Seminar 1 - Exerciții rezolvate

Echipa AA

19 octombrie 2022

Cuprins

Exercițiul 1

Fie mulțimile A, B disjuncte și complementare. Demonstrați că dacă A, B sunt recursiv-enumerabile, atunci acestea sunt și recursive.

Rezolvare

Rescriem ipoteza în termeni matematici:

1. $A \cap B = \emptyset$
2. $A \cup B = \mathbb{N}$
3. $A, B \in \mathcal{RE}$

Notăm cu $\mathcal{R}$ mulțimea tuturor mulțimilor recursive, cu $\mathcal{RE}$ mulțimea tuturor mulțimilor recursive-enumerabile, iar $\mathcal{P}$ cu este mulțimea tuturor programelor.

Din 1 și 2 ar fi util să ne facem un desen ca să ne putem gândi mai bine.

Din 3 rezultă că $\exists P_A, P_B \in \mathcal{P}$, astfel încât

$$P_A(x) = \begin{cases} 1 & \text{dacă } x \in A \\ \perp & \text{altfel} \end{cases}$$

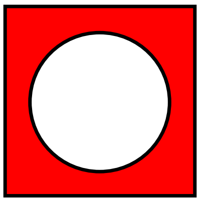

Figura 1: A este mulțimea albă, B cea roșie, iar chenarul mare negru mulțimea numerelor naturale.

$$P_B(x) = \begin{cases} 1 & \text{dacă } x \in B \\ \perp & \text{altfel} \end{cases}$$

Pentru a demonstra că $A \in \mathcal{R}$ trebuie să găsim $Q_A \in \mathcal{P}$, astfel încât

$$Q_A(x) = \begin{cases} 1 & \text{dacă } x \in A \\ 0 & \text{altfel} \end{cases}$$

Pentru a îl construi pe Q_A ne vom folosi de ce programe avem deja, P_A și P_B, și în funcție de outputurile lor să ne dăm seama dacă putem afirma/infirma apartenența la A.

A returnat P_A 1 pentru x?	A returnat P_B 1 pentru x?	$x \in A$?
DA	NU	DA
NU	DA	NU

Observați că nu am analizat cazul în care ambele programe returnează același rezultat. Acest fapt este imposibil, deoarece putem deduce că $\forall x \in \mathbb{N}, x \in A$ sau (exclusiv) $x \in B$, fie din desen, fie din ipotezele 1 și 2.

Pentru a putea scrie programul Q_A și nu știm care din P_A sau P_B se termină. Putem simula o paralelizare a celor două rulând un număr finit de instrucțiuni din primul program, apoi ruleazând un număr finit de instrucțiuni din al doilea, și tot așa. (Acest procedeu se cheamă în engleză dovetailing. Există o asemănare cu noțiunea de multitasking.)

```
// rulează funcția P asupra inputului x în steps pași
// dacă funcția a returnat în mai puțini pași, returnează valoarea ei
// altfel returnează o referință nulă
Boolean run_in_steps(BooleanFunction P, Input x, Integer steps);

bool QA(int x) {
  for (Integer t = 0; ; t++) { // t crește la infinit
    if (run_in_steps(PA, x, t) == true)
      return true;

    if (run_in_steps(PB, x, t) == true)
      return false;
  }
}

bool QB(int x) {
  return !QA(x);
}
```

Deoarce am găsit un program, atât pentru A, cât și pentru B, care răspunde într-un timp finit da sau nu legat de apartenența unui input la o mulțime, **ambele mulțimi sunt recursive**.

Generalizare

Fie $A_1, A_2, \ldots, A_n \in \mathcal{RE}$.

1. $\bigcap_{i=1}^{n} A_i = \emptyset$
2. $\bigcup_{i=1}^{n} A_i = \mathbb{N}$

$$P_A(i,x) = \begin{cases} 1 & \text{dacă } x \in A_i \\ \perp & \text{altfel} \end{cases}$$

```
bool Q_A(int i, int x) {
  for (Integer t = 0; ; t++) {
    if (run_in_steps(PA, (i, x), t) == true)
      return true;

    for (int j = 1; j <= n; j++)
      if (i != j && run_in_steps(PA, (j, x), t) == true)
        return false;
  }
}
```

Exercițiul 2

Fie $A \in \mathcal{R}$ și $B \in \mathcal{RE} \setminus \mathcal{R}$. Este $A \cup B \in \mathcal{RE}$?

Rezolvare

Fie P_A, P_B programele asociate mulțimilor A, B.

Știm că P_A se termină mereu, iar P_B se termină măcar doar când inputul aparține lui B. Construim $Q \in \mathcal{P}$ care poate afirma aparteneța la $A \cup B$,

```
bool Q(int x) {
  if (PA(x) == true)
    return true;
  return PB(x);
}
```

Exercițiul 3

Pornind de la exercițiul anterior, demonstrați că $A \cup B \in \mathcal{R}$ sau $A \cup B \notin \mathcal{R}$.

Rezolvare

1. $A = \emptyset$ și $B = \{(P, x) \in \mathcal{P} \times \mathbb{N} | P(x) \neq \perp\}$, adică:
   1. A este mulțimea vidă
   2. B este mulțimea tuplurilor de programe și inputuri, cu propietatea că acel program se oprește pe acel input
   3. $A \cup B = B, B \in \mathcal{RE} \setminus \mathcal{R}$[1]
2. $A = \mathcal{P} \times \mathbb{N}$ și $B = \{(P, x) \in \mathcal{P} \times \mathbb{N} | P(x) \neq \perp\}$, unde:
   1. A este mulțimea tuturor programelor și a inputurilor
   2. Deoarce verificarea că un șir este un program sau nu e o problemă decidabilă (spre exemplu compilatoarele acceptă sau nu un program), mulțimea A este recursivă
   3. $A \cup B = A, A \in \mathcal{R}$

Exercițiul 4

Mulțimea $A \in \mathcal{RE} \setminus \mathcal{R}$. Ce putem spune despre complementul lui A, $A^{\complement}$?

[1]Mulțimea B descrie problema opririi, care este semi-decidabilă.

Rezolvare

Presupunem prin absurd, $A^{\complement} \in \mathcal{RE}$. $\exists P_A, P_{A^{\complement}} \in \mathcal{P}$ astfel încât:

$$P_A(x) = \begin{cases} 1 & \text{dacă } x \in A \\ \perp & \text{altfel} \end{cases}$$

$$P_{A^{\complement}}(x) = \begin{cases} 1 & \text{dacă } x \in A^{\complement} \\ \perp & \text{altfel} \end{cases}$$

Construim $Q_A \in \mathcal{P}$ care decide apartenența la A.

```
bool QA(int x) {
  for (Integer t = 0; ; t++) {
    if (run_in_steps(PA, x, t) == true)
      return true;

    if (run_in_steps(PAC, x, t) == true)
      return false;
  }
}
```

$\Rightarrow A \in \mathcal{R}$. Contradicție!

$$A \in \mathcal{RE} \setminus \mathcal{R} \Rightarrow A^{\complement} \notin \mathcal{RE}$$

Exercițiul 5

Fie mulțimea $A \notin \mathcal{RE}$, și $B \subset A$ astfel încât $A \setminus B \in \mathcal{R}$. Ce putem spune despre B?

Rezolvare

Presupunem prin absurd $B \in \mathcal{RE} \Rightarrow \exists P_{AminusB}, P_B \in \mathcal{P}$, astfel încât

$$P_{AminusB}(x) = \begin{cases} 1 & \text{dacă } x \in A \setminus B \\ 0 & \text{altfel} \end{cases}$$

$$P_B(x) = \begin{cases} 1 & \text{dacă } x \in B \\ \perp & \text{altfel} \end{cases}$$

Atunci putem construi $P_A \in \mathcal{P}$ care afirmă aparteneța la mulțimea A.

```
bool PA(int x) {
  if (PA_minus_B(x))
    return true; // x face parte din A
  return Pb(x); // B inclus în A
}
```

Deci $A \in \mathcal{RE}$. Contradicție! $\Rightarrow B \notin \mathcal{RE}$

Exercițiul 6

Fie $A \in \mathcal{RE}$, și mulțimea $B = \{x + 1 | x \in A\}$. Dacă $A \cap B = \emptyset$ și $A \cup B \in \mathcal{R}$ demonstrați că $A, B \in \mathcal{R}$.

Rezolvare

Intuim că $B \in \mathcal{RE}$.

```
bool PB(int x) {
  return PA(x - 1);
}
```

$\Rightarrow B \in \mathcal{RE}$. Rezolvarea mai departe este asemănătoare cu exercițiul 1. Următoarele programe decid apartenența la mulțimile A și B.

```
bool QA(int x) {
  if (!PA_plus_B(x))
    return false;
  for (Integer t = 0; ; t++) {
    if (run_in_steps(PA, x, t) == true)
      return true;

    if (run_in_steps(PB, x, t) == true)
      return false;
  }
}

bool QB(int x) {
  if (!PA_plus_B(x)) // prog. care decide apartenența la reuniune
    return false;
  for (Integer t = 0; ; t++) {
    if (run_in_steps(PA, x, t) == true)
      return false;

    if (run_in_steps(PB, x, t) == true)
      return true;
  }
}
```

Exercițiul 7

Fie $A \subset N$ care nu are nicio pereche de elemente consecutive și $B = \{x+1 | x \in A\}$. Demonstrați că dacă $A \cup B$ recursivă atunci și A și B sunt recursive.

Rezolvare

Observație: $\forall x \in A, x+1 \notin A$, această regulă se respectă și pentru B. (Demonstrația se poate face prin reducere la absurd.)

Acum, trebuie să ne imaginăm cum sunt distribuite elementele din A, respectiv din B, în mulțimea $A \cup B$.

Dacă $x+1 \in A \cup B$, atunci:

$$x+1 \in \begin{cases} A & \text{dacă } x \in B \\ B & \text{dacă } x \in A \end{cases}$$

Dacă $x \in A \cup B$, atunci $y_x = max(\{n \in \mathbb{N} | n \notin A \cup B, n < x\})$ (cu alte cuvinte, y_x fiind cel mai mare număr mai mic decât x care nu aparține reuniunii), $y_x + 1 \in A$. (Altfel s-ar contrazice paragraful anterior și $y_x \in A \cup B$, contradicție!)

Următoarul program decide apartenența la mulțimea A.

```
bool PA(x) {
  Boolean prev = null;

  // căutăm primul nr. care nu aparține reuniunii
  int i = x;
```

```
  while (!PA_plus_B(i))
    i--;

  // cel mai mare număr mai mic decât x care nu aparține reuniunii este i
  bool res = false;
  for (int j = i + 1; j <= x; j++)
    res = !res;

  return res;
}
```

Asemenea se poate construi $Q_B \in \mathcal{P}$ pentru B.

Exercițiul 8

Mulțimea $A \notin \mathcal{RE}$. Ce putem spune despre complementul lui A, $A^{\complement}$?

Rezolvare

Presupunem prin absurd $A^{\complement} \in \mathcal{R}$. Precum $\mathcal{R}$ este închisă pe operația de complement, rezultă că $A \in \mathcal{R}$, contradicție!

Presupunem prin absurd $A^{\complement} \in \mathcal{RE}$ și o să contrazicem presupția printr-un exemplu.

Caz particular cunoscut

Fie $A = \{P \in \mathcal{P} | \forall x \in \mathbb{N}, P(x) \neq \bot\}$ și $A^{\complement} = \{P \in \mathcal{P} | \exists x \in \mathbb{N}, P(x) = \bot\}$.

Cu alte cuvinte A este mulțimea tuturor programelor care se termină pe orice input, iar $A^{\complement}$ este mulțimea programelor pentru care există cel puțin un input care duce la o execuție fără sfârșit.

Încă o reformulare poate fi: orice program din A, decide o problemă.[2]

Se poate demonstra că $A, A^{\complement} \notin \mathcal{RE}$. (Carl Mummert, f.a.)

Observație

$A = \{(P, x) \in \mathcal{P} \times \mathbb{N} | P(x) = \bot\}$, iar $A^{\complement}$ desemnează problema opririi, $A^{\complement} \in \mathcal{RE} \setminus \mathcal{R}$.

Deci putem spune doar că $A^{\complement} \notin \mathcal{R}$.

Bibliografie

Carl Mummert. f.a. „Show that the language TOT is not recursively enumerable nor its complement." Mathematics Stack Exchange. https://math.stackexchange.com/q/578107.

[2]Mulțimea A este un exemplu cunoscut de Π_2-completitudine.