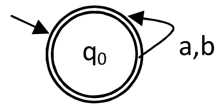
עידן דויטש
u.multinet.co.il

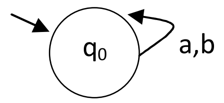
מודלים חישוביים – פתרון תרגיל 5

1. כתוב אוטומט דטרמיניסטי לשפות הבאות מעל הא"ב $\Sigma=\{a,b\}$.

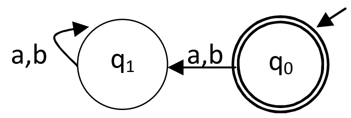
א. Σ^*



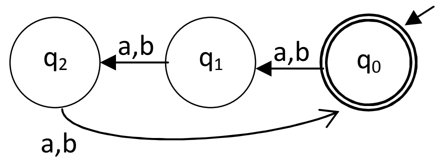
ב. $\emptyset$



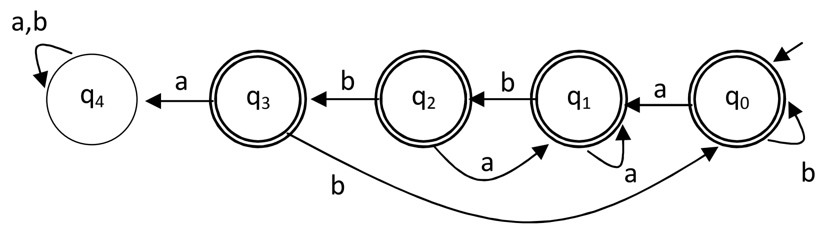
ג. $\{\varepsilon\}$



ד. $\{w \mid |w| \bmod 3 = 0\}$



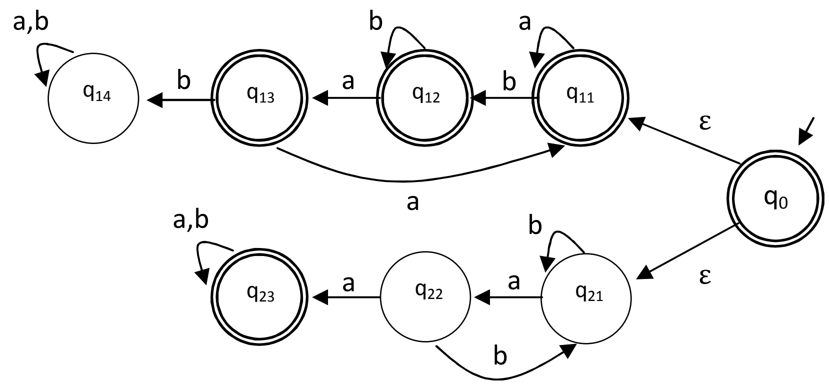
ה. {w | w does not contain the substring 'abba'}



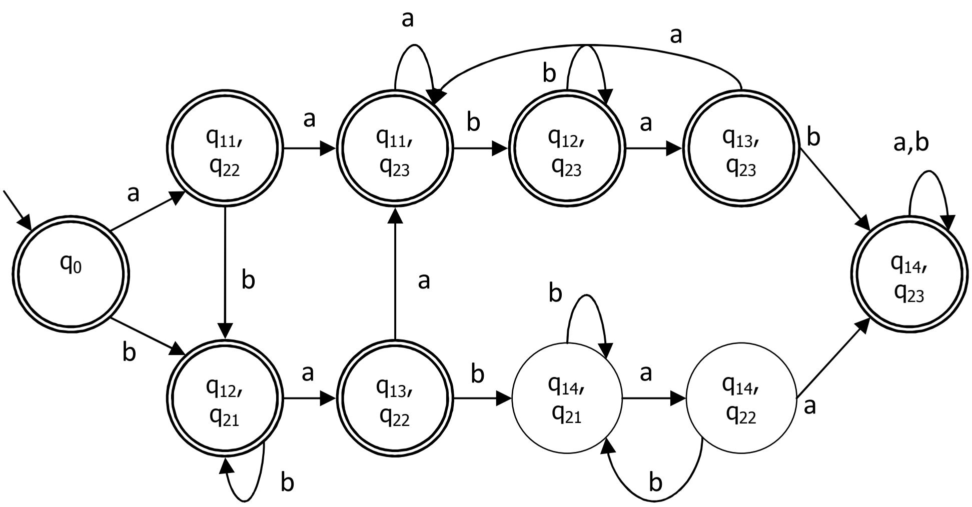
2. כתוב אוטומט דטרמיניסטי ותרגם אותו לאוטומט לא דטרמיניסטי לשפות הבאות מעל הא"ב $\Sigma=\{a,b\}$.

א. {w | w contains the substring 'aa' or does not contain the substring 'bab'}

<u>NFA</u>

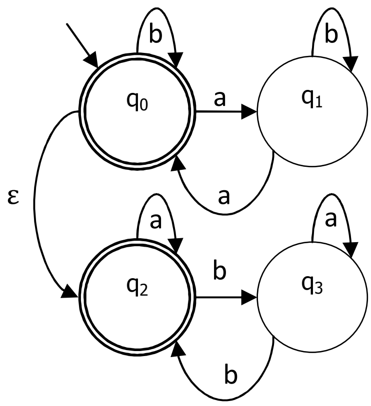


DFA

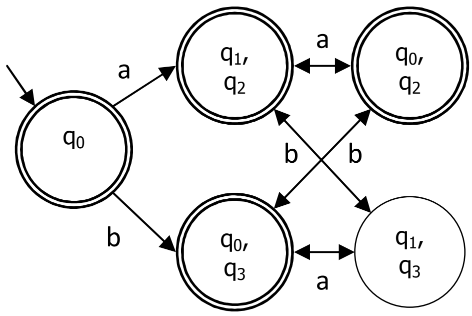


ב. {w∈{a,b}* | There exists a partitioning of w to w=xy s.t. x contains an even number of 'a's and y contains an even number of 'b's}

NFA



DFA



3. כתוב ביטוי רגולרי לכל אחת מהשפות הבאות מעל $\Sigma=\{0,1\}$

א. $\{w \mid |w| \bmod 4 = 0\} = [(0+1)(0+1)(0+1)(0+1)]^*$

ב. $\{w \mid w \text{ contains exactly four '1's}\} = 0^*\cdot1\cdot0^*\cdot1\cdot0^*\cdot1\cdot0^*\cdot1\cdot0^*$

4. עבור כל אחת מהשפות הבאות, הוכח שאם L רגולרית, גם 'L רגולרית:

א. $L'=\{ xy \mid x\notin L \text{ and } y\in L\}$

הוכחה:
נניח שהשפה L רגולרית, ונבנה אוטומט אי דטרמיניסטי 'A שיכריע את 'L. כיוון ש-L רגולרית, קיים אוטומט סופי דטרמיניסטי A המכריע אותה.

נגדיר את 'A:
$A' = (\Sigma, Q \times \{1,2\}, <q_0,1>, F \times \{2\}, \delta')$

כאשר

$$\delta'(<q,n> \in Q \times \{1,2\}, a\epsilon\Sigma \cup \{\varepsilon\}) = \begin{cases} \{<\delta(q,a),1>\} & n=1 \wedge a \neq \varepsilon \\ \{<\delta(q,a),2>\} & n=2 \wedge a \neq \varepsilon \\ \{<q_0,2>\} & n=1 \wedge a=\varepsilon \wedge q \notin F \end{cases}$$

הוכחת נכונות:

$\Leftarrow$ נניח שקיימת מילה $w\epsilon L'$. מכאן, שקיימת חלוקה $w=xy$ כך ש-$x\notin L$ ו-$y\in L$.
נקודת ההתחלה של האלגוריתם שלנו היא במצב ההתחלה של A. כיוון ש-$x\notin L$, לאחר התו האחרון שלו נגיע למצב t שאינו אחד ממצבי הסיום ב-A. כיוון ש-t אינו מצב סיום (ונמצא בעותק הראשון של A), קיים מעבר ε ממנו למצב ההתחלה של העותק השני של A. מצבי הסיום שהגדרנו באוטומט שבנינו הם מצבי הסיום A אך בעותק השני בלבד. כיוון ש-$y\in L$, לאחר בדיקתו ב"עותק" השני של A, נגיע למצב סיום של A ומכאן גם למצב מסיים של 'A.

$\Rightarrow$ נניח שקיימת מילה כך ש-$w\epsilon L(A')$. מכאן, שקיים הרצף:

$$<q_0,1>\longrightarrow\ldots\longrightarrow<q_k,1>\xrightarrow{\epsilon}<q_{0,2}>\longrightarrow\ldots\longrightarrow<q_f,2>$$

נגדיר את רצף האותיות עד המעבר ε להיות x ואת רצף האותיות לאחר מכן להיות y. x מתקבל ע"י רצף אותיות המתחיל ב-<q0,1> ומסתיים ב-$<q_k,1>$.
נשים לב, כי q_k אינו מצב מקבל באוטומט המקורי כיוון שיש ממנו מעבר ε. מעברים אלו לא קיימים באוטומט המקורי היות והוא דטרמיניסטי, ועל כן זהו מעבר שנוסף בעת בניית 'A. הוספנו מעברים אלו רק ממצבים שהוגדרו להיות "לא מקבלים" באוטומט A. מכאן, שהרצת x ב-A תסתיים במצב לא מקבל ולכן $x\notin L$. y מתחיל במצב התחלה של העותק השני של A ומסתיים במצב $<q_k,2>$.
כיוון ש-w שייך לשפה, זהו מצב מקבל. לכן, גם q_k הוא מצב מקבל ומכאן שהרצת y באוטומט A תסתיים במצב מקבל. מכאן, ש-$y\in L$.

ב. $L'=\{x_2x_4\ldots x_{2n} \mid x_2,x_4,\ldots,x_{2n}\in\Sigma, \exists x_1,x_3,x_{2n-1}\in\Sigma \text{ s.t. } x_1x_2\ldots x_{2n-1}x_{2n}\in L\}$

הוכחה:
נניח שהשפה L רגולרית, ונבנה אוטומט אי דטרמיניסטי 'A שיכריע את 'L. כיוון ש-L רגולרית, קיים אוטומט סופי דטרמיניסטי A המכריע אותה.

נגדיר את 'A:
$A' = (\Sigma, Q \times \{1,2\}, <q_0,1>, F \times \{1\}, \delta')$

כאשר

$$\delta'(<q,n>\in Q\ x\ \{1,2\}, a\epsilon\Sigma\cup\{\varepsilon\}) = \begin{cases}\{<\delta(q,a'),2>\mid a'\in\Sigma\} & n=1\wedge a=\varepsilon\\ \{<\delta(q,a)\,,1>\} & n=2\wedge a\neq\varepsilon\end{cases}$$

הוכחת נכונות:

$\Leftarrow$ נניח שקיימת מילה $w\epsilon L'$. מכאן, שב-A קיים רצף מצבים $q_0\to q_1\to\ldots\to q_{2n}$ מקבל. לכן, לפי הבנייה, באוטומט 'A קיים עבור המילה $x_2x_4\ldots x_n$ רצף מצבים $<q_0,1>\xrightarrow{\epsilon}<q_1,2>\to\ldots\xrightarrow{\epsilon}<q_{2n-1},2>\to<q_{2n},1>$.

כיוון ש-q_{2n} הוא מצב מסיים ב-A, $<q_{2n},1>$ הוא מצב מסיים ב-'A. כלומר, האוטומט 'A יקבל את w כנדרש.

$\Rightarrow$ נניח שקיימת מילה כך ש-$w\epsilon L(A')$. מכאן, שקיים הרצף:

$$<q_0,1>\xrightarrow{\epsilon}<q_1,2>\to\ldots\xrightarrow{\epsilon}<q_{2n-1},2>\to<q_{2n},1>$$

לפי הבנייה, עבור כל מעבר ε בין q_i ל-q_{i+1} ברצף זה, קיים מעבר בין מצבים אלו באמצעות אות השייכת לא"ב. נגדיר אות זו להיות y_i ונקבל מילה $w'=y_1x_2\ldots y_{2n-1}x_{2n}$ (כך שx_{2i} מסמל את האות ה-i ב-w). עבור מילה זו קיים ב-A רצף המצבים $q_0\to q_1\to\ldots\to q_{2n}$. כיוון ש-$<q_{2n},1>$ הוא מצב מקבל ב-'A, מצב q_{2n} הוא מצב מקבל ב-A. כלומר, A יקבל את 'w ולכן, w אכן שייך ל-'L.

5. הוכח אם השפות הבאות רגולריות.

א. $L_1=(w \mid w\in\{0,1\}^*, w=w^R\}$ (שפת כל הפולינומים מעל 0,1).

פתרון: זו <u>אינה</u> שפה רגולרית.

הוכחה:
נניח בשלילה שL_1 רגולרית ו-p הוא קבוע הניפוח שלה. נבחר $w=1^p01^p$. w מורכב מהא"ב {0,1} וכן מתקיים $w=w^R$ (w הוא פולינדרום) לכן, $w\epsilon L_1$. כמו כן מתקיים $|w|=2p+1>p$. עבור כל חלוקה $w=xyz$ כך ש-$|xy|\leq p$ מתקיים כי $y=1^k$ כך ש-$1\leq k\leq p$. נבחר $i=0$ ונקבל $w'=1^{p-k}01^p$. זהו אינו פלינדרום ולכן $w'\notin L_1$ בסתירה ללמת הניפוח.

ב. $\Sigma=\{0,1,\#\}, L_2=\{x\#y\#z \mid x,y,z\in\{0,1\}^*, x+y=z\}$ (x,y,z מיוצגים בינארית)

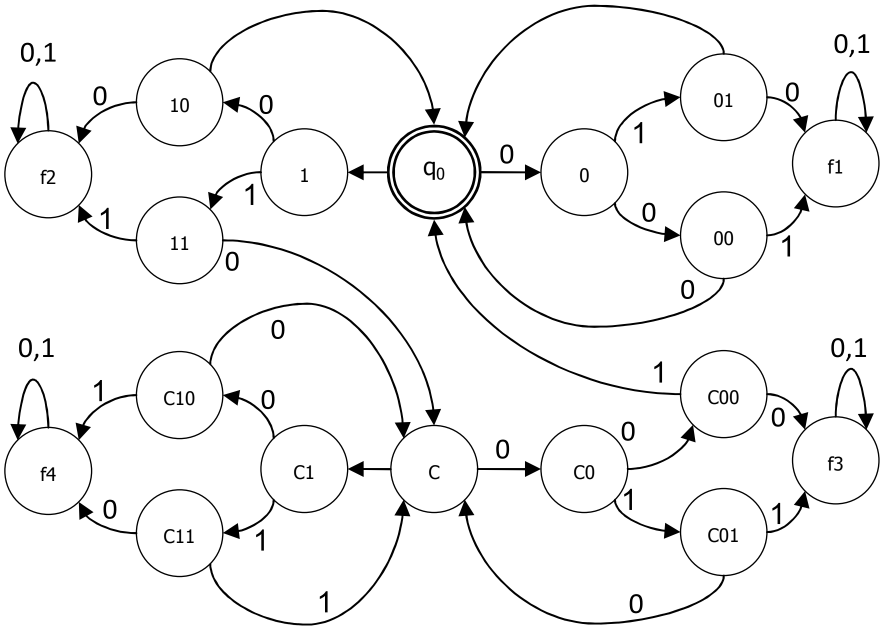
פתרון: זו <u>אינה</u> שפה רגולרית.

הוכחה:
נניח בשלילה שL_2 רגולרית ו-p הוא קבוע הניפוח שלה. נבחר $w=1^p\#0^p\#1^p$. w מורכב מהא"ב {0,1} וכן מתקיים $x=1^p$, $y=0^p$, $z=1^p$ כך ש-$x+y=z$. לכן, $w\epsilon L_2$. כמו כן מתקיים $|w|=3p>p$. עבור כל חלוקה $w=xyz$ כך ש-$|xy|\leq p$ מתקיים כי $y=1^k$ כך ש-$1\leq k\leq p$. נבחר $i=0$ ונקבל $w'=1^{p-k}\#0^p\#1^p$. נשים לב שקיבלנו כי $x+y=1^{p-k}+0^p=1^{p-k}\neq 1^p=z$. לכן, $w'\notin L_2$ בסתירה ללמת הניפוח.

ג. $\Sigma=\{0,1\}, L_3=(x_1y_1z_1\ldots x_ny_nz_n \mid x=x_1x_2\ldots x_n, y=y_1y_2\ldots y_n, z=z_1z_2\ldots z_n, x+y=z\}$

פתרון: זו שפה רגולרית.

הוכחה:
נראה אוטומט דטרמיניסטי המכריע את השפה.

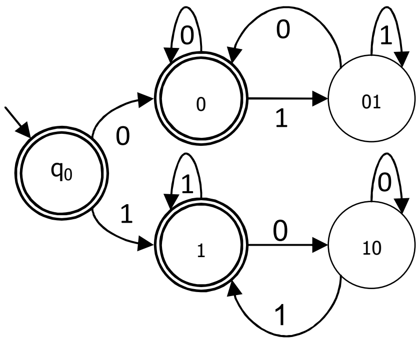


הוכחת נכונות:
האוטומט פועל למעשה כמו רכיב FA. הסדר המיוחד של הקלט מאפשר לבצע את החיבור תו אחר תו. במצב כזה, יש "לזכור" בכל שלב האם קיים קרי או לא. פרט לכך, מתבצעת בדיקה שחיבור כל 2 הביטים מתבצע כראוי (בהתחשב גם בקרי בכל שלב). במידה שאכן בוצע חיבור כראוי, האוטומט חוזר למצב ההתחלתי q_0 ומקבל את המילה כנדרש. במידה ובשלב כלשהו בוצע חיבור שגוי, האוטומט מגיע למצב "ללא מוצא" ולא מקבל את המילה.

ד. Σ={0,1}, L4={w | the number of ‘01’s in w is equal to the number of '10's}

פתרון: זו שפה רגולרית.

הוכחה:
נראה אוטומט דטרמיניסטי המכריע את השפה.



הוכחת נכונות:
נשים לב תחילה, שבכל עת, ההבדל מספר ההופעות של '10' ו-'01' הוא לכל היותר 1. הסיבה לכך היא כיוון שלאחר כל 01, יכול להופיע '1' מספר רב של פעמים (מבלי להוסיף לאף אחד מהצירופים) וברגע שמופיע '0', נוצרת, למעשה, הופעה של 10 (כלומר מספר ההופעות משתווה וההפרש בין הצירופים חוזר להיות 0). המצב מתקיים בצורה דומה גם עבור הצירוף 10. האוטומט שבנינו מבצע בדיוק בדיקה זו. לאחר התו הראשון, האוטומט שומר האם קיים יתרון יחסי לאחד הצירופים (מצבים 10 ו-01). כל עוד קיים יתרון כזה, הוא לא מקבל את המילה. במידה ומספר ההופעות משתווה, האוטומט מקבל את המילה (מצבים 0 ו-1).

6. סבוכיות מקום

א. הוכח שבעיית Clique נמצאת ב-PSPACE ע"י תאור אלגוריתם שפותר את הבעיה במקום פולינומי.

תאור האלגוריתם:

i. נבנה מערך A בגודל |V|. בכל תא במערך יהיה מצביע לקודקוד ייחודי (אין שני מצביעים לאותו הקודקוד) ודגל 0/1 (במונחי מחשב ניתן לתאר כ'ביט').

ii. נאתחל את כל הדגלים במערך להיות 0, פרט לדגל הראשון, אותו נאתחל להיות 1.

iii. נאתחל אינדקס ספירה s ל-0. נעבור על המערך, ונספור באמצעות s בכמה תאים מופיע 1. אם מספר הפעמים גדול מ-k נעבור לשלב v. אחרת, נבצע את שלב iv.

iv. נקדם את הדגלים במערך ב-1. הכוונה היא להסתכל על הדגלים של המערך כמספר בינארי (מתחילים ממצב בו הדגלים הוא 0...01 – כלומר המספר הוא 1 ובכל קידום מוסיפים למונה זה 1. למשל, בקידום הראשון של המונה מ-1 לשלב השני, ישתנה מצב הדגלים מ0...01 אל 0...10 וכך הלאה). לאחר קידום המונה, אם הגענו למצב 0...0 (כלומר היינו במצב 1...1 וקידמנו פעם נוספת את המונה), נחזיר F. אחרת, נבצע שוב את שלב iii.

v. אם s גדול או שווה ל-k, נרוץ על המערך באמצעות 2 אינדקסים: i ו-j. נוודא שכל צומת i במערך שהדגל שלו 1, מחובר לכל צומת אחר j במערך שהדגל שלו הוא גם 1. אם מצאנו 2 צמתים שדגליהם הוא 1 ואין קשת ביניהם נחזור לשלב iv. אחרת נקדם את j ב-1 עד שנגיע לצומת האחרון במערך (ואז נאפס את j ונקדם את i ב-1). אם i הגיע לצומת האחרון במערך, נחזיר T.

הוכחת נכונות:
באמצעות המערך אנו מוודאים שנבדוק כל צרוף אפשרי של הצמתים. לכן, אם עבור כל הצרופים שבדקנו, לא מצאנו צירוף מתאים, בוודאות לא קיים צרוף כזה ולכן נחזיר F כנדרש. אם במהלך הבדיקות, i הגיע לסוף המערך, הרי שכל הצמתים במערך נבדקו ומצאנו שכל הצמתים שהדגל שלהם הוא 1 מחובר לכל צומת אחר. כלומר, זהו אכן קליק. בשלב iii ווידאנו שגודל הקליק הוא לפחות כגודל k ולכן נחזיר T כנדרש.

סבוכיות מקום:

i. גודל המערך הוא כגודל מספר הקודקודים (והוא כולל מצביע ודגל בלבד עבור כל התא) ולכן – O(|V|).

ii. המונה s והאינדקסים i ו-j יכולים למנות לכל היותר כמספר הקודקודים ולכן עבור כל אחד מהם ידרש מקום – O(log|V|)

סה"כ סבוכיות מקום – **O(|V|)**

הראנו שניתן לפתור את הבעיה בסבוכיות מקום ליניארית ולכן זו בעיה בPSpace.

ב. הוכח ש-QSAT היא בעייה NP-קשה.

הוכחה:
נגדיר רדוקצית מיפוי $3SAT<_mQSAT$: נגדיר F, פונקציה המקבלת נוסחה φ.
הפונקציה F תסרוק את הנוסחה, ועבור כל ליטרל, תבדוק האם פגשנו בו קודם (באמצעות רשימה שנתחזק). אם לא פגשנו בליטרל קודם, נוסיף בתחילת הנוסחה כמת "קיים" עם הליטרל שבדקנו, נוסיף אותו לרשימת הליטרלים בהם נתקלנו ונמשיך לליטרל הבא (לדוגמה, אם הגענו אל x ולא פגשנו בו קודם, נוסיף לתחילת הנוסחה '$\exists x.$'). אם פגשנו בליטרל בעבר, נמשיך לליטרל הבא.
בסיום הסריקה של הנוסחה, תחזיר F את הנוסחה המעודכנת.

הוכחת נכונות:
עבור כל נוסחה $(x_1 \vee x_2 \vee x_3) \wedge \ldots \wedge (x_{n-2} \vee x_{n-1} \vee x_n)$, F תחזיר את הנוסחה $\exists x_1 \ldots . \exists x_n . (x_1 \vee x_2 \vee x_3) \wedge \ldots \wedge (x_{n-2} \vee x_{n-1} \vee x_n)$. נוסחה זו תועבר לQSAT. במידה ויש הצבה שמספקת את הנוסחה המקורית, הרי שקיימים $x_1 \ldots x_n$ שמספקים את הנוסחה ולכן QSAT תחזיר T כנדרש. אחרת, לא קיימים הצבות x כאלו ולכן QSAT תחזיר F כנדרש.

סבוכיות:
נניח שאורך הנוסחה הוא n ליטרלים (לאו דווקא שונים). עבור כל ליטרל, אנו בודקים האם נתקלנו בו בעבר באמצעות רשימת צבירה. בדיקה זו מבוצעת בזמן O(n) (סריקה פשוטה של הרשימה). במידה ולא נתקלנו בו בעבר, אנו מוסיף כמת עבורו בתחילת הנוסחה ומוסיפים אותו לרשימת הליטרלים בהם נתקלנו. 2 פעולות אלו מבוצעות ב-O(1). סה"כ כל טיפול בליטרל לוקח O(n).
אנו מבצעים זאת עבור n ליטרלים בנוסחה ולכן סבוכיות זמן ריצה של F הינה $O(n^2)$.

הראנו כי קיימת רדוקציה פולינומית מ- 3SAT אל QSAT. 3SAT היא NP שלמה ולכן בפרט NP קשה. מכאן, שגם QSAT היא בעיה NP קשה.