Cursul I

OPREA TUDOR
FMI GRUPA 341

Undecidabilitatea se referă la probleme care nu pot fi rezolvate de algoritmi.

Calculatoarele par a fi atât de evoluate încât să poată rezolva orice problemă, însă teorema următoare demonstrează contrariul. Există multe probleme care sunt computațional nerezolvabile de către calculator.

Am studiat în primul rând dacă o mașină Turing acceptă un input string. $A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a TM and } M \text{ accepts } w \}$.

A_{TM} - este undecidabil.
- este Turing Recognizable.

Constr. o TM D care preia inputul ⟨M⟩ și face exact opusul: dacă U dă acc => rej și invers. Dacă D există D
=> A_{TM} decidabil

Putem construi o mașină Turing U = On input ⟨M, w⟩ where M is a TM and w is a string:
1. Simulează M pe input w
2. M intră în stare de acc => acc
altfel reject

U buclează (loopează) pe inputul w <=> M loopează pe w
↓
deci în acest caz U nu ar decide A_{TM} pentru că nu ar ajunge la reject. Trebuie ca U să-și dea seama atunci când M nu se oprește pe w, pentru a-l respinge.

pag. 6 ~> demonstrație

Demonstrația undecid. pentru A_{TM} se face prin metoda diagonalizării, descoperită de Georg Cantor în 1873.

El a dorit să dea un nou înțeles mulț. Anume și s-a întrebat dacă 2 mulț. infinite pot fi comparate din pct. de vedere al cardinalului. A ajuns la o sol. pentru această

Problemă, și arume →

Două mulț. au același cardinal dacă putem împerechea fiecare element din prima cu fiecare din a doua, cu alte cuvinte să găsim o bijecție între cele două mulțimi.

O mulțime se ~~numește~~ numește numărabilă dacă are același nr. de elemente cu N, unde

$$N = \{1, 2, 3, \dots\}$$

ex. $E = \{2, 4, 6, \dots\}$ → are ac. nr de el. cu N sau cu alte cuv. este numărabilă, deoarece $\exists$ funcție $f: N \to E$, $f(n) = 2n$ bijectivă

Prin metoda diagonalizării vom arăta faptul că Q, unde $Q = \{\frac{m}{n} \mid m, n \in N\}$ este numărabilă.

Construim o matrice după regula $a[i][j] = \frac{i}{j}$

intr.

$$\begin{array}{cccccc}
\frac{1}{1} & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \dots \\
\frac{2}{1} & \frac{2}{2} & \frac{2}{3} & \frac{2}{4} & \frac{2}{5} & \dots \\
\frac{3}{1} & \frac{3}{2} & \frac{3}{3} & \frac{3}{4} & \frac{3}{5} & \dots \\
\frac{4}{1} & \frac{4}{2} & \frac{4}{3} & \frac{4}{4} & \frac{4}{5} & \dots \\
\frac{5}{1} & \frac{5}{2} & \frac{5}{3} & \frac{5}{4} & \frac{5}{5} & \dots \\
\vdots & \vdots & \vdots & \vdots & \vdots &
\end{array}$$

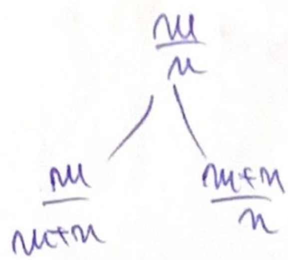
Matricea este infinită

Elementele se vor repeta $(\frac{1}{1}, \frac{2}{2}, \dots)$ așa că le vom sări pe acelea.

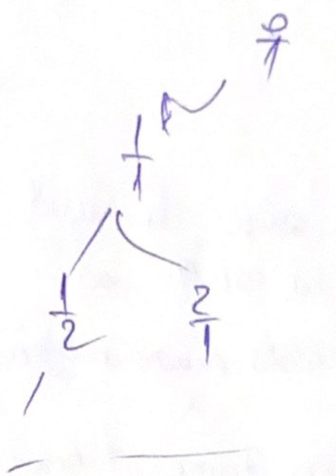
Astfel, parcurgând matricea, vom găsi o bijecție între toate elem. lui Q și N, deci Q este numărabilă

practic $f(1) = \frac{1}{1}$, $f(2) = \frac{2}{1}$, $f(3) = \frac{1}{2}$ și tot așa.

O altă dem ar fi str lui $\mathbb{Q}$ sub forma unui arbore, ~~...~~ de forma:



→ desc. în anul 2000



Mulțimile nenumărabile sunt mulțimi infinite pentru care nu se poate găsi o bijecție cu $\mathbb{N}$. ~~...~~ O astfel de mulțime ar fi $\mathbb{R}$, ~~...~~ Dem: Cantor

Vom arăta că nu există o bijecție între $\mathbb{N}$ și $\mathbb{R}$, prin construirea unui $x \in \mathbb{R}$ care nu are un corespondent în $\mathbb{N}$.

Cum îl construim?

Alegem $x \in (0,1)$.

Pt. orice $n \in \mathbb{N}$ punem a n-a zecimală a lui x să fie dif. de a n-a zecimală a lui $f(n)$ deja atribuită.

Astfel, x nu se va găsi în mulțimea $f(n)$ deoarece $\forall\, n \in \mathbb{N}$, a n-a zecimală a numărului e dif. de a n-a zecimala a lui x.

Limbaje non Turing-recognizable

O mașină rec. un limbaj

nr mas. < nr limbajelor

$\Rightarrow \exists$ limbaje care nu sunt T-r

Pt. a arăta că mulț. tut. limbajelor e nenumărabilă, arătăm că mulț. secv. binare ~~co~~ infinite e nenumărabilă prin metoda diagonalizării.

Fie B = mulțimea tuturor sec. binare infinite e nenum. dar că B e nenum. prin met. diagonaliz.

$\mathcal{L}$ e mulț. limb pe alfab. Σ — $\mathcal{L}$ = countable. Că dau corep cu

$\Sigma^* = \{s_1, s_2, \ldots\}$

constr. A de forma ~~...~~ alfab. care încep cu

$x_A = \begin{cases} 0, & s_i \notin A \\ 1, & s_i \in A \end{cases}$ secv. caract.

$x_a = 0$ dc. $s_i \in A$, 1 altfel.

$f: \mathcal{L} \to B$ bijectivă

B numărabilă $\to \mathcal{L}$ numărabilă $\to$ Mulț. limbajelor nu poate fi pusă în corep. cu mulțimea tuturor lui H mașinilor turing.

H decider $\to$ D face MU

D poate primi D ? contradicție

$\Rightarrow \nexists$ o mașină care

• ~~Un limbaj e decid. $\Leftrightarrow$ e Turing rec. și~~ co Turing rec

e turing rec dacă și el și complem sau sunt T.R.

e co-turing rec dacă e complementul unui lb. TR

A_{TM} undecidabil → demonstr.

R. prin reducere la absurd că ar fi

H = decider pt A_{TM} on input ⟨M, w⟩ { acc dacă M acc / resp dacă M resp

D = face qurul
D = la input rulează H pe input ⟨~~H~~, ⟨M⟩⟩ și face exact qurul

D(⟨M⟩) = { acc dacă M nu acc ⟨M⟩ / reject dacă M acc ⟨M⟩

ce s-ar întâmpla dacă am rula D pe el însuși? D(⟨D⟩) = ——.

contradicție + explicați. ⇒ nici D nici H nu pot exista

⇒ A_{TM} este undecidabil.

T

Subiectul al II-lea

OPREA TUDOR
FMI GRUPA 341

NP e clasa limbajelor care au verificatori în timp polin.
↳ Non deterministic polinomial time

Un limbaj este NP ⟺ este decis de o MT. non-det în timp polinomial.

Prob. rezolvabile în

~~P ⊊ NP~~
P ⊂ NP
& pp. că P ⊊ NP neadev.

SAT:
→ O prob NP-completă este prob satisfiability.
Avem o formulă booleană compusă din variabile care pot avea valorile 0/1 și operații boolene. De ex. avem

$\phi = (\neg x \wedge y) \vee (x \wedge \neg z)$

ea e sat ~~pentru~~ dacă după asignarea val. de 0 și 1 pt x, y, z va fi egală cu 1

ϕ = SAT pt. x = ... pt că e = 1

SAT = { <ϕ> | ϕ is a satisf. boolean formula

SAT ∈ P ⟺ P = NP

REDUCTIBILITY:
Def: Când o problemă A e eficient reductibilă la o prob B, o sol. eficientă a lui B poate fi folosită pt a rez. A eficient.

Reductibility:

O funcție $f: \Sigma^* \to \Sigma^*$ se numește funcție computabilă în timp polinomial dacă $\exists$ o mașină Turing ? în timp polinomial care se oprește având doar $f(w)$ banda sa, ~~la încep~~ unde ne este rezul.

Limbajul A este <u>reductibil</u> în timp polinomial la limbajul B (not. $A \le_p B$), dacă $\exists$ o funcție computabilă în timp polinomial $f: \Sigma^* \to \Sigma^*$ a.î. $\forall\ w \in A \Rightarrow f(w) \in B$ și reciproc

f = reducție în timp polinomial a lui A în B

=

- $\exists$ forme mai eficiente de reduct., dar t.tc polinomial e cea mai adecv pt noi
- casă testăm $w \in A$, folosim f — și test $f(w) \in B$
- dacă un lb. e reduct. în t. polin. la un lb. cu o sol. cunoscută în timp polinomial $\Rightarrow$ găsim o sol. în timp polin. și pt lb. original,

$A \le_p B$, $B \in P$ $\Downarrow$ $A \in P$

Fie M alg în timp polin. care decide B și f reducția în timp polinomial de la A la B.

Alg. N care decide A este:

1. calc $f(w)$
2. rulează M pe $f(w)$ și dă output ce dă M

SAT:

- literal: x $\bar{x}$
- clauză: $x_1 \vee x_2$
- formulă booleană: clauze conectate cu $\wedge$
- 3cnf-formula: când toate clauzele au 3 literali

3SAT = { $\langle \phi \rangle$ | ϕ is a 3cnf-formula }

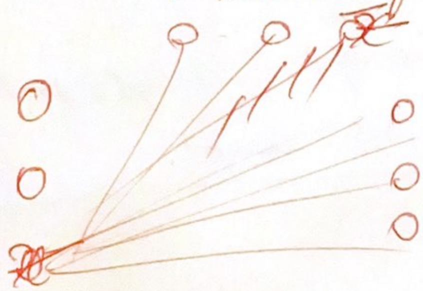
Th: 3SAT $\leq_p$ CLIQUE

pentru a demonstra, fie $\phi = (a_1 \vee b_1 \vee c_1) \wedge \ldots (a_k \ldots)$

→ Reducția f generează $\langle G, k \rangle$, unde G este un graf ~~neorientat construit astfel~~:

→ luăm fiecare ~~triplet~~ clauza, o facem triplet, iar fiecărui nod îi coresp. un literal.

Vom constr. muchii între toate nodurile, în afară de:
- nodurile din același triplet
- nodurile opuse $(x, \bar{x})$



Pt. a construi k-click-ul trebuie să selectăm din fiecare triplet câte un nod care este corespondentul unui literal cu val. de adv. 1.

UNIM LITERALII CU TRUE.

→ ...

(3)

Proof: Am ales valorile corect pt. ~~...~~ ,

ale clickului

p că garantat ∄ 2 noduri din același triplet deoarece nodurile din ac. triplet ~~contra exact~~ nu sunt legate cu muchii.

K TRIPLETE $\rightarrow$ K NODURI ÎN CL.

Asignăm val 1 lui ϕ $\Rightarrow$ fiecare literal al celui este (A).

NP completness:

CLIQUE $\in$ ~~NP~~ P $\Rightarrow$ 3SAT $\in$ P

Un limbaj B este NP-complet dacă

1. B este NP
2. $\forall A \in NP$, $A \leq_p B$

Th B = NP-complet, $B \in P$ $\Rightarrow$ $P = NP$.

dem: Teorema e dedusă din def. reductibilității în timp polinomial.

Th Dacă B = NP-compl., $B \leq_p C$ pentru C din NP $\Rightarrow$ C este NP-compl.

Dem:

$C = NP$

$B = NP \Rightarrow \forall D \in NP, D \leq_p B$

$B \leq_p C$

$A \leq_p B$, $B \leq_p C$ $\Rightarrow A \leq_p C$ $\Rightarrow$ Orice lb. din NP e polin. reductibil la C.

4

~~C~~ook - Levin th :

SAT is NP - complete.

3 SAT e NP complet :

dem: convertire SAT to CNF.

$$(a_1 \vee a_2 \vee a_3 \vee a_4)$$

$$(a_1 \vee a_2 \vee z) \wedge (\bar{z} \vee a_3 \vee a_4)$$

spargem in bec cate 3

1/2/~~3~~ : adaugam

3+ : spartire ~~x~~ adaugam

caz general : $a_1 \vee a_2 \;\text{--}\; \vee a_l$

$$\Downarrow$$

$$(a_1 \vee a_2 \vee z_1) \wedge (\bar{z_1} \vee a_3 \vee z_2) \wedge (\bar{z_2} \vee a_4 \vee z_3)$$

$$\ldots \wedge (\bar{z}_{l-3} \vee a_{l-1} \vee a_l)$$

$$(a_1 \vee a_2 \vee z_1) \wedge (\bar{z_1} \vee a_3 \vee z_2) \wedge (\bar{z_2} \vee a_4 \vee z_3) \wedge (\bar{z}_{l-3} \vee a_{l-1} \vee a_l)$$