

סיכום הרצאות למחן - מודלים חישוביים

23 בדצמבר 2025

גיא יער-און.

הסיכום נכתבו לאורך הרצאות *campus* והתוגולים בסמסטר א' שנת 2026 תשפ"ו, لكنו יתכו
שכלו טעויות לאורך כתיבת הסיכום - ככה של אחריותכם.

תוכן עניינים

| | | |
|----|---|-------|
| 3 | יחידה 2 - שפות | 1 |
| 4 | הא"ב | 1.1 |
| 4 | מחירות | 1.2 |
| 4 | פעולות על מחירות | 1.3 |
| 4 | שפות | 1.4 |
| 5 | פעולות על שפות: | 1.4.1 |
| 6 | הכליה ושוון בין שפות | 1.4.2 |
| 7 | פעולות | 1.5 |
| 7 | רישא, סיפא ותתי מילימ | 1.6 |
| 8 | יצוג בעיית הכרעה בשפה | 1.7 |
| 8 | יחידה 3 - אס"ד אוטומט סופי דטרמיניסטי | 2 |
| 10 | רכיבי האוטומט | 2.1 |
| 10 | הגדרה פורמלית של אוטומט סופי דטרמיניסטי | 2.2 |
| 11 | שפת האוטומט | 2.3 |
| 11 | בנייה אוטומטים | 2.4 |
| 15 | בנייה אבסטרקטית | 2.5 |
| 15 | אוטומט משלים | 2.5.1 |
| 15 | שפה רגולרית | 2.5.2 |
| 15 | אוטומט המכפלת | 2.5.3 |
| 16 | אוטומט מכפלת לאיחוד והפרש | 2.5.4 |
| 17 | אוטומט אתחול | 2.5.5 |
| 18 | שפת היזוג | 2.5.6 |
| 19 | יחידה 4: אוטומט סופי לא דטרמיניסטי | 3 |
| 19 | הגדרה פורמלית | 3.1 |
| 20 | שפת אוטומט לא דטרמיניסטי | 3.2 |
| 20 | בנייה אוטומט לא דטרמיניסטי | 3.3 |
| 21 | שקלות | 3.4 |
| 23 | מעברי אפסילון | 3.5 |
| 25 | בנייה אבסטרקטית של אוטומט לא דטרמיניסטי | 3.6 |

| | | | |
|----|--|-------|---|
| 26 | מצב מקבל יחיד | 3.6.1 | |
| 26 | ערוב שפות | 3.6.2 | |
| 26 | הכלאת שפות | 3.6.3 | |
| 28 | יחידה 5: שפות רגולריות | 4 | 4 |
| 28 | סגורות | 4.1 | |
| 28 | סגורות לשדרור | 4.1.1 | |
| 28 | סגורות לסגור קלין | 4.1.2 | |
| 29 | סגורות לרורס | 4.1.3 | |
| 29 | סגורות <i>prefix</i> | 4.1.4 | |
| 29 | ביטויים רגולריים | 4.2 | |
| 31 | המרת אוטומט לביטוי רגולרי | 4.3 | |
| 33 | למה הניפה | 4.4 | |
| 35 | הוכחת אי רגולריות באמצעות סגורות | 4.5 | |
| 36 | יחידה 6: שפות חסרות הקשר | 5 | |
| 36 | דקדוק חסר הקשר | 5.1 | |
| 37 | הגדרה פורמלית: | 5.2 | |
| 38 | גזרות | 5.3 | |
| 38 | שפה הדקדוק | 5.4 | |
| 38 | יצירת דקדוקים חסרי הקשר | 5.5 | |
| 41 | למה הניפה לשפות חסרות הקשר | 5.6 | |
| 43 | סגוריות לשפות חסרות הקשר | 5.7 | |
| 43 | סגורות לאחד | 5.7.1 | |
| 43 | סגורות לשדרור | 5.7.2 | |
| 44 | סגורות לסגור קלין | 5.7.3 | |
| 44 | אי סגורות לחיתוך | 5.7.4 | |
| 45 | יחידה 7: אוטומט מחסנית | 6 | |
| 48 | הגדרה פורמלית | 6.1 | |
| 48 | תהליך החישוב של האוטומט | 6.2 | |
| 48 | שפה האוטומטי: | 6.2.1 | |
| 49 | אוטומט מחסנית שקול לשפה חסרת הקשר | 6.3 | |
| 50 | חיתוך שפה רגולרית ושפה חסרת הקשר | 6.4 | |
| 51 | דקדוקים רגולריים ושפות רגולריות | 6.5 | |
| 52 | יחידה 8: מוכנות טיריניג | 7 | |
| 52 | מהי מוכנות טיריניג? | 7.1 | |
| 54 | הגדרה פורמלית | 7.2 | |
| 55 | קונפיגורציה: | 7.2.1 | |
| 55 | גרירה: | 7.2.2 | |
| 55 | קבלה ודחיה של מחרוזות: | 7.2.3 | |
| 55 | הכרעה של שפה | 7.2.4 | |
| 55 | קבלה של שפה | 7.2.5 | |
| 56 | תיאור מוכנות טיריניג באמצעות טבלת מעברים | 7.3 | |
| 58 | תיאור מוכנות טיריניג באמצעות פסודו קוד | 7.3.1 | |
| 59 | שימוש במוכנות טיריניג לחישוב פונקציות | 7.4 | |
| 60 | יחידה 9: וריאציות של מוכנות טיריניג | 8 | |
| 60 | מודל חישובי - הגדרה פורמלית | 8.1 | |
| 61 | סרט ימינה בלבד | 8.2 | |
| 62 | מודל <i>TS</i> | 8.3 | |
| 63 | מודל <i>OR</i> | 8.4 | |
| 65 | מוכנות טיריניג מרובת סרטים | 8.5 | |
| 66 | סגורות לאחד וחיתוך שפות כריעות/קבילות | 8.6 | |

| | | | |
|----|--|--------|-----|
| 67 | automat עם שתי מחסניות | P2 | 8.7 |
| 68 | סרט דו ממדדי - מודול 2D | 8.8 | |
| 69 | מכונית טיריניג שאינה דטרמיניסטיבית | 8.9 | |
| 70 | הכרעה וקבלה של שפות | 8.9.1 | |
| 71 | סגורות באמצעות אידטרמיניזם | 8.10 | |
| 72 | יחידה 10: התזה של צץ'-טיריניג | 9 | |
| 72 | סגורות | 9.1 | |
| 72 | היחס בין הכרעה לקבלה | 9.2 | |
| 72 | מכונות טיריניג שcolaה לתוכנית מחשב | 9.3 | |
| 74 | דקדוקים כלליים | 9.4 | |
| 75 | ההרככיה של חומוסקי | 9.5 | |
| 76 | התזה של צץ'-טיריניג | 9.6 | |
| 76 | יחידה 11: אי כריעות | 10 | |
| 76 | איימות תוכנה | 10.1 | |
| 77 | ATM קבילה | 10.2 | |
| 77 | ATM לא כריעה | 10.3 | |
| 78 | שפה שאינה קבילה | 10.4 | |
| 78 | בעיית העצירה | 10.5 | |
| 79 | שפות לא פתרות | 10.6 | |
| 79 | E השפה | 10.6.1 | |
| 80 | <u>EQ</u> השפה | 10.6.2 | |
| 80 | <u>EQ</u> השפה | 10.6.3 | |
| 81 | פונקציות לא חשיבות | 10.7 | |
| 81 | רדוקציות והוכחות ברדוקציות | 10.8 | |
| 83 | רדוקציה משפה כריעה | 10.8.1 | |
| 83 | NOT – REG השפה | 10.8.2 | |
| 85 | סיכון - השפות הלא כrüות/קבילות | 10.9 | |
| 85 | בעיית הנחש | 10.10 | |
| 86 | התוכנית של הילברט | 10.11 | |

1 יחידה 2 - שפות

ישן כמה סוגים בעיות שעשוית לעניין אותנו.

ישן בעיות חישוב - למשל, מה השורש הריבועי של 180? או מה התרגום לאנגלית של אני אוהב שוקולד?".

ישן בעיות אופטימיזציה - למשל, מה המסלול הקצר ביותר מכאן לתל אביב?", או בעיות תכנון דינמי למשל

ישן בעיות הכרעה - למשל, האם מס' הוא ראשוני. האם הגраф קשייר. בעיות אלו הן בעיות שהתשובה אליהן היא כן או לא. זו בעיות חישוב ספציפיות יותר, כיון שיש לה שתי תשובות בלבד. בקורס נתמקד בעיות אלו. מדוע? הן פשوطות להגדירה וניתנות, השאלה שטעני אותן בהמשך - האם יש בעיות שלא ניתן לפתורן באמצעות מחשב. סיבה נוספת להתusalem, היא שככל בעיות חישוב ניתן להמיר לביקורת הכרעה.

כיצד נמיר בעיות הכרעה לביקורת חישוב? נניח ואני יודעים להכריע האם $y = f(x)$. כתעת עבור על כל y בקבוצת הקלטים האפשריים ונבדוק האם $y = f(x)$. כאשר בכל שלב נקבל תשובה של כן או לא, כשנקבל כן פתרנו את בעיית החישוב.

1.1 הא"ב

תמיד הקלט לבעיות הכרעה יוצג ע"י סדרה של תווים. התווים שמהם נוצרת סדרה קורא האלפבית. או פשוט א"ב. נסמן באות Σ . בקורס זה הא"ב יהיה תמיד סופי. את אותיות הקלט נסמן לעיתים באותיות יוניות - σ, τ .
למשל - עבור הבעיה האם x ראשוני?", הא"ב יהיה $\{0, 1, 2, \dots, 9\} = \Sigma$. עבור הבעיה, האם הגרף קשור?" נזדקק לא"ב $\{1, \dots, 9, (\cdot), \{, \}, ", , x, y, u, v, w\}$ קבוצת קודקודים $\{(1, 2), (2, 3)\}$ לדוגמה וקשתות $\{1, 2, 3\}$

1.2 מחרוזות

נשים לב - לא כל סדרה מעל א"ב נתון מייצגת קלט תקין. סדרה כמו "12378" או "ססססססססססס" תקרא מחרוזת/מילה/סדרה. מילים יסומנו באותיות קטנות כדוגמך y, x, u, v, w .
 $|w|$ - אורך של מילה. מס' התווים ב-w. אורך המילה יהיה סופי, ובקורס הזה כל המילים יהיו סופיות.

המילה הריקה - מסומנת באות ϵ . מותקים $0 = |\epsilon|$. נשים לב כי $\emptyset \neq \epsilon$, המילה הריקה היא מילה בעוד הקבוצה הריקה היא קבוצה. כמו כן $\emptyset \neq \{\epsilon\}$ שכן הקבוצה משמאלה אינה ריקה, יש שם את המילה הריקה.

סימונו: עבור מילה w ואות σ נסמן $\#$ את מס' הפעמים שהאות σ מופיעה ב-w.

קובוצת כל המחרוזות מעל Σ - תסומן Σ^* . למשל, $ab \in \{a, b\}^*$ בעוד $ac \notin \{a, b\}^*$ וכן $\epsilon \in \Sigma^*$.
קובוצת כל המחרוזות הלא ריקות מעל Σ תסומן ותוגדר:

$$\Sigma^+ = \Sigma^*/\{\epsilon\}$$

הטענה לא נכונה עבור שפות - לא מותקים $L^+ = L^*/\epsilon$ אם $\epsilon \in L$.

1.3 פועלות על מחרוזות

a. **שרשור - יסומן** ○. השרשור היא פעולה פשוטה. למשל - $ab \circ bba = abba$, דוגמה נוספת: $ab = ab \circ \epsilon$. כלומר. קלומר - השרשור של מילה עם המילה הריקה נותן את המילה עצמה.

b. **חזקה - w^n .** ההגדירה היא -

$$w^n = \underbrace{w \circ w \circ \dots \circ w}_{n \text{ times}}$$

חזקה היא שרשור של המילה עם עצמה n פעמיים. למשל, $(ab)^3 = ababab$. מותקים כי $\epsilon^0 = w^0$. וכן $\epsilon = w^0$.

1.4 שפות

כעת נדבר על קבוצה של מחרוזות. שפה היא קבוצה של מחרוזות ונסמנה L . למשל, עבור $\Sigma = \{a, b, \dots, z\}$ אם נגדיר $L =$ כל המילים באנגלית במילון אוקספורד, L הינה שפה. מותקים כי $L \subseteq \Sigma^*$

ישן שפות אינסופיות. למשל, עבור $\Sigma = \{a, b\}$ אם נגיד $L = \{w \in \Sigma : |w| \% 2 = 0\}$ שהוא שפת כל המילים שאורך המילה הינו זוגי, היא שפה אינסופית.

1.4.1 פעולות על שפות:

- א. ראשית, כיוון ששפה היא קבוצה איזה כל הפעולות שנitin לבצע על קבוצות כגון חיתוך, איחוד ומשלים ניתן לבצע על שפה. המושגים יהיה על Σ^* .
- ב. שרשור - יהיו שפות L_1, L_2 . נגיד שרשור עליה:

$$L_1 \circ L_2 = \{u \circ w | u \in L_1, w \in L_2\}$$

כלומר כל המילים שמתabolicות ע"י שרשור מילה מ L_1 עם מילה מ L_2 .
לדוגמה:

$$\{ab, aa\} \circ \{b, ab\} = \{abb, abab, aab, aaab\}$$

ג. חזקה של שפה - שרטור השפה עם עצמה, n פעמים. ופורמלית:

$$L^n = \underbrace{L \circ L \circ \dots \circ L}_{n \text{ times}}$$

לדוגמה:

$$\{a, ab\}^2 = \{aa, aab, aba, abab\}$$

$$L^0 = \{\epsilon\}$$

ד. סגור כללי: האיחוד האינסופי של כל החזקות הסופיות של השפה. כלומר, ב L^* יש את כל המילים כך כשלוקחים מס' כלשהו, סופי, של מילים מ L וכותבים אותן אחת אחרי השנייה.
ופורמלית -

$$L^* = \bigcup_{n=0}^{\infty} L^n$$

לדוגמה: עבור $\Sigma = \{a, ab\}^*$ נקבל כי $aaabaa, ababaaaa$ וכו'. תמיד $\epsilon \in L^*$

חשיבותם לב: אם נסתכל על $\{a, b\} = \Sigma$, ונגדיר $L_1 = \{a\}$ ו $L_2 = \{b\}$, ונסתכל על $L_1 \cup L_2 = \{a, b\}$ זו שפת האיחוד, אם נסתכל על $(L_1 \cup L_2)^*$ זו השפה של כל המילים שנitin להרכיב מהאותיות a, b . לעומת זאת, אם נסתכל על $L_1^* \cup L_2^*$, כאן נקבל כי זו שפת כל המילים שמכילות רצף מסוים של a ואז רצף מסוים של b . זו כמובן לא אותה שפה.

כעת ניתן לשים לב, אם נסתכל על כל אות כמילה באורך אחד, אז Σ^* היא הסגור קlien של השפה Σ .
וכמו קודם, אם נרצה ללא המילה ה裏קה, נגדיר:

$$L^+ = \bigcup_{n=1}^{\infty} L^n$$

$$\text{נשים לב כי } (\emptyset^*)^* = \{\epsilon\}.$$

ה. נגדיר עבור א"ב Σ את Σ^n כקבוצת כל המילים מעל א"ב Σ שאורכן n .
דוגמה: עבור א"ב $\Sigma = \{1, 0\}$ נקבל: $\Sigma^0 = \{\epsilon\}, \Sigma^1 = \{1, 0\}, \Sigma^2 = \{10, 01, 11, 00\}$

$$\text{נשים לב } - \emptyset = \emptyset$$

טענה: עבור $x, y \in \Sigma^*$ מתקיים $(xy)^r = y^r x^r$
טענה: תהי שפה L . אז $(L^r)^* = (L^*)^r$.

1.4.2 הכללה ושוויון בין שפות

פעולה על שפות - יוצרת שפות חדשות, נרצה לבדוק הכללה / שוויון בין שפות. למשל, האם מתקיים תמייד

$$L_1(L_2 \cup L_3) = (L_1 L_2) \cup (L_1 L_3)$$

התשובה היא שכן. כיצד נוכיח דבר שכזה?
ההוכחה תהיה להוכיחה של שוויון בין קבוצות ע"י הכללה דו כיוונית. הפרכה? גם כמו בתורת הקבוצות, מצא מלא ששייכת לצד אחד של המשווה ולא בשפה החסנית. סתיירה. בום.
נוכיח את הטענה למטה כאן: יהי $w \in L_1(L_2 \cup L_3)$, לפि הגדרת שרשור,

$$\exists y, z \in \Sigma^* : w = yz, \text{and} : (y \in L_1) \wedge (z \in (L_2 \cup L_3))$$

מהגדרת האיחוד,

$$\exists y, z \in \Sigma^* : w = yz, \text{and} : (y \in L_1) \wedge (z \in L_2 \vee z \in L_3))$$

$$\exists y, z \in \Sigma^* : w = yz, \text{and} : (y \in L_1 \wedge z \in L_2) \vee (y \in L_1 \vee z \in L_3))$$

$$\exists y, z \in \Sigma^* : w = yz, \text{and} : (yz \in L_1 L_2) \vee (yz \in L_1 L_3))$$

$$\exists y, z \in \Sigma^* : w = yz, \text{and} : (w \in L_1L_2) \vee (w \in L_1L_3))$$

$$\exists y, z \in \Sigma^* : w = yz, \text{and} : (w \in L_1L_2 \cup L_1L_3))$$

$$w \in (L_1L_2) \cup (L_1L_3)$$

וההוכחה הייתה נחמדה יפה. זהו, מזכיר קצר שבוע שני בבדידה. הכוון השני זהה רעיוןית ואפיו אפשר לבצע את הגרירות עם אמ"מ.

דוגמא להפרכה, האם מתקיים: $L_1 = \Sigma = \{a, b\}$ וניקח $(L_1 \cdot L_2)^+ \subseteq L_1^+ \cdot L_2^+$? לא. נקח למשול כי $abab \in (L_1 \cdot L_2)^+$ אך $abab \notin L_1^+ \cdot L_2^+$ כי בשפה זו $abab$ אינו מתקבל מילאה של a -ים וachs"ב-ים כלומר פורמלית $L_1^+ \cdot L_2^+ = \{a...ab....b\}$ ולא ניתן שהמילאה תהיה בו.

1.5 פועלות reverse

נדיר פורמלית את פועלות *reverse* על מילה w :

$$w = \sigma_1 \dots \sigma_n : \left\{ \begin{array}{ll} \epsilon & n = 0 \\ \sigma_n \sigma_{n-1} \dots \sigma_1 & n \in \mathbb{N} \end{array} \right\}$$

הפעולה הופכת את סדר המילאה. למשל עבור $w = aabb$ נקבל $w^r = bbba$. נשים לב כי $\epsilon^r = \epsilon$.

שפת *reverse* תהי L^r , היא שפה שמקילה את *reverse* של כל המילים בשפה L . ופורמלית,

$$L^r = \{w | w^r \in L\}$$

$$\text{נשים לב כי } (L^r)^+ = (L^+)^r$$

1.6 רישא, סיפא ותתי מילים

תהי Σ^* . נגידיר:
א. שפת הרישות של L - שפת התחליות של המילים ע"י

$$\text{prefix}(L) = \{x | x \in \Sigma^*, \exists y \in \Sigma^* \wedge xy \in L\}$$

ב. שפת הסיפיות של L - שפת הסופיות של המילים ע"י

$$\text{suffix}(L) = \{y | y \in \Sigma^*, \exists x \in \Sigma^* \wedge xy \in L\}$$

ג. שפת תת-המילים של L - שפת כל תת-המילים ע"י

$$sub(L) = \{y | y \in \Sigma^*, \exists x, z \in \Sigma^* \wedge xyz \in L\}$$

לדוגמה: עבור $L = \{abc\}$ ו- $\Sigma = \{a, b, c\}$ נקבל

$$prefix(L) = \{\epsilon, a, ab, abc\}, suffix(L) = \{\epsilon, c, bc, abc\}$$

$$sub(L) = prefix(L) \cup suffix(L) \cup \{b\}$$

1.7 ייצוג בעיית הכרעה בשפה

נחוור להתחלה. נראה כעת, כיצד נוכל להעזר בהגדרות שאספנו על מנת ליעזג בעיית הכרעה. נסתכל על בעיית ההכרעה: האם מס' w הוא ראשוני? גדריר שפה -

$$Prime = \{w | w \text{ is prime}\}$$

כעת, בעיית ההכרעה האם מס' ראשוני תומר לבעה הבאה: האם $w \in prime$ - כל בעיית הכרעה תוכל להיות מומרת לשפה. ואז תמיד הבעיה תומר לשאלת בינהית: האם הקלט בשפה או שלא. **מבחן נגיד למסקנה נססת**, **כל בעיה היא שפה**.

נעיר כי ניתן להתייחס לאלה בעיה מעל א"ב שונה.

1. **ייצוג עשרוני** - מעל א"ב $\{0, \dots, 9\} = \Sigma$ ונקבל

2. **ייצוג בינארי** - מעל א"ב $\{0, 1\} = \Sigma$ ונקבל

3. **ייצוג אונרי** - מעל א"ב $\{1\} = \Sigma$ ונקבל **כאשר מס טبוי n מיווג ע"י רצף של n אחדות.**

2 יחידה 3 - אס"ד (אוטומט סופי דטרמיניסטי)

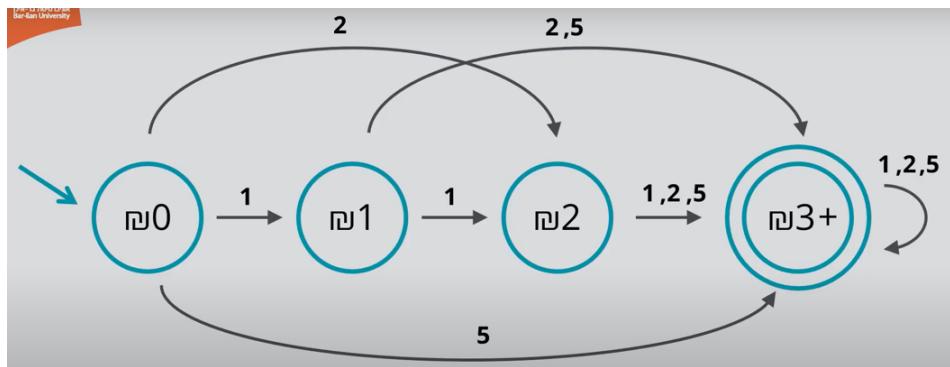
ביחידה זו נדון במושג המחשב. קודם לכן, דנו במושג בעיה=שפה. כעת נדבר על מודול חישובי כלשהו, לא מחשב ספציפי. היכולות החישוביות של האוטומט הסופי, די מוגבלות. אז מדוע נלמדו? יש לו שימושים מרכזיים בעולם האמיתי, והוא ישמש אותנו בחימום והכנה למודלים המורכבים בהמשך.

נתבונן בדוגמא: אנחנו משתמשים במכונית שתיהה, גם היא הרי, סוג של מחשב. לפניינו פחות קוקה-קולה שעולה 3 שקלים (דמיוני, אבל ניחא). אנחנו מכנים מטבעות למכוונה, ונבהיר - זו מכונה שלא מחזירה עודף. איך היא יודעת שהכנסנו מספיק? נתבונן ב"תרשים" מטה, המצביע ההתחלתי, הוא 0 שקלים, תמיד נהייה בו וונתחל ממנה, ולכן נסמננו בחץ - שנדע מהיקן להתחיל. המצביע הסופי, יקרה +. 3. לשם אנחנו שואפים להגיע, וכך ביעיגול פעמיים את מצב המטרה שלנו. כמו כן, נסיף לולאה פנימית, אם נגיע למצביע המטרה ונקבל מטבעות, נרצה להשאר בו. התרשימים מטה דיבורים, ככה פועלת מכונת

המשמעות. והתרשים מטה - הוא בדיק אוטומט סופי, עם מס' מצבים, סופי. זהו מכונה ש碼ריעה שפה. מהי השפה? אוסף כל המחרוזות, שסכום ערכן, גודל-שווה מ-3, ופורמלית:

$$\Sigma = \{1, 2, 5\}$$

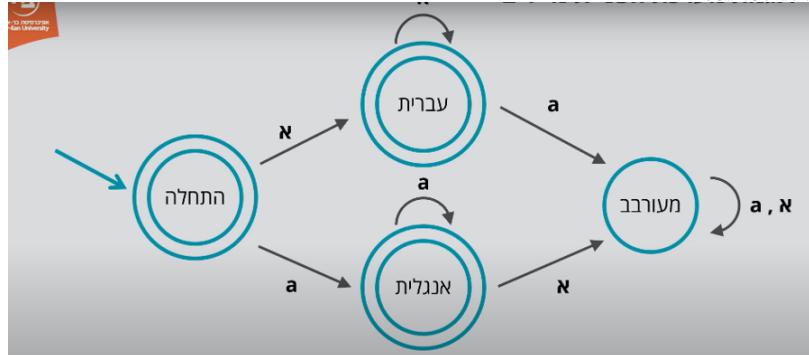
$$L = \{\sigma_1\sigma_2\dots\sigma_n \mid \sum_{i=1}^n \sigma_i \geq 3\}$$



דוגמה נוספת: בעיית הכרעת השפות
icut נרצה להכריע את השפה הבאה מעל א"ב $\Sigma = \{\text{א}, a\}$:

$$L = \{a\}^* \cup \{\text{א}\}^*$$

כלומר, השפה שמורכבת מאותיות א' בלבד, או a בלבד. קלומר - ללא ערבוב של השפות. כיצד ראה האוטומט?



כאשר נשים לב, כי כל מצב טוב מבחינתיו פרט למצב אחד - מעורבב. ולכן מצביו ה"הצלחה" שלנו, יהיו כולם, פרט למעורבב.

2.1 רכיבי האוטומט

כעת, נגדיר פורמלית ובאופן כללי את רכיבי האוטומט:

- א. מצבי האוטומט: אלו האפשרויות בהן יכול האוטומט להיות, אלו העיגולים שהקפנו בדוגמאות מעלה. פורמלית, נסמן מצב באות q_i כאשר הוא המצב ה- i -י באוטומט. ישנו כמה מצבים מוחדים:
 1. מצב התחלה, q_0 . הוא המצב בו האוטומט יחל והוא יחיד. כמובן, לא יתכן אוטומט עם שני מצבים התחלה (יהיה דו משמעי, מהיקן נתקיל?). נסיף חץ למספר התחלתי, שיתאר שמננו התחיל האוטומטי.
 2. מצב מקבל - מצב טוב, מצב זה יסמן בשני עיגולים והוא יתאר מצב שהקלט תקין וטוב מבחינתנו. אם בסוף קריאת הקלט סימנו במצב מקבל, אז האוטומט קיבל את המילה והיא חלה מהשפה. כמובן - התשובה בעברית ההכרעה היא כן. יכולם להיות מ'סיבות מסוימות' מצבים מקבלים.
 - ב. מעברים בין המצביעים באוטומט - מסומנים ע"י חצים, הם אומרים לנו לאיזה מצב עברו בקריאהאות קלט כלשהו. נשים לב כי מכל מצב, צריך לצאת חץ יחיד עברו כל אות קלט שהיא, שכן שוב, לא נרצה למצאו עצמוני במצב דו משמעי. אם נרצה להעיר שני חצים עברו אותן אות קלט אחת, יתכן שנוצרך להוסיף עוד מצב - כי זה יהיה מורכב ולא אפשרי בלי.



תהליך הריצת האוטומט על מחרוזת קלט: מתחילה במצב התחלתי, הקלט נקרא רק פעם אחת, אותן אותות, משמאלו לימין. בכל צעד, קריאה של אות מעבירה ממצב למצב לפי החיצים. החישוב מסתיים כשהקלט נגמר. המחרוזות מתקבעת אם' החישוב הסתיים במצב מקבל.

- ♡ - נשים לב: תנאי הכרחי ומופיע בשבייל שהamilha הריקה תתקבל ע"י האוטומט הוא שהמצב q_0 יהיה מצב מקבל.
- ♡ - לא בכל אוטומט סופי יש מצב מקבל.
- ♡ - אם לאוטומט יש שני מצבים מקבלים, אין זה הכרח שבבחירה יתקבלו שתי מיללים על ידי. למשל, יתכן מצב מקבל שאין שום מסלול אליו ודרך להגיע אליו מהמצב התחלתי, הוא כאילו מנוטק מהאוטומט. זה אונס מיותר, אך יכול להיות.
- ♡ - אם השפה של אוטומט היא סופית אז בהכרח יש מצב באוטומט שמננו אין מסלול למצב מקבל. אחרת, תמיד נגיע במצב מקבל והשפה תהפוך לאינסופית.
- ♡ - מספר המצביעים באוטומט חייב להיות לפחות מספר האותיות במילה הקרצה ביותר בשפת האוטומט.

2.2 הגדרה פורמלית של אוטומט סופי דטרמיניסטי

ובכן, כפי שאלעדי עטיה אמר, צירורים זה יפה, אבל לא שווה נקודות. נגדיר פורמלית את מושג האוטומט:

$$\begin{aligned}
 \text{אוטומט סופי דטרמיניסטי} &= \text{האומנות } A = (Q, \Sigma, \delta, q_0, F) \text{ כאשר} \\
 Q &= \text{קובוצת סופית של המצביעים שלנו} \{q_0, \dots, q_n\} \\
 \Sigma &= \text{הא"ב שלנו} \\
 \delta &= \text{פונקציית המעברים:}
 \end{aligned}$$

$$\delta : Q \times \Sigma \rightarrow Q$$

$q_0 = \text{ מצב תחيلي}$
 $F = \text{ קבוצת המ מצבים המתקבלים } \{q_i, q_j, \dots, q_k\}.$
 בעת, בהינתן אוטומט שיתואר בצורה מתמטית, נוכל לעבור לאוטומט בצורה ציור. שניהם כמפורט
 אוטומטיים. אחד פורמלי יותר ואחד פחות.

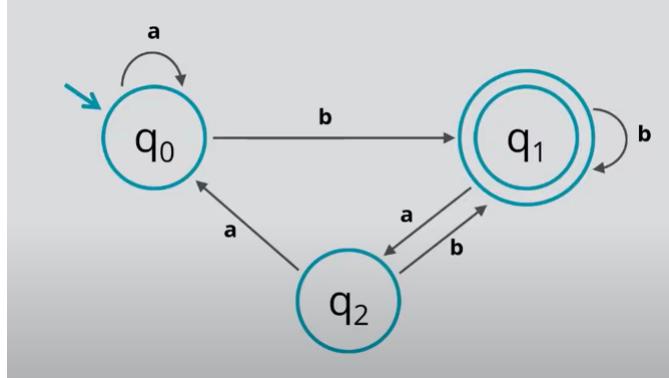
מה זה דטרמיניסטי? בכל שלב, ובכל מצב, ישנה אפשרות אחת בלבד עבור אות קלט لأن להתקדם.
 לא צריכה להיות בחרה של השימוש או של המחשב עצמו באיזה כיוון כדי לו להתקדם.

2.3 שפת האוטומט

באופן אינטואטיבי, זהו אוסף כל המילים שהאוטומט מקבל. נרצה להגיד מתמטית.
 לשם כך - נרჩיב את ההגדירה של פונקציית δ :

$$\delta^* : Q \times \Sigma^* \rightarrow Q$$

כלומר עת נשתכל על Σ^* , פונקציה שתקבע מחרוזת, לאתו בודד. כמובן, בהינתן לדוגמה לאוטומט הבא:



אם נרצה לחשב $\delta(q_0, ba) = q_2$. כמובן, הפונקציה מחשבת מה יקרה כאשר אני במצב מסוים, מפעיל מילט קלט מסוימת, והפונקציה מחירה לי לאיזה מצב אגיע לאחר הפעלת המילה. קל לראות
 כאן שאכן נגע ל q_2 . כמובן שמשים לב שערכי δ נקבעים באופן מלא בהתאם לערכי δ .

כעת, קל יהיה להגיד את שפת האוטומט:
 יהיו A אוטומט סופי דטרמיניסטי, השפה של A הינה הקבוצה:

$$L(A) = \{w | \delta^*(q_0, w) \in F\}$$

כלומר, כל המילים שכאשר נתחיל ב q_0 ונגיע למצב מקבל.

הערה: "יתכן כי נתקל בשני אוטומטים עם אותה השפה". כמובן, יהיו A, B אוטומטיים, "יתכן כי
 $L(A) = L(B)$ אך $A \neq B$

2.4 בניית אוטומטיים

בתרגילים בנושא נקבל שפה, ונctrיך לבנות לה אוטומט. נשים לב לדברים הבאים:

- א. חשוב להבין את השפה, לתרגם את המתמטיקה להבנה. נודא שברורו איזה מילים בשפה ואיזה לא.
- ב. בונה וצייר את האוטומט עצמו. לצורך זאת - זה סוג של מחשב, ולכן יתכונו כמה פתרונות לאותו התרגיל.
- ג. נבצע בדיקות הריצה של מילים בשפה ומילים שאינם בשפה. נודא שאכן האוטומט עובד.

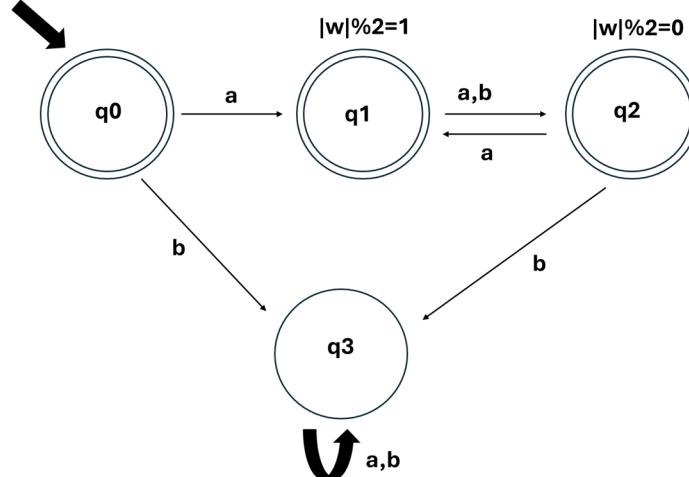
אוטומט שלד: נניח ונרצה לבנות אוטומט לשפה מעל $\{a, b\} = \Sigma = \{w | w = aa\sigma_1\sigma_2\dots\dots\}$. כלומר, כל המילים שנפתחות ברצף "aa". מהי המילה שבודאי תופיע שם? aa . אנחנו רוצה לבנות אוטומט שלד, שיברך את המילה aa , ואנו נרჩיב ונתאים למצבים השונים ולמקרי הקצה. למשל - **קיים מצב מלבדת:** כל מילה שלא פתחה בא", רצחה שתגע לשם, שכן לא מעניין אותנו מה המשך הקלט, לא קיימת את הכלל, את לא חלק מהשפה שלו.

אוטומטים לשפות הטריוויאליות:

- השפה הריקה - אוטומט לשפה שלא מכילה אף מילה, כלומר \emptyset . בצד יראה אוטומט זה? מצב יחיד, q_0 , שאינו מקבל, עם עצמי $loop$ על a .
- המילה הריקה - אוטומט עבור השפה $\{\}$. בצד יראה אוטומט זה? נזכיר q_0 מקבל, כאשר קיבל a, b באותיות קלט, נ עבר למצב מלבדת. ושם יהיה עצמי עבור אותיות הקלט. לעומת זאת, ברגע שנקבל $a^m b$, תס היספור - הולכנו למצב מלבדת.
- השפה * - כל המילים יתקבלו. בצד יראה אוטומט זה? בדוק כמו ב-1, רק עם מצב מקבל. כמובן, כאן אנחנו כן רצחה לקבל כל מילה שקיים בשפה.

דוגמא של אוטומט:

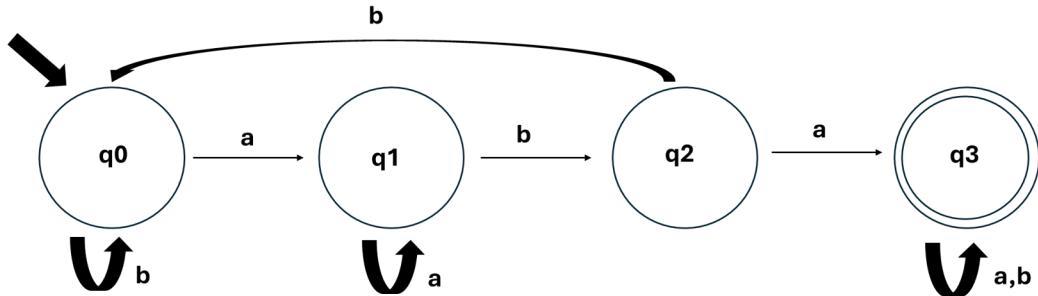
יהי $\Sigma = \{a, b\} = \Sigma$ שלנו. נגידר את L להיות השפה של כל המילים מעל Σ בהן האות b אינה במקומות הזוגי. נראה כי האוטומט שלנו יהיה:



עלים רמה, ובכן נשים לב שנרצה להגדיר מצבים שונים עבור אורך מילה זוגי ואי זוגי, וכן רצחה להפריד את האות הראשונה. אם החילנו באות b , איי אנחנו כבר רוצחים להקלע למצב מלבדת. נראה כי q_3 הוא ממש מצב מלבדת.

דוגמא 2. יהי $\Sigma = \{a, b\}$. רצחה לבנות אוטומט A שיקבל את כל המילים שמכילות את תת המחרוזת "aba" ופורמלית -

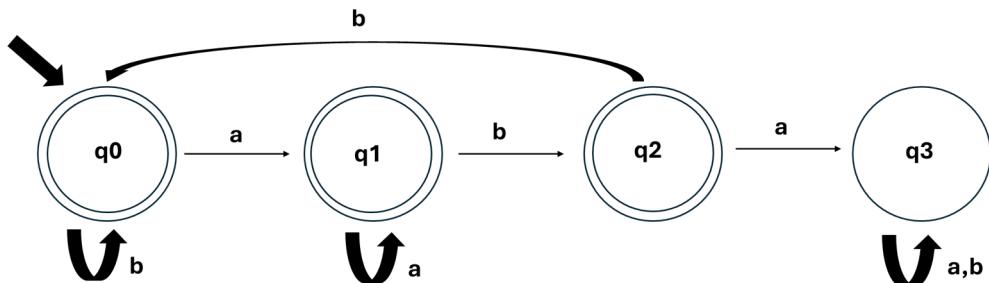
$$L = \{w = u_1 abau_2 : u_1, u_2 \in \Sigma^*\}$$



דוגמה 3. יהיו $\Sigma = \{a, b\}$. נרצה לבנות אוטומט A שיקבל את כל המילים שלא מכילות את תת המחרוזת "aba". פורמלית -

$$L = \{w = u_1u_2u_3 : u_1, u_2, u_3 \in \Sigma^* \wedge u_1, u_2, u_3 \neq "aba"\}$$

למעשה, מדובר במשלים של (דוגמה 2). לשפה זו יש שיטה מיוחדת. ניתן להתחיל כרגיל ולנסות לחשב על לוגיקה אך נראה שזה יהיה קשה. נשים לב כי נוכל להעתיק את האוטומט מדוגמה 2, ולמעשה - כל מילה שהשפה הקודמת לא קיבלה, השפה שלנו כן תקבל. ולהפוך, מילים שהשפה הקודמת קיבלה, השפה שלנו לא תקבל! ואיך זה יבוא לידי ביטוי? המצבים המתקבלים היפכו ללא מקבלים, והמצבים הלא מקבלים היפכו למקבלים:

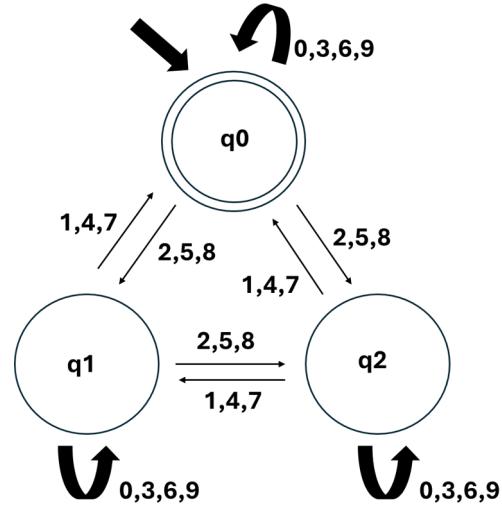


בשיטתה זו נוכל להשתמש **תמיד** כאשר נרצה לבנות אוטומט לשפה המשילימה, בהינתן שאנחנו ידעים את האוטומט לשפה המקורית. וכך, כמו בהסתברות למשל, נוכל לבנות אוטומט לשפה וואז להפעיל עליה שלילה", רק שכן במקום בצע $p - 1$ נשנה את המצבים המתקבלים.

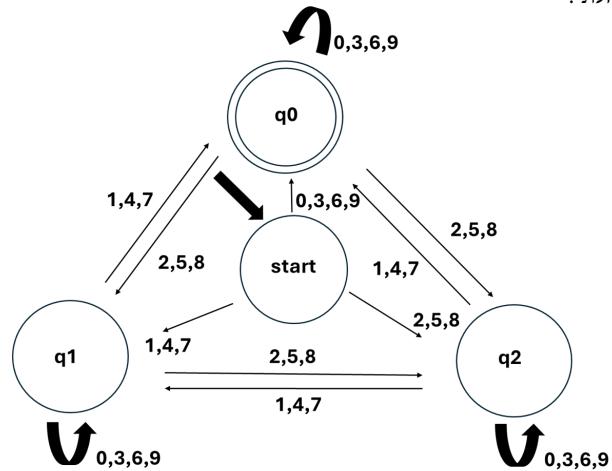
דוגמה 4. נרצה לבנות אוטומט A שיקבל את השפה הבאה מעל אל"ב $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$$L = \{w = \sigma_1\sigma_2\dots\sigma_n : w \% 3 = 0\}$$

נראה כי מתמטית, שקול הדבר לבדוק האם $(\sum_{i=1}^n \sigma_i) \% 3 = 0$. כמובן, מס' מתחלה בשילוש אם סכום ספרותיו מתחולק ב.3. مكانו, נגדיר את המצבים הבאים: לכל $2 \leq i \leq n$ נגדיר q_i להיות המצב באשר השאריות של סכום ספרות המספר עד כה הוא בדיק. i . مكان נקבע אוטומט הבא -

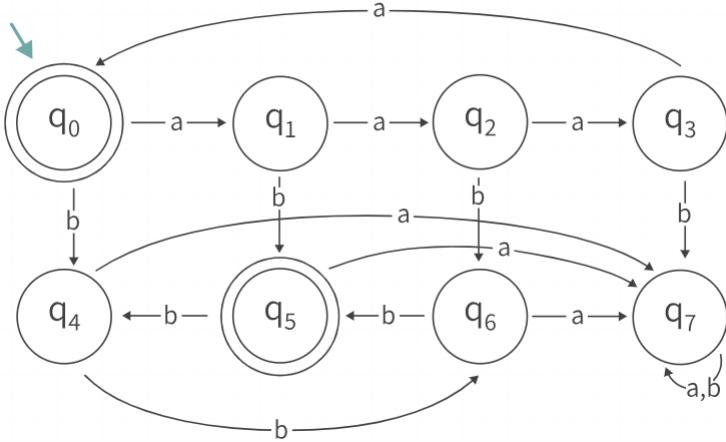


נשים לב, האם זה אוטומט מספק עבורנו? כמובן. מה עם המילה הריקה? מדוע היא במצב מקבל? הרי מילה ריקה איננה מתחלה בשליש. ולכן, נדרש להוסיף מצב נוסף להוות מצב נוסף להיות המצב ההתחלתי:



עכשו האוטומט שלנו, מוכן לקבל את הקלט.

דוגמה 5. יהיו $\Sigma = \{a, b\}$. בנה אוטומט שמקבל את $L = \{a^i b^j : i \bmod 4 = j \bmod 3\}$.



מדובר בדוגמה המורכבת ביותר עד כה ולכן נתעכט. בראשית, נרצה שהמצבים q_0, q_1, q_2, q_3 ישמרו את מס' מופעי a . כמובן כל אחד מהם שומר את תוצאת החולקה ב-4. לאחר רצף מופעי a , האוטומט חbucks זוכר למשעה את ההפרש בין שארית החלוקה ב-4 של מופעי a לבין שארית החלוקה ב-3 שצריכה להיות. כך q_4 הוא $z - 1$, q_5 הוא $z + 1$, q_6 הוא $z + 3$ בלבד הוא מצב מקביל - שכן ההפך זה יראה כי q_7 יהיה מצב מלכודת, אם במקרה ירצו לשולח אליו עוד a לאחר רצף של b או אם שארית החלוקה של המילה b היא שלוש, לא יוכל שהミילה בשפה, שכן נרצה שייהיה שוויון בין השאריות וכן שארית 3 בחלוקת שלוש היא אפס. לכן גם מצב זה יישלח למילכודת.

2.5 בניית אבסטרקטית

2.5.1 אוטומט משלים

טענה: هي A אוטומט, אז קיימים אוטומט B כך ש $\overline{L(A)} = \overline{L(B)}$
את האוטומט המשלים, מקבלים כתוצאה מהפיכת המקבילים של A , ללא מקבילים, ואת הלא מקבילים, למקבילים.
לשיטת זו קוראים **בנייה אבסטרקטית של אוטומטים**, אנו משתמשים בבנייה של אוטומט אחד לשימוש עבור אחד אחר.

2.5.2 שפה רגולרית

הגדרה: תהי L שפה. נאמר כי L רגולרית, אם קיים אוטומט סופי A שמקבל את השפה כך $L = L(A)$
טענה: אם L רגולרית, אז גם \overline{L} רגולרית.

2.5.3 אוטומט המכפלת

טענה: תהיינה L_1, L_2 שפות רגולריות. אז $L_1 \cap L_2$ רגולרית.
הוכחה: $L_1 \cap L_2$ רגולרית ולכן קיימים אוטומטים A, B כך ש $L_1(A) = L_2(B) = L_2$ וכנון $L_1(A) \cap L_2(B) = L_1 \cap L_2$ בונה אוטומט סופי C , כך ש $C = L_1 \cap L_2$
נשים לב - באוטומט סופי אין לנו שמי' של מושג מה תוצאת הקלט, פרט לרגע בו אנחנו מקבלים אותו. ולכן בשבייל לדעת האט מילה נמצאת בשני אוטומטים (ובפרט בחיתוך), יש להרים במקביל, ואז בסוף להשוות האם סיימנו בשני האוטומטים במצב מקביל.

נסמן את האוטומטיים:

$$A = (Q^A, \Sigma^A, \delta^A, q_0^A, F^A)$$

$$B = (Q^B, \Sigma^B, \delta^B, q_0^B, F^B)$$

כעת נרצה להגדיר את C פורמלית:

- .א. $Q^C = Q^A \times Q^B$: המכפלה הקרטזית של המ מצבים של A ושל B . כלומר, המ מצבים של C יהיו זוג, מצב מ- A ו מצב מ- B : (q_a, q_b) . על הניגיר זה נראה מוזר זוג מצבים, אבל בפועל מדובר בסימונו.
- .ב. אם הוא ב- A שני מצבים וב- B שלושה, ב- C יהיו 6.
- .ג. $\Sigma^A = \Sigma^B$ אז הוא שווה גם ל- Σ . אם לא, הוכחה שונה קצרה (בקמפוס).
- .ד. **פונקציית המעברים**: נסתכל על מצב כלשהו, (q_i^A, q_i^B) : נסתכל מה יקרה כאשר נרצה להעביר אותן σ ב- A , הגענו למצב q_k . כאשר נרצה להעביר אותן ב- B , הגענו למצב q_r . ולפיכך $\delta((q_i^A, q_i^B), \sigma) = (q_k, q_r)$ ומעט יותר פורמלי,

$$\delta^C((q, p), \sigma) = (\delta^A(q, \sigma), \delta^B(p, \sigma))$$

$$F^C = F^A \times F^B.$$

- כעת, הריצה במקביל על C מדמה ריצה על $A \cap B$. כלומר, כל מילה שתתקבל C יקבלו A וגם B .
- אוטומט זה נקרא אוטומט מכפלה, כיון שקבוצת המ מצבים שלו היא המכפלה הקרזיטית של קבוצת המ מצבים של A ושל B .**

2.5.4 אוטומט מכפלה לאיחוד והפרש

- .א. **איחוד** $L_1 \cup L_2$ אוטומטים לשפות הרגולריות L_1, L_2 בהתאם ויהי C אוטומט המניפולציה עלייהן. אז, גם כאן - נróż במקביל על שתי השפות. ההבדל היחיד בין החיתוך בטכניקה הוא בקבוצת המ מצבים המקבלים:

$$F^C = (F^A \times Q^B) \cup (Q^A \times F^B)$$

(כלומר מכפלה קרזיטית של מצב כלשהו ב- A עם מצב מקבל עם B , או מצב כלשהו ב- B עם מצב מקבל ב- A)

- .ב. **הפרש** $L_1 \setminus L_2$: גם כאן ריצה במקביל וההבדל היחיד הוא בקבוצת המ מצבים המקבלים:

$$F^C = F^A \times \{Q^B \setminus F^B\}$$

כלומר, מכפלה קרזיטית של מצב מקבל של A עם מצב כלשהו שאינו מקבל של B)

ג. הפרש סימטרי $L_1 \triangle L_2$ (תזרורת - הפרש סימטרי הוא $L_1 \setminus L_2 \cup L_2 \setminus L_1$) באופן לא מפתיע, גם כאן כל השינוי הוא בקבוצת המ מצבים המתקבלים:

$$F^C = (F^A \times \{Q^B \setminus F^B\}) \cup (Q^A \setminus F^A \times F^B)$$

כלומר מכפלה קרוטית של האיחוד של ההפרשים.

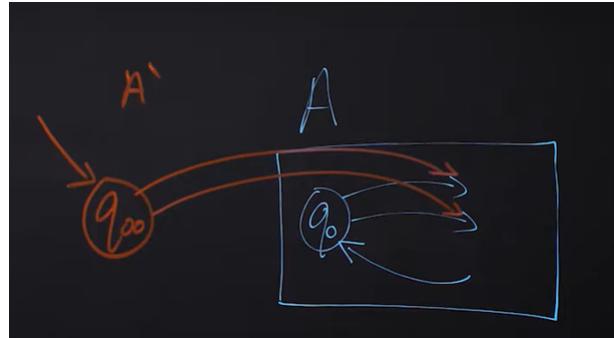
מסקנה: בהינתן שתי שפות רגולריות, Σ כל המניפולציות הבאות על השפות קיימים אוטומט שמקבל את המניפולציה: איחוד, חיתוך, משלים, הפרש סימטרי.

2.5.5 אוטומט אתחול

אוטומט אתחול הוא אוטומט שניינן לחזור למבוק החתמתי שלו לאחר קריאת קלט כלשהו שאינו ריק, ככלומר קיימת מילה $w \in \Sigma^*$ כך ש $w = q_0 \delta^*(q_0, w)$. ככלומר, יש מעבר שדרכו ניתן לחזור למבוק החתמתי.

יהי אוטומט סופי A שמקבל את השפה L . נרצה לבנות אוטומט חדש, A' שיקיים $L(A') = L(A)$. וכן A' לא יהיה אוטומט אתחול. ככלומר - לא ניתן היה לחזור למבוק החתמתי של האוטומט.

כיצד נבנה אותו?



כפי שנitinן לראות בתמונה, נוסיף מבוק q_{00} . את כל המעברים שיצאו מ- q_0 נוציא בעת מ- q_{00} . זהו, האוטומט לא ישתנה פרט לכך זה. בcut כשיינו מעברים חזרה, הם יהיו אל q_0 ולא אל q_{00} וכעת A' אינו אוטומט אתחול.

פורמלית: נסמן $A' = (Q', \Sigma', \delta', q'_0, F')$. נבנה $A' = (Q, \Sigma, \delta, q_0, F)$ כך ש: $Q' = Q \cup \{q'_0\}$ ו- $F' = F \cup \{q'_0\}$. וכן $\delta'(q, \sigma) = \delta(q, \sigma)$ ו- $\delta'(q_0, \sigma) = q'_0$.

$$\forall q \in Q, \sigma \in \Sigma : \delta^*(q, \sigma) = \delta(q, \sigma)$$

$$\forall \sigma \in \Sigma : \delta^*(q_{00}, \sigma) = \delta(q_{00}, \sigma)$$

באשר $\delta^*(q_{00}, \sigma) = \delta(q_{00}, \sigma)$

$$F' := \left\{ \begin{array}{ll} F \cup \{q_{00}\} & q_0 \in F \\ F & q_0 \notin F \end{array} \right\}$$

2.5.6 שפת הזוג

יהיו L_1, L_2 שפות מעל אותו א"ב Σ . נגידר את שפת הזוג להיות:

$$ZigZag(L_1, L_2) = \{w | w = a_1 b_1 a_2 b_2 \dots a_n b_n : a_1, \dots, a_n \in L_1, b_1, \dots, b_n \in L_2\}$$

נדגיש כי בשפת הזוג, נקח מילים $w_1 = |w_2|$ ונאגג בהםם. כלומר $w_1 \in L_1, w_2 \in L_2$ כך $w = w_1 w_2 \in L_1, L_2$ וכן $L_1 = \{abab, ababab\}$ ו $L_2 = \{bb, bbb\}$. $ZigZag(L_1, L_2)$ נקבל כי $L_2 = \{a, aa, aaa\}$ אם לב כי עבור a אין מקבילה ב L_2 באורך זהה ולבן איננה חלך משפט הזוג.

תרגיל. יהיו L_1, L_2 רגולריות מעל אותו א"ב Σ . הוכיח כי $ZigZag(L_1, L_2)$ רגולרית מעל Σ .
הוכחה: L_1 רגולרית לכן קיים אוטומט סופי דטרמיניסטי שמקבל אותה - נסמןו A , בדומה עבור B קיים אוטומט B . כך ש:

$$A = (Q^A, \Sigma, \delta^A, q_0^A, F^A)$$

$$B = (Q^B, \Sigma, \delta^B, q_0^B, F^B)$$

וכמוון $L(A) = L_1, L(B) = L_2$.
 נרצה לבנות אוטומט $C = (Q^C, \Sigma, \delta^C, q_0^C, F^C)$ כך שיתקיים $L(C) = ZigZag(L_1, L_2)$. מה יהיה רעיון ההוכחה? נרצה להשתמש באוטומטים שקיים לנו, כיוון שכל מילה בשפת הזוג מורכבת משתי מילויים, אחת של A וזוות של B . לפעשה ויזיות אליו ותבצע ויהה "מקבילי" - על האוטומט של A ושל B . לכן נגידר:

$$Q^C = Q^A \times Q^B \times \{A, B\}$$

כאשר: נפעיל מכפלה קרטזית על קבוצת המיצבים, כמו באוטומט מכפלה, וכן נוסיף מכפלה קרטזית עם האותיות A, B . לשם מה? נרצה בהינתן מצב מסוים, לדעת להיכן אני צריך לעבור כרגע. למשל, אם אני קורא אותן קלט של B , אני אצטרך לעבור לאחר מכן לאותן קלט של A , וכך הסימון יהיה A .
 שיבירר לי - אתה חולץ A .
באשר למכב ההתחלתי -

$$q_0^C = (q_0^A, q_0^B, A)$$

כיוון שכל מצב הוא שלשיה, ונרצה להיות כרגע בתחום A ולאחר הסימון של A כיוון שכל מילה בשפה פותחת באות A .
פונקציית המכביים: היא אמורה לקבל מצב ואות ולהחזיר מצב. כפי שאמרנו מצב אצלנו הוא שלישיה סדרה, לכן הפונקציה מקבל שלשיה ואתה ותחזר שלשיה.

$$\forall p \in Q^A, q \in Q^B, \sigma \in \Sigma : \delta^C((p, q, sign), \sigma) = \begin{cases} (\delta^A(p, \sigma), q, B) & sign = A \\ (p, \delta^B(q, \sigma), A) & sign = B \end{cases}$$

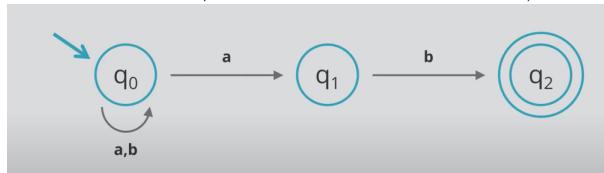
הטבר: כאשר אנחנו נמצאים בתוך A , נרצה לעcor בקורסיה הקלט הראה אל A וכן נפעיל את דלתא של A , נשאר בפ' כי לא צו B וכן ה $sign$ ישתנה ל B כי אנו כרגע ב A וכקורסיה הקלט הראה נרצה לעcor אל B . **בזומה, עכו המקרה השני.**
cut נגע אל קבוצת המצביעים המקבילים:

$$F^C = F^A \times F^B \times \{A\}$$

מדוע כך הגדרנו? נרצה להיות במצב מקלט של A , מצב מקלט של B וכן שהמעבר הבא צריך להיות אל A , ככלומר המילה הסטימית cut ב B , וזה הדרך היחידה שחוקית.
זה"כ - הוכחנו שקיים אוטומט סופי דטרמיניסטי C שמקבל את שפת הזוג, וכך הינה רגולרית. ■.

3 ייחידה 4: אוטומט סופי לא דטרמיניסטי

איך שלא, נתחל על הדוגמה. נסתכל על הדוגמה מטה, על פניו - נראה ליטימי. מה ההבדל? נשים לב, מה מצב q_0 ישנו שני>Statusים שונים של a . או להשתאר באותו מקום, המותbeta ע"י לולא עצמית או להתקדם הלאה אל q_1 . וזה לבדוק ההבדל - לא דטרמיניסטי: ישנו כמה אפשרויות עברו אותן קלט לאן להתקדם. כמו כן, נשים לב כי לא בכל מצב מטופלות כל אותיות הקלט. למשל - ב q_1 אין טיפול באות הקלט a . כפי שהיא במודל הדטרמיניסטי. כמו כן, ישנו מילים שלולות לא יסתימו במצב מקלט: למשל, $aa = aa$ לא הגיעו למצב מקלט.



באוטומט הסופי הלא דטרמיניסטי, תתקן הרצה זהה שתוביל לתוצאות קלט שונות. **האוטומט הלא דטרמיניסטי לא קובע תוצאה בזורה חד חד ערכית.**
נשים לב - כיון שלא בכל מצב חיברים לטפל בכל אותיות הא"ב, אם נגע למושך עם המחרוזת $w = aba$ דרך המסלול שמתחל מ $q_1 \rightarrow q_0$, נגיע אל q_2 כאשר נותרה בידינו אות קלט - a . אבל q_2 לא לטפל בה, ולכן החישוב ייתקע. זה לא שהוא ישר ב q_2 אלא החישוב ית��ע למגררי (אפשר להסתכל על זה כמו קriseה של התוכנית".)

הגדרה: יהי אוטומט סופי לא דטרמיניסטי A מעל א"ב Σ . תהי $w \in \Sigma^*$ מקבל את w אם"מ **קיימת** הרצה של האוטומט על המילה, שמסתיימת במצב מקלט.

הערה - נשים לב כי בשביל להכריע האם w לא שייכת לשפה, צריך לעבור על כל הרצות האפשרות ולודא שבסכל אחת מהן לא מופיעים במצב מקלט.

3.1 הגדרה פורמלית

אוטומט סופי לא דטרמיניסטי הינו חמשייה: $N = \{Q, \Sigma, \Delta, q_0, F\}$. כאשר Q היא קבוצת המצביעים, Σ הוא הא"ב, Δ היא פונקציית המעברים, q_0 הינו המצב ההתחלתי וכן F היא קבוצת המצביעים המקבלים וכמוון $F \subseteq Q$.

ההבדל המרכזי הוא בפונקציית המעברים. נגידרה: $\Delta : Q \times \Sigma \rightarrow P(Q)$

כשההבדל הוא שהפונקציה הפעם לא הולכת אל Q , אלא אל קבוצת החזקה של Q . כיוון שייתכן שקלט a יಲך לתת קבוצה של מעברים, כמו שדנו קודם לכן. למשל, מהדוגמה לעיל כאן לעיל, נקבל כי: $\{q_0, q_1\} = \Delta(q_0, a)$ וכן $\emptyset = \Delta(q_1, a)$ (כיוון שאין דרך לעבר מ q_1). באופן דומה, נוכל להרחיב את פונקציית המעברים מאותיות למחרוזות:

$$\Delta^* : Q \times \Sigma^* \rightarrow P(Q)$$

למשל, מהדוגמה לעיל מעלה: קריאה של ab יכולה להוביל לשני מצבים שונים, וכך $\Delta^*(q_0, ab) = \{q_0, q_2\}$. כמו כן, $\Delta^*(q_0, \epsilon) = \{q_0, q_2\}$

3.2 שפת אוטומט לא דטרמיניסטי

עבור אוטומט לא דטרמיניסטי N , השפה של N הינה הקבוצה:

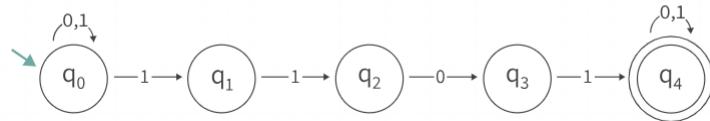
$$L(N) = \{w \mid \exists f \in F : f \in \Delta^*(q_0, w)\} = \{w \mid \Delta^*(q_0, w) \cap F \neq \emptyset\}$$

הנה המקום להציג - כיצד האוטומט הלא דטרמיניסטי יודע כיצד לבחור בין האפשרויות שלפניו? הכל אמרו להתבצע אוטומטיות הרי, כיצד הוא בוחר, לפי גחומיות האישיות? וכאן נסביר: אין באמות דבר כזה בחיים האמיטיים, מדובר מודול מתמטי בלבד. זה קיים במוח בלבד, בהגדרות המתמטיות בלבד. המודול מתמטי - ויפשط לנו בהמשך כל מיני טענות.

3.3 בניית אוטומט לא דטרמיניסטי

כמו אוטומט רגולרי, נתקל לעיתים בתרגילים בהם נדרש לבנות אוטומט סופי לא דטרמיניסטי.

דוגמא 1. בניית אוטומט לא דטרמיניסטי מעל $\{0, 1\}^*$ שיקבל את כל המילims שמכילות את המחרוזת "1101".

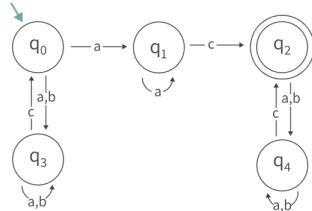


דוגמא 2. בניית אוטומט לא דטרמיניסטי שיקבל את השפה הבאה:

$$L = \{w_1 c w_2 c w_3 c \dots \dots w_k c \mid k \geq 1, \forall 1 \leq i \leq k : w_i \in \{a, b\}^+, \exists 1 \leq i \leq k : w_i \in \{a\}^+\}$$

נראה מרכיב, בפועל סה"כ מנוסים לבלבל אותנו. מה הם רוצחים? זיגז של w -ים עם אות c , כך שתמיד w -ים הם חלק מ $\{a, b\}^+$ וכן קיים איזשהו מחרוזות שהיא $\{a\}^+$ בלבד, לפחות a בלבד. איך נגשים לזה? נראה כי בידינו אוטומט לא דטרמיניסטי, ולכן ניתן לו האפשרות להחליט לאן ללכת: בניית מסלול מ q_0 שמאלה, אל q_1 ובו נdag לרצף של a -ים ולאחריו c . אפשרות אחרות, תהיה להתחיל

מרוצפים של a , b ולאחר מכן c וחזר חלילה. כך וידאו כי יהיה רצף של a -ים ולאחריו c במסלול. כמו כן, נראה כי לאחר שהגענו אל q_2 הבטחנו כי בידינו רצף של a -ים בלבד ולאחר מכן c , אך יתכן שימשכו רצפי ab , וכך טיפלו בהז במאובט q_4 . סה"כ היה נראה מפהיד - אבל המודל הלא דטרמיניסטי עזר לנו לגשת לזה בצורה טוביה.



3.4 שיקולות

מה הקשר בין המודל הלא דטרמיניסטי למודל הדטרמיניסטי? האם ישנו תשובות שנייתן לקבל באמצעות אוטומט לא דטרמיניסטי, אך ניתן באמצעות אוטומט דטרמיניסטי התשובה, המפתיעה, היא - לא.

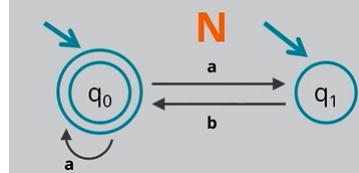
כל שפה שניתן לקבל באמצעות אוטומט לא דטרמיניסטי, ניתן לקבל באמצעות אוטומט דטרמיניסטי.
נראה להוכיח טענה זו. לפני כן, נזכיר את המושג של אוטומט לא דטרמיניסטי.
נדיר כעת אוטומט לא דטרמיניסטי $N = \{Q, \Sigma, \Delta, Q_0, F\}$ כאשר Q_0 היא קבוצת המ מצבים ההתחלתיים. ככלומר - אפשר יותר ממצב ההתחלתי אחד. לשם כך, המודל יבחר בצורה לא דטרמיניסטיבית בהתאם, באיזה מצב להתחילה. נראה כי המודל הקודם שראינו, עם מצב ההתחלתי אחד, הוא מקרה פרטי של מודל זה.

טענה: *יהי N אוטומט סופי לא דטרמיניסטי (עם קבוצת מצבים ההתחלתיים, איי ($L(N)$ רגולרית).
כלומר, קיימים אוטומט סופי דטרמיניסטי D כך ש $L(D) = L(N)$).
הערה. כמובן שהטענה נכונה גם במקרה הפרטי, בו $1 = |Q_0|$.*

הוכחה:

יהי N אוטומט סופי לא דטרמיניסטי: $N = \{Q^N, \Sigma^N, \Delta^N, Q_0^N, F^N\}$. נבנה אוטומט סופי דטרמיניסטי $D = \{Q^D, \Sigma^D, \Delta^D, Q_0^D, F^D\}$ כך ש $L(N) = L(D)$.
נדיר היבט את מרכיבי D .

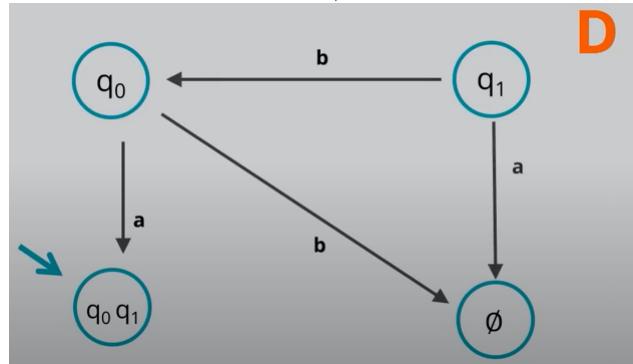
ראשית, נתחל בדוגמה שתעזר לנו במבנה ההוכחה.



מה ההבדל בין אוטומט לא דטרמיניסטי לבין דטרמיניסטי? בהינתן קלט כלשהו, ללא דטרמיניסטי ניתן להגיע לקבוצת מצבים. אנחנו נרצה להתחיל בקבוצות אוטומט שմצביעו שם בזוויק כל הקבוצות האפשרות. למשל, בהינתן קבוצת מצבים שניתנו להגעה אליה $\{q_0, q_1\}$ בקריאה קלט ניתן להגעה לקבוצות הבאות $\{q_0, q_1, \emptyset\}$. כעת יוצר אוטומט שימושתו יהוו בזוויק - שמות הקבוצות הללו. נראה כי לאוטומט בדוגמה שלו, ישנס שני מצבים ההתחלתיים, q_1, q_0 . כלומר, נרצה לבנות אוטומט חדש, דטרמיניסטי, עם מצב ההתחלתי אחד, מי זה יהו? q_0, q_1 - באוטומט החדש שלו, אכן קיימים מצבים אלה. ובאריך לקבוצה הריקה? נכון לשם, עכשו המילוי שככל מקרה - האוטומט המקורי היה נתזע בחישוב בזוויק להגעה לשם.

מה באשר לפונקציית המעברים שלו?

נסתכל על q_0 . נרצה להעכור קלט a . נראה כי ישנו שתי אפשרויות כיצד a יועכור - או שיישאר ב- q_0 או שייעכור אל q_1 , וכן קבוצת המ מצבים אליו יוכל לעכור היא $\{q_0, q_1\}$ - וכן באוטומט החדש, הדטרמיניסטי, D , a יועכר אל המצב q_0q_1 . מה באשר לקרויה של b מ- q_0 ? נראה כי אין אפשרות להמשיך - וכן b יועכר באוטומט הדטרמיניסטי, אל הקבוצה הריקה.



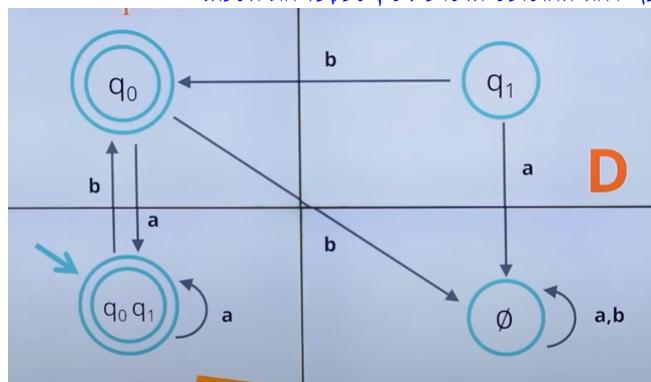
ובכן, זה האוטומט שלנו. נשים לב כי הוא עדין איננו דטרמיניסטי. עדיין, לא כל מילך מטופל בכל אפשרויות הקלט.

נזכיר ואנו גם במצב q_0q_1 . כמובן, אם היינו בפזול הלא דטרמיניסטי, יש חישוב שהוא מביא אותנו ל- q_0 ווש חישוב שהוא מביא אותנו ל- q_1 . נראה לטפל באות הקלט b שתצא מ מצב q_0q_1 . נשים לב שהגענו למצב q_0 או למצב q_1 . מהמצב q_0 או זו דڑ להמשיך עם b , ואילו מהמצב q_1 ניתן להמשיך אל q_0 בפזול הלא דטרמיניסטי, וכן סה"כ אם אנחנו הגיעו ל- q_0q_1 נוכל להגיע רק למצב q_0 וכן נבער מעבר של אותן הקלטים b מ- $q_0 \rightarrow q_0q_1 \rightarrow q_0$. עבור a מ- q_0q_1 אפשר להמשיך אל q_0 או q_1 ולא ניתן לעשות כלום - וכן את הקלט a תשאor בollowה עפ"ת במאז q_0q_1 .

נעיר, כי תמיון כשנינו למילך של הקבוצה הריקה - נבער לו לא עפ"ת עם כל ס' - כיוון שבאוטומט המקיים נתנו פירוש לשלב זה, וכן ייתן להסתכל על ה-"מצב מלוכחת".

מה באשר למיצבים המתקבלים? נראה כי פילה הרויה בשפה שאוטומט N קובל אפ"ע הוה קיומ מסלול בו היא התקבלה, נראה כי bN ורק q_0 הוא מצב מקבל - וכן גם אצלנו הוארה מצב מקבל. כמו כן, אם הגיעו למצב q_0q_1 משמע שכאותומט המקורי היה דורך למליה להתקבל שהסתימה ב- q_0 וגם ב- q_1 בפרט ב- q_0 , וכן זה גם מילך מקבל.

כך נראה האוטומט הדטרמיניסטי, שמקבל את השפה:



נראה להוכיח. נכליל את הדוגמה לדוגמה הכללי, נרצה להגדיר את מרכיבי D :

- קבוצת המ מצבים -

$$Q^D = P(Q^N) = \{R \subseteq Q^N\}$$

כלומר, קבוצת המיצבים החדש החזקה על קבוצת המיצבים הקודמת.
ב. $\Sigma^N = \sum^D$ - הקלטים לא משתנים
ג. פונקציית המעברים:

$$\delta^D(R, \sigma) = \bigcup_{q \in R} \Delta^N(q, \sigma)$$

כלומר, פונקציית המעברים תוגדר להיות: בהינתן קבוצה R)זה הרि שם המיצב) ואות σ , נרצה לעבור לקבוצה שהיא איחוד כל המיצבים, כך שבהינתן $q \in R$, הפונקציה שולחת אליו. ובמילים פשוטות יותר - נאخد את כל הדריכים האפשריות למעבר הקטל ששייך לקבוצה, זו קבוצה שהיא וודאי חלק מ(Q^N) P וכעת היא שם של מצב ב D לפי א', וזה המצב אליו עברו.
חשוב להתעכ卜 על חלק זה בהוכחה כי זה קצת מסורבל אבל אחרי شيء קריואת אפשר להבין.
ד. המיצב ההתחלתי - $q_0^D = Q_0^N$. ככלומר, המיצב ההתחלתי של D הוא ייחיד, והוא פשוט הסימון של קבוצת המיצבים ההתחלתיים. (בדטרמיניסטי, יש מצב ההתחלתי אחד).
ה. המיצבים המתקבלים

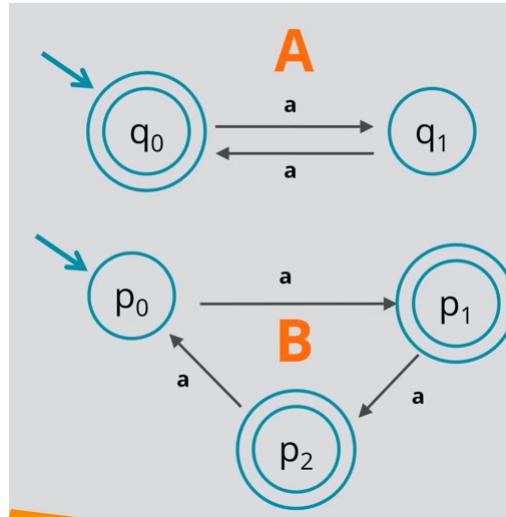
$$F^D = \{R \in Q^D : R \cap F^N \neq \emptyset\}$$

כלומר, קבוצת המיצבים המתקבלים זו קבוצת כל המיצבים ב Q כך שאם נסתכל על תת קבוצה שלנו עם קבוצת מיצבים מתקבלים של N , לא נקבל קבוצה ריקה.
(ישים לנו, בהוכחה לא הסבירו מזווע אכו האוטומט D מתקבל את אותה השפה, אלא רק הגדרו אותו. יספיק לנו כרגע, ההוכחה המלאה בקמפוס. המטרה בהבאת ההוכחה לכאן היא לדעת כיצד לעכבר בו אוטומטios בשאלות טכניות)
אכן D הוגדר היטב, והטענה הוכחה. כנדרש.

מסקנה. מעתה, נוכל תמיד לבנות אוטומט לא דטרמיניסטי, ובאמצעות האלגוריתם שתואר כאן לעיל, נוכל להמיר לאוטומט כן דטרמיניסטי. לפי הטענה כאן ("ראיינו בהרצאה"), נוכל תמיד לבצע מעבר זה.

3.5 מעברי אפסילון

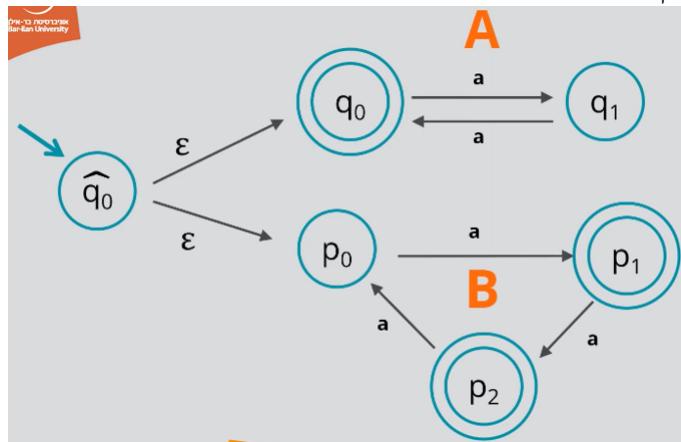
נרחיב את המושג של האוטומט הסופי. ובפרט, את המודל הלא דטרמיניסטי (שלפי טענה לעיל שקיים לאוטומט דטרמיניסטי).
הרעיוון הבסיסי הוא לאפשר מעבר באוטומט ללא קריית אף אותן קלט, כשנרצה לעשות זאת, נסמן זאת באמצעות האות ϵ . על קשותות אלו ניתן לעبور ללא קריית קלט באוטומט. נדוגש כי לא **חייבים** לעبور על מעברי האפסילון, וכן **יתכן מצב שייהי לנו על מעבר ϵ** , למשל ונרצה לבחור האם לדגל או לקרוא את האות a . זהרי אפשרות לא דטרמיניסטיות.
מדוע נצורך מעבר אפסילון? הוא מפשט לנו את בניית האוטומט. נתבונן בדוגמה -



אוטומט A מקבל את כל המילים באורך זוגי ואילו אוטומט B מקבל את כל המילים שאורכו לא מתחלק ב-3. נניח ונרצה לבנות אוטומט שיתאר את האיחוד של שתי שפות אלה. ככלומר,

$$C = \{w \in \Sigma^+ : |w| \% 2 = 0 \vee |w| \% 3 \neq 0\}$$

אפשרות אחת תהיה לעבוד קשה ולהסתבך לפי האלגוריתם שרריאנו או שם מעלה בסיכום במבנה אבסטרקטית. אפשרות קלה הרבה יותר תהיה לבנות אוטומט לא דטרמיניסטי, באמצעות שני מעברי אפסילון:

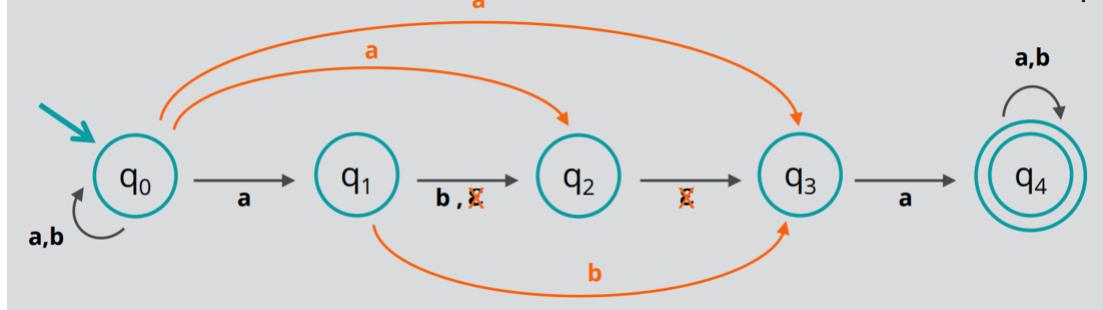


icut האוטומט מקבל את השפה C , ובחירה באופן לא דטרמיניסטי להיכן לכלכת. ביטלו את המ מצבים p_0, q_0 ו q_0 מכובדים התחלתיים והוספנו \hat{q}_0 כמצב התחלתי חדש, עם שני מעברי אפסילון.icut, אם נרצה גם אוטומט דטרמיניסטי - נוכל להפוך אותו לכך באמצעות המעבר לעיל שתואר בהוכחה מעלה.

טענה: יהיו N^ϵ אוטומט סופי לא דטרמיניסטי עם מעברי אפסילון. אז קיים אוטומט סופי לא דטרמיניסטי N בלי מעברי אפסילון, כך ש $L(N) = L(N^\epsilon)$.

הוכחה: יהיו $N^\varepsilon = \{Q^\varepsilon, \Sigma^\varepsilon, \Delta^\varepsilon, Q_0^\varepsilon, F^\varepsilon\}$ אוטומט סופי לא דטרמיניסטי עם מעברי אפסילון: $\{Q^N, \Sigma^N, \Delta^N, Q_0^N, F^N\}$ בנו אוטומט סופי לא דטרמיניסטי בלי מעברי אפסילון: $N = L(N^\varepsilon)$.

ראשית נקבעו בדוגמה.



בתמונה מתואר אוטומט הכלול מעבריו אפסילון. נשים לב כי ישנו מעבר אפסילון $q_3 \rightarrow q_2$. ומה זה אומר? כל קלט יכול לעבור שס. כמו כן, ישו מעבר אפסילון $q_2 \rightarrow q_1$. מה הוניבע בהיפוך האוטומט ללא מעברי אפסילון? נראה כי האות a למשל, יכולה לעבור אוטומטי מ q_1 ל q_2 תמי, בנסיבות מעבר האפסילון, וכן היא עוכרת קודס לו מ q_1 → q_0 . לנו הרעיון והוא לאפשר מעבר $q_2 \rightarrow q_0$ מראש של a , וכך גם ותרנו על מעבר האפסילון וס האוטומט נשאר כשרה. באופו דומה, יתו להעבור מ $q_3 \rightarrow q_0$ את האות a וכן נמנע מעבר האפסילון $q_3 \rightarrow q_2$ וכן אותו דבר על מעבר b ב $q_3 \rightarrow q_1$. כתעת נראה לנוכח זאת פורמלית.

ראשית, לכל $q \in Q^\varepsilon$ כאשר p הוא כל מי שניתן להגעה מ q בלבד נקרא קלט. ככלומר זו קבוצה. בפרט תמיד $CL(q) \subseteq Q^\varepsilon$. למשל, בדוגמה מעלה, $CL(q_1) = \{q_1, q_2, q_3\}$.

נפרט את מרכבי N :

$$\begin{aligned} Q^N &= Q^\varepsilon \\ \Sigma^N &= \Sigma^\varepsilon \end{aligned}$$

ג. פונקציית המעברים: בהינתן מצב q ואות σ ,

$$\Delta^N(q, \sigma) = \bigcup_{p \in \Delta^\varepsilon(q, \sigma)} CL(p)$$

ובעברית: בהינתן אותן קלט ומצב, ממנו נוכל להגעה להרבה מצבים, אך זו תהיה קבוצה. נרצה לעבור על כל המצבים אליהם ניתן להגעה באמצעות N^ε עם אותן והמצב הנקובי, ובכל אחד מהם לעבור על כל המצבים q בהם ניתן להגעה מ q , כך שלא נקרא קלט. כך למעשה, נבעץ כמו בדוגמה מעלה שיגור של האוטומת מצב נוכחי למצבים אחרים, ללא הצורך במעבר אפסילון.

ד. $Q_0^N = \bigcup_{p \in Q_0^\varepsilon} CL(p)$ - ככלומר קבוצת המצבים הראשוניים תהייה עבור על כל איבר בקבוצת המצבים הראשוניים, ולאסוף את כל המצבים שמשמו ניתן להגעה אליהם ללא קריאת קלט. אלו יהיו - המצבים הראשוניים.

$$F^N = F^\varepsilon$$

אכן N הוגדר היטב, מדובר הם מקבלים אותה שפה, קריגיל - לא רלוונטי לכך, והטענה הוכחה, כנדרש. ■.

3.6 בניית אבסטרקטית של אוטומט לא דטרמיניסטי

ניתן לבצע בניית אבסטרקטית גם על אוטומטים לא דטרמיניסטיים, עם מעברי אפסילון. נתבונן במס' דוגמאות.

3.6.1 מצב מקובל ייחיד

יהי A אוטומט דטרמיניסטי. נרצה לבנות אוטומט לא דטרמיניסטי, A' עם מצב מקובל ייחיד, שמקובל את אותה השפה כמו A .
 כיצד נפתרו את הבעיה? נסתכל על המฉบבים המקבילים הקיימים, הרוי הם קבוצה, $Q_0 = \{q_1, q_2, \dots, q_n\}$. נבנה מצב מקובל חדש, נקרא לו q_s . עבור כל $i \leq n$ נסיף מעבר אפסילון מ q_i אל q_s . כמו כן, נחפוץ את כל המฉบבים המקבילים ללא מקבילים. סימנו - יש לנו מצב מקובל ייחיד.
 אם נרצה מעט יותר פורמלי, $A = (Q, \Sigma, \delta, q_0, F)$, נגיד $A' = (Q', \Sigma', \Delta', q'_0, F')$ כידלמן: $A' = (Q', \Sigma', \Delta', q'_0, F') = Q \cup \{q_s\} = \Sigma' = \Sigma, q'_0 = q_0$ וכן $F' = F \cup \{q_s\}$ וכן נרצה להוסיף מעבר אפסילון, $\Delta'(q, \sigma) = \{\delta(q, \sigma)\}$

$$\forall q \in F : \Delta'(q, \varepsilon) = \{q_s\}$$

נשים לב כי Δ איננה דטרמיניסטית, ולכן ממצב ואות עוברים לקבוצה של מฉบבים.

3.6.2 ערובות שפות

יהיו L_1, L_2 שפות רגולריות. נרצה להוכיח כי השפה הבאה רגולרית:

$$L_3 = \{w_1 w_2 w_3 | w_1, w_2 \in L_1, w_3 \in L_2\}$$

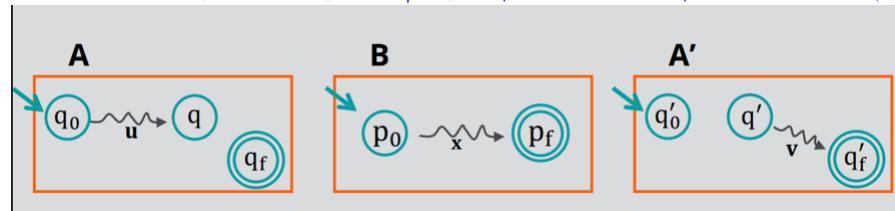
נראה כי ניתן לכתוב גם את $L_3 = L_1 \circ L_2 \circ L_1$. הרעיון יהיה פשוט: L_1, L_2 רגולריות ולכן יש אוטומטים שמקבלים אותם. כל שנרצה, הוא לשגר את האוטומטים. נסכו בהתחמזה את האוטומטים המקבילים את השפות B, A . כמו כן, ניצור אוטומט $A' = A$ (משם שכפול). הרעיון (לא פורמלי) יהיה כך -
 א. מצב ההתחלתי יהיה המצב ההתחלתי של A . ממשנוначילה. שם נקרא את w_1 .
 ב. מכל מצב מקובל של A , נעביר מעברי אפסילון אל המצב ההתחלתי של B . שם נקרא את w_2 .
 ג. מכל מצב מקובל של B , נעביר מעברי אפסילון אל המצב ההתחלתי של A' . שם נקרא את w_3 .
 ד. המฉบבים המקבילים של A' יהיו המฉบבים המקבילים של האוטומט שלנו. המฉบבים המקבילים של A, B יהפכו למฉบבים רגולרים. כך קיבלונו - את השפה.

3.6.3 הכלאת שפות

יהיו L_1, L_2 שפות רגולריות. נרצה להוכיח כי השפה הבאה רגולרית:

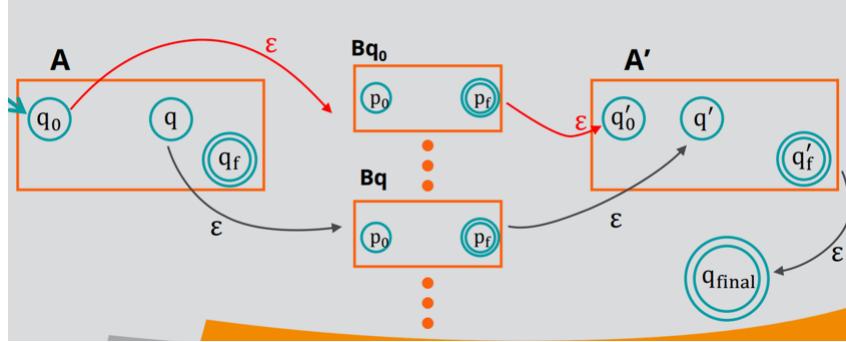
$$L = \{w | w = u \cdot x \cdot v : u, v \in L_1, x \in L_2\}$$

נכונה אוטומט לא דטרמיניסטי שיקבל את השפה. מטענה לעיל, אם נוכחה שקיים אוטומט לא דטרמיניסטי שיקבל את השפה, קיים גם אוטומט דטרמיניסטי שמקבל את השפה. וזה אכן הינו רגולריות, לכן קיימים אוטומטים A, B שמקבלים אותו בהתקאה. נתוך גם $A' = A$. הרעיון יהיה כמו בתרגול הקוקוס: את u רצוחה לקרה על A , את x על B ואת v על A' .



כיצד תתכצע קריאת מיליס? כידלמן:

כיצד נוכל לנוע בתוך האוטומטים? למשל, לעבר מ- q אל q' ? באמצעות מעבר אפסילון. כמו כן, נסמן מעבר אפסילון מ- q אל q' . ואז קירiat המילה תהיה חלקה. שיש לנו שזוגמא זו היה כלית מז. הרו, מהו q הזה בתוך A ? זה סתם הווה ייחוש שאסור לנו לצעע. לנו יותר שקיימות מעבר אפסילון אחרים. הכוונה שבודינו: כאשר גניע למק' p_0 , נטרץ לאזכור ולשומו, אותה מק' ב- A הוביל אותנו אליו. לנו הפטרון יהה:



כלומר, נשמר לכל מ麦克' B מסוים. קלומר ונכפל את אוטומט B כמ' המצביע שיש לנו במק' A . וכה, כאשר עברו אל אוטומט A' מ- B נדע מהיכן הגענו. כמו כן, נסמן מעבר אפסילון מחוץ ל- A' עבר המקב' הסופי שלו, שיצא מהמקב' המתקבל של A' (ייתכנו כמה), והוא יהיה מ麦克' מתקבל.

פורמלית:
נסמן את האוטומטים בהסתממה

$$A = (Q, \Sigma, \delta^A, q_0, F^A)$$

$$B = (P, \Sigma, \delta^B, p_0, F^B)$$

$$A' = (Q', \Sigma, \delta^{A'}, q'_0, F^{A'})$$

בנייה אוטומט C כדלקמן:

$$C = (Q^C, \Sigma, \Delta^C, q_0^C, F^C)$$

ונדריך את רכיביו להלן:
א. $\{q_{final}\} \cup Q' \cup (Q \times P) \cup Q^C = Q \cup (Q \times P) \cup Q^C = Q$ מדוע מכפלה קרטזית? נזכיר כי היה לנו שכפול של אוטומט B , וכן המצביע בו תלויים במצבים השונים של Q
ב. $q_0^C = q_0$
ג. $F^C = \{q_{final}\}$
ד. פונקציית המעברים:

$$\forall q \in Q, \sigma \in \Sigma : \Delta^C(q, \sigma) = \{\delta^A(q, \sigma)\}, \Delta^C(q, \varepsilon) = \{(q, p_0)\}$$

$$\forall (q, p) \in Q \times P, \sigma \in \Sigma : \Delta^C((q, p), \sigma) = \{(q, \delta^B(p, \sigma))\}, \forall (q, p) \in Q \times F^B : \Delta^C((q, p), \varepsilon) = \{(q')\}$$

$$\forall q' \in Q', \sigma \in \Sigma : \Delta^C(q', \sigma) = \{\delta^{A'}(q', \sigma)\}, \forall q' \in F^{A'} : \Delta(q', \varepsilon) = \{q_{final}\}$$

4 יחידה 5: שפות וגולריות

אנו מתקדמים בדרך כלל לפתרון השאלה: **אייזה בעיות ניתנות לפתרון באמצעות מחשב?**. ובכן: בעיה הוגדרה כשפה, מחשב כרגע הוא אוטומט סופי, ופתרון הוא קבלת ע"י האוטומט. כמובן - השאלה הומהה, **אייזו שפה ניתנת לקבלת ע"י אוטומט סופי?** שפות אלו - נקראות **שפות וגולריות**. ביחידה זו נחקור את התכונות של שפות וגולריות, ונבדוק האם ניתן להוכיח על שפות מסוימות, שאין גולריות.

4.1 סגירות

קובוצת ניתנות לסגירות על פעולה מסוימת, אם הפעלת הפעולה על האיברים בקובוצה תשאיר אותנו בקובוצה. למשל: עבור \mathbb{N} הפעולה $+$ "היא פעולה סגורה תחת הפעולות. לעומת זאת, " — " איןנה בהכרח תביא מ' טבעי, וכן הפעולות לא סגורים לחיסר.

טענה: לפי היחידה הקודמת, קובוצת השפות הגולריות, סגורה תחת משלימים, איחוד וחיתוך.
הערה: הכוון החפוץ לא נכון, אם $L_1 \cup L_2$ רגולרית למשל, זה לא גורר גולריות של השפות בלבד. למשל נקח $L_1 = \overline{L_1} = \{a^n b^n\}$, $L_2 = \Sigma^*$, מתקיים כי $L_1 \cup L_2$ שנרגולרית, כי האוטומט שלו הוא לולא עצמית עם כל הא"ב.

4.1.1 סגירות לשורש

יהיו L_1, L_2 רגולריות. אז $L_1 \circ L_2$ רגולרית גם כן.
הוכחה לא פורמלית: הבניה תהיה פשוטה, קיימים אוטומטים לא דטרמיניסטיים A, B שמקבלים את השפות בהתאם. נבנה אוטומט לא דטרמיניסטי C שיקבל את השפה החדשה ע"י ליקחת האוטומטים המקוריים, נניח כי B קיים מצב מקבל יחיד, ואם הוא לא יחיד, נסיף מצב מקבל, מכל מצב מקבל קודן נסיף מעבר אפסילון, ונטול את המצבים המקוריים הקודמים. כתע, נסיף מעבר אפסילון מהמצב המקביל של A למצב התחלתי היחיד של B , וסיימנו. כל המצבים המקוריים יתבטלו באוטומט החדש, המצב המקביל היחיד יהיה המצב המקורי של B שוב, נניח שקיים אחד כי אם לא נסיף מעבר אפסילון, והמצב התחלתי יהיה המצב התחלתי של A .

4.1.2 סגירות לסגור קלין

תהי L רגולרית. אז, L^* רגולרית גם כן.
הערה - הכוון השני, לא נכון: למשל $L = \{a^{2^k} | k \geq 0\}$ שאינה רגולרית (נראתה בהמשך) אך $L^* = \{a\}^*$ שכן רגולרית.

הוכחה לא פורמלית: רגולרית שכן קיימים אוטומט A שמקבל אותה. נבנה אוטומט לא דטרמיניסטי שקיבל את L . בדומה, נניח שקיימים מצב התחלתי יחיד ומצב מקבל יחיד. (אם לא נגעים לשם). בשביל לקלול את L^* , כל שימוש שהוא הוא להוסף מעבר אפסילון מהמצב המקביל q_f אל המצב התחלתי q_0 . כך, אכן נוכל לשגר מס' סופי של פעמים מילים מהשפה, וזה בדיקת הסגור-קלין. נשים לב שבנוינו מכונה עבר L^+ . עבר קבלת L^+ , נסיף מצב התחלתי נסוף, q_{00} , והוא יהיה מצב מקבל,

וממנו יהיה מעבר אפסילון אל המצב ההתחלתי q_0 .

טענה: ניתן לשלב בין סגירותים שונות. כאשר, מס' הפעמים שנבצע פועלות סגירותים שונות הוא סופי.

דוגמא. אם $R_{\text{גולרית}} = q_1 \cup q_2 \cup (L_1 \cup L_2 \cup L_3)^*$ רגולרית גם כן. עם זאת,

$L_1 \circ L_2 \circ L_3 \circ L_1 \circ L_2 \circ L_3$ איננה בהכרח רגולרית, כי מס' הפעולות שנבצע אינו סופי.

טענה: כל שפה סופית, הינה רגולרית. (כלומר, כל שפה עם מס' מילים סופי, הינה רגולרית).
הוכחה: כל מחרוזת בודדת בשפה w הינה רגולרית. מדוע? נבנה לה אוטומט לפי שיטת השלד.
כיוון שמס' המילים בשפה סופי, נבנה אוטומט אחד לכל המילים בשפה. לפי טענה לעיל, האיחוד
ישמר על הרגולריות של השפה, ולכן השפה כולה, הינה רגולרית.

4.1.3 סגירות לורוס

תהי L רגולרית. אז גם L^r רגולרית.

הוכחה לא פורמלית: סמן את האוטומט שמקבל את L כ- A) קיים כי רגולריתן. יש לו מצב התחלתי, q_0 ומצב מקבל q_f . אנחנו נבנה אוטומט חדש' A' בו המצב ההתחלתי יהיה q_f , כל ח' שנכנס אל q_f icut יצא ממנו, וכל ח' שיצא ממנו יכנס אליו. בדומה, עבור כל מצב אנחנו נחפץ את סדר החצים. וכן, המצב q_0 יחפץicut למבוקש.icut בידינו אוטומט' A' שמקבל את שפת הרורס. נשים לב, כי האוטומט החדש, יתכו ויהיה לא דטרמיניסטי. באופן פורמלי, פונקציית המעברים תראה כך:

$$\forall q \in Q^{A'}, \sigma \in \Sigma : \Delta^{A'}(q, \sigma) = \{p | \delta^{A'}(p, \sigma) = q\}$$

כלומר, כל המצבים p כך שפונקציית המעברים העבירה אותם למצבנו הנוכחי.

4.1.4 סגירות prefix

תהי L רגולרית, אז $\text{prefix}(L)$ רגולרית גם כן.

הוכחה לא פורמלית: L רגולרית לכן קיים אוטומט סופי דטרמיניסטי שמקבל אותה, נסמן A . יש בו מצב התחלתי q_0 , ומצב מקבל q_f (ייתכנו כמה). הראינו בהוכחה להסתכל על כל המסלולים שיכולים להוביל אותנו אל מצב מקבל, נסתכל על המסלולים האלו ונחפץ את כל המצבים בהם להיות מצבים מקבלים, כך כל תחילית של מילה תתקבל בשפה. פרט לקבוצת המצבים מקבלים, לא נשנה דבר. כיצד נגיד פורמלית את קבוצת המצבים מקבלים?

$$F^B = \{p | \exists u \in \Sigma^* : \delta^{A^*}(p, u) \in F^A\}$$

כלומר, זו תהיה קבוצת כל המצבים p , אשר קיימת מילה שתוביל אותנו למצב מקבל ממה, ככלומר היא חלק מהמסלול שיביל למצב מקבל.

4.2 ביטויים רגולריים

אדם מגיע אלינו ברוחב, וננסה להסביר לו על שפה רגולרית. לא מבין. למה לא מבין? כי לא מכיר מה זה אוטומט סופי. האם יש דרך לתאר שפה רגולרית, ללא אוטומט סופי? התשובה היא כן, עליה נדברikut.

ביטוי רגולרי הוא דרך מקוצרת לתאר תבנית של מחרוזות. למשל, a^*ab מתאר את כל המילים שמתחלילות ב- a , מסתiemיות ב- b ובאמצע יש מס' סופי של a -ים.

דוגמה נוספת: מתאר את כל המחרוזות שפותחות ב- c , ולאחריה יש או aa או b . כלומר $\{caa, cb\}$. ודוגמה אחת נוספת נספთ, אותו דבר עם כוכב: $\{aa|b\}^*$ מתאר את כל המחרוזות שפותחות ב- a , ולאחריה מס' סופי בהם נבחר את aa או b . למשל $caabaaaab$ וכו'.

הגדרה: ביטוי r נקרא רגולרי אם הוא אחת מהצורות הבאות:

א. בסיס

σ כאשר $\Sigma \in \sigma$.

\emptyset , ϵ - גם הן, ביטויים רגולריים.

ב. פעולות שנייתן לביצוע:

1. r_1r_2 , באשר r_1, r_2 ביטויים רגולריים

2. $r_1|r_2$ באשר r_1, r_2 ביטויים רגולריים

3. r_1^* באשר r_1 ביטוי רגולרי. הערת, יתכן גם ויתבצע אפס פעמים, כי יתכן $\epsilon = r_1^*$.

4. באשר r_1 ביטוי רגולרי.

מינוח. סינגלטון = מילה עם אות בודד.

*יינכנו מס' ביטויים רגולריים לאוותה השפה.

הגדרה: יהי r ביטוי רגולרי. השפה של r תסומן $L(r)$ הינה:

$$L(r) = \left\{ \begin{array}{ll} \{\sigma\} & r = \sigma \\ \{\epsilon\} & r = \epsilon \\ \emptyset & r = \emptyset \\ L(r_1) \circ L(r_2) & r = r_1r_2 \\ L(r_1) \cup L(r_2) & r = r_1|r_2 \\ (L(r_1))^* & r = r_1^* \\ L(r_1) & r = (r_1) \end{array} \right\}$$

דוגמה. כך למשל, השפה של הביטוי $r = ab^*a \circ \{b\}^*$ הינה $\{a\} \circ \{b\}^*a$.

נשים לב כי, יתכן ובמהלך הבניה אנחנו נחבר קודם ab ואז נפעיל כוכבית, במקום לקחת את a ו- b בנפרד, ולהפעיל כוכבית על b . **לכן נגיד סדר פעולות**, כמו סדר פעולות חשוב:

א. סוגרים

ב. כוכבית

ג. שרשור

ד. איחוד

ה. משמאל לימין

דוגמה 1. כתבו ביטוי רגולרי לשפט כל המילים מעל $\{a, b\} = \Sigma$ שטכילות רצף bb .
פתרונות: זו פשוט, a או b בהתחלה, ובסיום, באמצע bb חיית, זה הכליה הניל -

$$L = (a|b)^* \circ (bb) \circ (a|b)^*$$

דוגמה 2. כתבו ביטוי רגולרי לשפט כל המילים מעל $\{a, b\} = \Sigma$ בהסבוקים הזוגיים מופיע b בלבד.

פתרונות: רמה גבוהה יותר. נשים לב כי ההנחה היא שככל מקוט זוגי יהיה b בלבד, וכן במקומות האוי זוגי מה שורצים. גם נראה כי אפסלוון בשפה.
נשים לב כי האות הראשונה תהיה a או b , ואילו האות השנייה - b בלבד. על ביטוי זה יוכל להזוז כמה פעמים שנרצה. וכך:

$$L = ((a|b) \circ b)^*$$

נשים לב שזה לא מספיק, כיצד המילה aba תתאפשר למשל? היא לא. נרצה לאפשר מילים באורך או זוגי, لكن השפה תהיה:

$$L = ((a|b) \circ (a|b|\varepsilon))^*$$

דוגמא 3. שפט כל המילים שאורכו מתחלק ב-3 או ב-2 ללא שארות.
פתרון: נשים לב כי נרצה לבחור או כלומר והו כמת' | בו שתי אפשרויות. כשאתה - אורך המילה 3, השנייה אורך המילה 2. לכן

$$L = (((a|b) \circ (a|b)) \circ ((a|b) \circ (a|b)))^*$$

דוגמא 4. שפט כל המילים שלא מסתיימות ברצף ba
פתרון: נפצל לנקודות. אם המילה בעלת אורך אחד - סכבה, שווה $\varepsilon|b|a$. אחרת - שהו רצף כלשהו של $b|a$. כאשר אנחנו רוצים סיטים רק מהאפשרויות $\{aa, bb, ab\}$. שום דבר כי אם נדרש רצף של $a|b$ ובסופו אחד מהשווים aa או b בלבד, נעה על הדורש. מזען aa יכנס ממש, bb יכנס עס b אחת מ- $(a|b)$ ואחת נוספת לסתום, ab בדומה עם a מה $(a|b)$ ו- b בסופו. כלומר הבינו היה -

$$L = (a|b|\varepsilon) | ((a|b)^* \circ (aa|b))$$

4.3 המרת אוטומט לביטוי רגולרי

טענה: תהי L שפה. אז

$$L \text{ רגולרי} \iff \text{קיים ביטוי רגולרי } r \text{ כך ש } L(r) = L$$

הוכחה:

ונכיח את הטענה שתראה לנו כיצד בהינתן אוטומט, ניתן להמירו לביטוי רגולרי.
 \Rightarrow נניח $L = L(r)$ ונוכיח L רגולרי. הוכחת צד זה לא קשה יותר מדי, מכיון באינדוקציה על $|r|$ כשהבסיס הוא על ההגדלה לעיל, ובצעד מנוחים שבבקרה קיבלונו זודל שגדל מחדך שהתבצע ע"י אחת מהמניפולציות: איחוד, ששורר, סגוררים וכובב. וכך נמיין סגירותם שליהם, והנחת האינדוקציה - הכל רגולרי זה ממש לא היה פורמלי, אני עצמן, סלחו לי, זה פשוט לא המתהה.
 \Leftarrow נניח L רגולרי, כולם, קיים אוטומט סופי דטרמיניסטי A עבורו $L(A) = L$.

נכיל את מושג האוטומט הסופי:

אוטומט סופי מוכפל: נאפשר לאוטומט כעת לא רק להיות לא דטרמיניסטי עם מעברי אפסיילון, אלא גם שעל הקשתות לא יהיו רק אוטומיות בודדות אלא גם ביטויים רגולריים על הקשתות. באוטומט כזה, ניתן לעבור בהינתן קלט שמתאים לביטוי. מה באשר לשאלת הקבוצה הריקה? עליה לא ניתן לעبور כלל כי אין אף מחרוזות שמתאימה לה. באוטומט כזה נדרש כמו דברים:
 א. יש מצב התחלתי בודד ייחיד ביל' כנסיות - נסמן s
 ב. יש מצב מקבל בודד ביל' יציאות - נסמן f
 ג. ישנה קשת בין כל שני קודקודים, כולל קשתות עצמאיות (חו"ץ מכניות ויציאות f) ובסופה).

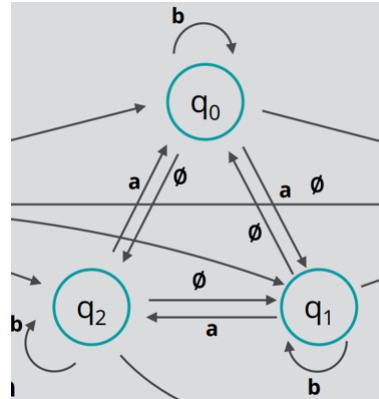
הקשות מסומנות בביטויים רגולריים.

- המטרה תהיה לעבורה בהינתן אוטומט רגיל, לאוטומט מוכלל. האלגוריתם יהיה כדלקמן:**
- מוסיפים מצב התחלתי חדש, s , ממנו מוסיפים מעבר אפסילון במצב התחלתי המקורי, וمبטלים את המצב התחלתי המקורי כמצב התחלתי.
 - מוסיפים מצב מוכל חדש, f , ומוסיפים ממנו מעבר אפסילון במצבים המקוריים, וمبטלים אותם כמצבים.
 - מוסיפים קשת של קבוצה ריקה, בין כל שני קודקודים שאמורה להיות בניהם קשת, אךCut איין.

מסקנה: לכל אוטומט סופי, A , קיים אוטומט מוכלל B כך ש($L(A) = L(B)$)

השלב הבא, יהיה לבצע דילול מצבים" - המטרה תהיה להגיע במצב בו יש לנו שני מצבים בלבד, s ו- f . ובניהם קשת אחת, עם ביטוי רגולרי, שהוא יהיה הביטוי השקול לאוטומט המקורי. כיצד נבצע דילול?

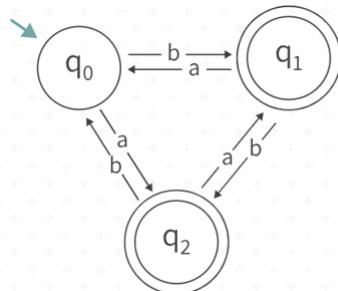
דוגמא לא תזקיק. נסתכל על האוטומט המקורי הבא. נראה כי אם נרצה למחוק את המצב q_1 למשל, עליו לטפל בשתי קשיות. $q_1 \rightarrow q_0$ עס "a", ולכן עצמאית של b -ים, וקשת נוספת $q_1 \rightarrow q_2$ עס ".ab". ככלומר, אם נרצה להעלים את המצב q_1 , נרצה להוסיף קשת מ- q_0 לש q_2 של הביטוי הרגולרי a^*a . האם זה מסוייך? כן. כיוון שלן שאר הקשיות שיוצאות מ- q_1 הם של הקבוצה הריקה. כמו כן, הינו נשים את הביטוי הרגולרי על הקשת $q_2 \rightarrow q_0$ עס הקבוצה הריקה.



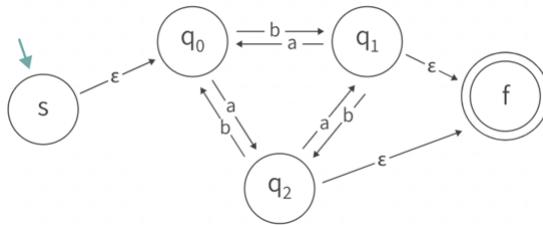
לאחר שביצענו דילול מצבים - נשארנו עם ביטוי רגולרי סופי, זהו הביטוי הרגולרי שמקיים $L(r) = L$.

דוגמא:

נרצה להמיר את האוטומט הבא, לביטוי רגולרי:

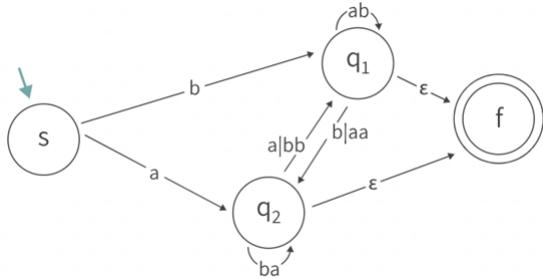


שלב א. נמירו לאוטומט מוכלל:

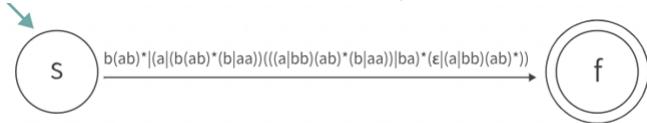


שלב ב. דילול מצבים:

נתחילה מדילול q_0 : נשים לב כי מ- q_1 ניתן לצאת עם a ולהזור עם b מ- q_0 ולכן נסיף לו לא q_1 עצםית עם ab . באופן דומה ba עם q_2 . כמו כן, יכול לצאת b מ- q_2 אל q_0 ומשם להמשיך לו ולכן בערך caa ממנו bb ובדומה.



לאחר דילול כל המצבים - קיבל את האוטומט:



והסדרוב הזה זה x .

4.4 למת הניפהו

אנו מתקדמים לפתרון הבעיה שלנו. האם לכל בעיית מוחשב קיים פתרון? כמובן, האם כל שפה הינה רגולרית? התשובה היא לא. כתע נמצוא כלי שיאפשר לנו להוכיח אירוגליות.

הлемה: תהי L שפה רגולרית. אז, קיים מס' $N \in \mathbb{N}$ כך ש $\forall w \in L$ אם $|w| > N$ אז קיים פירוק

- ג. $w = xyz$
- ב. $|y| > 0$.
- א. $|xy| \leq N$.
- ג. $xy^i z \in L$ לכל $i \in \mathbb{N}$ מתקיים

מדוע אנחנו עפין על הлемה? הлемה נותנת תנאי ששפה רגולרית מקיים. אם נמצא שפה, שלא מקיים את אחד התנאים, היא בהכרח אינה רגולרית.

דוגמה. נסתכל על השפה

$$L = \{w \in \{a,b\}^* \mid \#a_w = \#b_w\}$$

זו שפה שאינה גולרית. נב"ש שהוא כו וגולריות. לפי הلمה, קיוס N שפקיות את כל הזרוש. תהז $w = a^N b^N$ שהוא חלק מהשפה והוא $> N = |w|$. עבורה קיוס פירוק xyz פורק $x = a^k$, $y = a^j$ ו $z = a^l$ כך $|xy| \leq N$ ו $|y| < N$ ו $|yz| \leq N - j$ והוא $l = N - j$ כאשר $k + l = N$. אבל $y = a^j$ מופיע מ"מ בלב. כלומר, $l > 0$ כי $0 < |y| < N$ והוא $l > 0$ לפי תכונה 3, לכל i טבעי, $xy^i z \in L$, נסתכל על $i = 0$ (מסתכל שכקורת זהה אפס טכני? מורות, אפשר לחתך גס 2 = 2, אך נקבע כי $xy^0 z = xz = a^{j+l} b^N$, נשים לב כי $j + l = N$ וכן $0 < j + l < N$ ולכן $N \neq j + l$ וכן $\#a_w \neq \#b_w$, בסתיו לאמת הטעות. לנו $-L$ איננה גולרית.

הערה 2. אם שפה מקיימת את תכונת למת הניפוי, זה לא גורר שהיא רגולרית.
הערה 2. ראיינו כי שפה סופית היא רגולרית, ככלומר אוסף השפות הרגולריות מוכל באוסף השפות הסופיות. כמו שנשאש לב, L שבחרנו בדוגמה אכן איננה סופית. ככלומר – לא נוכל בחאים להוכיח כי שפה סופית, אכן איננה רגולרית, כי ההפק הוא הנכו.

מבנה הוכחה כללי לשאלות עם למת הניפוי: נניח בשיליה כי L רגולרית, יהא N שלא לבחירתו, נבחר w בבחינה שישichtet לשפה וכן $|w| > N$. נבחן פירוקים אפשריים שיקיימו את תנאים של הлемה, לפי 3 של הлемה נבחר $\mathbb{N} \in i$ ונגע לסתירה. נשים לב לשלב בחרית המילים שהוא חלק קרייטי בהוכחה, ונשים לב שבשלב הפירוק – הוא אינו נתון לבחירתו, וצריך לבדוק מראש את כל האפשרויות לפירוק) או לעבד חכם כמו בדוגמה מעלה, ולאחר מכן פירוק ייחיד).

דוגמה נוספת. נוכח כי שפת הפליארווטס מעלה א"ג $\{a, b\}$ איננה גולרית.
פתרון: נניח בשיליה כי L כו גולריות. אז לפי למת הניפוי קיוס N שיעונה על הזרוש. נבחר את המילה $w = a^n b a^n$, אכן, w היא פלייארווטס, וכן $n > 2N + 1$. לכן, קיוס פירוק $w = xyz$ כך $|y| \geq N$ ו $|x| \leq N$. מכיוון, בהכרח x וכן y מוככבים מ"מ בלב, ככלומר $i = 0$, $xy^i z \in L$, לפי למת הניפוי לפחות $k > 0$ ו $y = a^k$ וכך $x = a^j$ ו $z = a^l$. נקבע $i = 0$, $xy^0 z = xz = a^k a^l b a^N$ וכן $k + l = N - j \neq N$ לא שייכת לשפה, כי $i > 0$, מכך שקיים סתיו והשפה L איננה גולרית.

דוגמה 2. נוכח כי השפה $L = \{a^p : p \text{ prime}\}$ איננה גולרית.
פתרון: נב"ש כי L כו גולריות. אז, לפי למת הניפוי קיוס N שיעונה על כל הזרוש של הлемה. נבחר את המילה $w = a^p$ כאשר $p \geq N$ ראשווי (בזהות קיוס כזו, יש אנסוף מספרים ראשוניים). אכו $w \in L$. מתקיים כי $p \geq N = |w|$ וכן קיוס פירוק $w = xyz$ כך $|y| \geq N$ ו $|x| \leq N$. נסמן $x = a^j, y = a^k, z = a^{p-j-k}$. בהכרח מתקיים כי $j + k \leq N$ וכן $j > 0$.icut, לפי הлемה, לכל i מתקיים $xy^i z \in L$, בפרט עבור $i = 0$ נקבע $z = a^{p-k}$, $xy^0 z = xz = a^j a^{p-j-k} = a^{p-k}$, האם זה עוזר לנו? לא. זו לא שאלה טטודורית. נסה שוכ. נבחר $i = p + 1$. מה נקבע?

$$xy^{p+1} z = a^j a^{k(p+1)} a^{p-j-k} = a^{p-k+kp+k} = a^{p+kp} = a^p a^{1+k} \notin L$$

כיוון שאינו ראשוי, מכפלה של ראשוי במשהו אחר, תתו לא ראשוי. בסתיו.

דוגמה 3. נוכח כי השפה $L = \{a^i b^j c^k : i, j, k \geq 0 \wedge j = \max\{i, k\}\}$ איננה גולרית.
פתרון: נב"ש כי L כו גולריות. אז, לפי למת הניפוי קיוס N שיעונה על כל הזרוש של הлемה. נבחר את המילה $w = a^n b^n c^{n-1}$, מתקיים לפחות $k = n > n - 1 = i = n > n - 1 = r$. לכן, $j = \max\{i, k\} = n$ ו $|w| = 3n - 1 > N$. וכן $n > 0$ ו $|w| = 3n - 1 > N$. נבחן, לפי למת הניפוי, קיוס פירוק $w = xyz$ כך $|y| \geq N$ ו $|x| \leq N$. אנו וודעים כי בהכרח $x = a^l, y = a^r, z = a^{n-l-r} b^n c^{n-1}$. וכן $l + r \leq n$. מכיוון, $l + r > 0$ ו $l + r \leq n$, נקבע $i = 0$, $xy^0 z = xy$. בפרט עבור $i = 1$ נקבע

$$xz = a^l a^{n-l-r} b^n c^{n-1} = a^{n-r} b^n c^{n-1}$$

מתקיים כי $\{n - r, n\} = \max\{n - 1, n - r, n\}$, כיון ש- $xz \in L$. אם כן, כיון ש- $r > 0$, לא יוכל
כי הפעולה זו מתקינה, שכן $n - r = n - 0 = n$ אמ"מ $r = 0$, וכיון שהוא לא יכול להיות $1 - n$.
מכאן שינקל סטרוה כמשואה היל', והשפה L איננה רגולרית.

הערה מההובחה של הלמה: N של הלמה, הוא מס' המ מצבים של האוטומט. מה הרעיון בעצם? אם לוקחים מילה שאורכה גדול ממס' המ מצבים, יש מצב שאליו חזרים פעמיים, לפי עקרון שובך הינו. למעגל בתוך האוטומט נקרא u (בודדות ונוצר מעגל, מהסביר לעיל). לחلك לפני המעגל, נקרא x ולחחלק לאחרר המעגל נקרא z . על המעגל - אפשר לחזור כמה פעמים שרוצים, ולבן ניתן לנפח את המילה, וכל ניפוח על המעגל, יביא אותנו תמיד למצב מקובל.

4.5 הוכחת אי רגולריות באמצעות סגירות

הוכחנו הרבה על סגירות ונדחא להשתמש בתכונות הסגירות להוכיח אי רגולריות באמצעות דרך אחרת.
למשל, נדחה להוכיח כי:

$$L = \{w \in \{a, b\}^* \mid \#a_w \neq \#b_w\}$$

איןנה רגולרית. ובכן, נשים לב כי $\bar{L} = \{w \in \{a, b\}^* \mid \#a_w = \#b_w\}$, שפה זו איננה רגולרית
כפי שראינו כבר מעלה. כיון ש- \bar{L} איננה רגולרית, גם L איננה רגולרית. (כיון שגם L רגולרית, גם \bar{L}
רגולרית. זהה).

זוגמה 1. נגדיר את הפעולה $perm$ (פרמוטציה) על השפות הרגולריות כשפה של כל המילים w , כך
שקיים $y \in L$ שהוא פרמוטציה של w .
הוכח הף: אם L רגולרית, היא סגורה תחת פעולה $perm$ (כלומר $perm(L)$ רגולרית.
פתרון: האינטואיציה היא הרכה. פה, בהינתן כל שפה, כל הפרמוטציות על המילים היא גם שפה
רגולרית? שמע שיטות. ואנו שוטות.
נכ"ש כי אם L רגולרי אז גם $perm(L)$ רגולרי. נבחר $w = (ab)^*$
ישים לב כי

$$perm(L) = \{w \mid \#a_w = \#b_w\}$$

ואנו קודם לכו, כי שפה זו איננה רגולרית. בסתיו.

בדאי לזכור - השפה $L = \{w \in \{a, b\}^* \mid w = a^n b^n\}$ איננה רגולרית.

נשים לב, אם $L_1, L_2 = \bar{L}$ אינן רגולריות - יתכן כי $L_1 \cup L_2$ רגולרית, למשל:
נקבל $L = \Sigma^* \cup \bar{L} = \Sigma^*$ שהוא רגולרי (אוטומט עם מצב יחיד - ללא עצמיה עם כל
הא"ב).

הוכח/הפרק: איחוד אגסוז שפות רגולריות היא שפה רגולרית. נפרק -

$$L_1 = \{\varepsilon\}, L_2 = \{ab\}, L_3 = \{aabb\}, \dots, L_n = \{a^n b^n\}$$

בכל שפה מילה אחת, لكن ניתן לבנות אוטומט שלד, והיא רגולרית, עם זאת האיחוד הוא $\{a^n b^n\}$ שאינו רגולרי.

◊ נשים לב, ביטוי רגולרי ללא אופרטור קלין * הוא ביטוי שמייצג שפה סופית, אופרטור קלין מייצג לולאה אוטומט, אם אין לולאה עצמית אוטומט, שפטו סופית.

5 יחידה 6: שפות חסרות הקשר

כעת נגדיר את המושג שפות חסרות הקשר, ובהמשך נפתח מודל חישובי - מכונת טיריניג, שתפתור את הבעיה עבור שפות אלו. בדומה למה שעשינו עם אוטומט ושות רגולריות.

5.1 דоказך חסר הקשר

נתבונן בדוגמה. נרצה לראות כיצד ניתן לייצג את אוסף המחרוזות התקיינות של השעון (נספור עד 21 בבלב). ככלומר, 08 : 11 תקין, ואילו 94 : 28 לא תקין. יכולנו לייצג זאת באמצעות אוטומט סופי, נראה דרך אחרת:

$$T \rightarrow H : M$$

כאשר T זה הזמן, וכן H = Hours

$$M \rightarrow LD$$

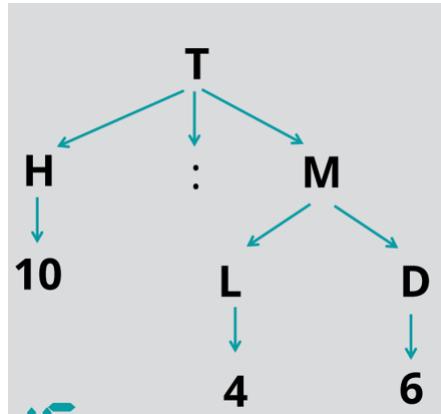
$$D \rightarrow 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$$

$$L \rightarrow 0, 1, 2, 3, 4, 5$$

כאשר D מייצג את ספרת היחידות, ו L את ספרת העשרות.

$$H \rightarrow D|10|11|12$$

ככלומר H הוא ספרה בודדת, או אחד מהספרות 10, 11, 12. העץ הבא ייצור את המחרוזות 49 : 6 :



כך יזכירנו, מחרוזת תקינה בצורה היררכית. **כללים אלו** נקראים **כלי יצירה/כלי גזירה**. העץ המיציג את המחרוזות, נקרא **עץ גזירה**. את העץ אנחנו קוראים משמאלי, למטה, לימין, כלפי מעלה.

דוגמה נוספת: בעיית הסוגרים המאוזנים. נגדיר **כלי יצירה** הבאים:

$$\begin{aligned} P &\rightarrow () .1 \\ P &\rightarrow PP .2 \\ P &\rightarrow (P) .3 \end{aligned}$$

נראה למשל, תהליכי קבלת המחרוזת (((())) יהיה -

$$P \rightarrow (P) \rightarrow ((P)) \rightarrow (((())))$$

שפה השעות חמודה, אך סופית ולא מעניינת. בהמשך נראה כי כל שפה רגולרית ניתנת לתיאור באמצעות **כלי יצירה**. אבל נרצה יותר - האם יש שפות לא רגולריות שניתן לתארן באמצעות כללי יצירה? כן:

דוגמה. נתבונן בשפה $\{a^n b^n | w = a^n b^n\} = L = \{w \in \{a, b\}^* | w = a^n b^n\}$. היא איננה רגולרית, ראיינו זאת כבר. נסתכל על **כלי יצירה** הבאים:

$$\begin{aligned} P &\rightarrow \varepsilon .1 \\ P &\rightarrow aPb .2 \end{aligned}$$

כך למשל, קיבל את $a^3 b^3 : a^3 b^3$

למערכת זו, הכוללת סימון (P) וכלי יצירה, נקראת **דקדוק חסר הקשר**.

5.2 הגדרה פורמלית:

דקדוק חסר הקשר הינו ריבועייה $G = (V, \Sigma, R, S)$ באשר:

- א. R הוא אוסף כללי יצירה
- ב. Σ הוא הא"ב הסופי, נקראים גם טרמינלים.
- ג. V הוא קבוצת המסתנמים, אוסף השמות שנתנו באשר נתנו את כללי יצירה.
- ה. $A \in V$, כלל היצירה יהיה $z \rightarrow A \in (V \cup \Sigma)^*$.
- ד. S הוא הסימן ההתחלתי, $s \in V$, והוא השורש ממנו מתחילה את הגירה (שורש העץ).

5.3 גיירות

יהי $G = (V, \Sigma, R, S)$ דקדוק חסר הקשור, וכי $A \rightarrow z \in R$ (אך $A \Rightarrow z$) כלל יצירה של הדקדוק. לכל $uAv \in (V \cup \Sigma)^*$ נסמן $uzv \Rightarrow_G uAv$ את הגיירה של המחרוזת uAv בפי G .
עבור מחרוזות $(V \cup \Sigma)^*$ נסמן $u \Rightarrow_G^* v$ אם ניתן לעבור מ- u על ידי 0 או יותר גיירות של G .

5.4 שפת הדקדוק

יהי $G = (V, \Sigma, R, S)$ השפה של G הינה

$$L(G) = \{w \in \Sigma^* | S \Rightarrow_G^* w\}$$

כלומר, כל המילים Σ^* שנitinן לעבור אליהן מ- S (שורשן ע"י 0 או יותר גיירות של G).

הגדעה: נאמר כי שפה היא שפת הקשר, אם קיים דקדוק חסר הקשור כך ש-

לכן, השפות סוגרים "אווניות", $\{a^n b^n | n \in \mathbb{N}\}$ הן שפות הקשורות הקשר, כי ראיינו שקיים עבורן דקדוק G כנ"ל.
מאייפה השפות האלו הגיעו? חוקר שפה חקרו וגילו שיש שפות רבות בעלות מבנה מחרורי ורקורסיבי, באגף בלשנות רצוי לפרט את המחרוזיות זו. אמונם המקור של שפות אלו בבלשנות, אך הן חשובות מאוד במדעי המחשב.

נשים לב - כל שפה רגולרית, היא בהכרח חסרת הקשר.

5.5 יצירות דקדוקים חסרי הקשר

נתבונן במס' דוגמאות, בהינתן שפה, ליצור דקדוקים חסרי הקשר.

דוגמה 1. כתוב דקדוק חסר הקשור לשפה הבאה:

$$L = \{a^x b^y a^z b^{x+y+z} | x, y, z \geq 0\}$$

נשים לב שהשפה שוקלה כתיבתה כך - $L = \{a^x b^y a^z b^z b^y b^x | x, y, z \geq 0\}$. מדובר נעדיף ככה לפטור את התרגילים? נראה רקורסיבי יותר.

$$S \rightarrow aSb|A$$

$$A \rightarrow bAb|B$$

$$B \rightarrow aBb|\varepsilon$$

זהו דקדוק חופשי הקשור.

דוגמה 2. כתוב דקדוק חסר הקשור לשפה הבאה:

$$L = \{a^n b^m \mid n \neq m\}$$

פתרונות: נראה כי $m \neq n$ פירושו $m > n$ או $n > m$. נרצה להגדיר כלל גזירה כדלקמן:

$$S \rightarrow aSb|aA|bB$$

$$A \rightarrow aA|\varepsilon$$

$$B \rightarrow bB|\varepsilon$$

מדוע זה עונה על הדרישת המחשבה היא כזו - תמיד נרצה לשמור על התבנית sa משמאלו ו- b מימין, אבל נרצה גם שתמיד לא יתקיים שוויון, כלומר S באמצעות יכול להיות רק תוספת של a -ים או b -ים, על מנת לדאוג שהשוויון לא ישמר. אם $m > n$, נרצה להוסיף aA , בתוך A נוכל לבחור אם להוסיף עוד a או שלשים אפסיילון. בדומה, אם $n > m$ נרצה להוסיף bB , ושם בתוך B נוכל לבחור אם להוסיף עוד b או שלשים אפסיילון.

דוגמה 3. כתוב דקדוק שפת הקשר לשפת כל המחרוזות מעל $\Sigma = \{a, b\}$ שאינן פליינדרום. כלומר $L = \{w \mid w \neq w^r\}$.
פתרונות: נרצה להתחיל כמו בפלינדרום, כלומר לאפשר aSa ו- bSb . אבל, כיון שהוא לא פליינדרום, הסדר חייב להיות מופר. וכן בין ההפרות, לאחר שכבר קرتה ההפרה, אנחנו נאפשר כל דבר בא"ב באמצעות. לכן ככל היצירה יהיה

$$S \rightarrow aSa|bSb|bAa|aAb$$

$$A \rightarrow aA|bB|\varepsilon$$

דוגמה 4. כתוב דקדוק חסר הקשור לשפת כל המספרים המפוסקים מעל א"ב $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, , ,\}$. הבירהה - מספרים לא יכולים להתחיל ב-0. למשל, 12, 456, 678 בשפה. ואנו לא 3, 34, 02, 456 בשפה.

פתרונות: נגדיר שני משתנים,

$$A = 1|2|3|4|5|6|7|8|9$$

$$A_0 = 0|A$$

כאשר נרצה שתהייה הפרדה בין להתחילה עם 0 או שלא. נראה כי כלל הגזירה ההתחלתי יהיה פשוט: מילה באורך אחד, מילה באורך שניים, מילה באורך שלוש, או גדול משלוש - שיכלול רקורסיבית את S . כלומר:

$$S \rightarrow A|AA_0|AA_0A_0|S, A_0A_0A_0$$

זוגמה 5. כתוב דקדוק חסר הקשור לשפט כל המילים מעל $\Sigma = \{a, b\}$ שמכילו את המחרוזת $.abba$.

פתרונות: נראה כי בשפה גם $abba$, וכן כל צירוף של אותיות מימין או משמאלו יהיה תקין, וכך גם מושגנה A שהוא יכול להיות מה שנרצה מאחד הצדדים

$$S \rightarrow AabbaA$$

$$A \rightarrow aA|bA,\varepsilon$$

זוגמה 6. כתוב דקדוק חסר הקשור לשפה $L = \{w \in \{a, b\}^* | \#a_w = 2\}$

פתרונות: נרצה למשהו לאפשר לבדוק שתי אותיות a , ובניהם לאפשר או רצף b או קלום. כולם אפסיון, ופורמלית זה יראה כך

$$S \rightarrow AaAaA$$

$$A \rightarrow bA|\varepsilon$$

5.6 למת הניפוח לשפנות חסروفות הקשר

בדומה לממת הניפוח עבור שפות רגולריות, משתמש בلمת הניפוח להוכחת אי חסروفות-הקשר של שפה.

תהי L שפה חסروفת הקשר.

אזי, קיים N טבעי כך שכל $w \in L$ כך ש- $|w| \geq N$ קיים פירוק $w = tuxyz$ כך ש:

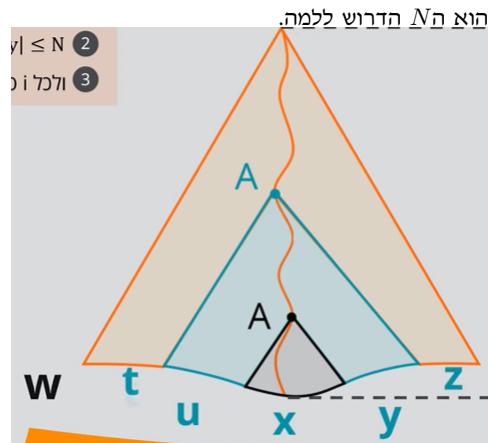
- א. $|uy| > 0$
- ב. $|uxy| \leq N$
- ג. $\forall i \in \mathbb{N} : tu^i xy^i z \in L$.

רעיון הוכחת הлемה: נסתכל על מילה w ונסתכל על עץ הגזירה שלה G . נבטיח כי גודל המסלול שלו $\leq |w|$. בהכרח, יהיה קודקוד (M שתנה) בעץ אליו נזרק פעמים. נסמן A . את החלק של תת העץ מהפעם השנייה של A מטה נסמן x , החלק מייננו ומתחתיו למטה העץ הגדל של A יקרא y ומשמאלו u . החלקים מעל A הראשון מיימן ומשמאלו בהתאם יהיו t ו- z . וכך - קיבלנו את הפירוק (בתמונה). כעת, כיצד נוכל $N \in \mathbb{N}$? באינדוקציה. נראה כי עבור $i = 0$ כל שצרייך הוא לחתות את החלק הכהיל, להוציאו החוצה, ולהעלות מעלה את החרורה, וזה עץ גירה תקין. מה עם $i = 1$? נוציאה את השחרורה, נוסיף פעם נוספת את הכהילו, ואז נחזיר את השחרורה. וכן הלאה - נוכל להוסיף את החתיכה הכהולה כמו פעמיים שנרצה, ומתחתיו את השחרורה. נראה כי אם נסמן

$$b = \max\{|u||A \rightarrow u \in R\}$$

אזי

$$N = b^{|V|+1}$$



זוגמה 1. הוכח הפרך - השפה $L = \{w \in \{a,b,c\}^* : w = a^k b^k c^k | k \in \mathbb{N}\}$ היא חסروفת הקשר.
הפרכה - נב"ש כי L חסروفת הקשר. אזי לפי הלמה לעיל, קיים n שיעונה על הדריש. נגיד $w = a^n b^n c^n$.
 מתקיים $n > |w| = 3n$, שכן קיים פירוק $w = tuxyz$ כך ש- $|uy| > 0$ וכן $|uy| \leq n$. נשים לב כי מהדרישה השנייה, בפרט לא יתכן כי uy ייחידי, ולכן חסירה לפחות אחת אות uy . כמו כן, מהדרישה $0 > |uy|$ בפרט יש שםאות אחת.

נגידר $i = 2$ וננפח z^2xy^2tu . זו מילה שנייפחנו אותיות מסוימות, ולא את כולם, ולכן התנאי לא ישרר, כלומר מס' a, b, c במחוזות יהיה שונה. מכאן, ש- L אינה חסרת הקשר.

דוגמא 2. הוכח הפרך - השפה $L = \{w \in \{a, b, c\}^* : w = a^i b^j c^k | i > j > k\}$ חסרת הקשר.

הפרכה - נב"ש כי השפה L חסרת הקשר. אז, לפי הולמה לעיל, קיימים n שעונה על הדורוש. נגידר $w = a^{n+2}b^{n+1}c^n$, אכן $w \in L$ וכנ $|w| = 3n + 3 > n + 2$. מכאןקיימים פירוק $w = tuxyz$ כך $|uy| \leq n$. נפצל לקרים -
א. uy מורכב ממשיים בלבד: נבחר $i = 0$, בהכרח מס' הא"ים ירד, ויהיה $\#a_w < n + 2$.
כעת, במילה החדשה עם הנינפה $\#a_w = n + 1$ או $\#a_w = n + 2$, כמו כן $i > j > k$ וקיים קטע יותר מ-2.
ובכלנו סטירה לדרישת $i > j > k$, פרטית יותר,

$$w' = tu^0xy^0z = txz = a^{n+2-|uy|}b^{n+1}c^n$$

כיוון ש- $|uy| \leq n + 2 - |uy| \leq n + 1$, בהכרח $i = 0$, בסטירה.
ב. uy מורכב ממשיים בלבד: ננפח עס $i = 2$. בהכרח $|uy| > 0$ וכנ קיימים שם b אחד לפחות.
כיוון שנייפחנו עבור מס' a -ים חיובי ממש, מס' b -ים יגדל-cut, ככלור $\#b_w > n + 1$ ושוב, סטירה לכך שמס' a -ים צריך להיות גדול יותר ממש.
ג. uy מורכב ממשיים בלבד: ננפח עס $i = 2$. בהכרח $|uy| > 0$, וכן קיימים שם c אחד לפחות.
נראה כי

$$w' = tu^2xy^2z = a^{n+2}b^{n+1}c^{n+|uy|}$$

כיוון ש- $|uy| \geq 1$, מתקיים $n + |uy| \geq n + 1$ ואז $j > k > n + |uy|$ בסטירה.
ד. uy מורכב ממשיים ואז b -ים: אנו יודעים כי $|uy| > 0$, וכן יש גם a -ים וגם c -ים ולכן ננפח עס $i = 0$. נקבל

$$w' = tu^0xy^0z = txz = a^{n+2-|uy|}b^{n+1-|uy|}c^n$$

כיוון ש- $|uy| \geq 2$, נקבע $\#a_{w'} = n + 2 - |uy| \leq n = \#c_{w'}$ והוא מס' a -ים היה קטן שווה במס' c -ים, בסטירה.
ה. uy מורכב ממשיים ואז c -ים: בדומה, יודעים כי $|uy| > 0$ וכן יש גם b -ים וגם c -ים ולכן ננפח עס $i = 2$ ונקבל $|uy| \geq 2$

$$w' = tu^2xy^2z = a^{n+2}b^{n+1+|uy|}c^{n+|uy|}$$

בדומה, נראה כי $|uy| \geq 2$, ולכן $\#a_{w'} = n + |uy| \geq n + 2 = \#c_{w'}$ בסטירה.
נשים לב שאלוי כל האפשרויות, לא ניתן רצף של שלושת האותיות כיוון ש- $n \leq |uy| \leq n$.
שבכל אחת מהאפשרויות קיבלנו סטירה, seh"כ לכל פירוק אפשרי נקבל סטירה. מכאן - סטירה למלמת הנינפה. השפה אינה חסרת הקשר.

דוגמא 3. הוכח הפרך: השפה הבאה חסרת הקשר $L = \{a^{i!} | i \in \mathbb{N}\}$ מעל $\Sigma = \{a\}$

הפרבה: נב"ש L חופשית הקשר. כלומר, לפי למת הניפוח, קיימים n שיענה על תנאי הלהמה. נגידיר את המילה $w = a^{n!}$, מתקיים $n! \geq |w|$, ולכן קיים פירוק $w = tuxyz$ כך ש $|uy| \leq n!$ וכן $|uy| > 0$. נסמן $k = |uy|$. מתקיים כי $n! \leq k \leq 1$. נסתכל על ניפוח עם $i = 2$. כלומר,

$$w' = tu^2xy^2z = a^{n!+k}$$

נרצה להראות כי $(n+1)! + k < (n+1)! + n! + k < n!(n+1) = (n+1)!$ קיבלו סטירה.

$$n! \leq n! + k \leq n! + n < n! + (n+1) \leq n!(n+1) = (n+1)!$$

וסה"כ הניפוח עם $i = 2$ הוא $L \notin i$, וזה סטירה לлемית הניפוח.

5.7 סגירות לשפות חסרות הקשר

טענה: השפות חסרות הקשר סגורות לאיחוד, שרשור וסגור קלין.
 ♥ אין סגירות לחיתוך ומשלים.
 ♥ יש סגירות לreverse וכן prefix לפי תרגיל מהקמפוס.

5.7.1 סגירות לאיחוד

תהיינה L_1 ו- L_2 שפות חסרות הקשר. אז, לפי ההגדרה, קיימים דקדוקים חסרי הקשר $G_1 = (V_1, \Sigma_1, S_1, R_1)$, $G_2 = (V_2, \Sigma_2, S_2, R_2)$. $L_2 = L(G_2)$ ו- $L_1 = L(G_1)$ כך ש $(V_1, \Sigma_1, S_1, R_1), G_2 = (V_2, \Sigma_2, S_2, R_2)$ ו- $L(G_U) = L_1 \cup L_2$ שקיים $G_U = (V_U, \Sigma_U, S_U, R_U)$. בה"כ נניח כי $V_1 \cap V_2 = \emptyset$ ו- $S_U \notin V_1 \cup V_2$. מדוע נוכל להניח זאת? אחרת נוכל להתחילה גירה עם כללי יצירה של דקדוק אחד ולהמשיך עם כללי הדקדוק השני, וכך נוכל ליצור גירה עם כללי השפה S_U . ואם יש חפיפה בשמות המשתנים, נשנה את שמות המשתנים. נגידיר -

- א. $\Sigma_U = \Sigma_1 \cup \Sigma_2$.
- ב. $V_U = V_1 \cup V_2 \cup \{S_U\}$.
- ג. S_U - התחלתי.

ד. $S_U = R_1 \cup R_2 \cup \{S_U \rightarrow S_1|S_2\}$. כלומר איחוד כללי היצירה של שתי הקבוצות פולס כל כללי היצירה מהמצב התחלתי לאחד מהמצבים התחלתיים.

5.7.2 סגירות לשרשור

תהיינה L_1 ו- L_2 שפות חסרות הקשר. אז, לפי ההגדרה, קיימים דקדוקים חסרי הקשר $G_1 = (V_1, \Sigma_1, S_1, R_1)$, $G_2 = (V_2, \Sigma_2, S_2, R_2)$. $L_2 = L(G_2)$ ו- $L_1 = L(G_1)$ כך ש $(V_1, \Sigma_1, S_1, R_1), G_2 = (V_2, \Sigma_2, S_2, R_2)$ ו- $L(G_{\circ}) = L_1 \circ L_2$ שקיים $G_{\circ} = (V_{\circ}, \Sigma_{\circ}, S_{\circ}, R_{\circ})$. נגידיר -

- א. $\Sigma_{\circ} = \Sigma_1 \cup \Sigma_2$.
- ב. $V_{\circ} = V_1 \cup V_2 \cup \{S_{\circ}\}$.
- ג. S_{\circ} - התחלתי.

ד. $S_{\circ} = R_1 \cup R_2 \cup \{S_{\circ} \rightarrow S_1S_2\}$. כלומר איחוד כללי היצירה של שתי הקבוצות פולס כל כללי היצירה מהמצב התחלתי לשרשור הממצבים התחלתיים.

5.7.3 סגירות לsegueו קלין

תהיה שפה חסרת הקשר. אז, לפי ההגדרה, קיים דקדוק חסר הקשר $G_1 = (V_1, \Sigma_1, S_1, R_1)$ כך ש $L_1 = L(G_1)$. נזכיר $L(G_*) = L_1^*$ שקיימים $S_* = (V_*, \Sigma_*, S_*, R_*)$ ו $\Sigma_* = \Sigma_1$. א. $V_* = V_1 \cup \{S_*\}$. ב. $S_* = S_1$ - התחלה. ג. $S_1 S_* | \varepsilon$. ד. $S_* \rightarrow S_1 S_*$ מה קורה כאו? שרשור שהולך מ S_* אל $S_1 S_*$, בסוף ה מסתים באפסילון, וכל S_1 הוא מילה כלשהי מ L_1 , וסה"כ נקבל שרשור מס' סופי של מילים מ L_1 .

5.7.4 אי סגירות לחיתוך

נסתכל על השפות הבאות:

$$L_1 = \{a^n b^n | n \geq 0\} \cup \{c\}^*$$

$$L_2 = \{a\}^* \cup \{b^n c^n | n \geq 0\}$$

L_1 היא איחוד של שפות חסרות הקשר, גם L_2 היא כזו, ולכן שתיהן חסרות הקשר.

נראה כי $L_1 \cap L_2 = \{a^n b^n c^n | n \geq 0\}$ שאינה חסרת הקשר, כי שהראיינו לפיה למota הניפור.

מסקנה: אין סגירות תחת משלים, כיוון ש $\overline{L_1 \cap L_2} = \overline{\overline{L_1} \cup \overline{L_2}}$, יש סגירות לאיחוד, נב"ש שיש סגירות למשלים, אז השפה מימין חסרת הקשר, ושוואה לשפת החיתוך, וגם היא חסרת הקשר, בסתייה.

מסקנות לסיום היחידה:

- א. אם שפה היא סופית, היא רגולרית, ואז היא בפרט חסרת הקשר.
- ב. אם שפה היא רגולרית, אז היא חסרת הקשר.
- ג. לא מתקיים כלל שפה רגולרית מוכלת ממש בתוך שפה חסרת הקשר, למשל $\{a^n\}^*$ רגולרית, ולא מוכלת ממש בשפה חסרת הקשר.
- ד. אם אחד המשתנים גורר את המילה הריקה בדקדוק, המילה הריקה לא בהכרח בדקדוק. למשל: $\varepsilon \rightarrow aA, A \rightarrow aA, A \rightarrow S$. במקרה, המילה הקצרה ביותר בשפה והיחידה בה היא a ולא המילה הריקה. ה. ידי G דקדוק חסר הקשר, תהי w מילה בשפה של G . "תיכנו שני עצי גזירה שונים עבור w .
- ו. תהי L מעלה א"ב $\leq \Sigma$. אז, אם L מקיימת את תנאי למota הניפור לשפות רגולריות, היא לא מקיימת את תנאי למota הניפור לשפות חסרות הקשר. כיוון שאם יש רק אחת בא"ב אז האפשרויות לחלוקת של המילה והניפור של המילה הם למעשה שוקלים בשתי הלמאות: בשתי הלמאות אנחנו מנפחים רצף של אותות כלשהיא (האות היחידה בא"ב) שאורךו גדול שווה ל 1 וקטן שווה ל- N (הקבוע של הלמה).

6 ייחידה 7: אוטומט מחסנית

ביחידה זו נזכיר את המודל החישובי, אוטומט מחסנית, שמקבל את השפota חסרות ההקשר. המודל הינו הרחבה של מודל האוטומט הסופי.

עד היום - התיחסנו לקלט שמאגי אוטומט שמובן מלאי. כתת נרחב - הקלט שאנו חנו קוראים על גבי האוטומט, נמצוא על מה שנקרא: **סרט הקלט**. סרט הקלט מחובר לאוטומט. האוטומט קורא מהסרט את האותיות אחת אחרת משמאלו לימין. באוטומט סופי, סרט הקלט הינו *read-only*. כמובן, אפשר לקרוא ממנו ואי אפשר לכתוב עליו. העבודה שלא ניתן לכתב על האוטומט - היא החולשה של האוטומט סופי.

לכן, כאשר רצינו להכريع שפה כמו $\{w \in \{a,b\}^* | w = a^n b^n\}$ לא יוכלו לעשות זאת עם אוטומט סופי, כיון שנדרשו לזכור. לשם כך - קיבל את המחסנית.

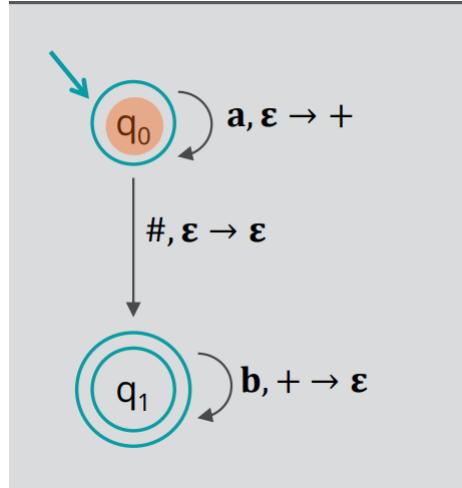
המחסנית היא רכיב זכרו שני תווים גם לכתוב בו, וגם לקרוא ממנו. המחסנית עובדת בשיטת *LIFO*. לאחר שנכנס, ראשוני שיצא. אם האוטומט רוצה, הוא כותב תוו בראש המחסנית ומכניס אותו, והוא יכול גם לקרוא תוו מראש המחסנית. אם המחסנית ברגע ריקה, יש שם ϵ בראש. וכך:

כאשר נרצה לדוחף אותן קלט, נבצע $\sigma \rightarrow \epsilon$.

כאשר נרצה לא לעת במחסנית, נבצע $\epsilon \rightarrow \epsilon$.

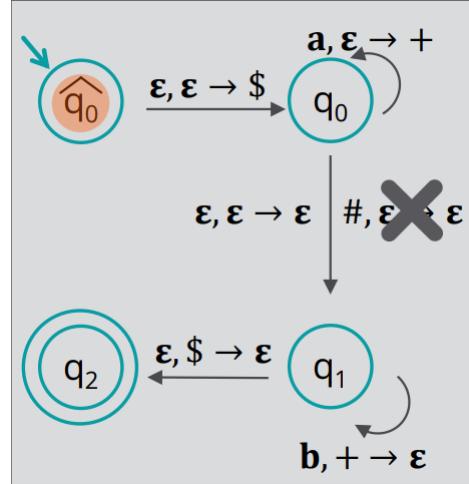
כאשר נרצה למחוק אותן מראש המחסנית, נבצע $\epsilon \rightarrow \sigma$.

דוגמא 1. נרצה לבנות אוטומט מחסנית עבור השפה: $L . L = \{w \in \{a,b\}^* : w = a^n \# b^k | n \geq k\}$ אינה רגולרית. נשים לב שאנו נדרש לזכור, בשליל לכור כמה a -ים היו עד כה. כיצד נסמן את המחסנית על גבי האוטומט? הסימן $+ \rightarrow \epsilon$ אומר: קח את האפסילון שיש ברأس המחסנית, שים שם פלוס. כמובן, תכניס פלוס בראש המחסנית. מה הסימן $\epsilon \rightarrow \epsilon$ אומר? אל תעשה כלום עם הקלט שבראש המחסנית. והסימן $\epsilon \rightarrow +$? קח את הפולוס בראש המחסנית, תחליף אותו באפסילון - כמובן, תוציא את הפולוס. זה הכל. מה יקרה כאשר נכנס עם המילה $a \# b b$? הקלט יגיע למצב q_1 עם אותן ϵ , וידרש להוציא b ממחסנית, אך המחסנית ריקה - ולכן נקבל שגיאה. כמובן: החישוב ית��ע.

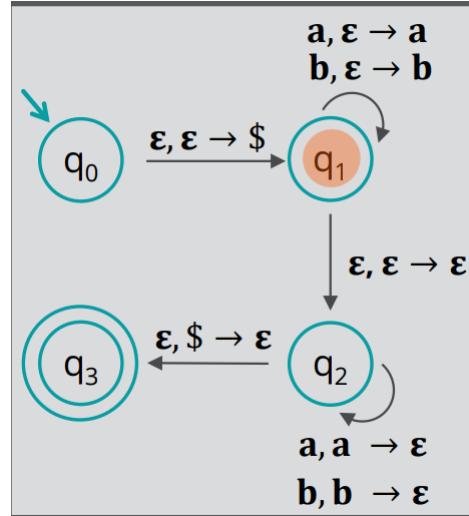


דוגמא 2. נרצה לבנות אוטומט מחסנית עבור השפה $L = \{w \in \{a,b\}^* | w = a^n b^n | n \in \mathbb{N}\}$. הרעיון יהיה כדלקמן: בדוגמה הקודמת - ידעו לדרוש מס' a גדול ממש מס' b . כתת נרצה לבדוק אותו מספר. הרעיון הוא זהה - אנחנו יודעים שкриיאת a דוחפת $+$. קרייאת b תוציא $+$. אם היה מס' שווה - המחסנית תהיה ריקה. כיצד נדע שהיא ריקה? התו הראשון שנדוחף יהיה $\$$. כך נתחיל את הקלט שלנו, במשיים את הקלט, אם בראש המחסנית יש $\$$, אז במס' a היה שווה למס' b , ולכן כת בראש המחסנית $\$$ וווציא אותו ונקבל את המילה. למעשה $\$$ יעזר לנו להתגבר על חוסר

הידיעה של מצב המחסנית. המעבר בין q_0 ל q_1 יהיה באמצעות מעבר אפסילון - בין רצף הא'-ים לרצף ה-ב'-ים.



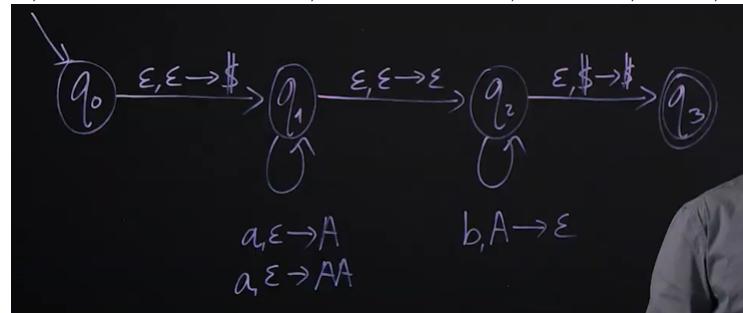
דוגמה 3: נרצה לבנות אוטומט מחסנית לשפה $L = \{ww^R | w \in \{a,b\}^*\}$. היא איננה רגולרית. הרעיון יהיה כמו בדוגמה הקודמת, נוציא $\$$ כמו קודם. באשר למעברים: אם קראנו a , נדוחו a למחסנית. אם קראנו b נדוחו b למחסנית. כשנרצה, נוציא מעבר אפסילון. אם קראנו a נוציא a מהמחסנית, ואם קראנו b נוציא b מהמחסנית.שוב, אם קיבלנו מחסנית ריקה - המחרוזות קייזו זה את זה, ולכן המילה בשפה, כלומר קיבל $\$$ ומשם למצב מקבל. (הרוי, נזכיר שאם הכנסנו מילה למחסנית, אם נקרה אותה מראשת המחסנית תחתית זה בדיק אבל בבדיקה reverseen).



דוגמה 4: נרצה לבנות אוטומט מחסנית עבור השפה

$$L = \{w \in \{a,b\}^* | w = a^n b^m : n \leq m \leq 2n\}$$

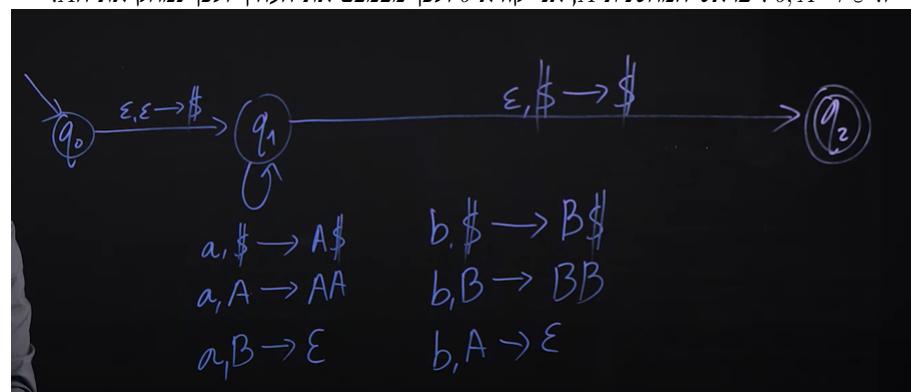
הרעין יהיה זה - מי בשפה? כל המילים שמתחלות ב a ומס' b שלם הוא לפחות מס' a , וכן לכל יותר פעריים מס' a -ים. כמובן, נסיף $\$$ בתחילת ובסוף, בשליל לדעת מתי התחתיות של המחסנית. כיוון שהאוטומט לא דטרמיניסטי: אפשר את ה"נירוש" או יותר נכון - הבחירה הלא דטרמיניסטיבית, לבין להכניס a לבין להכניס aa למחסנית. כאשר קיבל b נוצר a מהמחסנית. אם לאחר קריאת הקלט הגיעו ל\$, כלומר סיימנו את הקלט כנדרש - עברו למצב המקביל.



הוגה 5. נרצה לבנות אוטומט מחסנית עבור השפה

$$L = \{w \in \{a, b\}^* \mid \#a_w = \#b_w\}$$

הרעין יהיה זה: נתחל מ\$ ו גם נסיים אליו, בשליל לדעת מתי אנחנו בתחתית המחסנית. נרצה להוסיף סימן שיעזר לנו להבין מה המצב העודף שלנו כרגע. נשים לב כי יש 6 מצבים:
 א. $\$ \rightarrow A\$$. בראש המחסנית $\$$, ואני קורא a , יש לי עודף של a כרגע ולכן נרצה לזכור A בראש המחסנית.
 ב. בראש המחסנית A , אני קורא a , היתי בעודף של a , אני ממשיך בעודף של A ולכן נקרא שוב פעם a .
 ג. $\epsilon \rightarrow a, B$. בראש המחסנית B , אני קורא a ואני בעודף של B , لكن נמחק את הסימן של העודף B , ונמשך שם הלאה כי a קיא את B שצין את העודף על b ספציפי כלשהו.
 ד. $\$ \rightarrow b, B$. בראש המחסנית $\$$, אני קורא b ונכנס לעודף של B ולכן נכנס B .
 א. $b, A \rightarrow \epsilon$. בראש המחסנית A , אני קורא b ולכן מוצמצם את העודף ולכן נמחק את A .



- ***נעיר** - אוטומט המחסנית איננו דטרמיניסטי. יתכנו מסלולים שונים שיגמרו במקומות שונים. $L \in w$ אם קיים מסלול שמסתיים במצב מקבל.
- הערה** - את התוכן שבתוכן המחסנית, כותבים כך שהאות הימנית ביותר היא זו שבתחתיות המחסנית.

6.1 הגדרה פורמלית

הגדרה: אוטומט מחסנית הינו **שיישיה** $P = (Q, \Sigma, \Gamma, \Delta, q_0, F)$ באשר:

- Q היא קבוצת מצבים סופית.
- Σ א"ב הקלט הסופי.
- Γ א"ב המחסנית.
- Δ פונקציית המעברים:

$$\Delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow P(Q \times \Gamma^*)$$

בහינתן מצב, אותן בקלט, וסימון בראש המחסנית: הפונקציה קובעת מהו המצב החדש, והמחזזת שדווחים לראש המחסנית. האוטומט הינו לא דטרמיניסטי, ולכן הוא מחזיר קבוצה של מצבים אליהם ניתן לעבור. לכן מסמנים ב- P - קבוצת החזקה. הסימנים בהתאם הינם $\{ \varepsilon \} \cup \Sigma_\varepsilon \cup \Gamma_\varepsilon$ וכן $\{ \varepsilon \} \cup \Sigma = \Sigma_\varepsilon$. כיון שהאוטומט מאפשר מעברי אפסילון, הן בקלט והן במחסנית.

$$q_0 \text{ המצב ההתחלתי} \\ F \subseteq Q \text{ קבוצת מצבים מקבלים באשר}$$

6.2 תהליך החישוב של האוטומט

קודם לכן ראיינו כיצד ניתן לחשב קלט על אוטומט מחסנית באופן פיזי ופשוות. בהינתן אוטומט מורכב וגדול יותר, נרצה לתאר את החישוב באופן מתמטי.

קונפיגורציה: בהינתן רגע באוטומט, נרצה לדעת שלושה דברים: מצב שני אני נמצא בו, קלט שנוצר לי לקרווא, ומה יש בתחום המחסנית כרגע. ופורמלית - $i(q, u, x) = (Q, \Sigma, \Gamma, \Delta, q_0, F)$ אוטומט מחסנית.

קונפיגורציה הינה שלשה (q, u, x) באשר $q \in Q, u \in \Sigma^*, x \in \Gamma^*$ כאשר q הוא המצב הנוכחי, u היא יתרת הקלט ו- x זה תוכן המחסנית. קונפיגורציה זו תמונה רגעית של האוטומט.

כעת, נוכל לתאר פורמלית - התקדמות צעד על האוטומט:

גירה: הינו C_1, C_2 אוטומט מחסנית. והוא C_1, C_2 קונפיגורציות של P . נסמן $C_1 \mapsto_P C_2$ כלומר, C_1 גורר את C_2 אם כשנמצאים ב- C_1 ניתן לעבור C_2 לע"י מעבר יחיד.

נרחיב את המושג: נסמן $C_1 \mapsto_P^* C_2$ כלומר, C_1 גורר כוכבית את C_2 אם כשנמצאים ב- C_1 ניתן לעبور C_2 לע"י 0 או יותר צעדים.

6.2.1 שפת האוטומט:

הו $P = (Q, \Sigma, \Gamma, \Delta, q_0, F)$ אוטומט מחסנית. השפה של P מוגדרת להיות

$$L(P) = \{w \mid \exists f \in F, u \in \Gamma^* : (q_0, w, \varepsilon) \mapsto_P^* (f, \varepsilon, u)\}$$

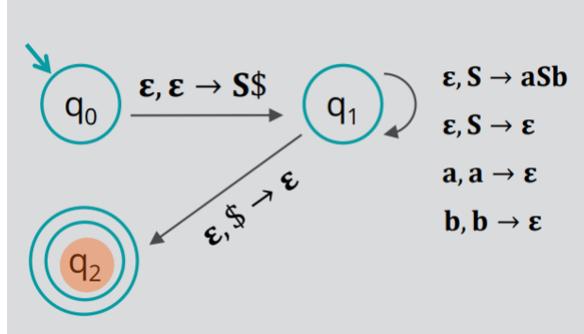
כלומר כל המחרוזות שניתן להתחיל מ q_0 , להגעה לpcb מכב, ככה שקרנו את כל הקטל, התחלו ממחסנית ריקה והסתמכו במחסנית כלשהי.

הערה - בנגדו לאוטומט סופי, באוטומט מחסנית אין שיקולות בין המודל הלא דטרמיניסטי למודל הדטרמיניסטי. ככלומר, ישן שפות שניתן לקבל במודל אחד ולא בשני. לכן הגדרנו את האוטומט כלל דטרמיניסטי כי עבר שפות הקשורות מסוימות, נדרש למודל הלא דטרמיניסטי.

6.3 אוטומט מחסנית שקול לשפה חסרת הקשר

טענה: שפה L הינה שפה חסרת הקשר אם ומ"מ קיים אוטומט מחסנית P כך ש $L = L(P)$.
מסקנה: כל שפה רגולרית הינה שפה חסרת הקשר. **כיוון** שבב אוטומט מחסנית הינו אוטומט סופי, אם לא נשתמש במחסנית.

הוכחה לטענה: נuir מראש שהוכחה כאן כיוון שבhocחה נгла אלגוריתם, בהינתן כללי דקדוק ← ליצירת אוטומט מחסנית.
 נראה כיצד בהינתן שפה חסרת הקשר, ניתן לבנות אוטומט מחסנית שייקבל אותה.
 נסתכל לדוגמה על השפה $\{N^n b^n \mid n \in \mathbb{N}\} = L$. רואה כי כל הזקוק ליצורה הימס $S \rightarrow \varepsilon$.
 לאוטומט המחשנית יהיו תמיד שלושה מצבים - לא משא מהם כלוי הזקוק.
 q_0 יהיה המצב ההתחלתי, q_1 באמצע ומייצג מצב q_2 .
 טו q_0 יצא מעבר ל q_1 בו נכניס $\$$ למחסנית. טו q_1 יצא מעבר בו נוציא $\$$ מהמחסנית. כמו כן, כל המעברים המעניינים באוטומט יתבצעו במעבר q_1 . הקשר והוא עם הפסחנית, רואה כי האוטומט עכשו השפה שלו יהווה:



נשים לב - תמיד וגם כאן, המעברים של q_1 :
 א. לכל כלל יצירה יתאים מצב בלולאה של q_1 - ללא קריאת קלט.
 ב. מטפלים בכל אותיות הא"ב כמו הדוגמה לעיל.
 נשים לב כי בתחילת, דחפנו $\$$. וכך במחסנית תמיד S כתנאי יצירה בסיסי, ממנו יוכל להתקדם.
 תמיד כאשר בראש המחסנית משתנה - נשתמש בכלל יצירה. כאשר אותן קלט - נשתמש בקריאה קלט.
 נשים לב - בדוגמה אעלו הרעיון הוא כזה: אם בראש המחסנית a וכן הקלט הוא a אוו יש התאמה אפשר למחוק את האות, נאוף דופה על b . כך מתקדים עד שהמחסנית מתרוקנת מכל היותר. לנוסף מחרוזת מתקצלת.

והרעיון הכללי הוא: מוחקים" את כללי היצירה, באמצעות דחיפת המשתנים אל המחסנית. כאשר בראש המחסנית משתנה, נפעיל כלל יצירה. כאשר בראש המחסנית אותן קלט נקרא את הקלט מסרט הקלט. תמיד נתחיל מ $q_1 \rightarrow q_0$ עם $\$S\$$. תמיד ב q יהיו כל כללי היצירה + אפשרות לקריאה והשווה עם כל אוויות הקלט, לפי התנאי $\epsilon \in a, a$ וכו'.

כעת, נראה כיצד בהינתן אוטומט מוחסנית, ניתן לבנות דקדוק חסר הקשר. \Rightarrow
נניח מספר הנחות לגבי האוטומט P .

א. נניח כי P יש מצב מוקל יחיד בשם q_{acc}

ב. נניח כי P מורקן את המחסנית לפני שהוא מקבל

ג. בכל מעבר, האוטומט בהכרח מבצע את אחת מן הפעולות הבאות (ורק אחת מהן):

1. הכנסה של אות בודד.

2. הוצאה של אות בודד.

הנחות אלו אינן מגבילות את כללויות הוכחה, מכיוון שכל אוטומט מוחסנית וודאי שקול לאוטומט מוחסנית שקיימים את כל התכונות הללו.

נתאר את רכיבי הדקדוק: G

משתני הדקדוק $V = \{A_{p,q} | p, q \in Q\}$ - כלומר לכל זוג קודקודים p, q של P יהיה משתנה בדקדוק $A_{p,q}$.

בහמץ, ונאה כי דקדוק בניו כך شامل המשתנה שכזה $A_{p,q}$ ניתן ליצור את כל (ורק) המחרוזות שבאוטומט P מעבירות מהצב p עם מוחסנית ריקה למצב q עם מוחסנית ריקה. (בדרכ ניתן לעבור ולזרקן את המחסנית ולעבור במצבים אחרים). לכן, נגדיר:

$$S = A_{q_0, q_{acc}}$$

בכללי היצירה יהיו שלושה סוגים:

א. לכל $p, q, s \in Q$ ישנו כלל היצירה:

$$A_{p,q} \rightarrow A_{p,s} A_{s,q}$$

ב. לכל Q $a, b \in \Sigma_\varepsilon$ ו $\sigma \in \Gamma$, $p, q, r, s \in Q$ אם מתקיים כי: $(q, \varepsilon) \in \Delta(s, b, \sigma)$ ו $(r, \sigma) \in \Delta(p, a, \varepsilon)$ אז ε י箝住 b על a .

$$A_{p,q} \rightarrow a A_{r,s} b$$

ג. לכל מצב $p \in Q$ ישנו כלל היצירה: $A_{p,p} \rightarrow \varepsilon$

6.4 חיתוך שפה רגולרית ושפה חסרת הקשר

טענה: יהיו C שפה חסרת הקשר ו R שפה רגולרית. אז $C \cap R$ הינה חסרת הקשר.

הוכחה: בה"כ הא"ב של השפויות זהה, אחרת ממילא מילימ לא יוכל להיות בחיתוך. נסמן א"ב Σ .

יהי $D = (Q^R, \Sigma, \delta^R, q_0^R, F^R)$ אוטומט סופי דטרמיניסטי המקבל את R .

יהי $P = (Q^C, \Sigma, \Gamma^C, \Delta^C, q_0^C, F^C)$ אוטומט מוחסנית המקבל את C .

נבנה אוטומט $U = (Q^U, \Sigma, \Gamma^U, \Delta^U, q_0^U, F^U)$ שייקבל את $C \cap R$.

נגדיר מרכיביו להלן:

$$Q^U = Q^R \times Q^C$$

$$q_0^U = (q_0^R, q_0^C)$$

$$F^U = F^R \times F^C$$

$\Gamma^U = \Gamma^C$
אננו משתמשים במחסנית עברו השפה C בלבד, ולכן
נדיר בעת את הפונקציה שלנו:

$$\forall q \in Q^R, p \in Q^C, \sigma \in \Sigma : \Delta^U((q, p), \sigma, \pi) = \{((\delta^R(q, \sigma), p', u) | (p', u) \in \Delta^C(p, \sigma, \pi))\}$$

$$\forall q \in Q^R, p \in Q^C : \Delta^U((q, p), \varepsilon, \pi) = \{((q, p', u) | (p', u) \in \Delta^C(p, \sigma, \pi))\}$$

הסביר: האוטומט הינו כמו אוטומט מכפלת. המעברים בין המ מצבים (שהם זוגות של מצבים מקוריים) ממש כמו באוטומט מכפלת. הפעולות על המחסנית תהיה של אוטומט P בלבד. בשורה הראשונה של הפונקציה, אלו מעברים עם קריאת קלט. זהו $\delta^R(q, \sigma)$ מעבר בשביל R והזוג (p', u) הינו עבור C , לדעת היכן לתקדם, באשר σ הוא הדופף למחסנית. הינו דטרמיניסטי שכן אין בו מעברי אפסילון, ובשורה השנייה מתיואר המצב הזה: ברכיב הראשון האוטומט מatkדム כמו באוטומט של R ובשני כמו באוטומט של C . סה"כ, האוטומט מקבל את החיתוך.

6.5 דקדוקים וגולריים וShapes רגולריות

כל שפה רגולרית כידוע היא שפה חסרת הקשר, כלומר קיימים דקדוק עבורה. נראה בעת כי לשפות רגולריות, יש מבנה מיוחד לדקדוק שיוצר אותה. דקדוק זה נקרא **דקדוק רגולרי** והוא מוחלקים לשני סוגים:

א. דקדוק רגולרי ימני

הכללים הם אחת מהצורות הבאות: $\forall A, B \in V, \sigma \in \Sigma$

$$\begin{aligned} A &\rightarrow \sigma B & .1 \\ A &\rightarrow \sigma & .2 \\ A &\rightarrow \varepsilon & .3 \end{aligned}$$

ב. דקדוק רגולרי שמאלי

הכללים הם אחת מהצורות הבאות: $\forall A, B \in V, \sigma \in \Sigma$

$$\begin{aligned} A &\rightarrow B\sigma & .1 \\ A &\rightarrow \sigma & .2 \\ A &\rightarrow \varepsilon & .3 \end{aligned}$$

טענה: שפה L רגולרית אם ומ"מ קיימים דקדוק רגולרי ימני או שמאלי כך ש $L(G) = L$.

הוכחה: ווכיח צד אחד, אם L רגולרית ונאה כיצד ניתן לבנות עברה דקדוק רגולרי ימני.

רגולרית ולכן קיימים אוטומט $L(A) = A = (Q, \Sigma, \delta, q_0, F)$ כך ש $L(A) = L$.

בננה דקדוק רגולרי ימני $L(G) = L(G) = (V, \Sigma^G, R, S)$ כך ש $L(G) = L$.

בננה דקדוק רגולרי ימני $L(G) = L(G) = (V, \Sigma^G, R, S)$ כך ש $L(G) = L$.

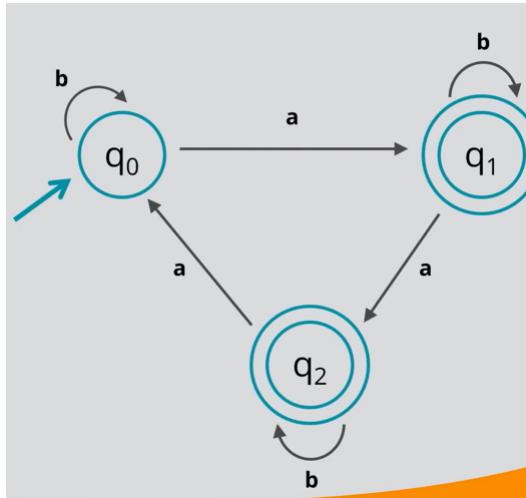
בננה דקדוק רגולרי ימני $L(G) = L(G) = (V, \Sigma^G, R, S)$ וכך נקבעת המצביעים כמשתנים. $S = q_0$.

בננה דקדוק רגולרי ימני $L(G) = L(G) = (V, \Sigma^G, R, S)$ וכך נקבעת העברות R .

1. לכל $p = p(\sigma, \delta)$ כלומר מ q עוברים עם σ למ'ן נגדיר את הכלל $p \rightarrow q$.

2. לכל $f \in F$ הכלל $\varepsilon \rightarrow f$ (מדוע? אם הגענו למצב מקבל, נוכל להפסיק עם אפסילון את הדקדוק כי זו מילה תקינה).

דוגמא. נתנו אוטומט הבא המיוצג בתמונה



נראה כי לכל מ McGrail אוטומט יהיה כל יצירה מתאימים. אוטומט ישו מעיר מ q_0 אל q_1 עס a ולען בזקוז נקל $aq_1 \rightarrow q_0$. בזופה, סה"כ נקל את הזקוז:

$$q_0 \rightarrow bq_0, q_0 \rightarrow aq_1, q_1 \rightarrow bq_1, q_1 \rightarrow aq_2, q_2 \rightarrow aq_0, q_2 \rightarrow bq_2$$

וכאשר למכנים המקרים, נקל $\epsilon \rightarrow q_1 \rightarrow \epsilon, q_2$. סה"כ זה זקוז חסר ההקשר המתאים לאוטומט.

מס' הערות על היחידה:

- א. אוטומט מחסנית לא מחייב שהמחסנית תהיה ריקה בסוף החישוב, ויתריה מאותה - יתכן שהיא ריקה לאורך כל הריצה.
- ב. כל דקדוק רגולרי, הוא בפרט דקדוק חסר הקשר.
- ג. מס' המשתנים בכל דקדוק שהוא, סופי.
- ד. אוטומט מחסנית דטרמיניסטי אינו שקול לאוטומט מחסנית לא דטרמיניסטי.

7.1 מוכנת טיריניג 7

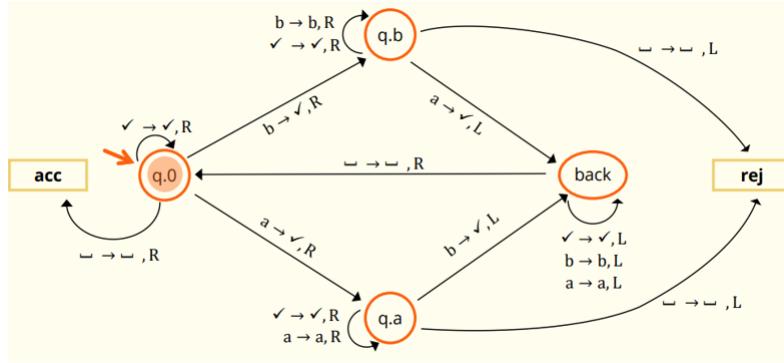
אנחנו ממשיכים במחקר שלנו סביבה השאלה: אילו בעיות ניתנות לפתרון ע"י מחשב?". ראיינו כבר כי יש שפות שאין אוטומט סופי והן אוטומטים מחסנית, לא מסוגלים לפתור. למשל השפה $w = \{w \in L^n \mid a^n b^n c^n\}$. די ברור, כי מחשב מסוגל להוכיח את השפה. מכיוון המשקנה היא שהבעיה במודלים שהצענו.icut נציג את המודל החזק מכולם. אכן טיריניג הוא אובי תורת מדעי המחשב - הוא פיתח מודל מתמטי שבמהלך השנים התבדר כמודול מרכזי שמתפרק מחשב: **מוכנת טיריניג**.

7.1 מהי מוכנת טיריניג?

כמו במודלים קודמים, הקלט יהיה על גבי סרט הקלט. בשונה ממודלים קודמים, ניתן לכתוב על המודול שלנו. **תמייד** ניתן לכתוב על המודול רק על הראש, בו אנו נמצאיםครגע. הרחבה נוספת ומשמעותית - במכנת טיריניג ניתן לזרז שמאליה. במכנת טיריניג אנו מנחים שסרט הקלט הוא אינסופי - בשני הצדדים. כאמור, גם אם התחלנו בנקודה כלשהי על קו הסרט, נזרז שמאליה ויש שם תווים, נזרז שמאליה

עד 69,420 צעדים, וגם שם יהיה תווים. לצורך הנוחות - אנחנו מניחים כי במקומות בהם לא כתבנו עוד מופיע התו "—" כלומר מCPF תחתון.

זוגמה 6. נתבגרו בשפה $\{w \mid \#a_w = \#b_w\} = L$. היא שפה חסרת הקשר. נרצה לראות כיצד המכונה תקבל את המילים בשפה: הרעיון יהיה פשוט - נתחיל בקירiatת המילה משמאלי, ונקרא אותה ראשונה. נסמן ✓ על האות שקראו, ונותנד בקלט ימייה לחפש b תואמת לה, נמצא אותה ונסמן בה ✓ גם כן. כעת נחזור לתחילה הקלט - מילג על ✓ המתאיםים ונוחש שוכ a או b תואם. נניח וממאננו ✓, נזכיר ונסמן ✓, ונמשיך בקורסית הקלט למליצאת a תואם וכו' הלאה. כך המכונה תעבד, בזרה מילולית. תהיי בסיסים נחזור לתחילה הקלט משמאלי. כיצד מעד של אותיות הקלט הוחלו? ✓? אם ורק אם תראה סירוקה של כל אוזור הקלט בה הכל הווה ✓, אז נוכל לסייע. متى נדע שהגענו לפחות לא טוב? כאשר בעהלך סירוקה הגענו לפחות שאון לה תואמת, ולכון הגענו לאוזור האסורה. וナルח לפiec מות. כיצד עבור זה את לתיוור גוף של מכונה?



ובכן, זוו מכונת הטירוג שמקבלת את השפה. נסביר קצת מה קורה כאן: a הוא מצב בו ראיינו a ומחפשים לו b תואם. בדומה, b הוא מצב בו ראיינו b ומחפשים לו a תואם. יהו מצב בו משתמש בשביי לחזור לקצה השמאלי של הקלט. ונסן שני מצבים מיוחדים: acc , $accept$, מצב זה קיבל פיליה. וש את המצב rej מפהילה $reject$. מצב זה זוכה מילה לגופרי. כאשר המכונה מגיעה לאחד המ מצבים - היא מפסיקת לפעול. בכל מצב אחר המכונה ברכבה ממשיכת.

כמו כן, יש את המצב ההתחלתי q_0 . בכל שלב במכונה: הינו עושה שני דברים - זהה לאנשeo, וכותבת משחו על הסרט. לנו עלינו לתאר שלושה דברים: מה הקלט שני רואה, מה נכתב על המכונה, ולאיזה כיוון זאת. לכן הסימנו R ל a מסמן: בהינתן שקרatoi a , שיש סש ✓ ותזה ומיינה. R מיינה ו L שפאללה.

חשוב - נשים לך, המכונה איננה יודעת שהגענו לקצה הקלט. נשים לך כי בהינתן קראה של מילה, המכונה לא יודעת שבתא אחד משאללה נגמר הקלט. ולכן מינעות הלולאות העצמיות $back$ של L ✓ → ✓. מה המשמעות? כאשר נרצה לחזור, אך שוכ שמאלה. אל תנסה כלום בסיטוון. ואז אנחנו הורה כבר במשמעותם רוח - וודע כי הגענו לאוזור לא טוב. וכשהגענו לאוזור הלא טוב, לאן נחזור? להתחלה. עם הסימנו R , ✓ → ✓ שימושית: ראות פקף, תשאיר אותו ולך מיינה. זה יכטיח לך שתגידי לאוזור הקלט.

כל עוד נראה ✓ נשאר ב q_0 עם לולה עצמיים כטובי. כי לא נרצה להתקಡ כי אין מה לזכור. אם אנחנו בפיגג $q.a$, אם נקרא a או ✓ נמשיך בקירה ימייה. רק b אנחנו מוחפשים. בדומה עבור המצב $q.b$ אם רואה b או ✓. כמו כן, בפיגג $back$ אם נקרא a או b , נרצה שייאשרו שס כי אנחנו רועים רק לחזור ולא לגעת בהם.

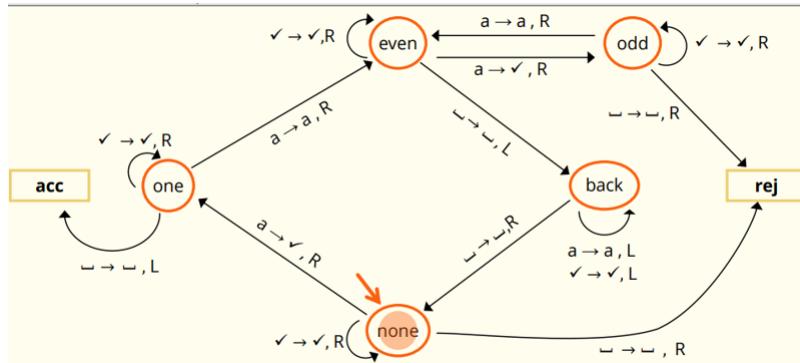
מתי נדע להגיאו למצב מתקבל? נראה כי יוצא מ q_0 מילג R , ✓ → ✓. משמעותו: אם אנחנו במאכל, וכל מה שעשינו היה לוו מיינה, או שלא נשאר קלט, וכן הגענו לסתו. אולי שאנו כבר קלט -

משעל המילה שלו חיבור להתקבל, וכן נגיע למצב מתקבל. מתי נגיע למילג? כאשר לא טוב לנו. כאשר אנחנו כב $q.a$ או כב $q.b$ וקיבלו -. משמע - אוון בזוג תואם למי ששמרנו שאנו חוצים למאכל לו בזוג תואם. במצב זה: יש לדוחו. אוון צריך להמשין.

זו הייתה דוגמה מורכبة - זה כל הרעיון. מודול חישובי זה חזק מאוד.

דוגמה 2. נרצה לנכונות מכונת טירוגינג שתקבע את השפה:

$$L = \{w = a^n \mid n = 2^k \wedge k \in \mathbb{N}\}$$



השפה L אינה גולרית, ואינה חסורת הקשר. נראה כי מכונת טירוגינג יודעת להכרע את השפה. הרעיון לבניית המכונה יהיה פשוט - מילה תתקבל אם מס' המ'ים שליה הוא חזקה של 2. הרעיון והו לעכור על קלט המילה ולמיחוק כל פעם a אחד, ואתה a אחריו להשאיר. כיצד נמיחוק? עם ✓. כך נמכו בפעם הראשון על הקלט, לפחות המילה a aaa \rightarrow ✓aaa. ואז, נמכו פעם נוסף על הקלט: שוב, פעם אחד נמיחק a ראשונה, ואת a אחרת ✓aaa. ומה עשוינו כאן בעצם? בכל פעם חילקו את מס' המ'ים פי 2, אם בסופו של דבר קיילתו מיצב בו נשאר a אחד בלבד, אז המספר המופיע של המ'ים היה חזקה של 2. מתי מילה לא בשפה? אם באיזשהו שלב, לפני שנגיע למ' האחרון, מס' המ'ים יהיה או זוגי, דבר שיפורע לרצף של אחד כו אחד לא.

נראה מה קורה במקרה: כרגע ישנו שני מצבים, acc ו- rej .
המצב $none$ הוא המצב ההתחלתי - עדין לא קראוו a בסביב סיריקה זה. המצב one משמשו שקראוו a בזוז. המצב $even, odd$ בהתאם כאשר קראוו מס' או זוגי/זוגי של a 'ים. המצב $back$ עכור חוזרת אחוריה בקליל.

נראה את המכונה בתמונה, שיענה בדיקות לתיוור לעיל. נראה כי אם הגיעו למצב one , או יותר אחד בלבד וכן המילה כו בשפה, כי חזקה של 2, וכן אם הגיעו לסוף הקלט - מלי' למצב acc . אם הגיעו למצב odd ולאחריו נגמר הקלט, מלי' rej כי המילה בהכרח לא בשפה אם מס' המ'ים או זוגי גדול מאשר, אפשרות נוספת להגעה rej היא אם אין a 'ים בכלל, כלומר מה המצב $none$. שכן 0 אינו טכני.

חשוב לציין - המכונה תמשיך לקרוא קלט כל עוד לא הגיעו לאחד משני מצביו העצירה: תמיד אנחנו זוזים על השרת, וכן תמיד כתבים עלי. גם אם לא רוצים לשנות מה שבכתוב, עדין יש כתיבה מחדש של מה שהיה קודם לכן.

7.2 הגדרה פורמלית

מכונות טירוגינג הינה שבעייה $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ כך ש:
 Q הינה קבוצת המצביעים, Σ הינו הא"ב, Γ הוא א"ב השרת הסופי, q_0 הוא המצביע ההתחלתי,
 δ הינו המצביע המקבל rej והוא המצביע הדוחה.
 acc הינה פונקציית המעברים המוגדרת:

$$\delta : (Q / \{acc, rej\}) \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

באשר הפונקציה מקבלת מצב ואות על הסרטן, ומחזירה מצב, אותן חדשה לשים בסרטן ולאיזה כיוון להתקדם.

7.2.1 קונפיגורציה:

קונפיגורציה נוטנת תיאור מתמטי ממזכה של כל הדברים שיש לדעת בעת התקדמות במכונה. מה עליינו לדעת בכל שלב? כיצד נראה הסרטן כת, באיזה מצב אנחנו נמצאים, ובאיזה חלק בסרטן אנחנו. כיצד נדע באיזה סרטן בחלק אנחנו? נניח ואנחנו עם הסרטן aab נמצאים במצב q_0 באוט a השנייה, אז הסימון aq_0ab יתאר כי כת אනחנו נמצאים באוט a השנייה, כלומר האות שלאחר המצב היא האות בה אנו נמצאים כת.

ופורמלית, תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ מכונת טירינגן. קונפיגורציה של M הינה מחרוזת w באשר $w \in \Gamma^*$ ו $\Gamma \in Q$ ו $\sigma \in \Gamma$ והא תוכן הסרטן שמאלי בראש, שאחורי יש אנסוף רווחים ו w הוא תוכן הסרטן מיימן לראש, שאחורי אנסוף רווחים. נשים לב כי $aq_0ab \sim aq_0ab \dots$ ככלומר הקונפיגורציות זהות, ניתן להוסיף כמה רווחים שנרצה לפני ואחרי. כל עוד מס' הרווחים סופי.

7.2.2 גיריה:

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ מכונת טירינגן ויהי C_1, C_2 קונפיגורציות של M . נסמן $C_1 \vdash_M C_2$ אם כשנמצביעים ב C_1 ניתן לעבור ל C_2 באמצעות צעד בודד. הכללה - נסמן $C_1 \vdash_M^* C_2$ אם כשנמצביעים ב C_1 ניתן לעבור ל C_2 באמצעות 0 או יותר צעדים.

7.2.3 קבלת ודוחיה של מחרוזות:

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ מכונת טירינגן. תהי $\Sigma^* w \in \Sigma$ מחרוזת. נאמר כי A מקבלת את w אם $\vdash_M^* u(acc)w$. באשר $u \in \Gamma^* \wedge \sigma \in \Gamma$ ו $u \in \Gamma^* \wedge \sigma \in \Gamma$ (כלומר, רק מעניין שהגענו למצב מקבל). לא מעוניין סרטן הקלטן. ב. דוחה את w אם $\vdash_M^* u(rej)w$. באשר $u \in \Gamma^* \wedge \sigma \in \Gamma$ (כלומר, רק מעניין שהגענו למצב דחיה). לא מעוניין סרטן הקלטן

נשים לב כי כאן, בניגוד למודלים קודמים, יש מצב דחיה. ומהו? המכונה המשיך לפעול עד אנסוף אם לא נעצר אותה. כאן, חישוב שאיןנו מקבל איינו בהכרח דוחה. וכך חיברים להגדר מה דחיה של מחרוזות, על מנת שמכונה לא תרוץ עד אنسוף.

7.2.4 הכרעה של שפה

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ מכונת טירינגן. ותהי $\Sigma \subseteq L$ שפה. נאמר כי M מכירעה את L אם לכל $w \in \Sigma^*$ מתקיים:
א. $w \in L \iff M \text{ מקבלת את } w$
ב. $w \notin L \iff M \text{ דוחה את } w$

מה אם יש מילוי שהמכונה לא עוצרת עכורים לא במצב דחיה, ולא במצב קבלה? כמובן - לולאו אינסופית על הקלטן, אז, המכונה לא מכירעה שום שפה.
ישנו מכונות טירינגן שלא מכירעות אף שפה.

7.2.5 קבלת של שפה

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ מכונת טירינגן. ותהי $\Sigma^* \subseteq L$ שפה. נאמר כי M מקבלת את L אם לכל $w \in \Sigma^*$ מתקיים:
א. $w \in L \iff M \text{ מקבלת את } w$

ב. $w \notin L \iff$ לא מקבלת את w
במקרה זה, נאמר כי $L(M) = L$

כאו, ה"זיהויה" הפכה ל"אי-זיהויה". מושג חלש יותר. לא מקלט = דוחה / לא מוגעה למצב קבלה.
כל מכונות טיורינג מקבלת שפה כלשהי, שהוא פשט שפת כל המילוטים שדרכו מגיעות ל'acc' במכונה.
זה אכן מודר הריבך.

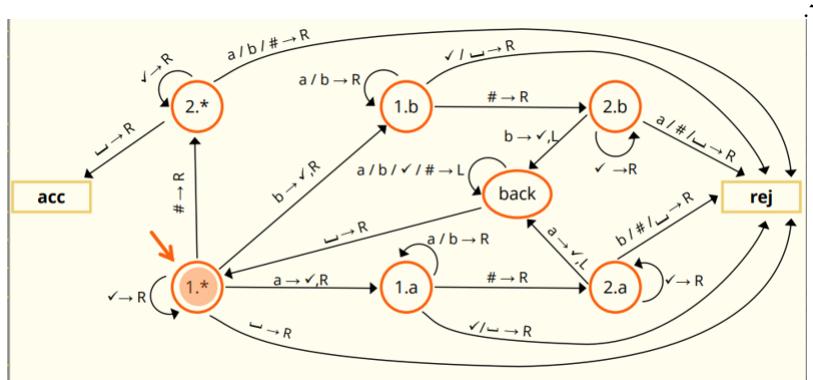
7.3 תיאור מכונות טיורינג באמצעות טבלת מעברים

כעת, לאחר שלמדנו הגדרות פורמליות - נלמד מס' טכניות וטריקים לבניית מכונות טיורינג.

דוגמה ראשונה - השפה $\{w\#w | w \in \{a,b\}^*\}$

נשים לב כי זו לא שפה רגולרית, ולא שפה חסרת הקשר. בקלות, לפי למת הנি�פוח, אם נבחר את המילה $a^n \# a^n = w$ שאורכה $1, 2n+1$, נקבל סטירה ע"י ניפוח עם $0, i$. עם זאת, נוכל לבנות לה מכונות טיורינג.

הרענון יהיה פשט - נתחילה בקורסית והסרת משMAL, נזכיר אם ראיינו a או b ונמחקה עם \checkmark . ממש נתקדם ונידלג על כל האותיות עד לאות הראנסונה שפונואה לאחר $\#$. אם הן שוות ממש, אם לא נלקח למצב דחיה. כך נמשיך עד שכ הסרת שלנו יהיה ריק, ואם אכן כך, המילה בשפה.
נשים לב כי אמרנו ממש גודל ומשמעותי - נזכיר שראיינו a , כיצד נזכיר? השתמש בתאי זכרון פנימי. בתא ראשון נזכיר באיזה צד של המחרוזת אנחנו, ובתא השני את אנחנו צרכיים לאזכור. כיצד ממשמשים את תא זכרון הפנימי? במצבי המכונה. לכן מצביו האוטומטי של המחרוזת $_1(w)$, ואנחנו צרכיים לציין שבתא השמאלי של הזכרון יש 1 (קרי, אנחנו בצד השמאלי של המחרוזת $_1(w)$, ואנחנו צרכיים לאזכור b . בדומה עבור שאר המצבים. המצב $*$, מציין שאנו בצד השמאלי של המחרוזת והתא ריק, כלומר $*$ מציין ריק. מכאן שהמצב ההתחלתי יהיה $1, 1$, שהרי אנחנו בצד השמאלי ואין צורך לאזכור דבר.



טבלת מעברים

ניתן לראות שמכונות הטיורינג הופכת למסריחה. בדוגמה הקודמת היו יותר מדי מצבים וייתר מדי מעברים. נרצה לתאר את מכונות הטיורינג באמצעות טבלה, מה שפשט את המודל הגրפי.
בහינתנו: מצב וסימון בסרטן. הטבלה תהיה: מצב חדש, כתיבה, תזוזה. לעומת מושג תיאור של פונקציית המעברים באמצעות טבלה. גם זו - דרך לתאר מכונות טיורינג.

באשר לדוגמה הקודמת, זה הייצוג הטבלאי, המתמטי והמדויר יותר של מכונות הטיורינג:

| מצב | סימן בסרט | מצב חדש | כתביה | позזה | |
|------|-----------|---------|-------|-------|------------------------|
| 1.* | σ | 1.σ | ✓ | R | $\sigma \in \{a, b\}$ |
| 1.σ | π | 1.σ | ↑ | R | $\pi \in \{a, b\}$ |
| 1.* | ✓ | 1.* | ↑ | R | |
| 1.τ | # | 2.τ | ↑ | R | $\tau \in \{a, b, *\}$ |
| 2.τ | ✓ | 2.τ | ↑ | R | |
| 2.σ | σ | back | ✓ | L | |
| 2.* | — | acc | — | R | |
| back | a ,b, ✓,# | back | ↑ | L | |
| back | — | 1.* | ↑ | R | |

$$L = \{w \in \{a, b, c\}^* | \#a_w = \#c_w\}$$

נראה כי אם נרצה לכתוב מודל גרפי שיפטור, יהיה לנו לפחות 40 מעברים. לכן נתאר בטבלה. נשים לב שהרעיון פשוט: בכל שלב נרצה לאזכור מי האותיות שראינו עד כה ביריצה ממשmaal לימיין, כאשר נגיד על קבוצה $\{a, b, c\}$ איזה שיש לנו את כל אותיות הקלט - וכך נחזור שמאללה. כאשר כל סרט הקלט מלא ברווחים וריק מקלט, איזה המילה בשפה. כאן בטבלה, S הינה קבוצה שיכולה להיות אחת מ8 הקבוצות $\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{c, b\}, \{a, b, c\}$

| מצב | סימן בסרט | מצב חדש | כתביה | позזה | תנאי | |
|-----------|------------|-------------|-------|-------|-------------------|--------------------------|
| q.S | σ | q.(S ∪ {σ}) | ✓ | R | $\sigma \notin S$ | $S \subset \{a, b, c\}$ |
| q.S | σ | q.S | ↑ | R | $\sigma \in S$ | $\sigma \in \{a, b, c\}$ |
| q.S | ✓ | q.S | ↑ | R | | |
| q.{a,b,c} | a, b, c, ✓ | back | ↑ | L | | |
| q.Ø | — | acc | ↑ | R | | |
| back | a ,b, c, ✓ | back | ↑ | L | | |
| back | — | q.Ø | ↑ | R | | |

כל השאר
עוברים
- rej

גם כאן, השתמשנו בזיכרון על מנת לתפעל את האוטומט. הזכרון אצלנו היה 8 מצבים - כל מצב זכר מה ראינו עד כה. **ניתן להשתמש בטכניקה זו רק כאשר מס' המ מצבים הינו סופי.**

$$L = \{x_1 \cdots x_k \# y_1 \cdots y_k \# z_1 \cdots z_k | x_i, y_i, z_i \in \{0, \dots 9\} \text{ ו } \forall i : x_i \geq z_i \geq y_i\}$$

בעת סריקת המחרוזות, علينا לזכור הפנימי מס' דברים: באיזה חלק של המחרוזות אני מבין החלושים x, y, z . בתחילת הסיריקה נזכר ספרות של x ושל y , ונבדוק אם z המתאימה נמצאת בינהם. הרעיון יהיה לסרוק ממשmaal לימיין את החלק הראשוני, לבחור את הספרה הראשונה ולזכור אותה בזכרון פנימי. כך נמשיך עד שנטקדם ל#. שם נמצא את ה z המתאים ונזכיר גם אותו בזיכרון פנימי. שם נמשיך עד ל# הבא ונמצא את ה z המתאים וגם אותו נזכיר בזיכרון פנימי. נבדוק את שלושת הספרות שלנו האם מקיימות את אי השוויון, ואם כן נחזור הלאה לתחילת הקלט.

כמובן שכל אות שקראננו תסומן ב✓. כאשר כל האותיות חוץ מהסמלויות יהו ריקות, נדע שהamilha צריכה להתקבל ושאכן הייתה התאימה. נראה כי יי'וג גרפי של מכונה זו תכלול המון מצבים. המון. ובטבלה: נעיר כי כל מצב יהיה עם שלושה>Status. יתאר את החלק בו נמצאים, האיבר השני בשלשה הוא תוכן תא האיקס, והשלישי הוא תוכן תא הוויא. * מסמל שעוד לא נקרה ספרה. נשים לב כי השורה החשובה ביותר בטבלהuesta היא שורת התנאי, אם ורק אם מתקיים התנאי $\tau_2 \geq \tau_1$ אז המילה התקבל. אחרת, במצב rej. איך התנאי היה בא ידי ביטוי במצבים "ui" אוטומט גרפי? אם היו ב.4.2.z ורצינו לקרוא 3, הינו חוזרים back כי אות חוקית, אם הינו מקבלים 9 הינו הולכים לדחיה כי היא לא בין ארבע לשתיים.

| מצב | מצב בסרט | סימן בסרט | מצב חדש | כתביה | תיזזה | תנאי |
|-----------------------------------|---------------|-----------------------------------|---------|-------|----------------------------------|------|
| X.*.* | σ | X.σ.* | ✓ | R | | |
| X.*.* | ✓ | X.*.* | ◻ | R | | |
| X.σ.* | 0,1,...,9,✓ | X.σ.* | ◻ | R | | |
| X.τ.* | # | Y.τ.* | ◻ | R | | |
| Y.τ.* | σ | Y.τ. σ | ✓ | R | | |
| Y.τ.* | ✓ | Y.τ.* | ◻ | R | | |
| Y.τ. σ | 0,1,...,9,✓ | Y.τ. σ | ◻ | R | | |
| Y.τ ₁ . τ ₂ | # | Z.τ ₁ . τ ₂ | ◻ | R | | |
| Z.τ ₁ . τ ₂ | ✓ | Z.τ ₁ . τ ₂ | ◻ | R | | |
| Z.τ ₁ . τ ₂ | σ | back | ✓ | L | $\tau_1 \geq \sigma \geq \tau_2$ | |
| Z.*.* | ◻ | acc | ◻ | R | | |
| back | 0,1,...,9,✓,* | back | ◻ | L | | |
| back | ◻ | X.*.* | ◻ | R | | |

כל השאר עוברים ל- rej

$$\begin{aligned} \sigma &\in \{0, \dots, 9\} \\ \tau &\in \{0, \dots, 9, *\} \end{aligned}$$

$$\tau_1, \tau_2 \neq *$$

7.3.1 תיאור מכונת טירינג באמצעות פסודו קוד

דרך נוספת לתיאור מכונת טירינג, היא באמצעות פסודו קוד. ניתן להשתמש בטכניתה זו רק כשברו שניתן למשתמש תיאור זה בטבלת מעברים, או שרטוט, פורמלים. נראה מ' דוגמאות:

- דוגמה ראשונה.** תאר מכונת טירינג שמכריעה את השפה $L = \{a^n b^n c^n | n \geq 0\}$ כמו שעשינו בעבר, מעבר איטרטיבי משמאלי לيمין ובכל שלב מורידים בהתאם באמצעות ו' a. מקבלים מילה אמ"ם כל הקלט שנותר הוא ו'. ובפסודו -
- א. כל עוד התו הוא ✓ - הוא את הראש ימינה.
 1. אם התו הנקרא הוא - קבל. (אין קלט)
 2. אם התו הנקרא הוא b או c - דחלה.
 3. אם התו הנקרא הוא a - כתוב במקומו ✓ והוא את הראש ימינה.
 - ב. כל עוד התו הנקרא הוא b או c או ✓ את הראש ימינה.
 1. אם התו הנקרא הוא c או - דחלה.
 2. אם התו הנקרא הוא b או ✓ כתוב במקומו ✓ והוא את הראש ימינה.
 - ג. כל עוד התו הנקרא הוא b או ✓ את הראש ימינה.
 1. אם התו הנקרא הוא a או - דחלה.
 2. אם התו הנקרא הוא c - כתוב במקומו ✓ והוא את הראש ימינה.
 - ד. הוא את הראש שמאליה עד לתו הראשון שמיini לרצף הרוחחים בתחילת הסרט.
 - ה. חזר לשלב א'.

דוגמה שנייה. תאר מכונת טירינג שמכריעה את השפה $L = \{a^j b^j c^{i+j} | i, j \geq 1\}$

נוצרך לחזור להגדרת הכפל - מהי ההגדרה של פעולת הכפל? $j \times i$ אומרו - בצע חיבור של j, i , $a^2b^3c^6$ נרצה על כל אות a , למחוק (לסמן וי) כמס' הפעמים שמופיע b . כיצד נעשה זאת? על כל אות a , נלק' לכל אות b וنمחק עבורה אחת מתאימה ב c , עד שלא ישארו אותיות b . ואז נוצרך להחזיר את אותיות b , למחוק את an האחת שבה השתמשנו, ולהתකם לאות הבאה bm , מתי נסיים? ככל שארו אותיות a , אנחנו יודעים שלא יותר בהתחלה וכן רק עליאנו לוודא שלא נותרו אותיות c בסוף הקלט. אם אכן לא נותרו a - המילה בשפה. נעריר כי אותיות b ימחקו עם סימון X למשל, בשביל שנדע לחזור לאחר מכן. והמחיקה של אותיות c יהיה עם V למשל. ובפודו:

- א. מחק אות a וכותב אותיות b . בשלב זה, אם האות האשינה אינה a , דחיה.
- ב. נסrok ימינה עד לאותיות b . בשלב זה, אם נתקל באות אחרת שאינה a או b נדחה.
- ג. כל עוד נותרו אותיות b שאין מחוקות:
 1. מחק אות b ראשונה וכותב X במקום.
 2. סרוק עד לאות c הראשונה, אם יש אותן מחק אותה וכותב V במקום.
 3. אם הגעת ל__ או a , דחיה
- ד. התקדם שמאלה עד לתחילת הקלט והחלף בדרך את כל אותיות X ב b .
- ה. חזר לתחילת הקלט, אם נותרו אותיות a חזר לשלב א'!

אחרון, התקדם לסוף הקלט. אם לא נותרו אותיות c קבל. אחרת: דחיה.

7.4 שימוש במכונות טיורינג לחישוב פונקציות

הברורה - הקורס עוסק בבעיות הכרעה. חישוב פונקציה היא בעיית חישוב. נהרג מנהל הקורס, ונראה כיצד אפשר לחשב פונקציות באמצעות מכונות טיורינג.

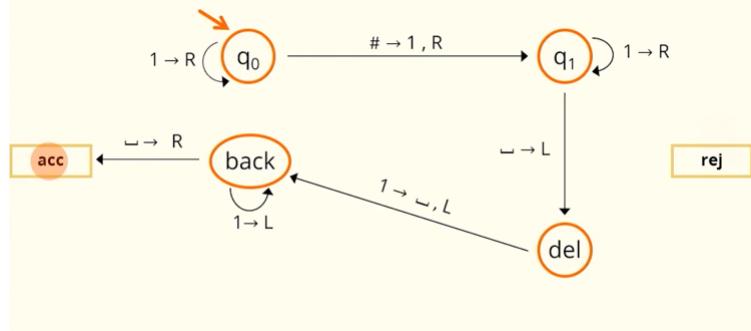
הגדרה: תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ מכונת טיורינג. ותהי $f : \Sigma_1^* \rightarrow \Sigma_2^*$. נאמר כי M מчисבת את f אם:

- א. $\Sigma = \Sigma_1, \wedge \Sigma_2 \subset \Gamma$.
- ב. לכל $w \in \Sigma_1^*$ מתקיים $accf(w) = rej$.

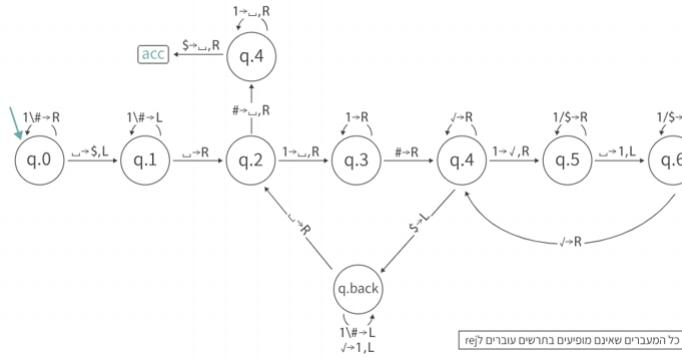
(הברורה - כל פונקציה שאנו מכירים תהיה חייבת להיות מקודדת בצורה מחרוזת).

נרצה לחשב פונקציות $N^k \rightarrow N^t$, אם כן נרצה לייצג באמצעות א"ב בשביל ש $\Sigma_2^* \rightarrow \Sigma_1^*$. f . כיצד נייצג את הפונקציות באמצעות א"ב? ביצוג אונרי. כך למשל: המספר 3 יוצג ע"י 111, המספר 7 ע"י 1111111 וכן הלאה. כמו כן $0 = \epsilon$. כיצד נפרק פרמטרים? באמצעות $\#$. למשל $(2,3) = 11\#111$.

דוגמא: יהיו מספרים $\mathbb{N} \in y, x$. כתוב מכונת טיורינג שמחשבת את הפונקציה $f(x,y) = x + y$. הרעיון יהיה פשוט: בגלל הייצוג האונרי, כל האחדות כתובות על הסרט. כל שיש לעשות הוא לחת את $\#$ שמספריד, לשים שם 1, ולהוריד 1 מהתו השופך וכך נקבל את התוצאה על הסרט שלנו.



דוגמא שנייה: יהיו מספרים $\mathbb{N} \in y, x$. כתוב מכונת טיורינג שמחשבת את הפונקציה $y \cdot x = f(x, y)$. הרעיון יהיה דומה מאוד לרעיון שבעצמו בכתיבת המכונה לשפה $a^i b^j a^{i+j}$ באשר אנחנו כעת רוצים לחשב את הפונקציה $f(1^i, 1^j) = 1^{i+j}$. עם זאת – זה חישוב ולא הרכעה ולכן יש לנו כמה הבדלים. מה יהיה הרעיון? נתקדם על הסרט עד סוף ונכתב תמיד \$. וכתע באופן איטרטיבי – נלך על האחדות של המסל' משמאל, בכל פעם נוריד אחת ממשם, ובהתאמה נ עבור על כל האחדות במס' השני שנמצא מימין יותר, ובזיג נעתיק אותם לאחר $\$$. כל אחד שסמננו, נשים עליו X למשל. וכך נדע מיהם האותיות בחלק השני של המספר. לאחר מכן החלק השני באיקסוי, נחזירו לאחורות. וכך נזרוק לחלק הראשון של המספר, שם נמשיך באופן איטרטיבי עד שלא יותרו ספרות בצד השמאלי של המספר, והפלט יזכה לנו לאחר סיכון $\$$. ובתיאור מכונה זה יראה כך –



- מס' הבהרות על היחידה:
- מכונת טיורינג יכולה להעביר מילה למצב מקבל, גם אם לא קראו את כל הקלט. למשל – כל המיללים שמתחלות בא, אין צורך להמשיך לקרוא פרט לתו הראשון.
 - השפה שמכונת הטיורינג מקבלת הינה יחידה.
 - יתכנו שתי קונפיגורציות $C_1 \neq C_2$ במהלך ריצת הקלט.
 - בבינתן מכונת טיורинг שמכירעה את L , יתכן שיש מכונת טיורינג אחרת שמקבלת את L ולא מכירעה אותה. למשל: $\{a\}^* = \{aw|w \in \{a\}^*\}$. בדור שהיא ניתנת להכרעה פשוטות. מצד שני, אפשר לבנות מכונה שבבינתן שהאות הראשונה a הולכים למצב מקבל ואחרת מתקדמים ימינה ללא הפסקה. היא תקבל את השפה, אך לא תכריע אותה כי יש מיללים שעבורן היא לא עוצרת.
 - במכונת טיורינג שמחשבת פונקציה, המכונה עוצרת בחלק השמאלי של הקלט.

8 ייחודה 9: וריאציות של מכונת טיורינג

8.1 מודל חישובי – הגדרה פורמלית

מודל חישובי: אוסף של מכונות שעוברות מוגדרים המושגים הרכעה וקובלה של שפות.

הגדרה: יהיו A, B מודלים חישוביים. נאמר כי A ו- B שקולים, אם לכל שפה L :

- קיימת מכונה במודל A שמכירעה את השפה L אם ומ"מ קיימת מכונה במודל B שמכירעה את השפה L .
- קיימת מכונה במודל A שמקבלת את השפה L אם ומ"מ קיימת מכונה במודל B שמקבלת את השפה L .

נשים לב, שקולות בין מודלים חישוביים היא יחס שקולות. (רפליקטיבי, טרנזיטיבי וסימטרי).

8.2 סרט ימינה בלבד

הבדל בין וריאציה זו למכונה הרגילה היא שהסרט הוא אינסופי רק בכיוון ימין, ולא בשני הכיוונים. מודל זה נראה מוגבל יותר, כיון שאין לו סרט בצד שמאל. במודל זה הקלט ממוקם בקצה השמאלי של הסרט וגם הראש נמצא שם. החישוב קורה אותו דבר פרט למספר אחד: אם הראש נמצא כבר במצב השמאלי, אנחנו נדרשים אז שמאלה - אז אנחנו נשארים במקום.

טענה: נסמן את המודל עם שני הכוונים באות (wo, T, O) . אז, O ו- T שולטים.

וככה:

א. נראה שלכל מכונה במודל O יש מכונה במודל T

$$M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, acc^T, rej^T) \text{ מכונת טיריג'ינג למודל } O. \text{ נבנה } T \text{ שקופה במודל } M_o = (Q^o, \Sigma^o, \Gamma^o, \delta^o, q_0^o, acc^o, rej^o) \text{ מכונת טיריג'ינג למודל } O.$$

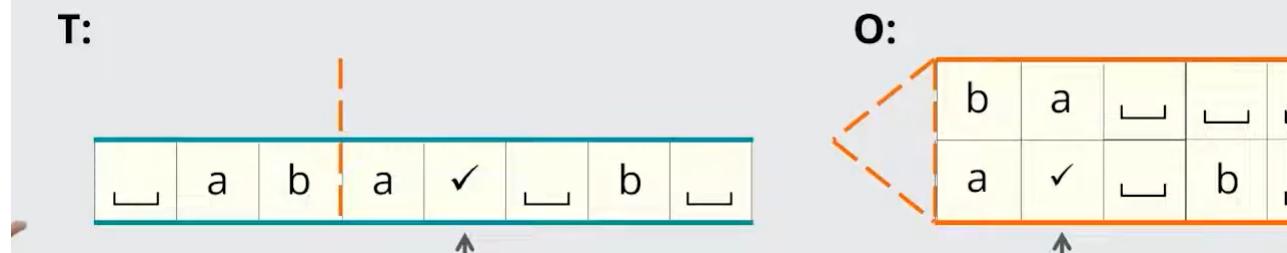
על פניו, נראה שהמכונות יהיו זהות ורק שלא משתמש הצד השמאלי. ובכן - כמובן. נזכיר כי אם אנחנו במצב התחלתי זים שמאלה, אנחנו נשארים במקום ואילו במודל הבסיסי זים שמאלה - יש לגשר על הפער הזה.

הרעיו יהיה כדלקמן - כל הריבאים יהיו זים למגרי. נסיף לטבלת המעברים את הדבר הבא: נסמן ב\\$ את האות הראשונה משמאלי לקלט, בשביל לדעתה היכן מתחילה הסרט השמאלי האינסופי. כל שנדרש לשנות הוא אם אנחנו נמצאים במצב הראשון, זים שמאלה: בצע שתי פעולות - אז שמאלה, אל \\$, וכשתראהدول: תלבץ ימינה ועל תשנה דבר. ככלומר - שתי פעולות אלו יבטיחו נשארם במקום.

ב. נראה שלכל מכונה במודל T יש מכונה במודל O

$$M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, acc^T, rej^T) \text{ מכונת טיריג'ינג למודל } T. \text{ נבנה } O = (Q^o, \Sigma^o, \Gamma^o, \delta^o, q_0^o, acc^o, rej^o) \text{ מכונת טיריג'ינג למודל } O.$$

הרעיו יהיה "לכע סטטוליה" כאלוינו כיסרט זו כיווני. ככלומר - נבחר מקום לתחילת מפנו את הסרט השמאלי, כל מה שהוא משפטו, נקפל, בזורה שתראה כך:



שאלת השאלה, כיצד זה חוקי? ובכן - אנחנו נקבע את האיך להיות זוגות סדורים. את נקודת הקיפול נסמן ב\\$.

מעבר לכך הקיפול: הזוג ימינה הסרט המקורי תורגם לתזהו ימינה הסרט החדש, ותזהו שמאלה לתזהו שמאלה הסרט החדש. (אווו כיוון

לאחר קו הקיפול: תזהה ימינה הסרט T תורגם לתזהה שמאלה הסרט O , ותזהה שמאלה הסרט T תורגם לתזהה ימינה הסרט O . (ההפקן

כמו כן - על המכונה לאזכור באיזה שורה היא נמצאת במכונה O : העליונה או התחתונה - וכן נשמש בתא זכרו פמייה: D יטמין להלך התחתון, ו- U לעליון. כאשר המכונה T תזהה מפונה לקו הקיפול נעהו U , והפקן מעבור ל- D .

כמו כן, נשלוט בתא זכרו פמייה את המצב בו היוו T . כל שינוי במכונה החדש בעלת שתי השורות ויהיה זוג אחר בשתי השורות, ככלומר - אם במכונה החדש היה לנו זוג $(a, -)$ והו יהולף ל- \emptyset כאשר אנחנו עדין פמייה לקו התהודה, נקבל במקצת את הזוג $(b, -)$ - ככלומר רק והשתנה. כמובן, כי משתנה מהזוג זה בהתאם L/U .

באשר המכונה הכתומה, O , תראה דילוי: הוא צריך לכע לו לאו שתחזיר אותה ימינה, ולשנות את הכוון - ככלומר אם היה D או U ולהפוך.

כעת למימוש הפורמלי:

$$Q^o = \{Q^T \times \{U, D\}\} \cup \{q_0^o, back\} \cup \{q.\tau | \tau \in \Sigma \cup \{-\}\} \cup \{acc^o, rej^o\}$$

הערה - החלק $\{q.\tau | \tau \in \Sigma \cup \{-\}\}$ הוא בשביל האתחול של המוכנה בעלת הזוגות.

$$\Gamma^o = \{\Gamma^T \times \Gamma^T\} \cup \Sigma \cup \{\$, -\}$$

$$\Sigma^O = \Sigma^T$$

וכן טבלת המיצבים:

| $\tau, \sigma, \pi \in \Gamma^T$ | מצב | סימן | מצב חדש | כתב | כתובת | תנאי | |
|----------------------------------|----------|------------------|----------|-------|---|------------------|--------------------|
| $q.D$ | π | $p.D$ | π | L | (q, π) M^T (p, τ , L) | תוויה שמאלה | תוויה מקורית שמאלה |
| | σ | $p.U$ | τ | R | | | |
| $q.U$ | π | $p.D$ | τ | L | (q, π) M^T (p, τ , L) | תוויה שמאלה | תוויה מקורית ימינה |
| | τ | $p.U$ | π | R | | | |
| $q.D$ | π | $p.D$ | π | R | (q, σ) M^T (p, τ , R) | תוויה ימינה | תוויה ימינה בקצה |
| | σ | $p.U$ | π | L | | | |
| $q.U$ | π | $p.D$ | τ | R | (q, π) M^T (p, τ , R) | תוויה ימינה | פינוי בקצה |
| | τ | $p.U$ | π | L | | | |
| $q.D$ | \$ | $q.U$ | π | R | $\tau \in \Sigma \cup \{-\}$ $\sigma \in \Sigma$ | תוויה ימינה בקצה | אתחול |
| | $q.U$ | \$ | $q.D$ | π | | | |
| q_0^o | τ | $q.\tau$ | \$ | R | $\tau \in \Sigma \cup \{-\}$ $\sigma \in \Sigma$ | תוויה ימינה בקצה | סיום |
| $q.\sigma$ | τ | $q.\tau$ | σ | R | | | |
| $q.-$ | - | back | - | L | $\tau \in \Sigma \cup \{-\}$ $\sigma \in \Sigma$ | תוויה ימינה בקצה | אתחול |
| back | τ | back | π | L | | | |
| Back | \$ | $q_0^o.D$ | π | R | $\tau \in \Sigma \cup \{-\}$ $\sigma \in \Sigma$ | תוויה ימינה בקצה | סיום |
| acc ^o .D | המל | acc ^o | | | | | |
| acc ^o .U | המל | acc ^o | | | $\tau \in \Sigma \cup \{-\}$ $\sigma \in \Sigma$ | תוויה ימינה בקצה | אתחול |
| rej ^o .D | המל | rej ^o | | | | | |
| rej ^o .U | המל | rej ^o | | | $\tau \in \Sigma \cup \{-\}$ $\sigma \in \Sigma$ | תוויה ימינה בקצה | אתחול |
| | | | | | | | |

נשים לב כי בטבלה מופיע מצב האתחול - שכן הקלט ניתן לנו כקלט אינסופי בשני הצדדים, ואנחנו, בהינתן אוטיות הקלט צריכים לתרגםו לאוגות על פניו סרטם עם ימינה בלבד. כמו כן, אם בסיסום המוכנה המקורית הגיעו גם אנחנו, ובדומה על rej .

הערה אחרונה - נשים לב כי מקצתה הסרט הימני, יש אנסורי רוחחים. לא יוכל לטפל בכלם ולהפוך אותם לאוגות: ולכונתאך מצב בו נפאו רוח חדש, ואנחנו נקבע שנטיעחס אליו אולי יהיה זוג רוחים.

8.3 מודל TS

וריאציה נוספת היא מכונת טיריניג שיכולה להשאר במקום, במודל זה הראש יכול להשאר במקום: קלומר ניתן להשאיר את הראש באותו מקום על הסרט גם כאשר עוברים בין מצבים. נזכיר כי במודל T המקורי הגדרנו את פונקציית המעברים כך:

$$\delta^T : (Q/\{acc, rej\}) \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

cut גדר את הפונקציה δ :

$$\delta^{TS} : (Q/\{acc, rej\}) \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\}$$

כאשר S - השארות במקומות.

כך למשל: אם אנו עובדים עם הרטט $abba$, המשמעות של S הינה - תעבור לנצח q_1 , תחליף את האות בה כרגע אתה נמצא a , הרי אנו בתחלת הרטט) לאות b , אבל ברטט - השאר במקומות.

טענה: המודל T והמודל TS הינם שקולים.

הוכחה (רעיון): ברור כי כל מכונה T הינה גם מכונה TS כי אפשר שלא להשתמש במצב S . בכיוון השני, הרעיון יהיה גם כן לא מסובך יותר מדי - לכל מצב, q_1 למשל, נוצר מצב בינוים q_1^L מה הרעיון? אנחנו נרצה לבנות מכונת TS ממכונית T ולפיכך - אם יש לנו את האות S , למשל, $\delta^T(q_1^L, b) = (q_1, b, L)$, והוא יתורגם ל- $\delta^{TS}(q_0, a) = (q_1, b, R)$ ולאחר מכן $\delta^T(q_1, b) = (q_1, b, S)$ כולם - זוו למצב הזמן שמאלה על הרטט. ואז אכן השגנו מה שרצינו: לא אזנו ברטט אבל אזנו במצב.

$$nismim leb shporimaliyot, \{q^L | q \in Q^{TS}\}$$

8.4 מודל OR

נדיר מודל מכונות טיריניג חדש בשם OR . במודל זה, הראש יכול לבצע בכל מעבר רק פעולה אחת: או אז על הרטט (ימינה או שמאליה) או לכתוב במקום הנוכחי ברטט - ללא תנואה ימינה או שמאליה. ככלומר, פונקציית המעברים במודל זה הינה:

$$\delta : Q \times \Gamma \rightarrow (Q \times (\Gamma \cup \{L, R\}))$$

מלבד הבדל זה, מודל OR זהה למודל T .

טענה: מודל OR ומודל T שקולים.

הוכחה: נוכיח ע"י שנראה כי OR שקול ל- TS , ומטרזנטיביות השקילות (הרוי זהיחס שיקילות) נקבע כי מודל OR שקול למודל T .
כיוון ראשון: לכל מודל OR קיימת מכונה שcolaה מודל TS .

תהי $M_{OR} = (Q^{OR}, \Sigma^{OR}, \Gamma^{OR}, \delta^{OR}, q_0^{OR}, acc^{OR}, rej^{OR})$ נבנה מכונה $M_{TS} = (Q^{TS}, \Sigma^{TS}, \Gamma^{TS}, \delta^{TS}, q_0^{TS}, acc^{TS}, rej^{TS})$. כל הרכיבים של M_{TS} יהיו זמינים לרכיבי M_{OR} . פרט לפונקציית המעברים.

א. מקרה ראשון: אם $\delta^{OR}(q, \sigma) = (p, move)$ באשר $move \in \delta^{OR}(q, \sigma) = (p, move)$ וכן $p, q \in Q^{OR}$ וכן $\sigma \in \Gamma^{OR}$: $\delta^{TS}(p, move) = (q, \sigma)$

$$\delta^{TS}(q, \sigma) = (p, \sigma, move)$$

כיוון שאנו במבנה המקורי צנו ללא כתיבת קלט על המוכנה, וכך נדמה זאת ע"י כך שנכתוב את אותן האות ונטקם באותו הכוון.

ב. מקרה שני: אם $\delta^{OR}(q, \sigma) = (p, \tau)$ באשר $\sigma, \tau \in \Gamma^{OR}$ מכילה את המעבר הבא: $p, q \in Q^{OR}$. המעבר המקביל ב- δ^{TS} יהיה:

$$\delta^{TS}(q, \sigma) = (p, \tau, S)$$

כיוון שאנו לא זים (נשארים במקום) וכן כותבים אותן קלט.

כיוון שני: לכל מבנה במודל TS קיימת מבנה במודל OR שקופה לה. תהי $M_{TS}(Q^{TS}, \Sigma^{TS}, \Gamma^{TS}, \delta^{TS}, q_0^{TS}, acc^{TS}, rej^{TS})$ מבנה מסוימת TS . נבנה מבנה מסוימת $M_{OR} = (Q^{OR}, \Sigma^{OR}, \Gamma^{OR}, \delta^{OR}, q_0^{OR}, acc^{OR}, rej^{OR})$ OR שבמעברים בהן המוכנה M_{TS} כותבת אותן וגם זה ימינה או שמאללה, אלא יתכן מעבר שכול יחיד במבנה M_{OR} . לכן, מירח חלק מהמעברים במבנה M_{TS} לשני מעברים עוקבים במבנה OR : מעבר ראשון כתוב אותן ובמעבר השני נבצע את התזוזה. לשם כך, נזדקק למצבי ביןים חדשים שיחברו בין המעברים. לכל מצב q נגדיר שני מצבים ביןים ייחודיים לו: q^L, q^R . וכך נקבל:

$$Q^{OR} = Q^{TS} \cup \{q^L | q \in Q^{TS}\} \cup \{q^R | q \in Q^{TS}\}$$

עת נגדיר את פונקציית המעברים.

מעבר הביניים תמיד יבצע תזוזה ימינה או שמאללה בלבד, לכל אותן שבסרט ולכל:

$$\forall q \in Q^{TS}, \sigma \in \Gamma^{TS} : \delta^{OR}(q^R, \sigma) = \{q, R\}$$

$$\forall q \in Q^{TS}, \sigma \in \Gamma^{TS} : \delta^{OR}(q^L, \sigma) = \{q, L\}$$

בحينו מעבר $p, q \in Q^{TS}$ $\sigma, \tau \in \Gamma^{TS}$ באשר $\delta^{TS}(q, \sigma) = (p, \tau, R)$ נגדיר:

$$\delta^{OR}(q, \sigma) = (p^R, \tau)$$

בحينו מעבר $p, q \in Q^{TS}$ $\sigma, \tau \in \Gamma^{TS}$ באשר $\delta^{TS}(q, \sigma) = (p, \tau, L)$ נגדיר:

$$\delta^{OR}(q, \sigma) = (p^L, \tau)$$

כמו כן ברור כי אם TS נשארים במקום זה טוב לנו כאן וזה טריוואלי.

8.5 מכונת טיורינג מרובת סרטים

וריאציה נוספת למודול, כאשר לכל מכונה יש מס' סרטים כלשהו גודל שווה מ-1, **קבוע מראש**. לכל סרט ראש כתוב משלו, אך כולם מחוברים לבקר המركזי של המכונה. בכל צעד המכונה קוראת וכותבת בכל אחד מהסרטים, וגם זה בכל אחד מהסרטים. הראשים הסרטים השונים יכולים לוזז באופן ב'ת'. במכונה עם מס' סרטים, הקלט תמיד בתחילת המכונה נמצאת על הסרט הראשון, ושאר הסרטים ריקים.

דוגמא. כתוב מכונת טיורינג עם מס' סרטים שתכרייע את השפה $\{w \in \{a, b\}^* | w = w^r\}$.
הרעיון, די פשוט. בתחילת: נתיק את הקטל מהסרט הראשון לסרט השני. בסרט הראשון הראש יהיה על האות השמאלית ביותר של המילה, ובסרט השני הראש יהיה על האות ימינית ביותר של המילה. בכל שלב, נבדוק האם השני הראים עם אותה אות - ואם כן נתקדם. אם יש שווין לאורך כל הדרך, אז המחרוזת היא פילינדרום. בשביל הפשטות, נניח שניין להשר במקומות שונים. ראיינו כבר שמודל TS שקו למודול T . זו המכונה -

| מצב | אות סרט 1 | אות סרט 2 | מצב חדש | כתב סרט 1 | כתב סרט 2 | תגובה סרט 1 | תגובה סרט 2 |
|-------|-----------|-----------|---------|-----------|-----------|-------------|-------------|
| copy | a | — | copy | ↑ | a | R | R |
| copy | b | — | copy | ↑ | b | R | R |
| copy | — | — | back | ↑ | — | S | L |
| back | — | a/b | back | ↑ | — | S | L |
| back | — | — | check | ↑ | — | L | R |
| check | a | a | check | ↑ | — | L | R |
| check | b | b | check | ↑ | — | L | R |
| check | — | — | acc | ↑ | — | S | S |

טענה: לכל $N \in k$, המודול של מכונת הטיורינג עם k סרטים שקו למודול מכונת הטיורינג המקורי.
הוכחה (לא פורמלית) בכלABL חשבוה: לכל k ומכונת טיורינג $(Q^k, \Sigma, \Gamma^k, \delta^k, q_0^k, acc^k, rej^k)$ עם k סרטים, נוכיח כי קיימת מכונה

$$M^1 = (Q^1, \Sigma, \Gamma^1, \delta^1, q_0^1, acc^1, rej^1)$$

נרצה לעשות סימולציה של המכונה עם k הסרטים על המכונה עם הסרט אחד. מה צריך בכל סימולציה?

1. ייצוג הקונפיגורציה של המכונה המסומלת ע"י או בתוך המכונה המסומלת (M^1)

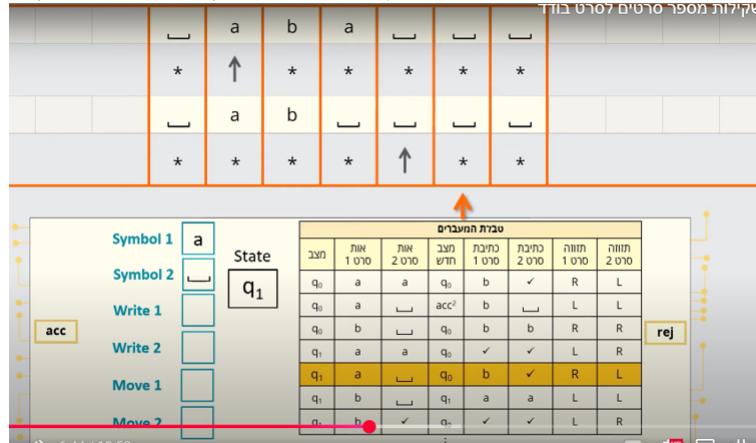
2. סמלץ המעברים בין הקונפיגורציות
הרעיון יהיה זה - נציג אותו עם $2 = k$: יש לנו שני סרטים, נhapok אוטם 4 כ' שמשודרים בשורות אחת אחרי השניה. הסרט הראשון, ומתחתיו סרט שמוסמן כ' - איפה שהה הראש יסומן חז, ובכל שאר מקומות יסומן *. השורה השלישי, תהיה הסרט החשי, והשורה הרביעית תהיה באופן זהה שורה של חז וכוכבויות. באופן דומה על k סרטים נשכפל למכונה עם $2k$ סרטים כשותחת לכל סרט סרט של * וחצים.

מדוע עשינו זאת? כיון שעליינו לדעת היכן נמצא הראש.

סימולציה של צעד בודד של M_2 במעבר ל- M_1 תעללה המון צעדים -

לשם כך נשתמש בתאי זכרון נוספים: $.symbol1, symbol2, write1, write2, Move1, Move2$ הרעיון הוא ש- $symbol1$ נשמר את הסימן שמוועץ בחז העליון ובהתאמה $symbol2$ הסימן שמוועץ בחז התחתון. $write1$ נשמר מה צריך לכתוב בכל אחד מהחצאים, $Move1$ נשמר לאיזה כיוון צריך להזיא את החז. בנוסף, M_1 תשמור את כל טבלת המעברים של M_2 .
בכל סימולציה, הראש של M_1 (המכונה עם סרט אחד שבנינו), יהיה במשבצת שמושمال למשבצת עם החzman השמאלי ביותר. נשמר תא זכרון נוסף בשם $state$ שיישמר את המצביע של M_1 .
המכונה מבצעת סריקה פעם אחת משמאלי לימיון - וממלאת את ששת התאים לעיל לפי מה שמצוין בסרט (כאשר היה תראה את החצים).

cut - המכונה מחזיקה במאה שהיא רואה בסדרת הראשון ובסדרת השני, אנחנו יודעים מה המצב בו אנו נמצאים - וכך cut ניתן לחפש את השורה המתאימה בטבלת המעברים (כאן מטה, בכתום) ולדעת מה יהיה הצעד הבא של המכונה. במקרה שלנו יכתוב ✓, בסרטים ונוזז R, L בהתאם.



כאשר המכונה סיימה סריקה ראשונה, ממשאל לימיין, היא עוברת לסריקה נוספת מימין לשמאל - כאשר היא תתקל בו המטאים, היא תכתוב ותזוז בהתאם למה ששמרנו בזיכרון הפנימי בששת התאים, ותשנה אותם בהתאם. נשים לב כי אם יש תזוזה L move2 במשול move1 (למשל, איז החץ שיוציא יהיה החץ בשורה הרביעית. החץ של M_1 זו רק ורק ממשאל לימיין ואז מימין לשמשאל ברצף. נעיר כי כאשר כותבים "משנים את כל הריבועיה. שחרי הא"ב הוא ריבועיות). בכל סירוקה מימין לשמשאל ברצף.

נעיר כי בכל שלב בסריקה - החץ צריך להתחיל משבצת אחת ממשאל לחץ השמאלי ביותר. שלב זה חשוב. מתי המכונה מפסיק לעבור? כשהיא ממשאל לימיין - כשהיא משבצת אחת מימין אחריו החץ הימני ביותר, וכשהיא מימין לשמשאל - משבצת אחת אחריו החץ השמאלי ביותר. בכל סירוקה שלב סריקה - אנחנו הולכים לקרוא בטבלת המעברים, מה עושים בהינתן מצב, ושתי אותיות קלות.

אם המצב הופך $state$ של M_1 עוברת למצב מקביל. אם המצב הופך rej או M_1 עוברת למצב דחיה. מה אם היא לא מגיעה לא לדחיה ולא לקבלת? איז M_1 כמו M_2 תמשיך לניצח לרוץ. בשלב התחלתי - תמיד יהיה אתחול והפיכת המכונה לכפי שתואר לעיל. ותמיד נתחיל מ q_0 . את אותו התהליך שהדגמוני כאן, ניתן לבצע על k סרטים - באינדוקציה או בכל דרך אחרת.

8.6 סגירות לאיחוד וחיתוך שפות כריעות/קבילות

cut נשותמש במודל ריבוי סרטים" באמצעות טכנית של ריצה במקביל להוכחת הטענות הבאות הבאה

טענה: יהיו $L_1, L_2 \cap L_1$ כריעה.

הוכחה: יהיו A, B שפות הcriuous והמכונות שמכרויות אותן בהתאם להסתrema M^A, M^B . נבנה מכונה שתכרע את $A \cap B$. רעיון הוכחה יהיה להשתמש בשני סרטים, ראשית נעתק את התוכן של הסרט הראשון לסרט השני ונוציא את הראש של שני הסרטים לשמשאל הקלט. הסרט הראשון יסמל את M^A והשני את M^B . נפעל כך:

א. נעתק את הקלט w הסרט השני והחזיר את הראש לתחילת הסרטים.

ב. נרים את M^A על גבי הסרט הראשון, אם דחפה - דחפה.

ג. אם M^A קיבל, M^C , תרים על הסרט השני את M^B - אם קיבל: קלט. אם דחפה - דחפה.

נשים לב כי מכונה זו תמיד עוצרת, כיון M^A, M^B מכריעות שפות וכלן תמיד עוצרות.

טענה: יהיו L_1, L_2 שפות קבילות, איז $L_1 \cap L_2$ קבילה.

הוכחה: יהיו A, B שפות קבילות והמכונות שמקבלות אותן בהתאם M^A, M^B . נבנה מכונה שתקבל את $A \cap B$ מ- M^C .

נפעל כך:

א. נעתק את הקלט w לסדרת השני והחזיר את הראש לתחילת הסדרתיים.

ב. נרץ את M^A על גבי הסדרת הראשון, אם דחפה - דחפה.

ג. אם M^A קיבלה, M^C תריץ על הסדרת השני את M^B - אם קיבלה: קבל. אם דחפה - דחפה.

נשים לב כי מכונה זו חיבת לקל את כל המילים שבספה, ולכן לכל מילה בשפה נקלט שהיא קיבלה בשתי המכונות אותן. וכך גם בחיון. כלומר, כל מילה בשפה מתתקבלת וכל מילה שאינה בשפה לא מתתקבלת.

טענה: יהיו L_1, L_2 שפות כrüוות, איזי $L_2 \cup L_1$ כrüה.

טענה: יהיו L_1, L_2 שפות קבילות, איזי $L_1 \cup L_2$ קבילה.

(רעיון ההוכחה לשני סעיפים אלו - בדיק כmo בשתי הטענות הקודמות. נרץ במקביל, אם נקלט באחת המכונות שהמילה התקבלה איזי גם המילה שלאו התקבלה. אם שתי המכונות ידחו - המילה לא בשפה.).

8.7 אוטומט עם שתי מחסניות P2

נדיר מודול חדש, אוטומט מחסנית עם שתי מחסניות. במודול זה, בכל מעבר ניתן להכניס/להוציא אוטיות בשתי מחסניות שונות, במקביל. פונקציית המעברים תוגדר להיות:

$$\Delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma^* \cup \{\varepsilon\}) \rightarrow P(Q) \times (\Gamma^* \cup \{\varepsilon\})$$

כלומר, המעבר $\Delta(p, a, \$) = \Delta(q, a, \$, \varepsilon)$ אומר כאשר אנו במצב q וקוראים a , אם בראש המחסנית הראשונית $\$$ אפשר לעبور למצב p ובו נזיא את התו $\$$ מהמחסנית הראשונית (האפסילון מוחק אותה), ובמחסנית השנייה שרים, כתוב $\#$. מודול זה נראה ממבט ראשון כסמל למודול האוטומט המקורי - אך לא.

טענה: מודול P2 שקול למכוון טיריניג במודול T.

הוכחה:

צד אחד פשוט, בהינתן אוטומט מחסנית שכזה קל לדמות אותו כמכונה טיריניג עם שני סרטים.

נכיה כבייחיתן מכונת טיריניג, ניתן לבנות ממנה אוטומט עם שתי מחסניות.

רעיון הבנייה הוא שנדמה כל קונפיגורציה של הסרט במכונה M ע"י שתי המחסניות יחד ב- P : במחסנית הראשונית נשמר את האוטיות (למעט אינסוף הרוחות) בצד שמאל של הסרט עד מיקום הראש (לא כולל), כאשר האות השמאלית ביותר בסרט תהיה בתחלתית המחסנית. במחסנית השנייה נשמר את האותיות מהראש וימינה בסרט (למעט אינסוף הרוחות), כאשר האות העליונה במחסנית תהייה האות עלייה מצעע והאות הימנית ביותר בסרט תהיה האות בתחלתית המחסנית. בהתוצאות של כל מחסנית יהיה הסיכון \$.

אוטומט P מתחילה את הריצה כאשר שתי המחסניות ריקות. כדי לדמות את הקונפיגורציה ההתחלתית של M נדרש שלב אתחול שבו נמלא את המחסניות מຕוך תוכן הקלט של P (נכיה שאוטומט P , ע"פ ההגדרה, קורא את הקלט באופן סדרתי, ואני יכול להגיד ולקרוא שוב את האותיות שנקרואו)

נדיר מצב אחד שираה את אוטיות הקלט אחת אחת, ולכל אות שנקרואת מהקלט - האוטומט יכניס אותה למחסנית 1. אחר כך נעבור למצב אחר, שיעביר את כל האותיות מחסנית 1 למחסנית 2.

המשך ההוכחה - בקמפוס מפורט.

8.8 סרט דו ממדוי - מודל 2D

המודל הדו ממדי הוא מעין מטריצה עם אנסוף עמודות ואנסוף שורות, אך מודל זה חסום בסדר מצטצ שמאלי ומימינית. כמפורט בתמונה:

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|
| | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | |
| a | b | a | | | | | | | | | | | | | | | | | |

הריאש הקורא יכול לזרז ימינה, מטה, שמאליה ומעליה.
פונקציית המעברים תוגדר כך:

$$\delta : (Q/\{acc, rej\}) \times \Gamma \rightarrow Q \times \Gamma \times \{R, L, U, D\}$$

$U - DOWN$, תנועה מטה, $U - UP$, תנועה מטה.

דוגמה. נגידר מושג מתמטי, מס' x יקרא משולשים אם, ורק אם קיימים $n \in \mathbb{N}$ כך ש- i $.x = \sum_{i=0}^n$ $L = \{x | x \text{ הוא מספר משולשי}\}$ Cut, נתכלל על השפה x הוא מספר משולשי. נרצה להציג את L באמצעות $2D$.

מה הרעיון? תחילה, נגידר שהקלט שלנו יהיה כמספר אונארי. כל הקלט בהתחלה יהיה בשורה התחתונה של המטריצה. בלאה חזורת: בכל שלב i נעתיק את כל האחדות בשורה הקודמת, פרט לכך שהאחדות ביזורה. ככלומר בהתחלה נעתק את כל המספרים פרט לאחד בודד, אח"כ את כלם פרט לשני אחדות, שלוש אחדות וכן הלאה... מס' x יהיה שיך לשפה אם"מ בסוף תהליך זה, קיבל צורת משולש במטריצה.

טענה: מודל $2D$ שקול למודל T המקורי.

הוכחה: נרצה להמיר את $2D$ ל T . גיגיל. הכוון השני פשוט. נתחילה במספור הטבלה, כך שהתחא השמאלי ביותר מטה יהיה $(0, 0)$ וכן הלאה נתقدس. לאחר מכן נציג מייפוי חד-חד ערכני מהמטריצה לטבלה ע"י ייצירת הטבלה עם התאים $(1, 0), (0, 1), (2, 0), (1, 1), (0, 2)$ וכו'. וכך הלאה.... ככלומר מעתיקים בכל פעם את האלכסון הבא אל המטריצה, כאשר מתחילה להעתיק אותו ממטה של כל אלכסון. כמו כן במספר את התאים בהתאם, ככה שהתחא הראשון יהיה 0 , בהתאם לאלכסון השני תאים $1, 2$ וכו' הלאה.

נרצה להציג את התוצאות.

בහינתן ראש שנמצא בסרט הדו ממדי בתא מס' i , בשורה y ובעומודה x אז התא מימין לתא i הוא $(x + y + 1)$

התא משמאל לתא i הוא $(i - (x + y))$

התא מעל התא i הוא $i + (x + y + 2)$

התא מתחת התא i הוא $i - (x + y + 1)$

נבחן כי אם נצטרך לזרז שמאליה בעמודה השמאלית ביותר נשאר במקום כי הסרט חסום משמאל, ואם נצטרך לזרז למטה בשורה התחתונה ביותר נשאר במקום כי הסרט חסום מלמטה. ככלומר, אם x אז i ואם $0 = y$ זה גם כן i .

הרעيون יהיה ליצור מכונה עם 4 סרטיים, שלפי שיקוליות קודמות שcolaה ל- T .

בסרט העליון: יהיה המיפוי החד ערכי שיצרנו לסרט בווד מהתבלה הדו ממדיית. הסרט השני ישמור את מיקום עמודה x , הסרט השני את מיקום שורה y הסרט הדו ממדי והסרט האחרון את חישוב פונקציית המיפוי. כלומר, הוא יחשב לאיזה תא הראש צריך להציבו לאחר המעבר. נבחן כי פונקציית המיפוי היא ע"י ביצוע פעולות חיבור וחיסור ולכן ניתן לחישוב ע"י מכונת טיריניג.

דוגמה. נניח ויש לנו במקום 0 את האות a ונקלט $\{q_0, a, U\} = \delta_{2D}(q_0, a)$ אזי הסרט הראשון ישמר a הסרט השני ישמר 0 הסרט השלישי ישמר 0 ובסרט האחרון תחשב פונקציית המיפוי למליה $2 = 0 + 2 + 0 + 0 = 2$ כלומר מיקום.

8.9 מכונת טיריניג שאינה דטרמיניסטית

כידוע, במקרים זה יהיו מס' מצבים אליהם ניתן יהי להגעה עבור קלט נתון. נגידו היבט את מושג הדחיה והקבלת מודול מסווג זה:

הגדרה: תהי M מכונת טיריניג לא דטרמיניסטית וכן $w \in \Sigma^*$.
 M מקבלת את w , אם קיים חישוב של M על w שמנגד למצב $.acc$.
 M דוחה את w אם כל חישוב של M על w מגע למצב $.rej$.
נשים לב - בדיחה זה כל חישוב, בקבלה זה אחד החישובים.

הגדרה: עבור שפה $L \subseteq \Sigma^*$, נאמר כי:
 $w \in L$ מכירעה את L אם לכל $w \in \Sigma^*$ מתקבלת את w .
אם $w \in L$ מתקבלת את w .
אם $w \notin L$ דוחה את w .

הגדרה: M מקבלת את L אם לכל $w \in \Sigma^*$ מתקבלת את w אם "מ".

דוגמא. נגידו משלים על מהירות $0\overline{1} = \bar{1}$. וכך למשל $001101 = \overline{110010} = \bar{0}$. נגידו את השפה

$$L = \{w \in \{0, 1\}^* \mid w = uv, (u\bar{v})^R\}$$

כלומר, אוסף המילים בהם קיימת סופה של המילה, שכאשר נפעל עליה את המשלים נקבל פולידורות. למשל 11000 , אם נסתכל על $00 = u$ מתקיים $11 = \bar{u}$ ואז אכן מתקיים השוויון. נרצה לבנות אוטומט לא דטרמיניסטי שיקבל שפה זו.
הפתרון - יהווה להשתמש בטכנית המעכרים שיצרנו למכונית הטיריניג של שפת הפולידורות + התוספה כאו מטה:

| PAL: | מצב | טימון | מצב חדש | כתב | תוויה | תוויה |
|------|-------|-------|---------|------|-------|-------|
| | q_0 | - | acc | ת | R | |
| | | | | . | | |
| | | | | . | | |
| | | | | back | - | q_0 |
| | | | | - | - | R |

| FLIP: | מצב | טימון | מצב חדש | כתב | תוויה | תוויה |
|-------|-----------------|-------|-----------------|-----|-------|-------|
| | \widehat{q}_0 | 0,1 | \widehat{q}_0 | ת | R | |
| | \widehat{q}_0 | 0,1,- | flip | ת | S | |
| | flip | 0 | flip | 1 | R | |
| | flip | 1 | flip | 0 | R | |
| | flip | - | back | - | L | |
| | back | 0,1 | back | ת | L | |
| | back | - | q_0 | ת | R | |

הຽיון - נוסף מצב חדש שיסורק את כל הקלט משמאלי לימי, נקרא לו \widehat{q}_0 , וכן מצב *flip* שি�שנה כל 0 לאחד וכל אחד לאפס. הירויו והוה להפעיל כל פעע את *flip* על חלק אחר בקלטן, באופו סימטרי ביו הקצאות, ואז לקרווא ל-*PAL*, שתבזוק אס המחרוזות שהתקבלו היא פלינדרום - ואם, ורק אם זה יקרה אז נלך למסוב מקבל.

cut, המכונה תערור בוזאות על כל מילה ששיוכת לשפה ולא תערור על מילים שלא שיוכות לשפה.
ולכן המכונה מקבלת את השפה L .

השימוש במוגנות טירינג לא דטרמיניסטי שימושי במיוחד עבור קבלה של שפות.

8.9.1 הכרעה ו渴לה של שפות

עבור מכונה לא דטרמיניסטי N ושפה L :

N מכירעה את L אם N מקבלת את כל המילים בשפה ודוחה את כל המילים שלא בשפה.
 N מקבלת את L אם N מקבלת את כל המילים בשפה ולא מקבלת את כל המילים שלא בשפה.

שים לב להבדל - במוגנות דטרמיניסטי ישנו היישוב יחידת אם הוא עוצר במצב מקבל המילה מתקבלת. במוגנות לא דטרמיניסטי יש הרבה היישובים אפשריים - מילה מתקבלת אם באחד המסלולים היא התקבלה. ונחתית אם בכל החישובים נחתה.

טענה: מודל מוגנות הטירינג הלא דטרמיניסטי שקול למודל הדטרמיניסטי (הערה, זה לא נכון בזמני ריציה אך לא נדוע בכך).

הוכחה: הרעיון יהיה שהמכונה הדטרמיניסטי שנבנה M תעבור על כל הקלטים האפשריים, אם אחד התקבל נלך במצב מקבל.

בידיינו טבלת מעברים. נשים לב שעבור אות קלט s ומצב q ייתכנו (כי לא דטרמיניסטי) מס' מעברים שונים, א' למשל. נסמן אותם במספרים $k = 1, \dots, n$. בעת - ניצור מכונה עם שני סרטים. על הסרט הראשון נריץ את הקלט. בסרט השני נכתב תחילתה מס' אחדות כמספר מס' מס' מספר המעברים באוטומט. בעת, נבצע סימולציה על המוגנות באמצעות המספרים: תחילת, כל המספרים יהיו 1, וכן נפעל לפי אפשרויות אחד. אח"כ המצב השני יהיה 2 נניח, ונריץ את הסדרה $\dots, 1, 12111111, 112111111$, וכך הלאה - כך נעבור על כל המסלולים האפשריים. נקבל מילה ונעצור אמ"מ באחד המסלולים הגענו במצב מקבל. סה"כ מכונה עם שני סרטים כפי שראינו בעבר שcolaה למוגנה עם סרט אחד, ולכן ישנה שיקולות. נעיר כי פורמלית יש לעשות סרט נוספת, להעתיק אליו את הקלט, ושם לבצע את ה"עבודה" ובדיקת האפשרויות השונות בשוביל לשומר על הקלט המקורי בשמורות.

פורמלית -

1. כתוב 1 בסדר בחירות.

2. העתק קלט מסרט כסתה הקלט" לסרט עבודה.
3. הרץ את המcona N על סרט עבודה" לפי הסדרה שבסרט בחירות". בכל צעד בדוק -
 - אם המcona N הגיעו למצב acc^N עברו למצב $.acc^M$
 - 4. מחק תוכן סרט עבודה
 - 5. בסרט בחירות כתוב את המחרוזת הבאה לפי סדר מניה (שאלה, כמה לפי סדר מניה ולא לקסיקופי? יתכנו חישובים שלא ידחו או יתקבלו אלא ית��עו ואז לא נצורך לעולם, لكن אסור לנו להמשיך כל הדרך בחישוב אחד אלא צריך לשנות בכל פעם את המסלול לאחד חדש, שלא תתקע לעולם בתוך חישוב אחד).
 - 6. חוזר לשלב 2.

8.10 סגירות באמצעות אידטרמיניזם

טענה: תהי L שפה שמתאפשרת ע"י מכונת טיריניג לא דטרמיניסטיבית, אז גם $prefix(L)$ (תחילהות) גם היא מותאפשרת ע"י מכונת טיריניג לא דטרמיניסטיבית.

טענה: יהיו L_1, L_2 כריעות. אז $L_1 \circ L_2$ כריעה.
רעיון הוכחה: נחלק את המחרוזות לרישא וסיפא, ובכל פעם את הרישא נרץ על גבי המcona A שמקיימת $L(A) = L_1$ ואת הסיפא על B המקיים $L(B) = L_2$. כיון ש A, B מכירויות את השפטות בהתאם, בהכרח עבור כל מחרוזות נקלט או דחיה או קבלה. קבלה של מחרוזת תהיה אם המחרוזות של הסיפא ושל הרישא התקבלה, והחיה אם אחת מהן נדחתה. כך נרץ על כל הרישאות והסיפאות האפשריות. כיצד? בכל שלב נעתיק את האות השמאלית ביותר לסרט אחר, ונרץ את המcona. אם נקלט סיימני, אחרת נמשיך להעתיק אות הבא. כך נעבור על כל רישא וסיפא אפשרית. בכל שלב שchez נרץ על הסרט השני את M_A , אם קיבל נרץ על הראשון (S_i) סיפות את M_B . אם קיבלה נקלט, אחרת נדחה. נשים לב שכיוון שהשפטות כריעות, בכל אחת מהמכונות תמיד נדחה/נקבל וכך לא נגיע למצב של אי עצירה.

טענה: יהיו L_1, L_2 קבילות. אז $L_1 \circ L_2$ קבילה.

טענה: אם L כריעה, אז L^* כריעה, ואם L קבילה אז L^* קבילה.
הוכחה: תהי M מכונת טיריניג דטרמיניסטיבית שמכירה את L . נבנה מכונה N לא דטרמיניסטיבית שמכירה את L^* . המכונה N שני סרטים, שליהם נקרא: "קלט" ו"עובדה". הרעיון הוא N פרק את הקלט באופן לא דטרמיניסטי לחקלים. היא תעתק כל "קלט" מסרט ה"קלט" לסרט "עובדה", ואז תפעיל את המכונה M על סרט ה"עובדה". רק אם המכונה M קיבלה את כל החקלים, אז המכונה N גם מקבל. אחרת – היא תדחה.

ספקטיבית, המכונה N תפעל בזרה הבהא:
 אם הסרט "קלט" רואים רווח – קיבל כל עוד הסרט קלט לא רואים רווח, חזר על התהליך הבא
 בחר באופן לא דטרמיניסטי בין האפשרויות הבאות: א. העתק אות מסרט "קלט" לסרט "עובדה", והוא את שני הראשונים ימיה. אז את הראש הסרט "עובדה" לטו השמאלי ביחסו שאנו רוח ועובד לשלב 3, אם אין זה – בצע את שלב א" הרץ את המכונה M על "סרט עובדה", עד לעצירה אם ההשופט הסתומים במצב דחיה – דחה. מחק תוכן סרט "עובדה" וחזר לשלב 1.
 נשים לב שכיוון ש M מכונה להכרעה, שלב 3 בהכרח מסוימים, וכך כל התהליך בהכרח מסוימים. ולכן, זו היא מכונה להכרעה. אם קלט w ב- L^* אז יש לו פיצול $w = w_1 \dots w_k$ מותקים $w_i \in L$. לכן, החישוב של N שבשלב 2 בדוק מעתק בכל פעם את ה w_i המתאיםibia את המכונה למצב מקבל.
 ומצד שני, אם w לא ב- L^* , אז לכל ריצה, בהכרח יהיה חלק שמוועתק הסרט "עובדה" שהוא לא בשפה, ולכן המכונה תדחה בשלב 3.

טענה: תהי L כריעה. אז $reverse(L)$ כריעה.
הוכחה: אם L כריעה קיימת מכונת טיריניג M_L שמכירה אותה. המכונה M_R ראשית תהפוך את סרט הקלט, והוא תרץ את המכונה M_L שתתקבלו או תדחה בהתאם. שכן אם הפכו את סדר

המילה, המילה תהיה ב L).

9. ייחידה 10: התזה של צראץ'-טיירינג

האם מכונת טיירינג היא המודל החזק ביותר שנוכל למצוא, שמותאים למחשב מודרני? ביחידה זו נגלה.

9.1 סגירות

יהיו L_1, L_2 שפות כrüoot האם יש למיניפולציה הבאה סגירות?

- א. איחוד $L_1 \cup L_2$ - כן
- ב. חיתוך $L_1 \cap L_2$ - כן
- ג. שרשור $L_1 \circ L_2$ - כן
- ד. סגור קליין L_1^* - כן
- ה. שפת התחילה $\text{prefix}(L_1)$ - $\text{prefix}(L_1) = \text{Reverse}(\overline{L_1})$ - כן
- ו. רורס $\underline{\underline{L_1}}$ - כן
- ז. משלימים: $\overline{L_1}$ - כן

יהיו L_1, L_2 שפות קבילות האם יש למיניפולציה הבאה סגירות?

- א. איחוד $L_1 \cup L_2$ - כן
- ב. חיתוך $L_1 \cap L_2$ - כן
- ג. שרשור $L_1 \circ L_2$ - כן
- ד. סגור קליין L_1^* - כן
- ה. $\text{prefix}(L_1)$ - $\text{prefix}(L_1) = \text{Reverse}(\overline{L_1})$ - כן

אין סגירות למשלים ולא לרורס.

9.2 היחס בין הכרעה לקבלת

טענה: אם שפה כrüoot, היא בהכרח קבילה.

הוכחה: L כrüoot, יש מכונה שמקரיעה אותה. בהכרח, היא גם מקבלת אותה. כנדרש.

טענה: אם L וגם \overline{L} קבילות, אז L כrüoot.

הוכחה: יש מכונות שמקבלות את L ו \overline{L} . ניצור מכונה חדשה D , וביצע סימולציה של הריצה על המכונות. בהכרח, כל מילה שיכת או L או \overline{L} , ולכן את המכונות תקבל את המילה. אם מכונה אחת קיבלה את המילה, בהכרח המילה לא קיימת בשפה השנייה - ולכן המכונה השנייה תדחה את המילה. סה"כ לכל מילה בשפה היא תתקבל ע"י אחת המכונות, וכל מילה שלא בשפה תדחה כתוצאה מכ"כ -כלומר, אם המילה התקבלה במכונה של L אז המילה בשפה ואם התקבלה במכונה של \overline{L} המילה לא בשפה, ולכן נזחה וסה"כ לכל L כrüoot.

9.3 מכונות טיירינג שcoleה לתוכנית מחשב

סוף סוף הגיענו לטענה שלנו - מכונות טיירינג חזקה **לפחות** כמו תוכניות מחשב. נציגך ראשית להגדר מהו מחשב. במקומות לדבר עליון, נגיד תוכניות מחשב. מה שנעשה היה להראות כי כל בעיית הכרעה שניתן לפתור באמצעות תוכנית מחשב, ניתן לפתור באמצעות מכונות טיירינג.

הרעיון הוא להראות שכל דבר שניתן לעשות בשפת תכנות, ניתן לעשות במכונות טיירינג. באיזה שfat תכנות נבחור? אם נבחר את ג'אווה או פיאיטון לא נסימם בקרוב כי יש בהן המון דברים. כמו כן נשים לב לטענה: כל דבר שניתן לעשות בשפת תכנות A , ניתן לעשות בשפת תכנות B (פרט לשפות ספציפיות מאוד). לשם כך - נגיד שfat תכנות משלנו בשם *SIMPLE* שתכלול את כל הדברים החשובים לשפת תכנות. כל תוכנית בכל שפה תוכל להיות מומרת לשפה *SIMPLE*. השפה תכלול:

- א. משתנים: ישנו שני סוגי משתנים -
1. משתנים טבניים i, j, k , שערכם יכול להיות מס' טבאי $A[], B[], C[]..$. בכל תא ערך מותך א'ב ג'. המעריכים הם אין סופיים.
 2. מערכים - הקלט נמצא בתאים הראשוניים של המעריכים, המשתנים מאותחלים לאפס.
- ב. פעולות:
1. השמה - בקבוע, למשל $# = i, j = 3, A[2] = B[7]$.
 2. פעולות חשבון $a = x + y, b = x - y, c = xy$
 - ג. תנאים:
1. שווין בין תנאים במערך $A[i] == B[j]$ או לבדוק אם שווין $j = i$. כל משתנה מופיע רק פעם אחת בשורה, כלומר לא ניתן לכתוב $k + j$.
 2. זרימה: הפעולות ממושפרות, ומוצבעות אחת לאחר השניה. פרט לפוקודה *goto* שהולכת לשורה ספציפית שיכולה לבוא במהלך היריצה. כמו כן יש פעולה *stop(a)* שמחזירה ערך a ועוצרת.

דחיה וקבלת עזרה: עבור קלט w ותוכנית P בשפה SIMPLE נאמר כי:
 P מקבלת את w אם היריצה של P על w עצרת עם ערך חזרה 1.
 P דוחה את w אם היריצה של P על w עצרת עם ערך חזרה 0.

הerule וקבלת של שפות: עבור שפה L ותוכנית $P \in SIMPLE$ מכראעת את L אם היא מקבלת את כל המילים שב L ודוחה את כל המילים שאינן בו. P מקבלת את L אם היא מקבלת את כל ורוק המילים בו.

המודל החישובי SIMPLE: אוסף כל התוכניות P התקינות בשפה SIMPLE.

הטענה: המודל של מכונת טירינג ומודל SIMPLE הינם שקולים.
הוכחה:
 כיוון ראשון: לכל מכונת טירינג יש תוכנית P שקופה. לא נתעכט על כיוון זה כי די ברור שכל שפת עילית יכולה לדמות באמצעות מבנה נתונים כלשהו את מודל מכונת הטירינג.
 כיוון שני: לכל תוכנית בשפה SIMPLE יש מכונת טירינג שקופה.
 נראה כיצד למשם את השפה מכונות טירינג.

משתנים - לכל משתנה יהיה סטרט נפרד. במערכות: לכל תא במערך יהיה תא בסרט הקלט. במספרים הטבעיים: יהיה ייצוג אונארי של אחדות. כל הסטרטים מאותחלים לרוחם שיתאר לנו ערך אפס.

פעולות: השמה בין משתנים טבעיות היא העתקה מסטרט אחד לשני. השמה בין ערכים במערך $B[j] = B[i], A[i] = B[j]$, ראשית הולכים ל- j , כיוון שהוא ייצוג אונארי זה קל: הולכים למערך A ולסרט של משתנה i ומתקדים בהתאם על הסרט של המשתנה כל פעם צד ובמקביל במערך, עד שנגיע למיקום הרלוונטי, בדומה על j במערך B ומשנים בהתאם את העריכים בסרטים. השמה בקבוע נעשית באופן דומה. השמה בקבוע למשנה טבוי - ראשית מוחקם את כל הקלט הקיים בסרט המשנה, ואז מושפים אחדות בהתאם לייצוג האונארי המתאים. באשר לפעולות חשבון - הן כפל הן חיבור והן חיסור ראיינו בעבר כבר כיצד למשם מכונת טירינג (מופיע כאן מעלה).

תנאים: כיצד בודקים אם שווין? ריצה במכונות מההתחלת, ובודקים היכן מגיעים לרשותה. מס' גדול יותר אם הגענו מאוחר יותר - כי יש בו יותר אחדות. ואיך בודקים שווין? אם הגענו לרוחם באותו הזמן. השוואה בין תאים במערכות - מגעים אליהם פשוט ובודקים אם הערך בפנים זהה. באמצעות שווין שתואר לעיל.

זרימה: במכונת טירינג כל פוקודה ממושפרת. מצבי המכונה הם הפקודות השונות. לכל מצב גם טבלת המעריכים קבועת לאיזיה מצב עוברים - ולכן כל למשג $acc = stop(1)$ וכן $stop(0) = rej$. סה"כ, לכל תוכנית מחשב יש מכונת טירינג עם מס' סטרטים שקופה, שלא כדי שראינו יש מכונת טירינג עם סרט אחד שקופה. ננדרש.

■

מעתת ואילך, מחשב=מכונת טיורינג. כל שפה שבריעה/קבילה ע"י מחשב כריעעה/קבילה ע"י מכונת טיורינג.

הערה חשובה. על פניו, מכונת טיורינג עם זכרון אינסופי. כיוון שלמחשב זכרון כה גדול, אי אפשר למדל אותו לאוטומט סופי ללא זכרון. לכן מותיחסים למחשב כאשר צאילו היה עם זכרון אינסופי, כמו מכונת טיורינג.

פסודו קוד: כיוון שכל מכונת טיורינג שcola לשפה simple, מעתה נוכל לכתוב את מכונת הטיורינג בשפת simple או בכל שפת תכנות שהיא. ויתרה מזאת - ניתן וכך נעשה: נכתב מעתה פסודו קוד לתרגילים. לצורך הדוגמה: נרצה להכרייע את השפה

$$L = \{w|w = uu, u \in \Sigma^*\}$$

לשם כך נכתבת התוכנית הבאה:

Double(w):

1. n=length(w)
2. if n mod2=1 return(0)
- 3.n=n/2-1
4. for i=0 to n do:
5. if w[i]!=w[n+1] return(0)
6. return (1)

נעיר כי נוכל לכתוב פסודו קוד, שאיןו דטרמיניסטי. אף על פי שאין לנו מושג כיצד המימוש נראה בפועל. בתוכניות אלו נאפשר פקודות guess בה התוכנית תבחר בזרה לא דטרמיניסטיבית אופציית מבין קבוצה סופית של אפשרויות.

9.4 דקדוקים כלליים

נרחיב את המושג דקדוק חסר הקשר".
בדקדוק חסר הקשר, משמאלי מופיע משתנה ולאחריו חץ לאות קלט. למשל $a|\varepsilon \rightarrow S$. בדקדוק כללי, גם מצד ימין וגם מצד שמאל יכולים להופיע מחרוזות, למשל $[aa] \rightarrow aa[a]$. המשמעות היא שבמהלך כלל היצירה ניתן להחליף את המחרוזות השמאלית, בימנית. כך למשל, בהינתן המחרוזות $[aa]$ לפי כלל היצירה $[aaa] \rightarrow [aaa]$.

דוגמה. נסתכל על השפה $L = \{w \in \{a, b\}^* | \#a_w = \#b_w\}$. דקדוק כללי עברוה יהיה:

$$S \rightarrow abS, S \rightarrow \varepsilon, ab \rightarrow ba, ba \rightarrow ab$$

בעזרת חזזה על הכללי השמאלי ניתן ליצור מחרוזות עם מס' זהה של a ו b , שני הכללים האחרונים מאפשרים סידור חדש של אותיות המחרוזות.

דוגמה 2. נסתכל על השפה $L = \{a^n b^n c^n | n \geq 0\}$. שפה זו אינה חסרת הקשר, אך דקדוק כללי עברוה יהיה:

$$S \rightarrow s'], S' \rightarrow aS'bC|\varepsilon, Cb \rightarrow bC, C] \rightarrow]c,] \rightarrow \varepsilon$$

הרעין יהיה שנazor מילה עם מס' a -ים שווה למס' c -ים, ולאחר מכן נדרש לסדר את המחרוזת בסדר הרצוי של קודם a -ים אחר b -ים ואחר c -ים. מחרוזת תתקבל רק אם סיימנו אותה ללא סוגרים. כל עוד הם מופיעות, לא סיימנו את תהליך הגירה. די להשתכנע שגירה זועובדת - מס' דוגמאות יאוששו זאת.

טענה: שפה הינה קבילה, אם וקें דקדוק כללי שיוצר אותה.
הוכחה: כיון ראשון - בהינתן דקדוק כללי בונה מכונת טיריניג שמקבלת את הדקדוק, אך בשל השקילות של מכונת טיריניג לתוכנית מחשב, ניתן לבנות תוכנית מחשב שיצרת את הדקדוק. בונה תוכנית לא דטרמיניסטית:

כלט: w

$u = S . 1$

repeat : .2

פצל באופן לא דטרמיניסטי את u ל xyz

בחר באופן לא דטרמיניסטי גירה $v \rightarrow t$ של G

אם $y \neq t$ אם דירה

$u = xvz$

אם $u == w$ קיבל

כיוון שני - נתונה מכונת טיריניג M , בונה ממנה דקדוק כללי G כך ש $L(M) = L(G)$. נסביר רעיון כללי - לצורך הבנה בלבד: במכונת טיריניג קוניגורציה היא מהצורה aq_0baa למשל. נניח ונסתכל בביטול המעברים ונראה כי

$$aq_0baa \vdash_M aaq_1aa$$

אזי, הרעיון יהיה להגיד את הדקדוק לפי הקוניגורציות, כלומר

$$aq_0baa \Rightarrow_G aaq_1aa$$

מה השתנה כאן בין המחרוזות? $aq_1 \rightarrow aq_0b$, ולכן זה יהיה כלל יצירה ב- G . וכך, באופן דומה, נגדיר את כלל הדקדוק לפי הקוניגורציות.
 ומה באשר לתזוזה שמאליה? למשל - $aaq_1ab \vdash_M aq_0abb$, זה יתורגם לכל $q_0ab \rightarrow q_0ab$ באופן כללי: אם $\delta(q, \sigma) = (p, \pi, R)$ או $\delta(q, \sigma) = (p, \pi, L)$ או $\delta(q, \sigma) = \tau$ אז $\pi \rightarrow p$ ו $R \rightarrow \tau$.

9.5 ההרכבה של חומוסקי

לפי היררכיה מותקיים:
 בתחתיות הפרמיידה, **השפות הרגולריות** שמתקבלות ע"י **דקדוק רגולרי**, וע"י **מודל האוטומט הסופי**.
 מעלייהם, השפות **חסירות ההקשר** שמתקבלות ע"י **דקדוקים חסרי הקשר** וע"י **מודל אוטומט מחסנית**.
 ומעלייהם, **השפות הקבילות**, שמתקבלות ע"י **דקדוקים כלליים** וע"י **מודל מכונת הטיריניג**.

כל שפה רגולרית - היא גם חסירת הקשר וגם קבילה.
 כל שפה חסירת הקשר - היא גם קבילה. (אך יש שפות קבילות שאינן חסירות הקשר).

טענה: כל שפה חסירת הקשר, הינה כריעה.

9.6 התזה של צרצ'טיוריינג

מכונת הטיוריינג הומצאה ע"י אלן טיוריינג במאה הקודמת. במאה הקודמת, בסביבות שנות העשרים - נכנס עולם המתמטיקה למשבר קיומי סבב הפורמליות של המתמטיקה. لكن הגיש דיוויד הילברט להניה מסמך עקרונות הבא:

1. **שלמות:** כל טענה שנכונה מתמטית, ניתנת גם להוכחה.

2. **נאותות:** רק טענות נכונות ניתנות להוכחה.

3. **כרייעות:** לפתח שיטה בעזרתיה יהיה ניתן לקבוע האם טענה נכונה או לא.

בහמשך, הוכח שיעד מס' 1 אינו ניתן להשגה. ויעד מס' 2 - כן ניתן להגשה. באשר לעיד מס' 3, אלן טיוריינג פיתח את מכונת הטיוריינג דרך לתאר אלגוריותם. המרצה שלו, צרצ'טיוריינג: כל אלגוריתם שנitin לティיאור כלשהו, ניתן גם לתיאור כמכונת טיוריינג. ובפרט, אין מודל חישובי חזק יותר מכונת טיוריינג. **ניסי**
לב שזו תזה ולא משפט שהוכח מתמטי.

מה באשר למטרה ?3? האם אפשר לפתח שיטה באמצעותה נקבע אם טענה נכונה או שלא? את זה נראה ביחידה 11.

10 יחידה 11: אי כרייעות

האם יש בעיות שלא ניתן לפתורן באמצעות מחשב? כמובן, האם יש שפות שאינן כרייעות? האם יש שפות שאינן קבילות?

10.1 אimotoות תוכנה

נאס"א שלחה ב-1998 Challiethal. אמהה אבד הקשר אליה בכניסה למאים. כשבדקנו מידע, גילו שעבדו על החללית שני צוותים שונים שהתייחסו ליחידות המידה באופן שונה: צוות אחד התייחס במטרים ואחד ב-yards. והותכוות ברורות. נאס"א רצתה למנוע זאת. כמובן בהינתן תוכנית של החללית ומפרט - האם התוכנית עומדת בדרישות המפרט? זו בעיה הכרעה.

בהינתן תוכנית P , ומפרט S , נגידר את השפה $\{P\}$ תוכנית, S מפרט וכן P עומדת בתנאי המפרט.

$$PS = \{(P, S) | S \text{ תוכנית } P\}$$

האם PS כרייעת?

נניח ונרצה לפתח תוכנית להכרעת PS . נקרא לה $D - PS$. שמקבלת תוכנית P ומפרט S . כיצד נקבל תוכנית? תוכנית היא רצף של תווים וכן הקלט לתוכנית מחשב יכול להיות תוכנית. (למשל, הקומפיילר מקבל תוכנית, ממיר אותו לשפת מכונה, ומהיר תוכנית).

האם נאס"א יכולה לבנות תוכנית כזו? קשה לדעת. נרצה להמיר את הבעיה לקללה יותר כי לא נרצה לתאר את מפרט הדרישות של נאס"א. נניח שהמפרט מתייחס רק לערך החזרה של התוכנה - וקובע עבור אילו קלטים ערך החזרה צריך להיות 1. נניח שנאס"א חשושה שהתוכנית לא עובדת טוב עבור קלט מסוים, w . במקרה זה, בעיית ההכרעה היא:

$$ATM = \{(P, w) | P(w) = 1\}$$

כלומר, האם בהינתן תוכנית ומחרוזת מתקיים $1 = (w)P$ (כלומר, שנרץ את w על התוכנית, קיבל 1). זו גרסה מנוגנת מזו של הבעיה המקורי: כיון שמתיחס רק לדבר אחד במפרט, ודורשת לעבור על כל קלט w אפשרי. אבל - זה אבל גדול, והוא תנאי הכרחי (ולא מספיק) לכל נסיו לטעות על הבעיה המקורי. נשים לב שדרישה היא גם שהתוכנית תסתיים, כי אם לא תסתיים ותרוץ באופן אינסופי בפרט היא לא תחזיר אחד. האם ATM כריעה? על כך נדון-cut.

10.2 ATM קבילה

האם ניתן לבנות מכונה כללית שבхиינתן זוג (P, w) תקבע אם הזוג ב-ATM? **טענה:** ATM קבילה.

הוכחה: כיצד בונה תוכנית שתקבל תוכנית אחרת? זה בדיק מה שעשוosa מערכת הפעלה של המחשב: היא בעצמה תוכנית, תוכנה, שמקבלת תוכניות ומריצה אותן. נסמן תוכנה זו שמריצה תוכניות שהיא מקבלת ב-U. ככלומר - $U(P, w)$ מರיצה את P על w ומהזירה את ערך החזרה ש- P החזירה. נשים לב כי:

$$(P, w) \in ATM \iff P(w) = 1 \iff U(P, w) = 1$$

מזהירה את ערך החזרה שהתקבל מהריצה של P על w . מריצה את התוכנה P על הקלט w (במקרה שבו P אינה תוכנית מחשב תקינה אז U מזהירה ערך 0). נשים לב שאם P לא עצרת על w או גם U לא עצרת על הזוג (P, w) . התוכנה U פועלת באופן דומה לאוון שבה מערכת הפעלה מפעילה תוכנות אחרות.

מסקנה - U מקבלת את השפה ATM.

הערה: סימנו את התוכנה של מערכת הפעלה ב-U כקיצור *Universal*, ישנה מכונת טירינג אוניברסלית - מכונת טירינג שהקלט שלה הוא תיאור מכונת טירינג + קלט למכונה, והמכונה האוניברסלית מבצעת סימולציה של המכונה על הקלט. היא אוניברסלית כי היא מכונה אחת שיכולה לבצע סימולציה של כל מכונה אפשרית.

10.3 לא כריעה ATM

טענה: ATM לא כריעה.

הוכחה: נב"ש כי ATM כריעה. תהי $D - ATM$ - התוכנית שמכריעה את ATM. התוכנית מקבלת (P, w) . נבנה תוכנית אחרת - z בשם $Stupid(z)$. כאשר z הינה מחרוזת הקלט של נתארה כך:

```
Stupid(z):
a=D-ATM(z,z);
return(!a)
}
```

נשים לב כי הקלט של $D - ATM$ הוא מחרוזות ותוכנית. אך שלחנו שתי מחרוזות. על פניו - זה חוקי כיון שלא שלחנו תוכנית, ולכן על הזוג הזה היא תחזיר תשובה כלשהי (אפס או אחד). הקוד של $D - ATM \subseteq Stupid$.

כעת, נרצה להריץ את $Stupid(Stupid)$. מה יחזיר? נשים לב שבפרט תמיד יוחזר ממשו, כיון ש- a תמיד מוגדר כי $D - ATM$ הינה תוכנית להכרעה. לכן תמיד נחזיר ערך כלשהו. נשים לב כי $stupid(stupid) = 1/0$ בלבד.

נב"ש $D - ATM(Stupid, Stupid) = 1$ או $Stupid(stupid) = 1$. וכך $Stupid(Stupid, Stupid) \in ATM$. ולכן $Stupid(stupid) = 1$. ולכן בשורה $a = D - ATM(z, z)$; $z = Stupid$ כאשר $z = 1$. ולכן, יוחזר סה"כ 1 . $1 = 0$ מסקנה - $Stupid(Stupid) = 0$.

נב"ש $D - ATM(Stupid, Stupid) = 0$. מכאן, $Stupid(Stupid) \notin ATM$. ככלומר $Stupid(Stupid) \neq ATM$. לכן בשורה לעיל כאשר $z = stupid$ נקבל $z = stupid = 0$. ולכן יוחזר $1 = 0$. ככלומר, סה"כ 1 בסתייה. סה"כ $1 = 0$ בסתייה כי $Stupid(Stupid) \neq ATM$.

■

קיבלונו לראשונה שפה לא כריעה - בעיה שלא ניתן לפתורן ע"י מחשב: בעיית אימות תוכנה לא פתירה ע"י מחשב. לא ניתן לקבוע אלגוריתם כללי שיקבע האם תוכנה ניתנת לאימות.
יש המון בעיות נוספות לא כריעות - המחשב שלנו לא יוכל לעשות הכל ():

10.4 שפה שאינה קבילה

נשים לב כי השפות הכריעות סגורות למשלים. וכך \overline{ATM} לא כריעה. עם זאת, M כן קבילה.
טענה: \overline{ATM} לא קבילה.
הוכחה: בשלילה נניח \overline{ATM} קבילה. גם ATM קבילה. לפי טענה () אם L קבילה ו- \overline{L} קבילה,
אז L כרעה (\overline{L} נקבע ATM כרעה. בסתיו).

מסקנה: כיצד נוכיח ששפה אינה קבילה? נוכיח שהשפה לא כריעה, והמשלים שלה כן קבילה. ואז
כמסקנה כמו בדוגמה כאן, השפה שלנו לא קבילה (אחרת בסתיו המשפט לעיל).
הערה: נסמן תוכניות להכרעה בקידומת D , ותוכניות לקבלה בקידומת A .

10.5 בעיית העצירה

נדיר את השפה הבאה: $\{\downarrow\} = \{(P, w) | P(w) \text{ אוסף כל הזוגות, כך שהריצה של } P \text{ על }$
המחזרות w עצרת (כלומר - התוכנית לא נתקעuta). החץ מסמל עצירה.

טענה: \overline{HALT} אינה כריעה.
הוכחה: נב"ש כי \overline{HALT} כרעה. תהי $D - HALT$ – התוכנית שמכריעה את $HALT$. (לא ידוע
מה הקוד.).

בנייה תוכנית $D - ATM$ שמכריעה את ATM , וזה תחיה הסתירה:

```
D-ATM(P,w):
if D-Halt(P,w)==0
return(0);
return(U(p,w));
```

מה קורה כאן? אם אין עצירה על w , אז התוכנית p לא מקבלת את w ולכן מחזירים אפס. אחרת,
התוכנית לא עצרת, מובטח לנו שיש עצירה של p על w . ואז – פשוט נróż את המכונה היררכו-אליטית
 U על התוכנית עם הערך w , ונחיזיר את הערך המתתקבל. נראה כי $U(P,w) = 1 \iff P(w) = 1$. סה"כ – צרנו תוכנית שמכריעה את ATM , ולכן יתקבל 1 אם"מ הריצה של P על w החזרה 1. סה"כ – צרנו תוכנית שמכריעה את ATM כרעה, בסתיו.

■

טענה: \overline{HALT} קבילה

הוכחה: בניית תוכנית שמקבלת את $HALT$:

```
A-HALT(P,w):
U(p,w)
return(1);
```

אם P עצרת על w , אז התהיליך בשורה השנייה יסתתיים וועברים לשורה השלישי. אם P לא
עצרת על w , אז נשאר תקווים בשורה 2 ולא נגע לשורה השלישית – ולכן לא נחיזר 1. כלומר סה"כ
התוכנית מחזירה 1 אם"מ התוכנית עצרת ולכן $HALT$ קבילה.

טענה: \overline{HALT} לא כרעה, ולא קבילה. (ההוכחה בדיקת ATM).

| קבילה | כריעה | |
|-------|-------|-------------------|
| ✓ | ✗ | <i>ATM</i> |
| ✗ | ✗ | \overline{ATM} |
| ✓ | ✗ | <i>HALT</i> |
| ✗ | ✗ | \overline{HALT} |

10.6 שפות לא פתרות

שפה שאינה כריעה וקבילה נקראת שפה לא פתרה.

E השפה 10.6.1

$$E = \{P | L(P) = \emptyset\}$$

כלומר, שפת כל התוכניות כך שהשפה שלחן ריקה. למשל, E : $Q(x) : \in E$ כי התוכנית לא עצרת אף קלט, ובפרט לא מקבלת אף קלט.

טענה: E לא כריעה.

הוכחה: נב"ש כי E כריעה. תהי $D - \overline{ATM}$ התוכנית שמכריעת את E . נבנה $D - \overline{ATM}$ שמכריעת את \overline{ATM} (שהיא אינה כריעה, ואז מקבל סטירה). כך:

$D - \overline{ATM}(P, w)$:

```
Q="Q(x){return (U("+"P+","+"w+","));};"
a=D-E(Q);
return (a);
```

נשים לב כי אם P על w מחזירה 1, אז $U(P, w)$ תחזיר 1 לכל קלט. ואם P על w לא מחזירה 1, אז Q לא מקבל שום קלט. ולכן השפה של Q היא אמ"מ P לא מקבלת את w . נשים לב כי

$$L(Q) := \begin{cases} \Sigma^* & P(w) = 1 \\ \emptyset & P(w) \neq 1 \end{cases}$$

כלומר, אם התוכנית P מחזירה 1 על הקלט w , אז השפה של Q היא של הא"ב. אחרת, שפה ריקה. ובמילים יפות: לא חזר בבדיקה החפץ. אם $P(w) = 1$ אז $Q(w) = 0$, כי השפה אינה ריקה. המטריה בשורת Q היא לתרגם את הזוג (P, w) לשורת קוד אחת אותה נשלח לתוכנית $D - E$.

לכן, $L(Q) \in \overline{ATM} \iff Q \in E \iff \overline{Q} \in \overline{ATM}$, בסטירה כי \overline{ATM} לא כריעה.

למבנה ההוכחה יהיה כאן - קוראים **רוצח** שתכח נטו על כך. מנחים בשליליה, מקבלים קופסה שחורה (לא ידועים מה קורה אליה בפנים, כאן $D - E$) וממנה יוצרים קופסה שחורה גדולה יותר שתוביל לסתירה.

טענה: E לא קבילה.

הוכחה: נוכיח ש \overline{E} קבילה, ואז נב"ש כי E קבילה, ונקלט כי לפי משפט L ו- \overline{L} קבילות $L \iff \overline{L}$ קבילות \overline{E} לא כריעה).

כעת נוכיח \overline{E} קבילה: $\{Q | L(Q) \neq \emptyset\} = \overline{E}$. נבנה תוכנית:

A- $\overline{E}(Q)$:

if q is not a program return 1
guess $w \in \Sigma^*$
return ($U(Q, w)$);

נעיר כי התוכנית ה"ל אינה דטרמיניסטית. תחילתה היא מוחשת מהירות w כלשהי, ואז היא שולחת אותו למוכנה הווירטואלית. אם השפה לא ריקה - לבסוף היא תצליח לנחש מילה כלשהי. אם המילה ריקה: לעולם לא יוזר 1. ולכן סה"כ \overline{E} קבילה.

EQ השפה 10.6.2

$$EQ = \{(Q_1, Q_2) | L(Q_1) = L(Q_2)\}$$

אוסף כל התוכניות, כך ששפנתן זהה.

טענה: נב"ש כי EQ אינה קבילה.

הוכחה: נב"ש כי EQ קבילה. תהי $A - EQ$ שמקבלת את EQ . נבנה $A - EQ$ שמקבלת את E , אז נקבל סתירה - כיון EQ לא קבילה.
מה הרעיון? בידינו קופסה שחורה, $A - EQ$. נרצה להכניס לה שתי תוכניות Q_1, Q_2 ונשים לב שהיא תחזיר כן/לא/תתקע ולא תעזר. מן הקופסה הזה, נבנה קופסה גדולה יותר $A - E(P)$ שמקבלת את E . נגדיר E כ:

A-E(P):

$Q_1 = P$
 $Q_2 = "Q(x)\{return0\}";$
return (A-EQ(Q_1, Q_2))

נשים לב כי Q_1 וה Q_2 היא תוכנית עם שפה ריקה, כי תמיד מחזרה אפס. השפה של P שווה לשפה של Q_2 אמ"מ השפה של P וריקה. כמובן, מתקיים $L(Q_1) = L(Q_2) = \emptyset$ וגם $L(Q_1) = L(Q_2) = \emptyset$ אמ"מ AE מחזירה על P . 1. כמובן סה"כ יצרנו תוכנית Q_2 ששפנתה ריקה, השפה שלנו תהיה ריקה אמ"מ היא שווה לשפה Q_1 , זו בעיה שנייה לפטור כי EQ קבילה מהשלילה, סה"כ קיבילנו כי הראודקציה מקבלת את E , בסתירה.

טענה: לא כריעה כי אם שפה היא כריעה, היא בהכרח קבילה. ולכן גם ההפוך נכון:
- לא קבילה גורר לא כריעה(ה).

\overline{EQ} השפה 10.6.3

$$\overline{EQ} = \{(Q_1, Q_2) | L(Q_1) \neq L(Q_2)\}$$

טענה: לא קבילה. \overline{EQ}

הוכחה: נב"ש \overline{EQ} קבילה, ותהי $A - \overline{EQ}$ שמקבלת את \overline{EQ} .
نبנה $A - \overline{ATM}$ שמקבלת את \overline{ATM} (או סתירה, כי היא לא קבילה). נזכר כי $\overline{ATM} = \{P(w) | P(w) \neq 1\}$. ושוב, ברודקציה: קופסה קטנה שלנו היא $A - \overline{EQ}$, שמקבלת זוג תוכניות. נרצה לבנות קופסה גדולה יותר, שבහינתה תוכנית וקלט תמים נכון. נכתב:

A- $\overline{ATM}(P, w)$:

$Q_1 =$
 $Q_2 =$
return $A - \overline{EQ}(Q_1, Q_2)$;

מה הרעיון מאחורי הוכחה? להתחיל כבסיס נ"ל, ואז להמיר נכונה את התוכניות. הרעיון התהיליך הוא מקבלים זוג (P, w) וזוג (Q_1, Q_2) ומשתמשים בהם לקבל את \overline{ATM} בסתרה. נרצה זוג כזה ש $L(Q_1) \neq L(Q_2) \iff P(w) \neq 1$.

A- $\overline{ATM}(P, w)$:

$Q_1 = "Q_1(x) : \text{return}(\text{U}(" + \text{P} + ", " + \text{w} + "));"$

$Q_2 = \{Q_2(x) : \text{return}(1);$

$\text{return } A - \overline{EQ}(Q_1, Q_2);$

כאשר Q_2 תחזיר תמיד 1, ככלומר השפה שלה היא כל ${}^*\Sigma$. ולעומת זאת, Q_1 תחזיר את תוצאת המוכנה האוניברסלית על הקלט (P, w) . וכך -

$$L(Q_1) := \left\{ \begin{array}{ll} \Sigma^* & P(w) = 1 \\ \emptyset & P(w) \neq 1 \end{array} \right\} = \left\{ \begin{array}{ll} \Sigma^* & (P, w) \in \overline{ATM} \\ \emptyset & (P, w) \notin \overline{ATM} \end{array} \right\}$$

מדוע? אם $P(w) = 1$, זה נכון לכל קלט, כי הרצנו תוכנית. ולכן השפה תהיה ${}^*\Sigma$. נשים לב כי $\emptyset \neq L(Q_2) \iff (P, w) \in \overline{ATM}, L(Q_2) \neq \emptyset$, מתקיים מכאן $L(Q_1) \neq L(Q_2) \iff (P, w) \in \overline{ATM}$. מכאן שההשובה שמוחזרת בשורה return $A - \overline{EQ}(Q_1, Q_2)$; הינה 1 אם $P(w) \neq 1$, ולכן המוכנה מקבלת את \overline{ATM} , בסתרה.

הערה חשובה: כיצד מוגדרת $\overline{EQ}(Q_1(x))$? לכל קלט x , תמיד יוחזר תוצאה המוגנה הוירטואלית - וכן אם התוצאה יצא אחד, ככלומר $(P, w) \in \overline{ATM}$, אז כל מילה בשפת הא"ב שננו התקבל ולפניהם השפה תהיה ${}^*\Sigma$, מנגד - אם לא יצא אחד אז אף מילה לא התקבל.

טענה: לא כריעה כי אם שפה היא כריעה, היא בהכרח קבילה. ולכן גם ההפוך נכון:

- לא קבילה גורר לא כריעה.

10.7 פונקציות לא חשיבות

קיימות פונקציות שלא ניתנות לחישוב ע"י אף מכונת טיריניג (כלומר, אף מחשב).

דוגמה. נתבון בפונקציה הבאה:

יהי Σ_1 הא"ב של האותיות שדרושות לכתיבה בשפה SIMPLE. יהי $\Sigma_2 = \{0, 1\}$. נגיד $X_E : \Sigma_1^* \rightarrow \Sigma_2^*$ ע"י

$$X_E(x) := \left\{ \begin{array}{ll} 1 & x \in E \\ 0 & x \notin E \end{array} \right\}$$

פונקציה זו תקרא הפונקציה המציינת של E או אותה E לעילן. פונקציה זו אינה ניתנת לחישוב ע"י אף מכונת טיריניג. שחררי, אם קיימות מכונות טיריניג M שמחשבת אותה, ניתן לבנות ממנה מכונה M' שמכריעה את E , בסתרה כי E לא כרעה.

הערה. האם קיימות פונקציות מהטבעיים לטבעיים גם שלא ניתנות לחישוב? ודאי. הפונקציה לעיל היא של מחזוזות, ניתן לקודם לערך מספרי לפ' טבלת אסקי למשל. מינוח. לפונקציה שנייתנת לחישוב ע"י מכונת טיריניג נקרא פונקציה חשיבה.

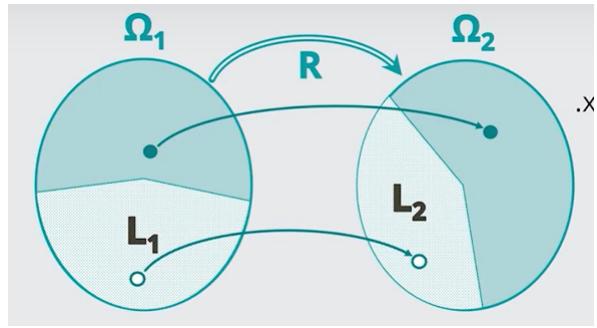
10.8 רדוקציות והוכחות ברדוקציות

כלל הדוגמאות שהשתמשנו להוכיחו של אי כריעות/אי קבילות בדוגמאות הקודמות נקראות רדוקציה.

הרעון של רדוקצייה הוא פתרון בעיה אחת, באמצעות פתרון מוכר של בעיה אחרת. בעת נתאר באופן פורמלי את מושג הרדוקציה:
גדירה: רדוקציית התאמה היא - יהו שני מרחבים, Ω_1 , Ω_2 .
 Ω_1 הוא מרחב הביעות אני מעוניין לפתרו, Ω_2 הוא מרחב הביעות שיש לי פתרון עבורם. תהי
 $L_1 \subseteq \Omega_1$, $L_2 \subseteq \Omega_2$.

$$R : \Omega_1 \rightarrow \Omega_2$$

כך לכל $x \in L_1$ מתקיים $R(x) \in L_2$ $\iff R(x) \in L_2$. כלומר - המקור ב- L_1 אמ"מ התמונה שלו ב- L_2 .
נשים לב - הפונקציה שומרת על השיקות לקבוצות המתאימות. נשים לב כי רדוקציה מ- L_1 אל L_2 אינה פונקציה מ- L_1 ל- L_2 . אלא מן המרחב ש- L_1 נמצא בו למרחב ש- L_2 נמצא בו.



סימונו: אם יש רדוקציית התאמה ניתן לחישוב $L_2 \preceq_m L_1 \rightarrow$ איזי נסמן $L_2 \preceq_m L_1$

טענה: תהיינה L_1, L_2 שפות. אם:
 L_2^* כריעה
 $L_1 \preceq_m L_2^*$
 איזי כריעה.

הוכחה: תהי $D = L_2 - L_1$ התוכנית שמכריעת את L_2 . בהינתן $x \in \Omega_1$ נרצה לדעת האם $x \in L_1$ נחישב $L_2(y) = R(x)$.
 1. נחישב $y = R(x) \in L_2$. כיון שגם $R(x) \in L_2$ ו- L_2 רדוקציה, $x \in L_1 \iff R(x) \in L_2$.
 2. נחישב $x \in L_2$. כנדרש.

טענה: תהיינה L_1, L_2 שפות. אם:
 L_2^* קבילה
 $L_1 \preceq_m L_2^*$
 איזי קבילה.

מסקנות: אנו משתמשים ברדוקציה בדרך השילילה וכך גם יהיה ב מבחנו. נתרגם את הטענות לעיל:
תהיינה L_1, L_2 שפות. אם: L_1 לא כריעה וכן $L_2 \preceq_m L_1 \rightarrow$ איזי כריעה.
תהיינה L_1, L_2 שפות. אם: L_1 לא קבילה וכן $L_2 \preceq_m L_1 \rightarrow$ איזי לא קבילה.

כיצד נוכחה שפה לא כריעה/קבילה? נמצא שפה L_1 לא כריעה/קבילה, ונבנה רדוקציית התאמה נתנת לחישוב L_1 מ- L_2 .

10.8.1 רזוקצייה משפה בריעת

טענה: תהי A שפה בריעת. אזי, קיימת רזוקצייה התאמה ניתנת לחישוב $.A \preccurlyeq_m ATM$

הוכחה: תהי A בריעת. אזי קיימת D^A המכריעת אותה. נבנה רזוקציה R מ- ATM ל- A כך:

$R(w)$:

```
if  $D^A(w) == 1$ :  
    return " $Q(x)\{return(1);\};$ " aba";  
else  
    return " $Q(x)\{return(0);\};$ " aba";
```

נשים לב כי הרזוקציה מקבלת מילה, ומוריצה את המכונה על המילה. אם $w \in A$ קיבלה, אזי $D^A(w)$ מוכנה שלא מקבלת דבר, ומילאה, ATM החרינו תוכנה שמקבלת כל מילה וアイיר - שבספרט בתוכנה כי היא מקבלת כל מילה. אחרת, D^A לא קיבלה ולכן גרצה להציג זוג שלא מתאים ל- ATM - מוכנה שלא מקבלת דבר, ומילאה, שודאי לא שייכת אליה כי היא לא מקבלת דבר. נשים לב כי הרזוקציה חישובית ומוגדרת היטב - והיא מכיריה, היא תמיד תעזר. לכן הרזוקציה R תמיד תחזיר קלט. כמו כן, ניתן לראות כי $x \in A \iff R(x) \in ATM$, כיון שאחרת לא נקלט זוג שמתאים ל- ATM . סה"כ בנו רזוקציה שתמיד עבדת וקיים.

מסקנה: באופן דומה, ניתן לבנות רזוקציה התאמה ניתנת לחישוב מכל שפה בריעת לכל שפה לא בריעת (פרט לשפה הריקה ול Σ^*). כמו כן, מכל שפה לא קיבלה לכל שפה לא קיבלה.

נשים לב כי שפה בריעת, לא ניתן למצוא רזוקציה ניתנת לחישוב אל השפה הריקה. שכן הרזוקציה צריכה להציג לכל פלט $\in A$, פלט בשפה שנשלחנו אליו. בשפה הריקה אין פלטים כלל, ולכן לא קיימת רזוקציה כזו. באופן דומה - לא קיימת רזוקציה משפה בריעת אל Σ^* .

נשים לב - פונקציית הזוזות היא רזוקציה ניתנת לחישוב כאשר $L_1 = L_2$ (כלומר, מאותו הקבוצה לאוותה הקבוצה).

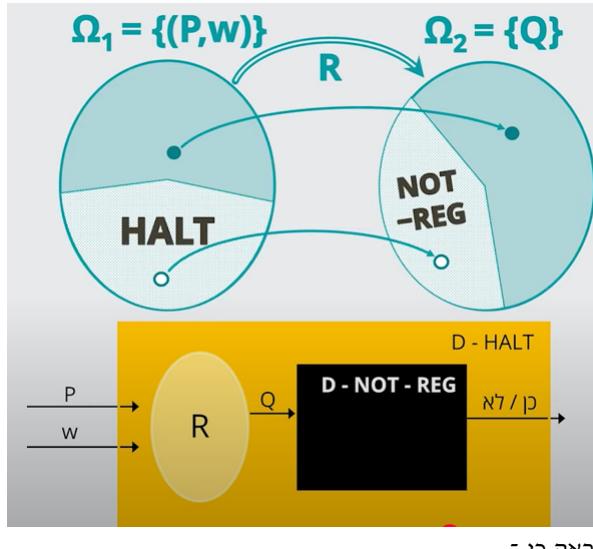
NOT – REG השפה 10.8.2

נדיר $\{Q\}$ לא רגולרית $.NOT – REG := \{Q | NOT – REG \in \{Q\}\}$.
למשל, שפת הפלינדרום, כיוון שלא רגולרית. עם זאת, $.palindrome \in NOT – REG$. עם זאת, $startA \notin NOT – REG$

טענה: $NOT – REG$ לא בריעת.

נראה כי $HALT \preccurlyeq_m NOT – REG$

נשים לב כי $(P, w) \in NOT – REG \in \Omega_2 = \{Q\}$ וכן $HALT \in \Omega_1 = (P, w)$ וכון $HALT \in NOT – REG$ ו- (P, w) תחזר $R(P, w)$ רצחה כי הפונקציה R מ- Q ש- $\in \Omega_2$ (כלומר, בהינתן זוג (P, w) רצחה כי $R(P, w) = P$) ו- (P, w) עוצרת. ובתיאור - $HALT \iff Q \in NOT – REG$



נראה כי -

$R(P,w)$:

```
"Q(x){  
U(P,w)  
return Palindrome(x));}
```

עונה על הדרוש. ראשית היא מקבלת זוג (P, w) , והיא קוראת למוכנה הווירטואלית, ואז מרים את התוכנית פלינדרום על הקלט איקס ומחרירה תשובה מתקבלת. זו רדוקציה שניתנית לחישוב, שכן מקבלים זוג מחרוזות, ומחרוזים מחרוזות בודדות.
חשוב לציין מה שקרה ב- R לא מרצה את המחרוזת (x) . נשים לב כי -

$$L(Q) := \left\{ \begin{array}{ll} \emptyset & P(w) \uparrow \\ L(palindrome) & P(w) \downarrow \end{array} \right\}$$

השפה של Q תלויה במקרה בשרה (P, w) . אם החישוב מסתיים, נגיע לשורה האחורונה ונחיזיר את שפת הפלינדרומים. אחרת, נעלם לא נעצור ונגע לשורת ההחזרה והשפה תהייה ריקה. סה"כ קיבלנו מה שרצינו - $L(Q)$ לא רגולרית אם $\downarrow P(w)$ עוצרת. מדוע בחרנו בשפת הפלינדרומים? כי היא לא רגולרית, יכולנו לבחור כל אחת אחרת שאינה רגולרית וניתנת לחישוב - וזה היה עובד.

טענה: $NOT - REG$ לא קבילה.

הוכחה: נראה כי $\overline{ATM} \not\leq_m NOT - REG$.

נשים לב כי $(P, w) \in \Omega_2 = \{Q\}$ וכן $\overline{ATM} \in \Omega_1 = (P, w)$

כלומר, בהינתן זוג (P, w) נרצה כי הפעונקציה R תחזיר Q כך ש -
 $P(w) \in L(Q) \iff \overline{ATM} \in NOT - REG$. כלומר, לא רגולרית אם $\downarrow P(w) \neq 1$.
נשים לב כי אכן \overline{ATM} לא קבילה, ולכן אם נוכחים שקיימות צו לפि טענה מההרצאה - אכן $NOT - REG$ לא קבילה גם היא.

אנחנו מוחשיים ליצור תוכנית Q , שהשפה שלה אינה רגולרית, אם $\downarrow P(w) \neq 1$. נסתכל על התוכנית הבאה (ונעיר, יש הרבה תוכניות שיכלות להתאים. כמו תמיד!)

$R(P,w)$:

```
"Q(x){  
if palindrom(x)==1
```

```

return(1);
return U(P,w);

```

ראשית התוכנית בודקת האם המחרוזת פלינדרום, אם כן היא מוחזירה אחד. אחרת - לא פלינדרום:
מראיצה מכונה וירטואלית עם התוכנית והקלט. נשים לב כי

$$L(Q) := \left\{ \begin{array}{ll} L(\text{palindrome}) & P(w) \neq 1 \\ \Sigma^* & P(w) = 1 \end{array} \right\}$$

כיוון שם $P(w) = 1$, אז יוחזר 1 עבור פלינדרומים ו 0 עבור שאר הקלטים - סה"כ כל Σ^* .
אם $P(w) \neq 1$, אז השפה שתתקבל היא שפת הפלינדרומים בלבד. סה"כ $L(Q)$ לא רגולרית (ושווה לשפת הפלינדרומים (אם $w \neq P(w)$) כי Σ^* כן רגולרית), ולכן הרדוקציה מתאימה ונכונה.

**עוד דוגמאות לרדוקציות - הייתה עצמן בשבייל לכתוב כאן. אבל בשיהיה כוח
להעתיק לבאן עוד דוגמאות מהקמפוס!!!!!!**

!!!!!!
!!!!!!
!!!!!!
!!!!!!

10.9 סיכום - השפות הלא כרייעות/קבילות

| קבילה | כריעה | |
|-------|-------|-------------------|
| ✓ | ✗ | ATM |
| ✗ | ✗ | \overline{ATM} |
| ✓ | ✗ | HALT |
| ✗ | ✗ | \overline{HALT} |
| ✗ | ✗ | E |
| ✗ | ✗ | EQ |
| ✗ | ✗ | \overline{EQ} |
| ✗ | ✗ | NOT - REG |

10.10 בעיית הנחש

הຮושים שנוצר עד כה - הוא שכל הבעיות הלא כרייעות קשורות לתוכניות. ובכן, זה לא נכון. דוגמה לכך היא בעיית הנחש.
נתונים לנו ריבועים כנ"ל:



אריחים עם צבעים שונים, כל אחד מורכב מ-4 אריחים שונים.
כמו כן, נתונות לנו שתי משבצות כמו זו:



השאלה היא, האם ניתן לשבץ שני אריחים בשתי המשבצות, כך שיתאימו זו לזו. כשהכוונה ביטאיםו - אפשר לחבר שני אריחים זה ליה רק אם הצבעים בפאות הנוגעות, זהים. למשל: בדוגמה מעלה, אפשר לחבר את הימני והאמצעי, כי הפאות הצדדיות שלהן הקרובותצבעם צהוב. כמו כן, אפשר גם בין המשבצות הללו ליצור צירוף של יותר משתי אריחים - כמה שרוצים, כל עוד הכלל נשמר ואנחנו מתחילה במשבצת הראשונה ומסימאים בשניה. נשים לב כי אסור לסובב את האריחים ומותר להשתמש כמו שרוצים באוטו סוג אריח. כמו כן - גם מסלולי נחש כאלו חוקיים:



יש מסלולים - ללא אפשרות לצירמת מסלול.

פורמלית, הבעה תתואר כך:

- * קבוצת אריחים (t_1, \dots, t_n) כ- $T = (t_1, \dots, t_n)$ כל אריח מיוצג ע"י 4 מס' טבעים שמייצגים את הצבעים (כasher כל אריח $t_i \in \mathbb{N}^4$) כל אריח מוצג ע"י 4 מס' טבעים שמייצגים את הצבעים)
 - * זוג נקודות קצה $\mathbb{N} \times \mathbb{N}$ בין $p, q \in \mathbb{Z} \times \mathbb{Z}$
- הבעיה היא: { יש נחש תקין של אריחי T בחצי המישור העליון שמחבר בין p לבין q }

$$|(T, p, q)|$$

טענה: השפה Half-Snake לא כריעה, אך כן קבילה (ניתן לנחש מסלול מחבר, ולבסוף אם הוא תקין).

נשים לב כי אם נגדיר { יש נחש תקין של אריחי T בכל המישור שמחבר בין p לבין q }, זו בעיה כריעה.

10.11 התוכנית של הילברט

נזכר בתוכנית הילברט מהיחידה הקודמת: לבנות יסודות פורמליים למתמטיקה. ראיינו כי היעד

השלישי - כריעות: למצוא אלגוריתם שקובע האם טענה היא נכונה או שלא, חיכה ליחידה הנוכחית.
ובכן, טירינג לא מצא את האלגוריתם שהילברט חיפש - טירינג הוכיח שאין אלגוריתם כזה, ולא
יכול להיות. ניתן להסביר את הדברים מנקודת מבטו של שריאנו ביחידה זו: ראיינו כי למשל *ATM* לא כריעה
- וכן אין אלגוריתם מתמטי או תכני, שידעו להכריע את הבעיה. (}):