CODURI – Seminar 6

Definitie Fie A o multime nevida. Se numeste cod peste A orice submultime nevida C a lui A^+

care genereaza liber un submonoid al lui A^*.

Daca C este un cod peste A atunci submonoidul liber generat de C se noteaza cu C^*.

Observatia 1. $C^+ = C^* - \{1_{A^*}\}$

Elementele din C se numesc cuvinte cod.

Observatia 2. C este un cod peste A, daca orice cuvant $w \in C^+$ poate fi descompus in mod unic in C. Aceasta proprietate face ca orice codificare a unui text (obtinand o secventa din C^+) sa poata fi decodificat in mod unic (altfel decodificarea este ambigua, deci C nu este un Cod).

Teorema lui Sardinas-Patterson

Fie A o multime nevida. $C \subseteq A^+$, $C \neq \emptyset$. Atunci C este un cod peste A daca si numai daca $C \cap C_i = \emptyset$, pentru orice $i \geq 1$, unde multimile C_i sunt construite astfel:

$\mathbf{C_1 = \{x \in A^+ |\ \exists c \in C}$ **astfel incat** $\mathbf{cx \in C\}}$ "traducere" exista un cuvant cod c in C care este prefix al unui alt cuvant cod din C. Notam acest cuvant cu cx. Punem x in multimea C_1.

Cum construim C_1 : Se ia primul cuvant din C si se verifica daca este prefix pentru celelalte cuvinte din C. Daca DA se pune fiecare sufix (ale cuvintelor pt. care c este prefix) in C_1. Acest lucru se repeta pentru al doilea cuvant din C, al treilea cuvant din C, si asa mai departe pana se epuizeaza toate cuvintele din C.

$\mathbf{C_2 = \{x \in A^+ |\ (\exists c \in C}$ **a.i.** $\mathbf{cx \in C_1)}$ **sau** $\mathbf{(\exists c \in C_1}$ **a.i.** $\mathbf{cx \in C)\}}$

"traducere" pt. $\mathbf{(\exists c \in C}$ **a.i.** $\mathbf{cx \in C_1)}$ exista un cuvant cod c in C care este prefix al unui alt cuvant din C_1. Notam acest cuvant cu cx. Punem x in multimea C_2.

"traducere" pt. $\mathbf{(\exists c \in C_1}$ **a.i.** $\mathbf{cx \in C)}$ exista un cuvant c in C_1 care este prefix al unui cuvant cod din C. Notam acest cuvant cu cx. Punem x in multimea C_2.

Cum construim C_2 : **Pas1.** Se ia primul cuvant din C si se verifica daca este prefix pentru cuvinte din C_1. Daca DA se pune fiecare sufix (ale cuvintelor din C_1 pt. care c este prefix) in C_2. Acest lucru se repeta pentru toate cuvintele din C.

Pas2. Se ia primul cuvant din C_1 si se verifica daca este prefix pentru cuvinte cod din C. Daca DA se pune fiecare sufix (ale cuvintelor din C pt. care c este prefix) in C_2. Acest lucru se repeta pentru toate cuvintele din C_1.

$\mathbf{C_3 = \{x \in A^+ \mid\ (\exists\ c \in C\ a.i.\ cx \in C_2) \vee (\exists\ c \in C_2\ a.i.\ cx \in C)\ \}}$

"traducere" pt. $\mathbf{(\exists\ c \in C\ a.i.\ cx \in C_2)}$ exista un cuvant cod c in C care este prefix al unui alt cuvant din C_2. Notam acest cuvant cu cx. Punem x in multimea C_3.

"traducere" pt. $\mathbf{(\exists\ c \in C_2\ a.i.\ cx \in C)}$ exista un cuvant c in C_2 care este prefix al unui cuvant cod din C. Notam acest cuvant cu cx. Punem x in multimea C_3.

Cum construim C_3 : **Pas1.** Se ia primul cuvant din C si se verifica daca este prefix pentru cuvinte din C_2. Daca DA se pune fiecare sufix (ale cuvintelor din C_2 pt. care c este prefix) in C_3. Acest lucru se repeta pentru toate cuvintele din C.

Pas2. Se ia primul cuvant din C_2 si se verifica daca este prefix pentru cuvinte cod din C. Daca DA se pune fiecare sufix (ale cuvintelor din C pt. care c este prefix) in C_3. Acest lucru se repeta pentru toate cuvintele din C_2.

"Redundanta" Presupunand ca s-a construit multimea $\mathbf{C_i}$, construim $\mathbf{C_{i+1}}$

$\mathbf{C_{i+1} = \{x \in A^+ \mid\ (\exists\ c \in C\ a.i.\ cx \in C_i) \vee (\exists\ c \in C_i\ a.i.\ cx \in C)\ \}}$

"traducere" pt. $\mathbf{(\exists\ c \in C\ a.i.\ cx \in C_i)}$ exista un cuvant cod c in C care este prefix al unui alt cuvant din C_i. Notam acest cuvant cu cx. Punem x in multimea C_{i+1}.

"traducere" pt. $\mathbf{(\exists\ c \in C_i\ a.i.\ cx \in C)}$ exista un cuvant c in C_i care este prefix al unui cuvant cod din C. Notam acest cuvant cu cx. Punem x in multimea $\mathbf{C_{i+1}}$.

Cum construim C_{i+1} : **Pas1.** Se ia primul cuvant din C si se verifica daca este prefix pentru cuvinte din $\mathbf{C_i}$. Daca DA se pune fiecare sufix (ale cuvintelor din $\mathbf{C_i}$ pt. care c este prefix) in $\mathbf{C_{i+1}}$. Acest lucru se repeta pentru toate cuvintele din C.

Pas2. Se ia primul cuvant din C_i si se verifica daca este prefix pentru cuvinte cod din C. Daca DA se pune fiecare sufix (ale cuvintelor din C pt. care c este prefix) in $\mathbf{C_{i+1}}$. Acest lucru se repeta pentru toate cuvintele din $\mathbf{C_i}$.

Observatia 3. Pentru fiecare multime $\mathbf{C_i}$ construita se verifica daca $C \cap C_i = \oslash$. Daca DA se trece la counstuirea urmatoarei multimi $\mathbf{C_{i+1}}$ (C inca este cod). Daca NU, adica $C \cap C_i \neq \oslash$, C nu este cod (nu satisface Teorema lui Sardinas-Patterson). Deci ne oprim cu constructia multimilor C_i . Raspunsul este **C nu este cod!**

Observatia 4. La un moment dat (incepand de la un anumit index i) fie se obtine $\mathbf{C_i}$ multime vida (ceea ce inseamna ca toate celelate multimi $\mathbf{C_j}$ $j \geq i$, vor fi vide, deci $C \cap C_j = \oslash$ pentru orice index $j \geq i$). Fie se obtin multimi egale, $\exists$ un $j > i$, a.i. $C_i = \mathbf{C_j}$. Cum $C \cap C_i = \oslash$ avem $C \cap C_j = \oslash$ pentru orice index $j \geq i$. Asadar nu mai are sens sa mai construim alte multimi. In ambele cazuri ne oprim cu raspunsul **C este cod!**

Observatia 5. Algoritmul lui Sardinas-Patterson se termina intotdeauna intr-un numar finit de pasi. Fie obtineti o intersectie nevida (exista i ai. $C \cap C_i \neq \oslash$ caz in care C nu este cod), fie obtineti o multime C_i

astfel incat aceasta este vida, si automat $C \cap C_i = \emptyset$, (si $C_i = C_j = \emptyset$ pentru orice index $j \geq i$) fie C_i este nevida, $C \cap C_i = \emptyset$, si $\exists$ un $j > i$, a.i. $C_i = C_j$. (In ultimele doua cazuri C este cod).

Exemple

Ex1. Fie $C = \{01, 110, 0001, 0110\}$ Verificati daca C este cod.

Calculam $C_1 = \{x \in A^+ \mid \exists\, c \in C \text{ a.i. } cx \in C\} = \{10\}$

Verificam $C \cap C_1 = \emptyset$ adevarat

Calculam $C_2 = \{x \in A^+ \mid (\exists\, c \in C \text{ a.i. } cx \in C_1) \text{ sau } (\exists\, c \in C_1 \text{ a.i. } cx \in C)\} = \emptyset$

Evident $C \cap C_2 = \emptyset$ (si asa va fi pt. orice alta multime C_i, $i \geq 2$)

Asadar C este cod.

Ex2. Fie $C = \{11, 1100, 00011, 0001, 101011\}$ Verificati daca C este cod.

Calculam $C_1 = \{x \in A^+ \mid \exists\, c \in C \text{ a.i. } cx \in C\} = \{00, 1\}$ deci $C \cap C_1 = \emptyset$

Calculam $C_2 = \{x \in A^+ \mid (\exists\, c \in C \text{ a.i. } cx \in C_1) \text{ sau } (\exists\, c \in C_1 \text{ a.i. } cx \in C)\} = \{011, 01, 1, 100, 01011\}$

Deci $C \cap C_2 = \emptyset$

Calculam $C_3 = \{x \in A^+ \mid (\exists\, c \in C \text{ a.i. } cx \in C_2) \text{ sau } (\exists\, c \in C_2 \text{ a.i. } cx \in C)\} = \{1, 100, 01011\}$

Deci $C \cap C_3 = \emptyset$

Calculam $C_4 = \{x \in A^+ \mid (\exists\, c \in C \text{ a.i. } cx \in C_3) \text{ sau } (\exists\, c \in C_3 \text{ a.i. } cx \in C)\} = \{1, 100, 01011\}$

Deci $C_4 = C_3$, (si asa va fi pt. orice alta multime C_i, $i \geq 3$) deci $C \cap C_i = \emptyset$ pt orice $i \geq 3$.

Asadar C este cod.

Ex3. Fie $C = \{b, aab, bba, aabbb, abbb\}$ Verificati daca C este cod.

Calculam $C_1 = \{x \in A^+ \mid \exists\, c \in C \text{ a.i. } cx \in C\} = \{ba, bb\}$, $C \cap C_1 = \emptyset$

Calculam $C_2 = \{x \in A^+ \mid (\exists\, c \in C \text{ a.i. } cx \in C_1) \text{ sau } (\exists\, c \in C_1 \text{ a.i. } cx \in C)\} = \{a, b\}$

Deci $C \cap C_2 \neq \emptyset$ Asadar C un este cod.

Ex4. Fie $C = \{01, 10, 011, 110\}$ Verificati daca C este cod.

Calculam $C_1 = \{x \in A^+ \mid \exists\, c \in C \text{ a.i. } cx \in C\} = \{1\}$ deci $C \cap C_1 = \emptyset$

Calculam $C_2 = \{x \in A^+ \mid (\exists\, c \in C \text{ a.i. } cx \in C_1) \text{ sau } (\exists\, c \in C_1 \text{ a.i. } cx \in C)\} = \{0, 10\}$

deci $C \cap C_2 = \{10\} \neq \emptyset$ Asadar C un este cod.

Observatie Daca veti lua cuvantul 01110 veti observa ca acesta acepta 2 descompuneri in C anume:

01110 = 01.110 = 011.10 ceea ce face ca sa existe texte ale caror codificare peste C sa accepte mai multe decodificari (decodificare ambigua). Asadar C un poate realiza a codificare buna.

Exercitii

Ex1. Fie C = {aa, aab, baa, baab} Verificati daca C este cod.

Ex2. Fie C = {bc, b, cba, cbba} Verificati daca C este cod.

Ex3. Fie C = {001, 00110, 1, 1010} Verificati daca C este cod.

Ex4. Fie C = {1, 011, 01110, 1110, 10011} Verificati daca C este cod.

Ex5. Fie C = {01, 1001, 1011, 111, 1110} Verificati daca C este cod.