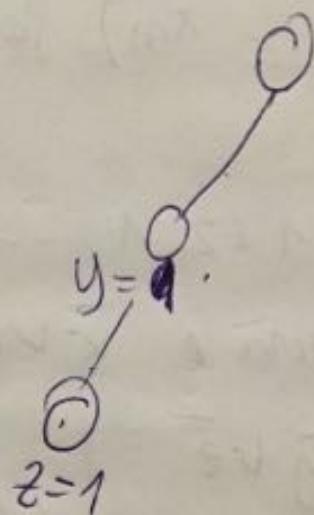
Regula 1: Dacă x literal pur îl alăt pe x ca variabilă
și nu fac backtracking după x

② x LITERAL UNITAR

$x \vee \bar{y} \vee \bar{z}$

$y \vee c_1$

$z \vee c_2$



[DPLL] backtracking

+

clăuse pure

+

clăuse unitare

transfarea de forță \rightarrow alg de tip DPLL

SAT SOLVERE

minisat

glucose

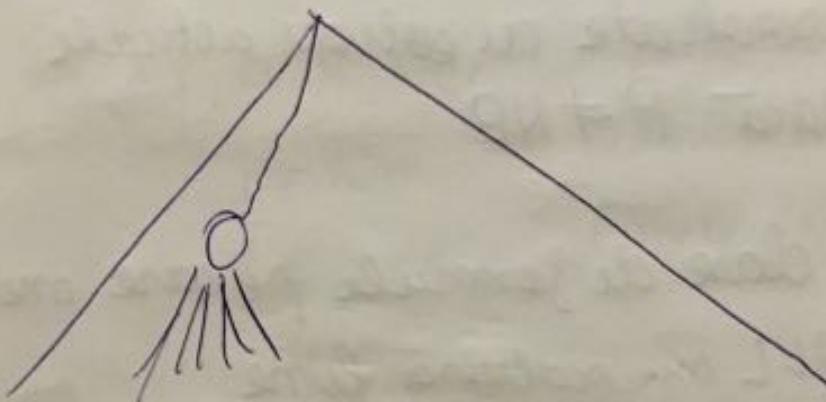
!

\searrow
casăile
pe DPLL

idee fundamentală pt accelerarea algoritmului DPLL

"conflict-driven clause learning" \rightarrow CDCL

Hu am găsit sol
Păt invata
cva dim eșec?



$$x = T \quad y = T \quad z = F \quad t = T$$

creare o contradicție

Păt invata săptul căcerice xul pt Ø subfațe

$$C' = (\bar{x} \vee y \vee \bar{z} \vee t)$$

adaug C' la Ø când explorez

Concluzii:

1. Reg practici pt SAT = rezolvare casati pe paradigma DPLL
2. Curse learning \rightarrow (C&CL) principala idee experimentala
3. Multe instante de SAT care vîm din practică pot fi rezolvate cu solvare actuale chiar dacă $P \neq NP$
4. Există clase de formule pe care metoda de tip DPLL nu scădere lume
 \downarrow
polynomial

De ce există clase de formule pe care DPLL nu le rezolvă?

=)

Ex: - (Principiul cutiei / principiul lui Dirichlet)

m permutări im $m-1$ cînd
formulă proporțională PHP $\xrightarrow{m-1}$ rezolvabilită $\} \Rightarrow$ imposibil să a
avea 2 permutări im
același răsta

| Pâră de curând $n=15$ prea mult pentru
• toate solverele din ziua de azi

$$\text{PHP}_{n,n-1} \quad x_{i,j} = \begin{cases} 1 & i \rightarrow j \quad i = \overline{1, n} \\ 0 & \text{altfel} \quad j = \overline{1, n-1} \end{cases}$$

Exp $n=15$ 15×14 variabile

c1 $x_{i,1} \vee x_{i,2} \vee \dots \vee x_{i,n-1}$ permutulul i merge de pe locul i într-o casă

c2 $\bar{x}_{i,a} \vee \bar{x}_{i,b}$ permutulul i nu merge simultan în acile a și b
 $i = \overline{1, n}$
 $1 \leq a, b \leq n-1$ (merge exact într-o casă)

c3 $\bar{x}_{i,a} \vee \bar{x}_{j,a}$ permutui i și j nu merge simultan în aceeași casă
 $1 \leq i < j \leq n$
 $1 \leq a \leq n-1$

$\Rightarrow \boxed{\text{PHP}_n^{m-1} \notin \text{SAT}}$

Timpul de rulare $DP(\text{PHP}_n^{m-1})$ - exponential cu n

Explicație: COMPLEXITATEA DEMONSTRATIILOR PRIN REZOLVĂRE

$$\underline{\text{rezolutie}} \cdot \underline{x v c_1} \quad \underline{\bar{x} v c_2}$$

\oplus $\emptyset \in SAT \Leftrightarrow \emptyset + \text{rezolutie } \square$

Mă interesată

$$(\Phi_n) \quad \text{Exp. } (\text{PHP}_n^{n-1})$$

$L_m = \text{nr de cluse în ea mai scurtă}$
 $\text{demonstrație de neatisfiabilitate}$
 din rezolutie

$$\underline{c_1 \ c_2 \ \dots \ c_m \ \text{rez } (c_1, c_j) \ \dots} \quad \square$$

L_m

lungimea cea mai mică de două
 păru formule nesat.

\oplus $\exists c > 0 \text{ astfel că } L_m(\text{PHP}_n^{n-1}) \geq 2^{c \cdot 1}$

("complexitatea demonstraților")

\Downarrow
 DPLL?

Θ $L_m \leq$ running time storice alg de tip DPLL

Dem

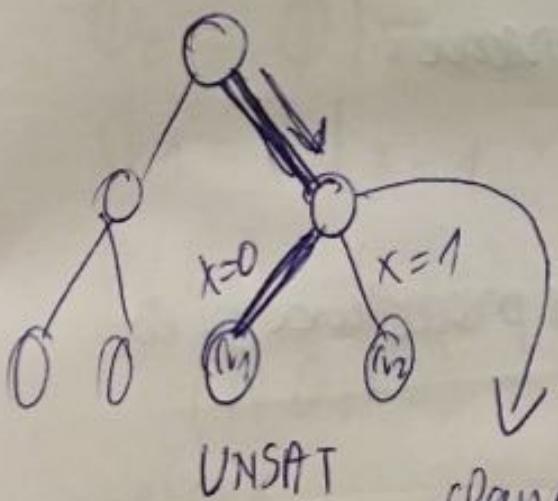
Idee: iau un arbore de BT pt $A(\emptyset_m)$

$A(\emptyset_m)$



construiesc o dem de resatisfabilitate
pt \emptyset_m prim rezolutie

in care m clause = m noduri in arbore



pt fiecare frunza există o clauză
CV în \emptyset menită să respectă

$$C_{V_1} = x \vee D_1$$

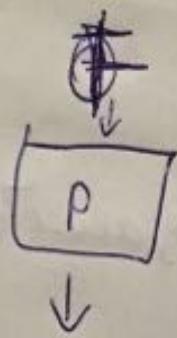
$$C_{V_2} = \bar{x} \vee D_2$$

clauză $D_1 \vee D_2$ (res((V_1, V_2)))
menită să dețină partea

În felul asta am dem

SEMINAR 5

IP: Presupunem că avem o procedură P

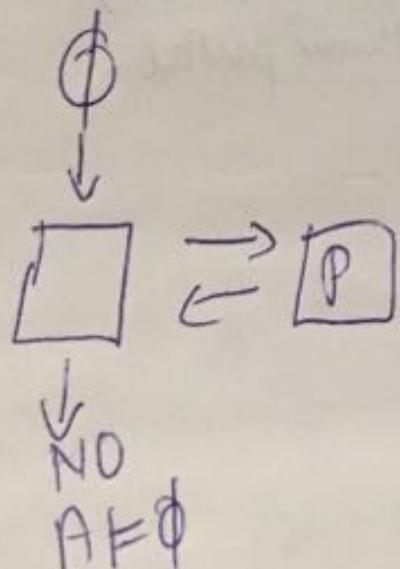


YES/NO

Pentru a face procedura de ori căte ori vreau

Emin: Scrieți un program care făcăsește procedura și ea
iese în serie HV sau o soluție.

Program eficient (HV Backtracking)

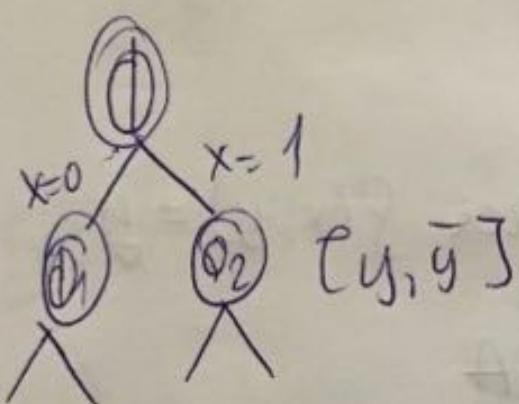


Văd pe ce ramură
procedura dă DA și
nu pe ramura altă.

Dacă pe nicio rami dă DA,
atunci se leag HV

$$\left\{ \begin{array}{l} x \vee y \\ \bar{x} \vee y \\ \bar{x} \vee \bar{y} \end{array} \right.$$

x	y	\emptyset
0	0	0
0	1	1
1	0	1
1	1	0



↓

dacă toate datele dau 1,
atunci primem 1 la \emptyset

INPUT \emptyset

Aleg \emptyset var X

$$\emptyset_0 = \emptyset |_{x=0}$$

$$\emptyset_1 = \emptyset |_{x=1}$$

$$P(\emptyset_0), P(\emptyset_1)$$

$$(NO, NO) \rightarrow NO$$

altfel

$$(DA, NO) \rightarrow x=0 \rightarrow \emptyset_0$$

$$(ND, DA) \rightarrow x=1 \rightarrow \emptyset_1$$

② Def: f este P -selektivă

$\Leftrightarrow \exists g(\cdot, \cdot)$ calculabilă în timp polynomial
așa că

• $\forall x, y \quad g(x, y) = x$ sau $g(x, y) = y$

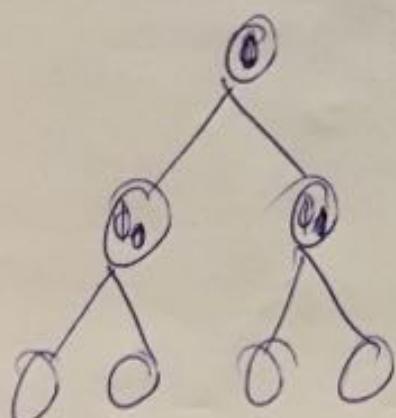
• $x \in A \vee y \in A \Rightarrow g(x, y) \in A$

\downarrow
dăm mai prealabil să
fie în A

De exemplu:

SAT este P -selektivă (SAT $\in P$)

Indicatie:



Dacă SAT ar fi P -selektivă $\Rightarrow g(\emptyset, F) = \emptyset \neq F$

Dar nu \Rightarrow nu ar fi o algoritm polynomial \Rightarrow SAT nu este un algoritm polynomial

DPLL

- backtracking

- lit pur: × apare doar pozitiv în formula
negativ

- lit unitar

Ex: $\emptyset = (P \vee Q \vee \bar{R}) \wedge (P \vee \bar{Q}) \wedge \bar{P} \wedge R \wedge U$

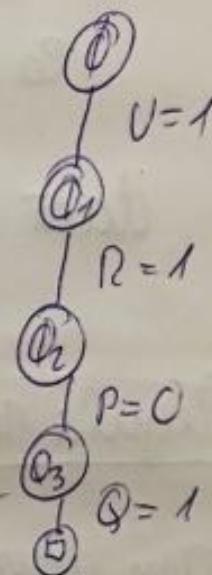
dacă $U=1 \Rightarrow \emptyset = (P \vee Q \vee \bar{R}) \wedge (P \vee \bar{Q}) \wedge \bar{P} \wedge R$

$R=1 \Rightarrow \emptyset = (P \vee Q) \wedge (P \vee \bar{Q}) \wedge \bar{P}$

$P=0 \Rightarrow \emptyset_3 = Q \wedge \bar{Q}$

$Q=0 \quad \left\{ \begin{array}{l} \emptyset_4 = \square \\ Q=1 \end{array} \right.$

\Rightarrow Formula \emptyset nu e SATISFACITĂ



Ex: $\emptyset = (P \vee Q) \wedge \emptyset \wedge (\bar{P} \vee Q \vee \bar{R})$

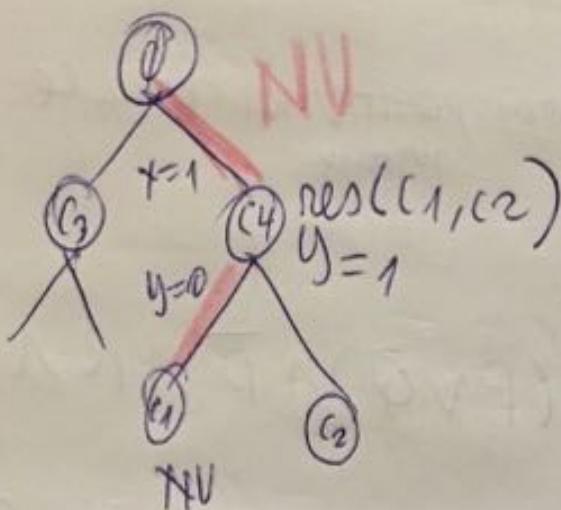
P nu e pură că apare și \bar{P} , Q nu e pură, R nu e
pură
Q e unitară

⇒ soluție OK, gen $\bar{R}=1$, e OK

$R=0 \quad \left| \begin{array}{l} \text{OK} \\ \Rightarrow \end{array} \right. \quad \emptyset_3 = P$

$P=1 \Rightarrow$ soluție

Ex Am un arbore DPLL care stiu că e unsat



\Rightarrow există o clauză care nu e satisfăcută de $x=1, y=0$
Ca ar fi clauza de genul $\bar{x} \vee y$ care dă 0

dacă $x=1$ și $y=1 \Rightarrow c_2 = \bar{x} \vee \bar{y}$ nu e sat

Regula rezolvării

Am clauzele $c_1 (\bar{z} \vee D_1)$ $c_1 (\bar{z} \vee D_2)$

$$\frac{}{D_1 \vee D_2}$$

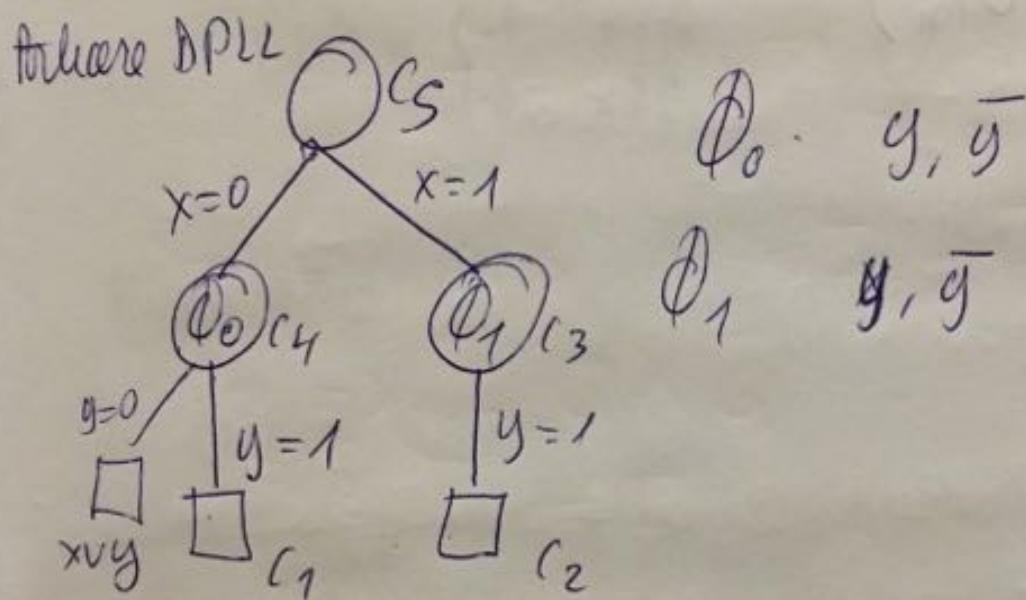
prin rezolvare (nu pun deluluri
dacă sunt)

$$res(\bar{x} \vee y, \bar{x} \vee \bar{y}) = \bar{x} \text{ și cum } x=1 \Rightarrow \text{nu e sat}$$

lungimea celui mai scurtă \leq nr moduri DPLL
rezolvării

$$\text{Ex } \emptyset = \begin{matrix} x \vee y \\ \bar{x} \vee y \\ x \vee \bar{y} \\ \bar{x} \vee \bar{y} \end{matrix} \quad \begin{matrix} \text{nu e SAT} \\ \text{ca se exclude una pe alta} \end{matrix}$$

Nu avem literal pur, nu impar



$$C_1 = x \vee \bar{y} \quad (\text{clasa rez de } x=0, y=1)$$

$$C_2 = \bar{x} \vee \bar{y} \quad (x=1, y=1)$$

$$C_3 =$$

$$\text{res}(\bar{x} \vee y, \bar{x} \vee \bar{y}) = \bar{x}$$

$$\text{res}(x \vee \bar{y}, x \vee y) = x$$

$$\text{res}(x, \bar{x}) = \square$$

③ Ø Clause de lungime 2 \Rightarrow există cel mult de
n sat și O(n²) cluse
m variabile

$$\text{res}(x \vee y, \bar{x} \vee \bar{y}) = x \vee \bar{z}$$

$$4(n^2) = O(n^2)$$

CURS 11

Problema de decizie,

- se dă un input arhital $A \subset \Sigma^*$
- se așteptă un răspuns de tip DA sau NU

$$f(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases}$$

Ex: NPC

CLIQUE

input: $G = (V, E)$ graf

diciu: $\exists S \subseteq V \quad |S| = k \text{ astfel încât } \forall v, w \in S \quad (v, w) \in E$

se dă un graf și vrem să găsim dacă există varfură conectate între ele (k varfuri)

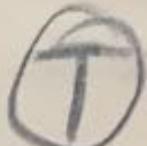
pt $k=4$



clique $\Theta(n^k)$

Algoritmul Brute Force incercă toate combinațiile

Nu e polinomial pt că k poate varia nici pe de fi oricăr



CLIQUE NPC

Pe naturală Găsim o clăcă maximă

$$\nexists T \mid |T| = |S| + 1 \quad T \text{ nu e clăcă}$$

$$\exists w P(G(v, w))$$

$$\exists w \forall T P(G, v, w, T)$$

Cânduzie: Problema Clique \neq Max Clique pentru că prima dpdv logic este de tipul existential }
și a doua e existată oricare $\exists x$

O paralelă imhe $\frac{\text{rec}}{\text{rec}} \text{ si } \frac{MP}{P}$

• Recursivă: O multime e recursivă când $P(x) = \{0\} \cup$
UT care se oprește pe orice input și mediu nu spune
DA sau NV în numar finit de pasi pt orice x

• multime $\in P$: $A(x) = \begin{cases} 1 \\ 0 \end{cases}$

- există o MT care se oprește imediat după $P(|x|)$ pasi
 \downarrow
 reacție

- în P nu interesează că MT să se oprească într-un timp mult foarte mare (nu doar să se oprească)

• multime = rec. binar : $A(x) = \begin{cases} 1 & x \in A \rightarrow \text{MT se oprește} \\ 0 & x \notin A \rightarrow \text{nu se întâlnește} \end{cases}$

- există o MT care se oprește numai în cazul în care răspunsul e DA

• multime $\in NP$: $A(x) = \begin{cases} 1 & x \in A \rightarrow \text{MT se oprește în} \\ & \quad \text{temp} \leq P(|x|) \\ 0 & x \notin A \rightarrow \text{nu se întâlnește} \end{cases}$

- există o MT care se oprește când $x \in A$ în temp polinomial, și disipație maximă care ia x și witnessul w $M(x, w)$.
 \downarrow pe care nici-l nu găsești

$|w| \leq p(|x|)$ pt ca $M(x, w)$ trebuie să se oprească în $\leq p(|x|)$

NP = clasa multimilor \mathcal{F} (prelimelor) pt care \exists un predicat $P(\cdot, \cdot)$ calculabil in timp polynomial

$x \in \mathcal{A} \Leftrightarrow \exists w \quad |w| \leq g(|x|) \quad P(x, w) = \text{TRUE}$
 lungimea witnessului nu este prelungită și fiind mare

Re = \exists predicat recursiv $P(\cdot, \cdot)$ astfel încât
 $x \in \mathcal{A} \Leftrightarrow \exists w \quad P(x, w) = \text{TRUE}$
 recursiv

- Nu poate fi NP deoarece R de RE -

Putem încerca să dem $P \neq NP$ prin analogie cu $R \neq RE$

- La să arăt $R \neq RE$ folosesc o UT universală
- Ceea ce la $P \neq NP$ nu poate face. O UT univ pt P nu o putem calcula în clasa P
- Nu poate găsi o UT care să genereze toate calculabilele polinomiale posibile

astă ar însemna că un algoritm de n^{700} să il
pot rezolva în n^2 ceea ce nu este posibil.

$A \inneq \bar{A} \inneq$

$$A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases} \quad \bar{A}(x) = \begin{cases} 1 & x \notin A \\ 0 & x \in A \end{cases}$$

sunt diferențe în cele două cazuri

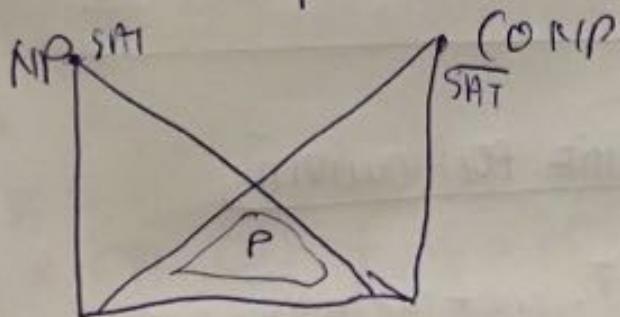
$$NP = (\exists^{< P(|x|)})P$$

$$co-NP = \{A \mid \bar{A} \in NP\}$$

$$co-NP = (\forall^{< P(|x|)})P$$

OBS: Dacă $P = NP \Rightarrow P = NP = co-NP$

Se crede că poate este: $NP \neq co-NP$



CURS 13

EXAMEN:

Partea 1: grilă (întreliări la nivel de cunoștințe generale)

Partea 2: 3 probleme de rezolvat - o oră

Partea 1: în jur de o oră - 26 întreliări aproximativ

30 min pauză între cele două părți

P_1 : metă 7 (grilele pot fi multiple, punctate parțial)

P_2 : pt metă > 7

P_1 : fără materiale

P_2 : cu materiale

IEFRARII POLINOMIALE

$\left\{ \begin{array}{ll} N & NP \rightarrow \text{mai multe reprezentări} \\ \text{Numărătore} & \text{Numitor} \end{array} \right.$

$\frac{x \in A}{\exists}$ \Rightarrow M7 re opere si dă
 un răspuns
 $x \notin A \Rightarrow ?$ (nu reține nimic)

NP
NP - pot verifica
 dacă asta dă
 griu și că nu este

NU - nu există niciun y
 cum să dă NU e SAT
 pot lucra toate valențile
 dar nu e eficient

$A \in RE \Leftrightarrow \exists$ predicat P recursiv (calalalile de M7 care
 numără răspuns) așa

$$\forall x \in \Sigma^* \\ x \in A \Leftrightarrow \exists a \in P(x, a) = \text{true}$$

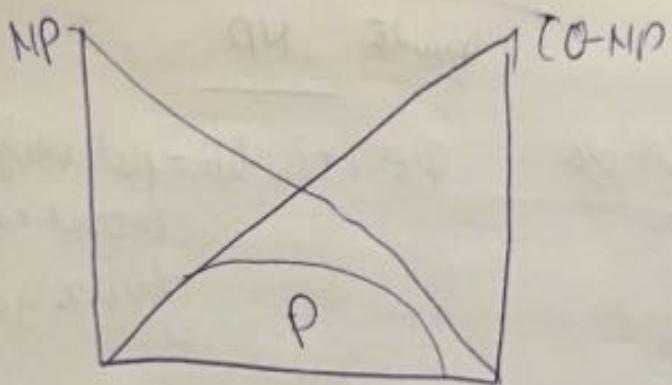
$A \in NP \Leftrightarrow \exists$ predicat P valabil în timp polynomial
 $P(x, y) \rightarrow (|x| + |y|)^{O(1)} \rightarrow$ complexitatea

$$\forall x \in \Sigma^* \\ x \in A \Leftrightarrow \exists y \in \{0, 1\}^{2^{|x|}} \text{ așa } P(x, y) = \text{TRUE}$$

NP reamănat cu RE

P reamănat cu R

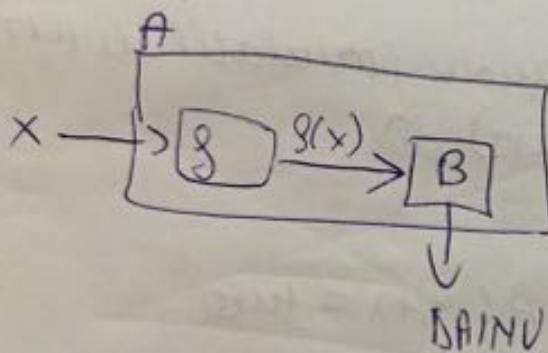
$A \in RE \nRightarrow \bar{A} \in RE \quad | \quad A \in NP \nRightarrow \bar{A} \in NP$
 Ex: HALT



$$CO-NP = \{ A \mid \bar{A} \in NP \}$$

Se arată: $CO-NP \neq NP \rightarrow P \neq NP$

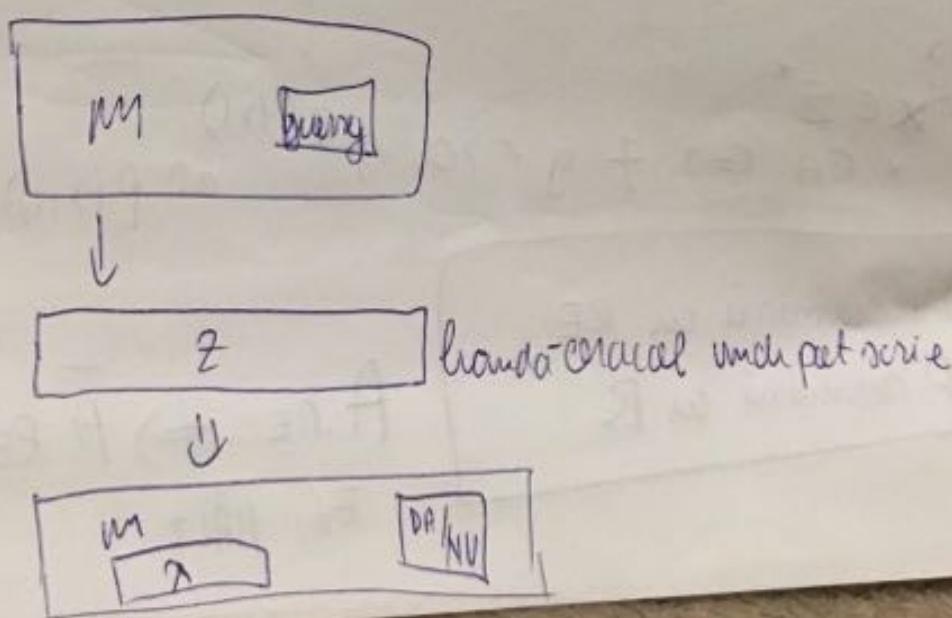
$$A \leq_m^P B \quad (A \text{ se reduce la } B)$$



$$A \leq_T^P B \quad (A \text{ se reduce la un singur } B)$$

A poate fi rezolvată cu o subrunitate pentru B

Mamă T
an Oracol



Când întâia im starea guilty \Rightarrow M se oprește din calcul
 și cuvătul de pe hârtie oracel se
 verifică dacă este în B sau nu.

După care sterge cuvătul și se face
 maximă întreaga stare care va
 decide dacă întâia im starea B
 $(\in B)$ sau NU ($\notin B$)

$$T_M(x) \leq g(n)$$

↓ polinom

\rightarrow maximă folosește intr. pt B
 \rightarrow poate primi mai multe de o imobilare

MT mod cu oracel

iterație polinomială

$$\sum_{k=1,+\infty}^P \pi_k^P = \{ \bar{A} \mid A \in \mathcal{Z}_k^P \}$$

$$\sum_{k=1}^P = \{ A \mid \exists M \text{ modul cu oracel a.t. } A = L(M; SAT) \}$$

Expresarea Σ_2^P

$$\Sigma_2 - SAT = \{ f \mid \exists x \forall y (f(x, y) \in sat) \}$$

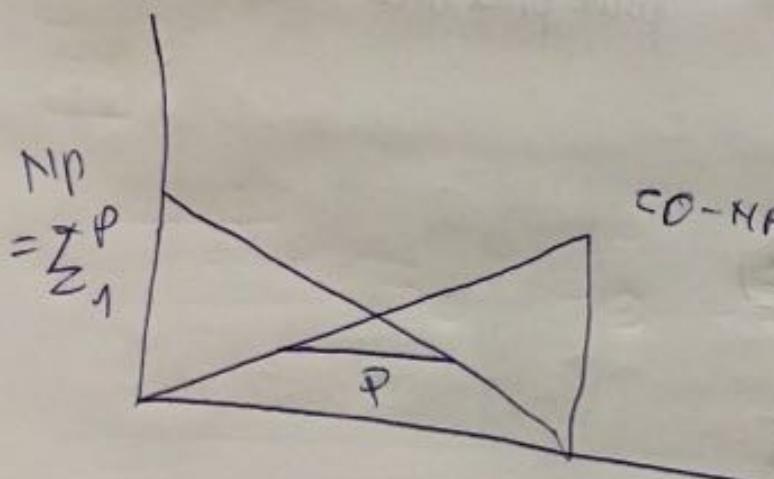
$$\Sigma_2\text{-SAT} \in \Sigma_2^P$$

⊕ $\Sigma_2\text{-SAT}$ este completă pt Σ_2^P

$$\forall a \in \Sigma_2^P \Rightarrow A \leq_m^P \Sigma_2^P \text{SAT}$$

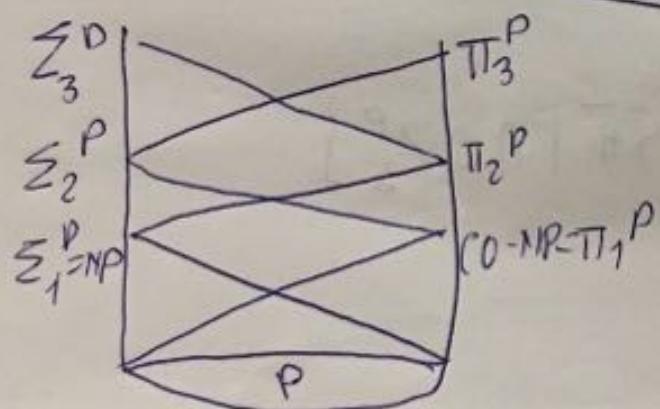
$$\Sigma_3^P = \{A \mid \exists B \in \Sigma_2^P \text{ s.t. } \text{MT med pol } M \text{ var } A = L(M, B)\}$$

$$\Sigma^P = \{A \mid \exists B \in$$



$$\Sigma_{k+1}^P, \Pi_k^P \leq \Sigma_{k+1}^P$$

$$\Sigma_k^P, \Pi_k^P \leq \Pi_{k+1}^P$$



$$\text{Se vede că } \Sigma_{k+1}^P + \Pi_{k+1}^P > \Sigma_k^P$$

fiecare din clase sunt pl complete

$$\Sigma_k^P = \{\emptyset \mid \exists x_1 \neq x_2 \neq x_3 \neq x_4 \dots Q \times K, \emptyset(x_1, x_k) = \text{TRUE}\}$$

Problema QBF → iau toate formule care sunt adiuvante mai grele din piramida plusnecunoscute

$$\exists x \forall y (x \neq y) \Rightarrow \text{formula e Falsă}$$

$$\downarrow \quad \downarrow$$

$$S_0, 13 \quad \{0, 13\}$$

$$\forall x \exists y (x \neq y) \Rightarrow \text{formula e Adiuvantă}$$

$$\exists x_1 \forall x_2 \exists x_3 \forall x_4 \dots \exists x_n \theta(x_1, \dots, x_n) \rightarrow \text{dolu sau Falsă}$$

Cum mă ducid dacă formula e A sau F?

- impreuna cu toate variabilele pt x și pt y

$$QBF = \{ \psi \mid \psi = \exists x_1 \dots \exists x_n \phi(x_1, \dots, x_n) \}$$

$$\phi = \forall / \exists$$

ψ adiuvantă

↑
iau o formulă booleană cuantificată și odată ce
verific formulă e adiuvantă

Să arde $QBF \notin \Sigma_k^P \vee k \geq 1$
 $\notin \Pi_k^P \vee k \geq 1$

$$A \in \Sigma_k^P \Rightarrow A \leq_m^P QBF$$

⑦ QBF e completă pt clasa PSPACE

QBF \in PSPACE

$\forall A \in$ PSPACE , $A \leq_m^P QBF$

PSPACE = $\{A \mid$ există o MT deterministă care
decide ~~NEA~~ A folosind spațiu de memorie
cel mult $g(n)$
 \nwarrow polinom

Se vede:

$P \neq NP \neq \Sigma_2^P \neq \Sigma_3^P \dots \subseteq$ PSPACE

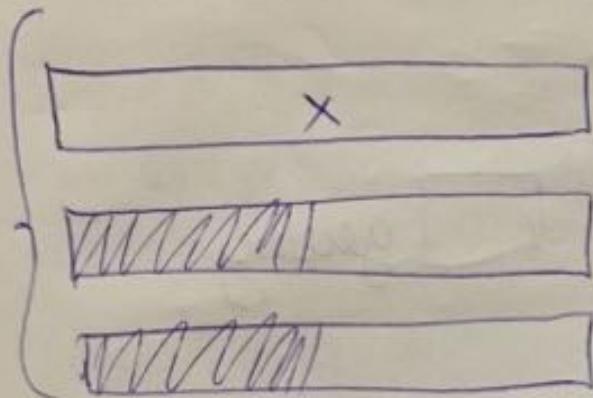
DSPACE $\{g(n)\}$ \rightarrow MT det care folosește spațiu $O(g(x))$
NSPACE $\{g(n)\}$ \rightarrow — mediată —

PSPACE = $\bigcup_{k \geq 1}$ OSPACE $\{n^k\}$

NLogSPACE $\stackrel{\text{def}}{=} NL = \bigcup_{k \geq 1} NSPACE[\log n]$

CURS 14

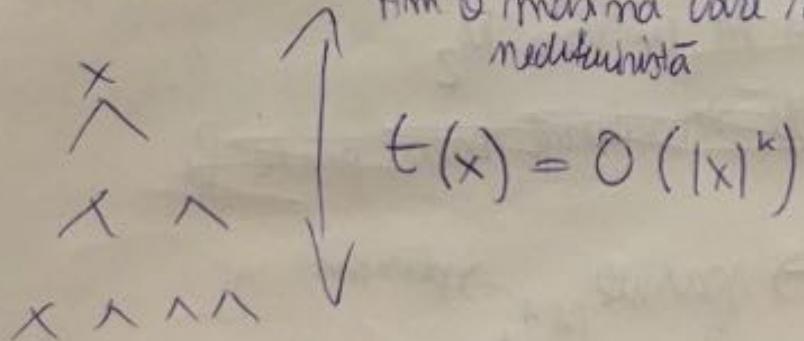
$\text{PSPACE} = \{ A \mid \text{există } M\text{T cu prop. că M7 recunoaște limbajul } A$
 și oricare ar fi x spațiul folosit pe intrarea x
 $\text{space}_M(x) = O(|x|^k) \ k > 0\}$



$\text{Space}_M(c)$

! Dacă o mașină face 5 pași, spațiul ei este 5

⑦ $P \subseteq \text{PSPACE}$



Pot să simulez o mașină mediterabilă în backtracking
 c.mai rapid

$S_P^P = \{ A \mid \text{există } M\text{TM M cu oracol în } A \vdash_1 (M, B)\}$

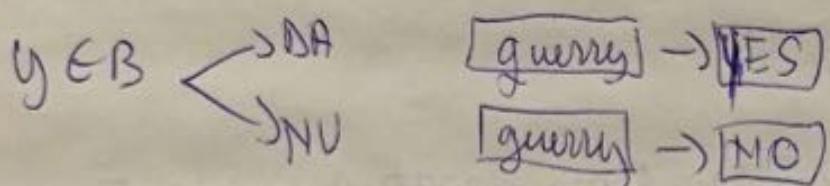
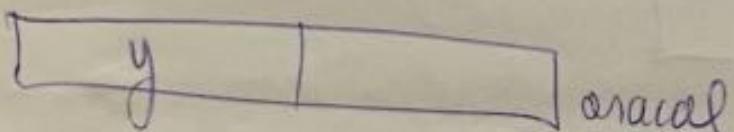
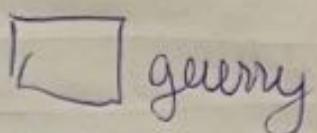
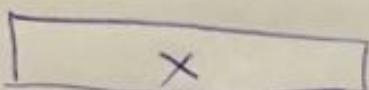
Maxima în oracol - SIMULARE

Am o mt memnă

Aceasta mai are o funcție oracol pe care scrie un cuvânt

Când împreună intr-o stare specială $\xleftarrow{\text{Query}}$

Nu spunea dacă cuvântul e în multimea B



Simulez oracolul cu M_1

Simulez maxima M în una M_2

~~Sunt făcute de simularea maxei poate și~~

$$\begin{aligned} \text{spatiul } m/x) &\rightarrow \text{spatiul } M_1 \xrightarrow{\text{plimare}} \\ &\rightarrow \text{spatiul } M_2 (1x1) \xrightarrow{\text{plimare}} \end{aligned}$$

Suma a două plimări este tot un plimare

Există NSPACE?

Nu avem mijloc de o astfel de clasă pt că PSPACE ar fi echivalentă cu NSPACE, deci nu mă ajută.

Motiv:

① (SAVITC+)

Fie A un joc de dicție care poate fi rezolvat de MMT nedictabil care pe intrarea x folosește spațiu maxim $S(x)$



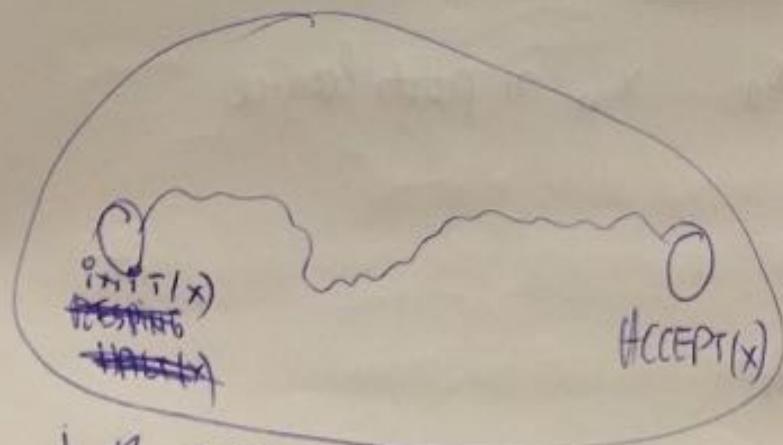
Fie o MMT deterministă M' care decide problema H și $m'(x)$ folosește spațiu maxim $S^2(x)$

Maximă med = 9 rânduri \Rightarrow Maximă alt = 81 rânduri

~~Dacă dăresc~~

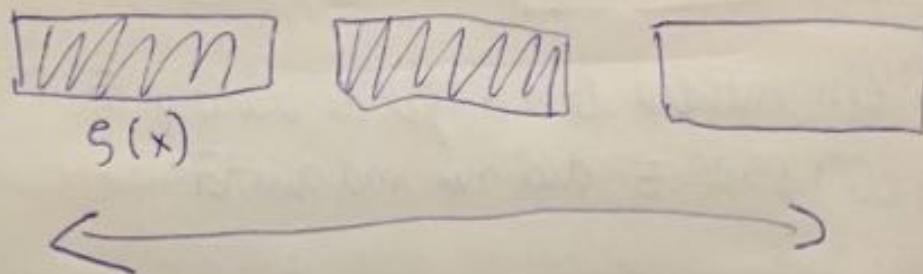
de dim:

Graful config maxim $m(x)$



~~NU SE STIU~~

Alg cu complexitate spațiu $O(S(x)^2)$



$$P \subseteq \text{NP} \quad \Sigma_2^P \quad \text{PH SPACE}$$

\swarrow \nearrow \swarrow \nearrow

$$\text{CONP} \quad \Pi_2^P$$

PROBLEMA

Se dă un graf G și varfurile a, b

Vream să văd dacă pot ajunge din a în b

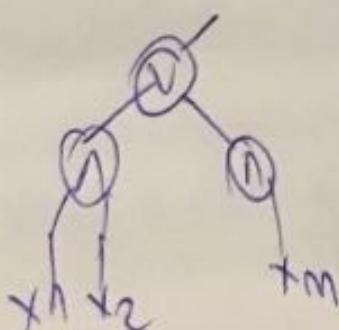
- de rezoluțional cu algoritmi

Ghișec învers (spațiu logarithmic)

a x_1 b

CIRCUITE BOOLENE

fie multe intrari x_1, \dots, x_m și porti logice



Nu se cunoaște o fermie booleană $f_m: \{0,1\}^m \rightarrow \{0,1\}$
 care să realizeze un circuit boolean cu m porturi $\geq m^2$

Funcția XOR este de fermie greu de calculat.

$$\text{PARITY}_n(x_1 \dots x_n) = \begin{cases} 1 & x_1 \oplus x_2 \dots \oplus x_n = 1 \\ 0 & x_1 \oplus x_2 \dots \oplus x_n = 0 \end{cases}$$

⊕ Nu putem calcula funcția PARITY cu circuite booleane
 ai ordinărilelor unui circuit \leq o constantă K , m de porturi
 necesită $\leq \text{poly}(m)$ liniar dacă dău valoare 1, V de oare
 paritate

$K \downarrow$ + porturi de tip
 modulo 6

$$\text{MOD}_6(x_1 \dots x_m) = \begin{cases} 1 & \sum x_i \\ 0 & \text{altfel} \end{cases}$$

RECAPITULARE

- Exemplu problema care nu poate fi rezolvată:

Teorema lui Matiasevic (C_1)

Sădă: Un polinom cu coeficienți întregi în variabilele $p(x_1, \dots, x_n)$

Dacă: Dacă are ecuația $p(x_1, \dots, x_n) = 0$ sau orice relație

$$\text{RECURSIVĂ: } f \text{ MT aș } M(x) = \begin{cases} 1 & x \in A \text{ și } x \text{ nu } \in M(x) \text{ reciproc} \\ 0 & x \notin A \text{ sau } M(x) \text{ finit de pasi} \end{cases}$$

NERECURSIVE: - proprietăți
- probleme
} Nu au alt
} Nu pot fi rezolvate
de MT

C e o MT?

Un automat care minge de la stătă la dr

Pot stații de accept, respingere sau mers la infinit

Pot fi stări de intrare nu pot să scriu