CALCULABILITATE ŞI COMPLEXITATE: PROBLEME FÃCUTE LA SEMINAR

GABRIEL ISTRATE

Problemele sunt prezentate pe scurt. Uitaţi-vã pe notiţele voastre pentru detalii.

1. Seminarul nr. 1

- Funcţia $< x, y >$, bijecţie intre $\mathbf{N} \times \mathbf{N}$ şi $\mathbf{N}$. Formula pentru funcţie, programe pentru inversele sale.
- Bijecţia intre $\cup_{k \geq 1} \mathbf{N}^k$ şi $\mathbf{N}$.
- Recapitulare noţiunea de automat finit.
- Automate finite pentru divizibilitatea cu 3,5,7.
- Automate celulare. Codificarea lui Wolfram. Demonstrat in Netlogo Game of Life şi regula 110.

2. Seminarul nr. 2

- Recapitulare: noţiunea de funcţie primitiv recursivã, precum şi a demonstraţie cã $f(x, y) = x + y$ este primitiv recursivã.
- Probleme: sã se arate cã urmãtoarele funcţii sunt primitiv recursive
  - $x \cdot y$
  - $f(x,y) = \begin{cases} 0 & \text{dacã } n = 0 \\ m & \text{altfel.} \end{cases}$
  - $sign(x) = \begin{cases} 0 & \text{dacã } x = 0 \\ 1 & \text{altfel.} \end{cases}$
  - $Pred(x) = \begin{cases} 0 & \text{dacã } x = 0 \\ x - 1 & \text{altfel.} \end{cases}$
  - $x \ominus y = \begin{cases} 0 & \text{dacã } x < y \\ x - y & \text{altfel.} \end{cases}$
  - Funcţia $< x, y >$ de la Seminarul 1.
- Noţunea de program in limbajul LOOP. Problemã: cum simulãm operaţiile de compunere şi recurenţã primitivã in limbajul LOOP ?
- Sã se scrie o maşinã Turing care recunoaşte cuvintele peste alfabetul $\{0, 1\}$ care sunt palindroame.

3. Seminarul nr. 3

- Exemplu de automat celular: care este starea automatului 110 care initial are pe banda un singur 1 si restul 0 la $t = 3$?

- Cum putem simula automatul finit pentru divizibilitatea cu 3 cu un automat celular unidimensional ? Cum se modificã rezultatul pentru simularea unei maşini Turing ?
- Douã exemple de pavaje Wang: sã se decidã dacã pot sau nu sã paveze planul.
- Noţiunea de reducere. Exemplu: sã reducem problema 3-colorãrii unui graf la problema satisfiabilitãţii.

4. Seminarul nr. 4

- Cum implementãm in Python rezolvarea problemei 3-colorãrii cu ajutorul reducerii la problema satisfiabilitãţii (via SAT-solving).
- Determinarea existenţei unui pavaj Wang periodic via SAT solving.

5. Seminarul nr. 5

- 2-COL in timp polinomial.
- HORN-SAT in timp polinomial.
- algoritmul bazat pe grafuri pentru 2-SAT.
- algoritmul bazat pe rezoluţie pentru 2-SAT.
- Majority 3-SAT e o problemã polinomialã.
- SAT: fiind datã o procedurã pentru determinarea satisfiabilitãţii unei formule, sã de determine un algoritm care sã **gãseascã** o soluţie pentru o formulã (dacã aceasta este satisfiabiliã).
- IP feasibility este NP-hard.

6. Seminarul nr. 6

- NAE-2SAT reduces to 2-COLORING.
- 3-SAT reduces to NAE-4SAT reduces to NAE-3SAT.
- satisfiability of linear equations over $\mathbf{Z}_2$ is in P.
- 3-SAT reduces to satisfiability of cubic equations over $\mathbf{Z}_2$ (which reduces to satisfiability of quadratic equations over $\mathbf{Z}_2$).
- SAT is not p-selective.
- Arãtaţi cã $A \oplus B$ este supremul lui A, B faţã de $\leq_m^P$.
- Dacã $A \leq_m^P C$ şi $B \leq_m^P C$ atunci $A \oplus B \leq_m^P C$, unde $A \oplus B = \{x0 : x \in A\} \cup \{y1 : y \in B\}$. Deduceţi faptul cã $A \in NP$ şi B in NP atunci $A \oplus B \in NP$.

7. Seminarul nr. 7

- Generarea permutarilor cu backtracking: nr biţi de memorie folosiţi este $O(n \log(n))$.
- Tic-tac-toe on $n \times n$ boards: determining a winner is in PSPACE.
- Geography in PSPACE (Moore & Mertens, pp. 330).
- Arborele de decizie pentru 2-player SAT (figura 8.12, Moore & Mertens)
- Algoritmul de aproximare pentru Vertex Cover.
- Kernelizarea pentru Vertex Cover.