

Calculabilitate și complexitate

Sujectul I: Mașini Turing deterministe și nedeterministe;

DEFINIMĂLĂ:

Informal, o mașină Turing este un "ansamblu" format din o bandă nemărginită și un cap de citire-scriere. Capul de citire-scriere se poate deplasa atât la stânga cât și la dreapta.

Mașinile Turing pot fi folosite ca dispozitive de acceptare atunci când la intrare pe bandă se află un cunoscător și dacă mașina se poate opri și să accepte sau să nu accepte într-o reacție, dacă poate să nu se aprezeze. Totodată, mașinile Turing pot fi folosite și ca dispozitive de calcul atunci când este practic o funcție particulară $f: \{1\}^k \rightarrow \{0,1\}$. Dacă mașina se oprește pe intrarea $(x_1 \dots x_k)$ atunci soluție este rezultatul pol $f(x_1 \dots x_k)$. Dacă mașina nu se oprește, atunci funcția nu este definită în $(x_1 \dots x_k)$.

DEFINIMĂLĂ:

$M(Q, V, U, \delta, q_0, F, B)$, unde

$Q \rightarrow$ multimea finită de stări;

$V \rightarrow$ alfabetul de intrare

$U \rightarrow$ alfabetul de lizier $V \subseteq U$;

$q_0 \in Q \rightarrow$ starea initială a mașinii;

$F \rightarrow$ multimea stărilor finale

$B \rightarrow$ Blank

$\delta : (Q \setminus F) \times U \rightarrow 2^{Q \times (U \setminus \{B\}) \times \{L, R\}}$;

$(s, b, L) \in \delta(q, a) \rightarrow$ mașina se află în starea q , citeste simbolul a , poate obține niciun să scrie b , pe lângă a , schimbă stare în s și se deplasează la stânga

Mașinile Turing pot fi deterministe și nedeterministe.

Pentru mT deterministe avem $M(Q, V, U, \delta, q_0, F, B)$ cu proprietatea $\delta: Q \times V \rightarrow (Q \times U \times \{L, R\})$ [dacă $\text{card}(\delta(q, a)) \leq 1 \quad \forall q \in Q, \forall a \in U$ și $mT \in \text{det}(\delta)$];

MASINA TURING CU MAI MULTE BENZI: $Q \times (U \setminus \{B\})^n \times \{L, R\}^n$

$M(m, Q, V, U, \delta, q_0, F, B)$, $\delta: (Q \setminus F) \times U^m \rightarrow 2^{Q \times (U \setminus \{B\})^n \times \{L, R\}^n}$

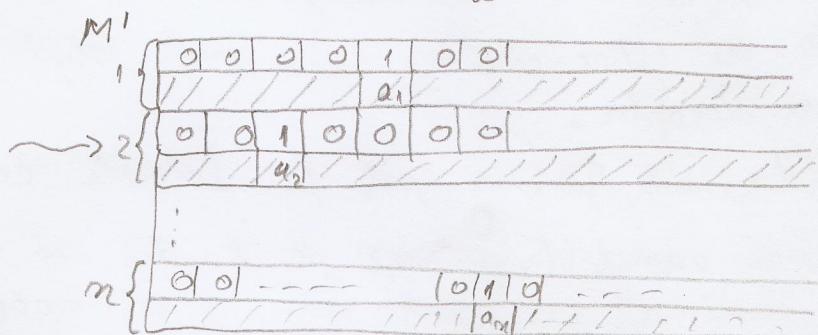
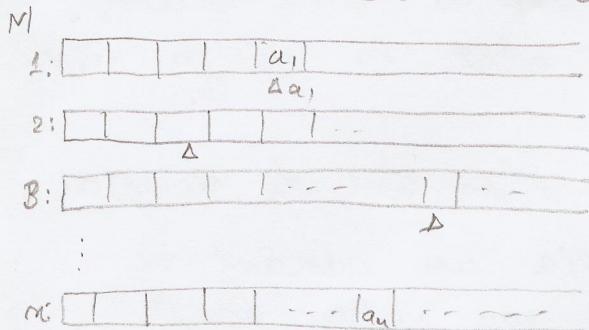
$(s, b_1 \dots b_m, x_1 \dots x_m) \in \delta(q, a_1 \dots a_m) \rightarrow$ mașina se află în starea q , capul 1 citește $a_1 \dots$ capul m citește a_m . Atunci mașina trăiește în starea s , scrie $b_1 \dots b_m$ și capurile de citire se mișcă în constantă

TEOREMA 1:

Oare masina Turing M cu n benzi poate fi simulata cu o masina Turing M' cu o singura banda? In plus, daca M este deterministica, atunci si M' este deterministica.

DEMONSTRATIE

Fie M o mT cu n benzi: $M = (n, Q, V, U, S, q_0, F, \beta)$.



M' : o singura banda, pe care o impartim in pisti. Fiecare set de 2 pisti corespunde unui set de benzi din masina M . Pe banda de gas a setului punem tot continutul benzelor, iar pe banda de sus punem 1 doar in poz simbolului pe care se afla I/O in M .

Pas 1: Positioneaza copul I/O pe prima celula a benzelor.

Pas 2: Scanarea banda si memorarea stora de la copie a inceput si simbolurilor de pe pistile care apar sub simbolul 1;

Pe stora (gasit de la stanga) ($0, 0, -0, 0, -0$) se inlocuiesc zeroii de pe pozitia i (din primul set de n zeroi) cu simbolul ai atunci cand se detecteaza simbolul care are 1 pe pista $i+1$;

Pasul 2 se termina atunci cand primele n zeroi au fost inlocuiti cu simboluri din V ;

Pas 3: Prim pozsurge de la dreapta la stanga se actualizeaza valorile pistilor corespunzatoare benzelor masinii M . Fiecare actualizare corespunzatoare benzii care K din masina M este marcată cu 1 în ultima serie de n zeroi, pe poz K .

Alegem multimea stariilor finale din M' ca fiind aceeași cu multimea stariilor finale ale lui M .

Asadar observam că masina M' simuleaza perfect masina M .

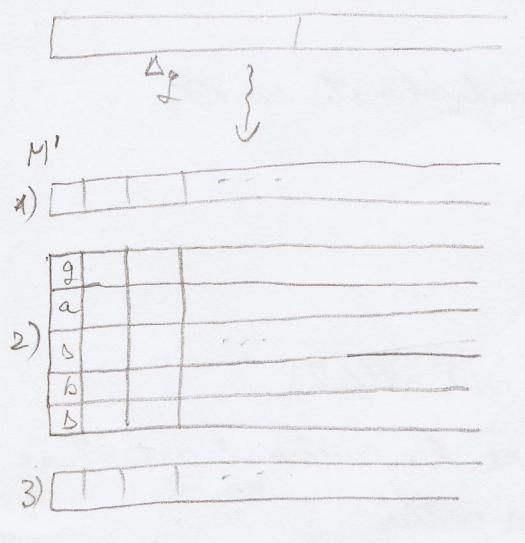
TEOREMA 2:

Oare masina Turing nedeterminista cu o banda poate fi simulata de o masina Turing determinista cu 3 benzi;

DEMONSTRATIE

Fie $M = (Q, V, U, \delta, q_0, F, B)$ o mT. nedeterminista. Ieșirea sa construim o mT. M' determinista cu 3 benzi care să simuleze M.

M



Fie $W = \{(q, a, s, b, \Delta) / q \in Q \setminus F,$
 $s \in Q, a \in U, b \in U \setminus \{B\}, \Delta \in \{L, R\}\}$

W este multime finita

(W^*, \leq)

ordinea lexicografica

$$x, y \in W^*, x \leq y \Leftrightarrow \begin{cases} 1) |x| < |y| \\ 2) \exists i \in \omega \{x_i = y_i \wedge i < j\} \\ \quad \quad \quad x_j < y_j \end{cases}$$

OBS: $x \in W^*$, x se gaseste pe banda unei masini Turing \tilde{M} , \tilde{M} poate construi succesorul lui x pe banda sa? DA.

$M' \rightarrow$ pe prima banda are acel de intrare W

\rightarrow copiază W de pe banda 1 pe banda 3.

\rightarrow scrie pe banda 2 primul element din W^* .

\rightarrow intră în starea q_0 .

M' efectuează verificarii pozitive:

1) Există simbolul curent de pe banda 2
 $fie (q, a, q', b, \Delta)$

2) Verifică dacă starea sa este q . Dacă nu, gata R

3) Verifică dacă simbolul citit pe banda 1 este a . Dacă nu, gata R

4) Scrie Δ pe banda 1;

5) Schimbă starea în q' ;

6) Repeteaza capul de citire de pe banda 1 după valoarea Δ

7) Repeteaza capul de citire pe banda 2 obligatorie ACEPTA

Se uria pasul 1 dacă există simbol pe banda 2. Dacă nu, gata R

R: Copiază W de pe 3 pe 1

scrie pe banda 2 succesorul elementului oflat pe banda 1
 intră în starea q_0 .

Masina M' acceptă doar dacă ajunge într-o stare finală a lui M.
 M' este deterministica.

Gurile 2: Funcții recursive. Funcții Turing calculabile. Funcții calculabile cu programe standard.

Programe standard:

limbajul abstract S va calcula $f: \mathbb{N}^k \rightarrow \mathbb{N}$

Variabile de intrare x_1, \dots, x_m

 → de ieșire y

 → locale z_1, \dots, z_n

Variabile de lucru și cele de ieșire sunt inițializate cu 0;

ETICHETE: E, A_1, A_2, \dots, A_n ;

INSTRUCȚIUNI:

- $V \in V$ (efect nul)

- $V \in V+1$ (incrementare)

- $V \in V-1$ (efect de decrementare dacă $V \neq 0$, nul altfel)

- IF $V \neq 0$ GOTO $L: \rightarrow V=0$ efect nul, se trece la instrucția următoare
 $\rightarrow V > 0$ transferul se face astfel:

- la prima instrucție cu etichetă L dacă $L \neq e$ și există L ;

- se termină programul dacă $L=e$ sau nu există L .

Programul standard este format dintr-un set de instrucțiuni și terminarea se face fil prim săt la eticheta E , fil prim săt la o etichetă inexistentă și transferul se face la sf. instrucție. Outputul este valoarea lui y la sf. programului.

Functii Turing calculabile

Să numesc funcții Turing calculabile funcțiile pentru care se poate scrie o masină Turing.

Functii recursive:

1) Functii elementare:

 → succ: $\mathbb{N} \rightarrow \mathbb{N}$ $\text{succ}(x) = x + 1$

 → constante: $C_K^{(m)}: \mathbb{N}^m \rightarrow \mathbb{N}$, $C_K^{(m)}(x_1, \dots, x_m) = K$

 → Proiecții $P_x^{(n)}: \mathbb{N}^m \rightarrow \mathbb{N}$, $P_x^{(n)}(x_1, \dots, x_m) = x_k$

2) Operatii:

 1) compunerea funcțională

funcția $f: \mathbb{N}^k \rightarrow \mathbb{N}$ se obține din compunerea funcțională a funcțiilor $g_i: \mathbb{N}^k \rightarrow \mathbb{N}$, $i = \overline{1, m}$ și $h: \mathbb{N}^m \rightarrow \mathbb{N}$ dacă

$$f(x_1, \dots, x_k) = h(g_1(x_1, \dots, x_k), g_2(x_1, \dots, x_k), \dots, g_m(x_1, \dots, x_k));$$

2) recurentă primitară:

$f: \mathbb{N}^{k+1} \rightarrow \mathbb{N}$ se obține prin recurentă primitară din funcții $g: \mathbb{N}^k \rightarrow \mathbb{N}$ și $h: \mathbb{N}^{k+2} \rightarrow \mathbb{N}$ dacă

$$f(x_1, \dots, x_k, 0) = g(x_1, \dots, x_k)$$

$$f(x_1, \dots, x_k, t+1) = h(x_1, \dots, x_k, t, f(x_1, \dots, x_k, t));$$

3) minimisore nemărginită.

$f: \mathbb{N}^k \rightarrow \mathbb{N}$ se obține din funcția $g: \mathbb{N}^{k+1} \rightarrow \mathbb{N}$ prin minimisare nemărginită dacă $f(x_1, \dots, x_k)$ este cel mai mic t pentru care

$$f(x_1, \dots, x_k) = \min_t \{g(x_1, \dots, x_k, t) = 0\};$$

Teorema 1:

Orică funcție calculabilă cu programe standard este Turing calculabilă.

Teorema 2:

Orică funcție recursive este calculabilă cu programe standard.

Teorema 3:

Orică funcție Turing calculabilă este recursive.

Demonstratie:

Fie $M = (Q, V, U, \delta, q_0, F, B)$ un TM determinist.

$\overline{x_1}$	$ 0 $	$\overline{x_2}$	$ 0 $	$\overline{x_3}$	$ 0 $	\cdots	$ 0 $	$\overline{x_k}$	$ 0 $
Δ									
\downarrow					$f: \mathbb{N}^k \rightarrow \mathbb{N}$ $f(x_1, \dots, x_k)$				
$f(x_1, \dots, x_k)$					B				

Configurație: $\times 2^B$

→ numărătăm cuburile benzii $0, 1, 2, \dots$

→ numărătăm stările din Q : $q_0 = 0, q_1 = 1, \dots, q_{2^B-1}$

→ numărătăm simbolurile din V : $0=0, 1=1, \dots$

Numește atosat unei configurații

$q \rightarrow i'$

$i \times 1 = j \Rightarrow$ poziția pe bandă a capului este j'

Continutul benzii: $\alpha_B = p_1, p_2, \dots, p_n \rightarrow$ nr atosat este

$p_1^{#_1}, p_2^{#_2}, \dots, p_m^{#_m}$, unde $#_k$ este nr simb p_k

Configurația $\rightarrow \langle x, \langle y, z \rangle \rangle \rightarrow x = \text{nr de stări}$

$y = \text{poz capului I/O}$

$z = \text{nr atosat continutului Benzii}$

$\langle a, b \rangle$ f. perioade $= 2^a (2b+1) - 1$

$C_M(x, m)$, $x = (x_1, \dots, x_k)$

nr atosat configurației de la pasul m al maximi Turing M pe intrarea (x) ;

Ne propunem să demonstreăm că $C_M(x, m)$ este f. recursivă;

$C_M(x, 0) = \langle 0, \langle 0, p_1, p_2, p_{x_1+1}, p_{x_1+2}, \dots, p_{x_1+x_2+2}, \dots, p_{x_1+x_2+\dots+x_k} \rangle \rangle$

$h_1(z) = \begin{cases} \text{nr atosat stării în care trec } m \text{ din config. curentă, cu nr } z, \\ \text{dacă } z \text{ este nr unei config. valide} \\ \uparrow, \text{ altfel (nechiarită)} \end{cases}$

$h_2(z) = \begin{cases} \text{poziția capului de citire/scriere în care af. } m \text{ din config. curentă} \\ \text{dacă } z \text{ este nr unei config. valide;} \\ \uparrow, \text{ altfel} \end{cases}$

$h_3(z) = \begin{cases} \text{numărul atosat continutului Benzii în care trec } m \text{ din} \\ \text{config cu nr } z, \text{ dacă } z \text{ este nr unei config. valide} \\ \uparrow \text{ altfel.} \end{cases}$

$C_M(x, m+1) = \langle h_1(C_M(x, m)), \langle h_2(C_M(x, m)), h_3(C_M(x, m)) \rangle \rangle$

h_1, h_2, h_3 recursive?

$g_1(a, b) = \text{nr stării în care trec } M \text{ din starea cu nr a citind simbolul}$
 $\text{cu nr } b$

$g_2(a, b) = \begin{cases} 0, M \text{ se deplacează la stânga din st. cu nr a citind obiectul} \\ 1, M \text{ se depl. la dreapta din starea curățind obiectul cu nr } b; \end{cases}$

$g_3(a, b) = \{ \text{nr obiectul de } M \text{ din starea cu nr a citind obiectul cu nr } b \}$

Afl: g_1, g_2, g_3 sunt funcții de suport finit $\Rightarrow g_1, g_2, g_3$ fct recușivă

$$h_1(z) = \ell\left(\ell(z), (r(r(z)))_{\ell(r(z))}\right) \rightarrow f \cdot \text{recușivă}$$

$$h_2(z) = \ell\left(\ell(z) + g_2(\ell(z), (r(r(z)))_{\ell(r(z))})\right) \stackrel{?}{=} \rightarrow f \cdot \text{recușivă}$$

$$h_3(z) = \left(r(r(z)) \mid P_{\ell(r(z))}^{g_3(\ell(z), r(r(z)))} \cdot P_{\ell(r(z))} \right) \rightarrow h_3 \text{ fct recușivă}$$

$C_M(x, m_0) \rightarrow M$ se oprește după m_0 pasi

$C_M(x, n) \nvdash n > m_0$

$$\mu(x) = \min_m [C_{pq}(x, m) = C_M(x, m+1)]$$

= nr de pași după care M se oprește pe intrarea x

$$f(x) = L_\epsilon\left(r(r(C_M(x, \mu_x)))\right) - 1$$

//

$f(x_1 \dots x_k) \Rightarrow f$ este funcție recușivă

Subiectul III: Funcția universală, Păopirii. M. neur, sec. enum, Nere

Funcția universală:

Fie P un program standard.

$$\#(P) = \{\#(I_1), \#(I_2), \dots, \#(I_m)\} - 1$$

$$\#(I_i) = \langle a, \langle b, c \rangle \rangle$$

$\varphi_P(x_1, \dots, x_m) \rightarrow$ funcția calculată de programul P .

$\varphi^{(m)}(x_1, \dots, x_m, t) \rightarrow$ funcția universală de m variabile;

$$\varphi^{(n)}(x_1, \dots, x_m, t) = \varphi_P(x_1, \dots, x_m), \#(P) = t;$$

Teorema 1:

pentru orice $m \geq 1$ funcția $\varphi^{(m)}$ este calculabilă ✓

$\text{HALT}(x, y) = \begin{cases} 1 & \text{dacă programul } P \text{ cu } \#P=y \text{ se termină pe intrare } \\ 0 & \text{oarecum;}\end{cases}$

Teorema 2:

Funcția HALT nu este calculabilă.

Demonstratie

Pp R. A că HALT este calculabilă. Fie P' programul care calculează HALT .

Considerăm programul:

$$z_{2t+1} \leftarrow \text{HALT}(x_1, x_1) \quad \{ P,$$

$$A: \text{IF } z_{2t+1} \neq 0 \text{ GOTO } A$$

$$\#(P') = t$$

$$\text{HALT}(x, t) \Leftrightarrow z_{2t+1} = 0 \text{ în } P'$$

$$\Leftrightarrow \text{NOT } \text{HALT}(x, x)$$

$$\text{dor } x = t : \text{HALT}(t, t) \Leftrightarrow \text{NOT } \text{HALT}(t, t) \quad \text{X}$$

Teorema 3: Pt orice $m \geq 1$ funcția $\varphi^{(m)}$ este calculabilă

Demonstratie

$T \leftarrow x_{m+1} + 1$ // numărul codificării lui P

$S \leftarrow \prod_{i=1}^m P_{2i}$ // starea programului

$K \leftarrow 1$ // contor pt instrucțiunea curentă

C: IF ($K > L_T(T)$) $\vee (K=0)$ GOTO F

$I \leftarrow (T)_k$ // nr instrucție următoare a se executa

$V \leftarrow L(I)$ // numărul asociat etichetei instrucțiunii vizibile

$U \leftarrow R(I)$ // tipul instrucțiunii

IF $U=0$ GOTO N // efect nul ~~N~~

IF $U=1$ GOTO A // incrementat

IF $U=2$ GOTO D // decrementat

IF $P_V \neq S$ GOTO N

$K \leftarrow \min\{k \mid s = \ell((T)_e^k)\}$ // nr de ordine al primei instrucțiuni et cu
eticheta cu nr $n-k$ dacă există

GOTO C

A: $S \leftarrow \cancel{S} S * P_V$

GOTO N

D: IF P_V / S GOTO N

$S \leftarrow S / P_V$

GOTO C

F: $y \leftarrow (S),$

Problema oprii:

Nu există nicio mașină Turing care primind codul unei mașini M și codul lui și să poată decide dacă mașina M se oprește pe intrarea .

Def

O mulțime X se numește recuziv enumerabilă (Σ, Σ) dacă funcția ei caracteristică este Turing calculabilă

Def

O mulțime \mathbb{X} se numește recursivă dacă funcția caracteristică este Turing calculabilă dor mașina Turing se oprește pe fiecare inter

Teorema:

Dacă A este o mulțime recursivă, atunci A este recuziv.

Teorema:

1) X este o mulțime recursivă $\Rightarrow X$ și CX sunt recursiv enumerabile

2) X și Y sunt recuziv sau recursiv enumerabile \Rightarrow

$X \cup Y$ și $X \cap Y$ sunt recursiv sau recuziv enumerabile;

Găurile 4:

Clasa de complexitate timp:

Pentru calculul complexității timp se folosește modelul mașinii Turing cu K-benzi inf. la ambele capete și care se oprește pe f. intrare, iar capul de citire scriere poate sta.

Pt o mt numată $\text{Time}_M(n) = m$ maximul de pasi pe care îl face maxima M pe o intrare de lungime n.

$$\text{DTIME}_K(f(m)) = \{L \mid \exists M \text{ mt obtinută, cu } K \text{ benzi astfel că } L(M) = L \text{ și } \text{Time}_M(n) \leq f(m) \text{ și } n \geq n_0\}$$

$$\text{NTIME}_K(f(m)) = \{L \mid \exists M \text{ mt nedeterministă cu } K \text{ benzi astfel că } L(M) = L \text{ și } \text{Time}_M(n) \leq f(m) \text{ și } n \geq n_0\}$$

ELIMINAREA CONSTANTELOR:

$$\text{Th} \lim_{n \rightarrow \infty} \frac{f(n)}{n} = +\infty \Rightarrow (N)(\Delta) \text{TIME}_K(f(m)) = (N)(\Delta) \text{TIME}_K(c \cdot f(n));$$

COMPRIHARE benzilor:

$$(N)(\Delta) \text{TIME}_K(f(m)) \leq (N)(\Delta) \text{TIME}_K(f^2(n))$$

Demonstratie

Construim o mt

Procedăm la fel ca la transformarea unei mașini Turing cu mai multe benzi în mt cu o sg. sondă d.p.d.v. al construcției.



al mult $f(n)$ celule

$$1 \text{ tranziție} \Rightarrow f(n) + f(n) + 2f(n) \leq 4f(n)$$

$$(N)(\Delta) \text{TIME}_K(f(m)) \leq (N)(\Delta) \text{TIME}_K(4f(n));$$

$$= (N)(\Delta) \text{TIME}_K\left(\frac{1}{2}f(n)\right) \leq (N)(\Delta) \text{TIME}_K\left(4 \cdot \left(\frac{1}{2}f(n)\right)^2\right)$$

IERARHII DE CLASE DE COMPLEXITATE TIMP

$$(N)(\Delta) \text{TIME}_K(f(m)) \leq (N)(\Delta) \text{TIME}_2(f(m) \cdot \log_2(f(n)))$$

Teorema:

Fie $f(n)$ o funcție recursive.

Atunci $\text{REC} \setminus \text{DTIME}(f(n)) \neq \emptyset$

Demonstrare:

Fie $f(n)$ o funcție recursive.

Definim $L = \{\#(x) \mid \#(x) \notin L(M) \text{ în cel mult } f(n) \text{ pasi}, \#(M) = \#(x)\}$,
 M deterministă

1) $L \in \text{REC}$:

Construim o mașină Turing deterministă, cu $L(M) = L$ care se oprește pe fiecare intrare;

M : INPUT: $\#(x)$

→ calculează lungimea lui $\#(x) = n$ pe o bandă auxiliară

→ calculează $f(n)$ pe același bandă;

→ găsești M' așa că $\#(M') = \#(x)$

→ simulează mașina M' pe intrarea $\#(x)$ pînă la final

→ Acceptă dacă $\rightarrow M'$ se oprește și nu acceptă

→ M' depășește timpul slotat

Mașina M este deterministă și se oprește pe fiecare intrare;

2) $L \notin \text{DTIME}(f(n))$

Pp că $L = L(\hat{M})$

$\hat{M} \rightarrow$ deterministă

$\text{time}_{\hat{M}}(n) \leq f(n)$

$\#(\hat{M}) = j$

$\exists x \text{ așa că } \#(x) = j$

$\#(x) \in L \Leftrightarrow \#(x) \notin L(\hat{M})$ în maxim $f(n) \Rightarrow \#(x) \notin L$

Relații între clase de complexitate:

1) $(D)(N)\text{TIME}(f(n)) \subseteq (D)(N)\text{Space}(f(n))$

2) Dacă $f(n) \geq \log_2 n \Rightarrow \#L \in \text{DSPACE}(f(n)) \exists c_L \text{ așa că } L \in \text{DTIME}(C_L^{f(n)})$

3) $\#L \in \text{NTIME}(f(n)) \exists c_L \text{ așa că } L \in \text{DTIME}(C_L^{f(n)})$

4) Dacă $f(n) \geq \log n$ și $f(n)$ este spațiu construibilă complet \Rightarrow
 $\text{NSPACE}(f(n)) \subseteq \text{DSPACE}(f^2(n))$ [TH. SAVITCH]

SUBIECTUL 5: Clase de complexitate spațială

Pentru colțul complexității spațiale se folosește modelul maxim Turing offline care are o bandă de intrare și k benzi auxiliare. Bandă de intrare este doar pentru citire, iar benzile auxiliare sunt mărginite la unul dintr-o capătă.

Masina se oprește pe fiecare intrare;

$\text{SPACE}_M(n) \rightarrow$ numărul maxim de căile folosite de masina M pe o bandă auxiliara pînă la oprirea sa pe o intrare de dimensiune n .

⊗⊗

Comprimarea benzilor:

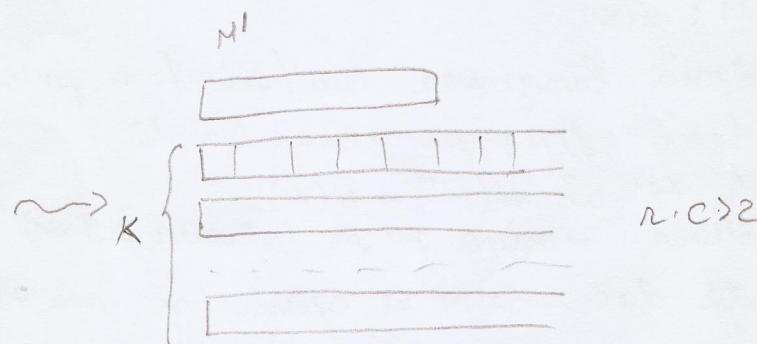
$$(N)(\Delta) \text{SPACE}_k(f(n)) = (N)(\Delta) \text{SPACE}_1(f(n))$$

ELIMINAREA CONSTANTELOR: $(N)(\Delta) \text{SPACE}_k(f(n)) = (N)(\Delta) \text{SPACE}_k(C \cdot f(n)) + k \geq c > 0$

M mînt cu k benzi;



$$\text{SPACE}(n) \leq C \cdot f(n)$$



O mișcare a masinii M' simulează mișcările masinii M pe intervalul de lungime n corespunzător dim M .

$$\text{SPACE}_{M'}(n) \leq \lceil \frac{\text{SPACE}_M(n)}{n} \rceil \leq \frac{C \cdot f(n)}{n} + 1 \leq f(n)$$

$$\Leftrightarrow \frac{f(n)}{2} + 1 \leq f(n) \Leftrightarrow f(n) \geq 2$$

$$\text{Dacă } f(n)=1 \Leftrightarrow \text{SPACE}_M(n) \leq C \Rightarrow \text{SPACE}_{M'}(n)=1 \leq f(n)$$

Teorema:

S_1 și S_2 sunt spații construibile complet și $\lim_{m \rightarrow \infty} \frac{S_1(m)}{S_2(m)} = 0$,

Atunci $\text{DSPACE}(S_2(n)) \setminus \text{DSPACE}(S_1(n)) \neq \emptyset$

⊗⊗ $\text{DSPACE}_k(f(n)) = \{L \mid \exists \sigma \text{ mînt deterministă cu } k \text{ benzi cu } L=L(M) \text{ și}$
 $\text{SPACE}_M(n) \leq f(n) + m > m_0\}$

$N\text{SPACE}_k(f(n)) = \{L \mid \exists \sigma \text{ mînt nedeterministă cu } k \text{ benzi cu } L=L(M) \text{ și}$
 $\text{SPACE}_M(n) \leq f(n) + m > m_0\}$

$$(N)(\Delta) \text{SPACE}(f(n)) = \bigcup_{k \geq 1} (N)(\Delta) \text{SPACE}_k(f(n))$$

Subiectul 6: NP-Complexitate

Def

L' se reduce în timp polynomial la L ($L' \xrightarrow{tp} L$), dacă există o mașină Turing M deterministă care are o bandă de ieșire, și care pentru orice x la intrare produce un y pe bandă de ieșire, în timp polynomial în $|x|$ astfel că $x \in L' \Leftrightarrow y \in L$.

Def

L' se reduce în spațiu logaritmic la L ($L' \xrightarrow{se} L$) dacă există o mașină Turing cu o bandă de ieșire cu porțegeze numai la dreapta și care pentru orice x la intrare produce un y pe bandă de ieșire, folosind spațiu mărginit logaritmic cu excepția spațiului de pe bandă de ieșire.

$$P = \bigcup_{k \geq 1} \text{DTIME}(n^k)$$

$$\text{PSPACE} = \bigcup_{k \geq 0} \text{DSPACE}(n^k)$$

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

$$\text{NSPACE} = \bigcup_{k \geq 0} \text{NSPACE}(n^k)$$

$$\text{DLOG} \subseteq \boxed{\text{P} \subseteq \text{NP}} \subseteq \text{NSPACE} = \text{PSPACE}$$

//

$$\text{DSPACE}(\log n)$$

TEOREMA 1:

Fie $L' \xrightarrow{tp} L$.

a) Dacă $L \in P$ atunci $L' \in P$

b) Dacă $L \in NP$ atunci $L' \in NP$

DEMONSTRATIE:

Construim mașina Turing M' deterministă, cu $L(M') = L'$, și $\text{Time}_{M'}(x) \leq p(|x|)$.

Dacă $L \in P \Rightarrow \exists \underline{mT} M$ deterministă, cu $L(M) = L$ și $\text{Time}_M(a) \leq p_1(a)$

$L' \xrightarrow{tp} L \Rightarrow \exists \underline{mTM}_0$ deterministă, cu $\text{Time}_{M_0}(x) \leq p_2(x)$ și $x \in L' \Leftrightarrow y \in L$

Mașina M' primește la intrare x .

→ simulează M_0 pe intrarea x și $\{p_2(|x|)\}$;

→ obține y , simulează $mT M$ pe intrarea y și $\{p_1(|y|)\}$;

→ acceptă ($\Leftrightarrow M$ acceptă pe y);

$$\begin{aligned} \text{Time}_{M'}(x) &= p_2(|x|) + p_1(|y|) \leq p_2(|x|) + p_1(p_2(|x|)) \leq p(|x|); \\ |y| &\leq p_2(|x|) \end{aligned}$$

Analog pt b, doar că mașinile sunt nedeterministe.

Teorema 2:

Dacă $L' \xrightarrow{\text{de}} L$ atunci

a) dacă $L \in P$ atunci $L' \in P$

b) dacă $L \in \text{SPACE}(\log^k m)$ atunci $L' \in \text{SPACE}(\log^k m)$

c) dacă $L \in \text{NSPACE}(\log^k m)$ atunci $L' \in \text{NSPACE}(\log^k m)$

Definitii:

Fie C o clasă de limbi.

a) L se numește C -difícil în raport cu o reducere (cp/sl) dacă $\forall L' \in C$ avem $L' \xrightarrow{\text{cp/sl}} L$;

a) L se numește C -complet dacă L este C -difícil și $L \in C$.

PROBLEMA ACOPERIRII UNUI GRAF (ACG)

$G = (V, E)$ graf neorientat și aciclic

$E \subseteq P_2(V)$, $\{x, y\} \in E$

$x \in V$

Ge cere să se decidă dacă există o submulțime X a lui V , cu $\text{cord}(X) = k$ cu $\forall \{x, y\} \in E : \{x, y\} \cap X \neq \emptyset$;

Teorema: ACG este NP-completă;

Demonstrație:

1) ACG este în NP;

"Ghiozdan" $x \in V$ (timp 1) cu $|X| = k$.

Verificăm dacă X este acoperire (timp cord(E))

$V = \{x_1, \dots, x_m\}$

$E = \{e_1, \dots, e_m\}$, $e_i = \{x_{i1}, x_{i2}\}$

Graful este introdus pe baza unei máxiuni Turing astfel

$x_1 x_{10} x_{11} \dots x_{m1} \{x_{12} x_{13} x_{14}\} \{x_{22} x_{23} x_{24}\} \dots$

2) Demonstrația că o problemă NP-completă se reduce în sl. la ACG

$3\text{-SAT} \xrightarrow{\text{de}} ACG$

Fie $\alpha = C_1 \wedge C_2 \wedge \dots \wedge C_n$

$C_i = \{y_{i1}, \vee y_{i2}, \vee y_{i3}\} \quad i = \overline{1, m}$

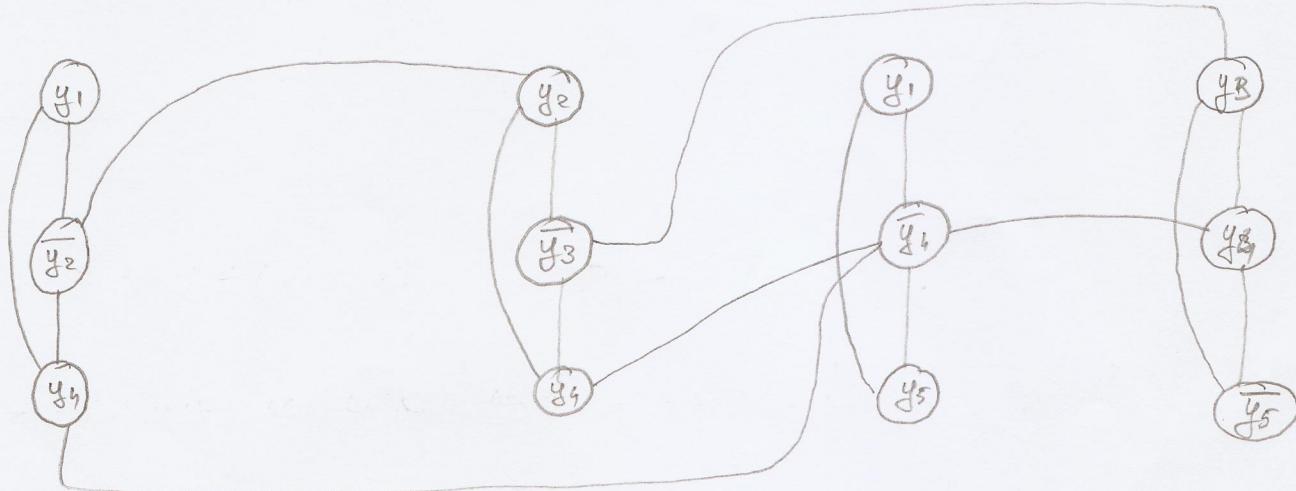
Construim graful $G = (V, E)$, $V = \{y_{it} \mid i = \overline{1, m}, t = \overline{1, 3}\}$

$E = \{y_{i0}, y_{ir}\} \in E \quad i = \overline{1, m} \quad \wedge 1 \leq r \leq 3$

$\{y_{i0}, y_{jt}\} \in E \Rightarrow y_{i0} = \overline{y_{jt}}$

Afirmatie: α este satisfiabilă dacă și numai dacă graful G are o acoperire de dimensiune $2m$;

$$\alpha = (y_1 \vee \bar{y}_2 \vee y_4) \wedge (y_2 \vee \bar{y}_3 \vee y_4) \wedge (y_1 \vee \bar{y}_4 \vee y_5) \wedge (y_3 \vee y_4 \vee \bar{y}_5)$$



"Dacă"

Eliminând nodurile din acoperire ($2m$) rămân m noduri. Ele m noduri sunt distribuite câte trei în fiecare subgraf complet de 3 noduri. Atribuirea valoarei 1 literelor lui corespunzătoare. Atribuirea făcută este consistență și în urma ei α este satisfiabilă.

"Numai dacă"

α satisfiabilită \Rightarrow un literal din f. obiectă are val 1

Dând la o parte din graf nodurile ce corespund literelor cu valoarea 1 rămân exact $2m$ noduri ce reprezintă acoperirea (fiecare subgraf complet este acoperit și tot graful este acoperit).

Spatiu logoricomic:

muchile grafului se construiesc următoare la nodurile din obiectă și imediat că terminăm să clătim sau nu mai întârcem \Rightarrow pe baza cărora vom merge mereu la dreapta. (se va ajunge la spațiu logoricomic)