

# סיכום הרצאות למחן - מודלים חישוביים

23 בדצמבר 2025

גיא יעראן.  
הסיכום נכתבו לאורך הרצאות *campus* והתוגולים בסמסטר א' שנת 2026 תשפ"ו, لكن ייתכו  
שכלו טעויות לאורך כתיבת הסיכום - כה שעלה אחריהם.

## 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   |   |
| 50 | דקדוקים רגולריים ושפות רגולריות . . . . .          | 6.5   |   |
| 52 | יחידה 8: מכונות טיריניג . . . . .                  | 7     |   |
| 52 | מהי מכונות טיריניג? . . . . .                      | 7.1   |   |
| 54 | הגדרה פורמלית . . . . .                            | 7.2   |   |
| 54 | קונפיגורציה: . . . . .                             | 7.2.1 |   |
| 54 | גרירה: . . . . .                                   | 7.2.2 |   |
| 54 | קבלה ודחיה של מחרוזות: . . . . .                   | 7.2.3 |   |
| 55 | הכרעה של שפה . . . . .                             | 7.2.4 |   |
| 55 | קבלה של שפה . . . . .                              | 7.2.5 |   |
| 55 | תיאור מכונות טיריניג באמצעות טבלת מעברים . . . . . | 7.3   |   |
| 58 | תיאור מכונות טיריניג באמצעות פסודו קוד . . . . .   | 7.3.1 |   |
| 59 | שימוש במכונות טיריניג לחישוב פונקציות . . . . .    | 7.4   |   |
| 60 | יחידה 9: וריאציות של מכונות טיריניג . . . . .      | 8     |   |
| 60 | מודל חישובי - הגדרה פורמלית . . . . .              | 8.1   |   |
| 60 | סרט ימינה בלבד . . . . .                           | 8.2   |   |
| 62 | מודל <i>TS</i> . . . . .                           | 8.3   |   |
| 63 | מודל <i>OR</i> . . . . .                           | 8.4   |   |
| 64 | מכונות טיריניג מרובת סרטים . . . . .               | 8.5   |   |
| 66 | סגורות לאחד וחיתוך שפות כריעות/קבילות . . . . .    | 8.6   |   |

|    |  |        |       |
|----|--|--------|-------|
| 67 | automat עם שתי מחסניות . . . . .             | P2     | 8.7   |
| 67 | סרט דו ממדדי - מודול 2D . . . . .            | 2D     | 8.8   |
| 69 | מכונית טיריניג שאינה דטרמיניסטיבית . . . . . |        | 8.9   |
| 70 | הכרעה וקבלה של שפות . . . . .                | 8.9.1  |       |
| 70 | סגורות באמצעות אידטרמיניזם . . . . .         | 8.10   |       |
| 71 | יחידה 10: התזה של צץ'-טיריניג . . . . .      |        | 9     |
| 71 | 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    |       |
| 75 | התזה של צץ'-טיריניג . . . . .                | 9.6    |       |
| 76 | יחידה 11: אי כריעות . . . . .                |        | 10    |
| 76 | איימות תוכנה . . . . .                       | 10.1   |       |
| 76 | ATM קבילה . . . . .                          | 10.2   |       |
| 77 | ATM לא כריעה . . . . .                       | 10.3   |       |
| 77 | שפה שאינה קבילה . . . . .                    | 10.4   |       |
| 78 | בעיית העצירה . . . . .                       | 10.5   |       |
| 78 | שפות לא פתרות . . . . .                      | 10.6   |       |
| 79 | E השפה . . . . .                             | 10.6.1 |       |
| 79 | <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  |
| 82 | 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 סוגי בעיות שעשוויות 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 קשיר. 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 ונויותו, השאלה שתענין אותנו בהמשך - האם יש בעיות שלא ניתן לפתורן באמצעות מחשב. סיבה נוספת להתusalem, היא שככל בעיות חישוב ניתן להמיר לבאית הכרעה.

כיצד נמיר בעיות הכרעה לבאית חישוב? נניח ואנו ידעים 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 ייזכג ע"י סדרה של תווים. התווים **שמהם נוצרת סדרה קרוא האלפבית**. או פשוט **א"ב**. נסמן באות  $\Sigma$ . **בקורס זה הא"ב יהיה תמיד סופי**. את אותיות הקלט נסמן לעיתים באותיות יווניות -  $\sigma, \tau$ .

**למשל** - עבור הבעיה האם  $x$  ראשוני?", הא"ב יהיה  $\{0, 1, 2, \dots, 9\} = \Sigma$ . עבור הבעיה, האם הגף קשור?" נזדקק לא"ב  $\{\cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot, \cdot\} = \Sigma$  כיון שגרף מייצג ע"י קבוצת קודקודים  $\{(1, 2), (2, 3)\}$  לדוגמה וקשותות  $\{1, 2, 3\}$

### 1.2 מחרוזות

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

המילה הריקה - מסומנת באות  $\epsilon$ . מתקיים  $\epsilon \neq \emptyset$ , 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  $\sigma$  נסמן  $\#$  את מס' הפעמים שהאות  $\sigma$  מופיעה w.

**קבוצת כל המחרוזות מעלה  $\Sigma$  - תסומן  $\Sigma^*$ .** למשל,  $\{a, b\}^* = ab$  בעוד  $ac \notin \Sigma^*$  וכן  $a \in \Sigma$ .  
**קבוצת כל המחרוזות הלא ריקות מעלה  $\Sigma$  תסומן ותוגדר:**

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

$$\text{הטענה לא נcona עבור שפות - לא מתקיים } L^+ = L^*/\epsilon \text{ אם } \epsilon \in L.$$

### 1.3 פעולות על מחרוזות

**א. שרשור - יסומן  $\circ$ .** השרשור היא פעולה פשוטה. למשל,  $ab \circ bba = abbba$ . **דוגמא נוספת:**  $ab = ab \circ \epsilon$ . כלומר, השרשור של מילה עם המילה הריקה נותן את המילה עצמה.

**ב. חזקה -  $w^n$ .** ההגדרה היא -

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

חזקה היא שרשור של המילה עם עצמה  $n$  פעמים. למשל,  $(ab)^3 = ababab$ . מתקיים  $\epsilon^0 = \epsilon$  ו-  $\epsilon^0 = w$ .

### 1.4 שפות

cutת 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. מתקיים 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  $\Sigma^*$ .
- 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  $\Sigma^*$  היא הסגור קlien של השפה  $\Sigma$ .  
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  $\Sigma$  את  $\Sigma^n$  כקבוצת כל המילים מעל א"ב  $\Sigma$  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 למעלה 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  $\Sigma^*$ . 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? נגידר שפה -

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

כעת, בעיית ההכרעה האם מס' ראשוני תומר לבעה הבאה: האם  $w \in Prime$ ? המסקנה - כל בעיית הכרעה תוכל להיות מומרת לשפה. ואז תמיד הבעיה תומר לשאלת בינהית: האם הקלט בשפה או שלא. **מבחן נגיד למסקנה נספח, 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)

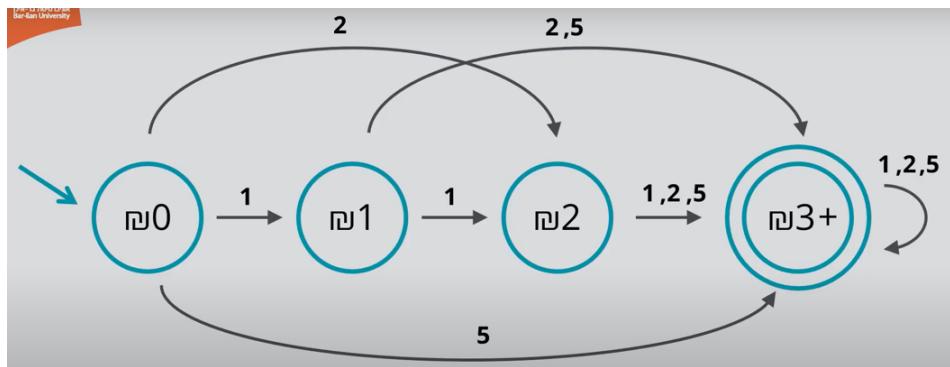
ביחידה זו נדון במושג המחשב. קודם לכן, דנו במושג בעיה=שפה. כעת נדבר על מודול חישובי כלשהו, לא מחשב ספציפי. היכולות החישוביות של האוטומט הסופי, די מוגבלות. אז מדוע נלמדו? יש לו שימושים מרכזיים בעולם האמיתי, והוא ישמש אותנו בחימום והכוה למודלים המורכבים בהמשך.

**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 שהנכנסנו מספיק? נתבונן ב"תרשים" מטה, המצביע ההתחלתי, הוא 0 שקלים, תמיד נהייה בו וначילה ממנו, וכך נסמננו בחץ - שנದע מהיקן להתחיל. המצביע הסופי, יקרה +. לשם אנחנו שואפים להגיע, ולכן נקייף בעיגול פעמיים את מצב המטרה שלנו. כמו כן, נסיף לולאה פנימית, אם נגיע למצביע המטרה ונקבל מטבעות, נרצה להשאר בו. התרשימים מטה די ברור, 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:

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

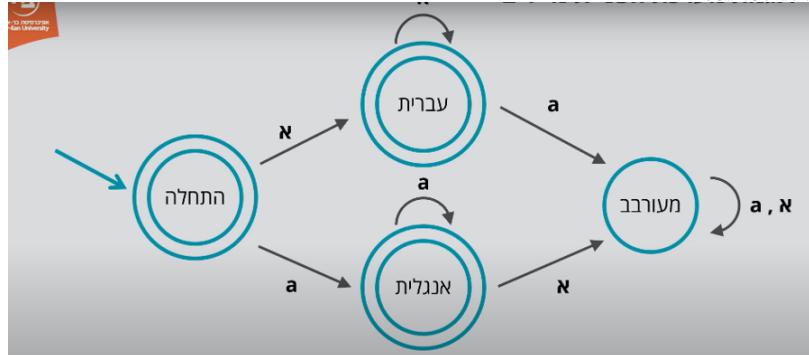
$$L = \{\sigma_1\sigma_2\dots\sigma_n \mid \sum_{i=1}^n \sigma_i \geq 3\}$$



**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. ככל 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 למציב. 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 שאלעד 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.

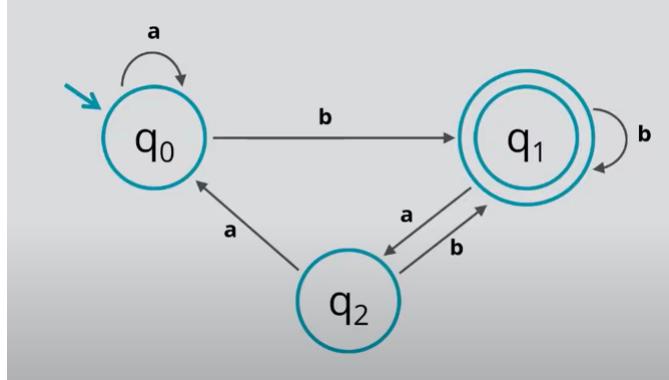
**מה זה דטרמיניסטי?** בכל שלב, ובכל מצב, ישנה אפשרות אחת בלבד עבור אות קלט لأن להתקדם.  
 לא צריכה להיות בחרה של השימוש או של המחשב עצמו באיזה כיוון כדי לו להתקדם.

### 2.3 שפת האוטומט

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

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

כלומר עת נשתכל על  $\Sigma^*$ , 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  $\delta$  נקבעים באופן מלא בהתאם לערכי  $\delta$ .

**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, ונתפרק 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.

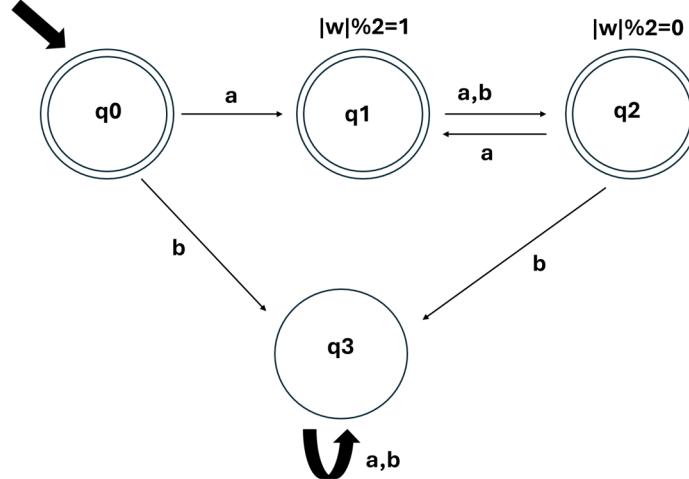
**אוטומט שלד:** נניח ונרצה לבנות אוטומט לשפה מעלה  $\{a, b\}^*$   $= \Sigma = \{w | w = aa\sigma_1\sigma_2\dots\dots\}$ . כלומר, כל המילים שנפתחות ברצף "aa". מהי המילה שבודאי תופיע שם?  $aa$ . אנחנו רוצה לבנות אוטומט שלד, שיקבל את המילה  $aa$ , ואנו נרჩיב ונתאים למצבים השונים ולמקרי הקצה. למשל - **קיים מצב מלבדת:** כל מילה שלא פתחה בא", נרצה 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? ניצור  $q_0$  מקבל, אשר קיבל  $a, b$  באותיות קלט, נ עברו במצב מלבדת. ושם יהיה עצמי עבור אותיות הקלט. לעומת זאת, ברגע שנקלט  $a, b$ , תס היפויו - הלבנו במצב מלבדת.
- השפה  $* \Sigma$  - כל המילים יתקבלו. בצד יראה אוטומט זה? בדוק כמו ב-1, רק עם מצב מקבל. כמובן, כאן אנחנו כן נרצה לקבל כל מילה שקיים בשפה.

#### דוגמא של אוטומט:

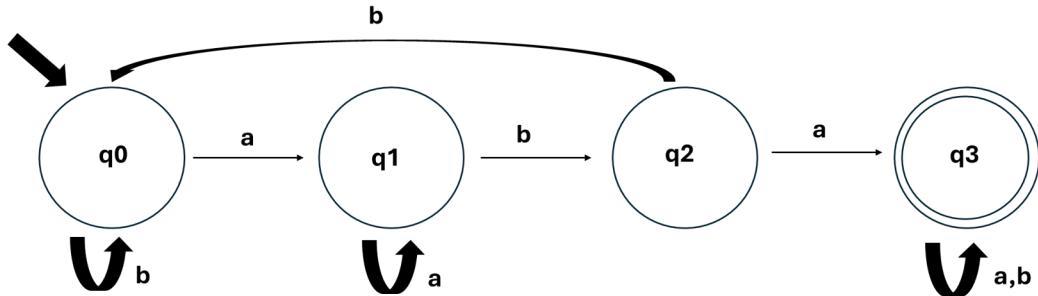
יהי  $\Sigma = \{a, b\}^*$ . נגיד את  $L$  להיות השפה של כל המילים מעלה  $\Sigma$  בהן האות  $b$  אינה במקומות הזוגי. נראה כי האוטומט שלנו יהיה:



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

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

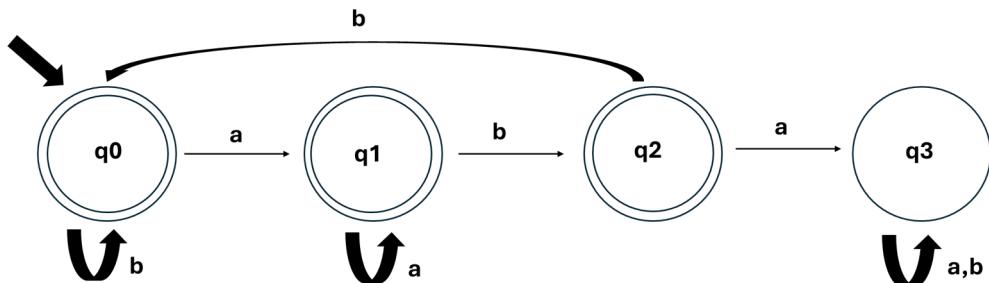
$$L = \{w = u_1aba u_2 : u_1, u_2 \in \Sigma^*\}$$



**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:

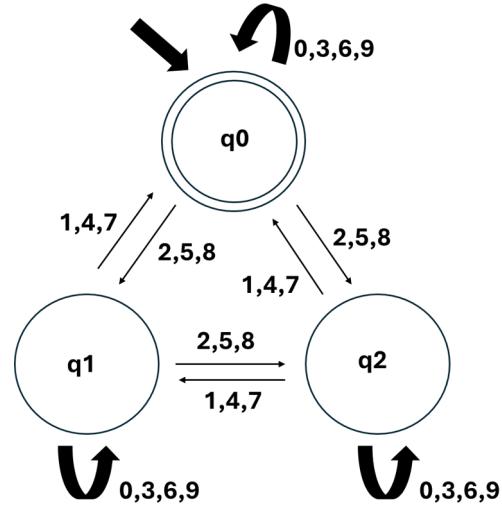


בשיטתה זו נוכל להשתמש **תמיד** כאשר נרצה לבנות אוטומט לשפה המשילימה, בהינתן שאנחנו ידעים את האוטומט לשפה המקורית. וכך, כמו בהסתברות למשל, נוכל לבנות אוטומט לשפה וואז להפעיל עליה שלילה", רק שכן במקום בצע  $p - 1$  נשנה את המצבים המתקבלים.

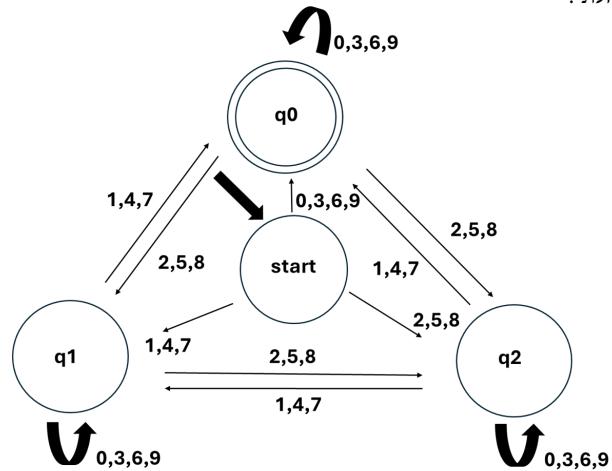
**דוגמה 4.** נרצה לבנות אוטומט  $A$  שיקבל את השפה הבאה מעל אל"ב  $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

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

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 -

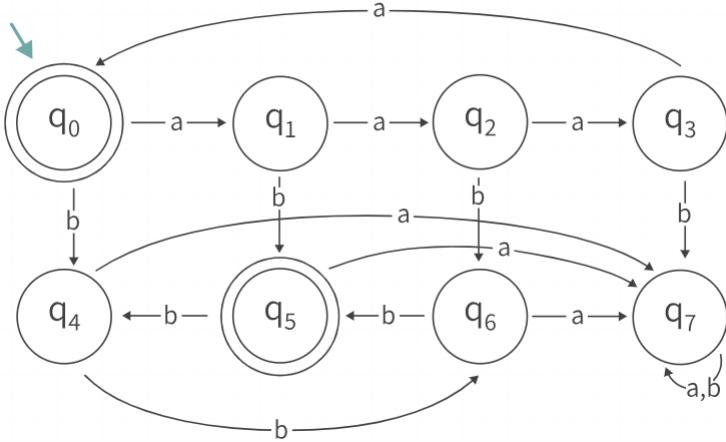


נשים לב, האם זה אוטומט מספק עבורנו? כמובן. מה עם המילה הריקה? מדוע היא במצב מקבל? הרי מילה ריקה איננה מתחילה בשלוש. ולכן, נצטרך להוסיף מצב נוסף להיוות המצב ההתחלתי:



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 נתעכבר. 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:  $\exists A$  אוטומט,  $\exists B$  אוטומט  $B$  כך  $L(A) = \overline{L(B)}$   
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  $L_1(A) = L_1$  ו-  $L_2(B) = L_2$ .  
בננו 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- $\Sigma$ . 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  $\sigma$  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.

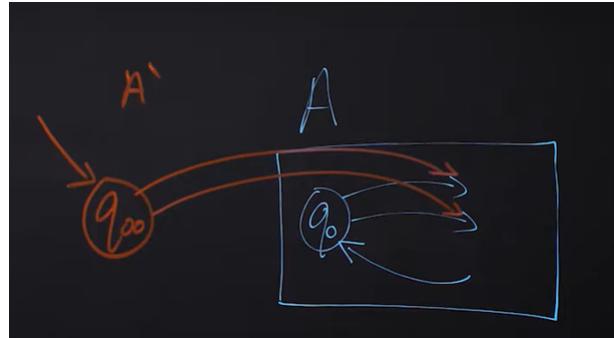
**מסקנה:** בהינתן שתי שפות רגולריות,  $\Sigma$  כל המניפולציות הבאות על השפות קיימים אוטומט שמקבל את המניפולציה: איחוד, חיתוך, משלים, הפרש סימטרי.

#### 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  $w = q_0 \delta^*(q_0, w)$ . 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  $\Sigma$ . 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  $\Sigma$ . הוכיח כי  $ZigZag(L_1, L_2)$  רגולרית מעל  $\Sigma$ .  
**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 ותגבע 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 שלישיה, 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 שלישיה ואות ותחזיר שלישיה.

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

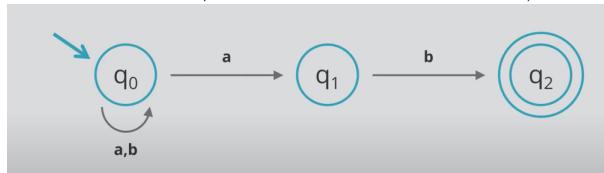
הטבר: כאשר אנחנו נמצאים בתוך  $A$ , נרצה לעcor בקורסיה הקלט הראה אל  $A$  וכן נפעיל את דלתא של  $A$ , נשאר בפ' כי לא צו  $B$  וכן ה  $sign$  ישתנה ל  $B$  כי אנו כרגע ב  $A$  וכקורסיה הקלט הראה נרצה לעcor אל  $B$ . **בזומה, עכו המקרה השני.**  
cut נגע אל קבוצת המצביעים המקבילים:

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

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 דוגמה. 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 עצמת 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 מקובל: למשל,  $aa = a$  לעומת  $w = aa$  לא הגיעו למצב מקובל.



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  $\Sigma$ . תהי  $* \in \Sigma^*$  מקבל את  $w$  אם ומן **קיימת** הרצה של האוטומט על המילה, שמסתיימת במצב מקלט.

הערה - נשים לב כי בשביל להכריע האם  $w$  לא שייכת לשפה, צריך לעבור על כל ההוראות 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,  $\Sigma$  הוא הא"ב,  $\Delta$  היא פונקציית המעברים,  $q_0$  הינו המצב ההתחלתי וכן  $F$  היא קבוצת המצביעים המקבלים וכמוון  $F \subseteq Q$ .

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

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:

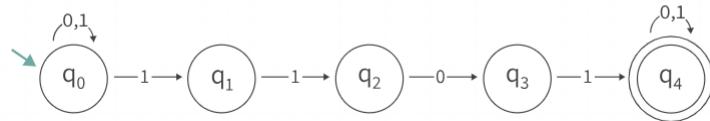
$$L(N) = \{w \mid \exists f \in F : f \in \Delta^*(q_0, w)\} = \{w \mid \Delta^*(q_0, w) \cap F \neq \emptyset\}$$

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 רגילים, 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".

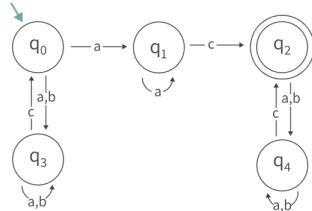


**דוגמא 2.** בניית אוטומט לא דטרמיניסטי שיקבל את השפה הבאה:

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

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: נבנה 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.

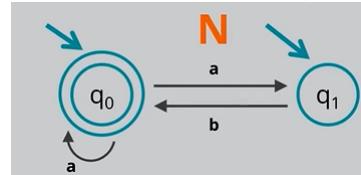
**כל שפה שניתן לקבל באמצעות אוטומט לא דטרמיניסטי, ניתן לקבל באמצעות אוטומט דטרמיניסטי.**  
נראה להוכיח טענה זו. לפני כן, נזכיר את המושג של אוטומט לא דטרמיניסטי.  
נדיר כעת אוטומט לא דטרמיניסטי  $N = \{Q, \Sigma, \Delta, Q_0, F\}$  כאשר  $Q_0$  היא קבוצת המ מצבים ההתחלתיים. ככלומר - אפשר יותר ממצב ההתחלתי 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  $D$ .

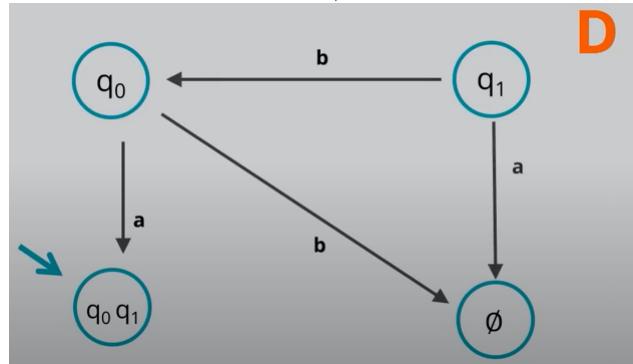
ראשית, נתחל בדוגמה שתעזר לנו במבנה ההוכחה.



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? נגעים לשאלה, עכשו המילוי שכלל מקרה - האוטומט המקורי היה נתקע 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.

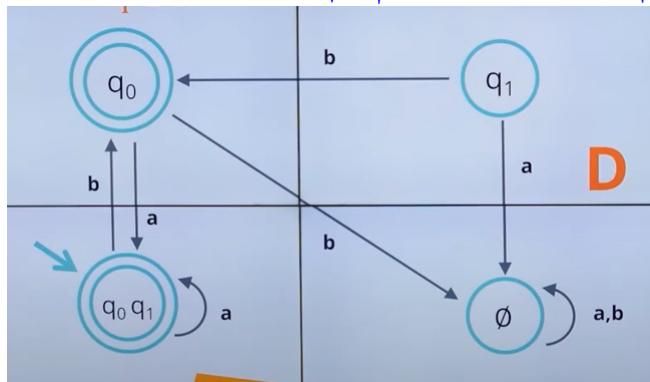


ובכן, זה האוטומט שלנו. נשים לב כי הוא עדין איננו דטרמיניסטי. עדיין, לא כל מילך מטופל בכל אפשרויות הקלט.

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- $q_0 \rightarrow q_0q_1 \rightarrow \dots$ . 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 תheid כשנגיון למילך של הקבוצה הריקה - נבער לו לא עפ"ת עם כל ס' - כיוון שבאוטומט המקיים נתקענו מפילה שאנו לשכז זה, וכן יויתן להסתכל על ה-"מצב מלוכחת". מה באשר למיצבים המתקבלים? נראה כי פילה הייתה בשפה שאוטומט  $N$  קובל אפ"ע היה קיים מסלול בו היה התקבלו, וכך כי  $bN$  ורק  $q_0$  הוא מצב מקבל - וכן גם אצלנו הוא והוא מצב מקבל. כמו כן, אם הגיעו למצב  $q_0q_1$  משמע שכאוטומט המקור היה דורך למתיקל 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  $D$ :  
א. 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 הרו' שם המיצב) ואות  $\sigma$ , נרצה לעבור לקבוצה שהיא איחוד כל המיצבים, כך שבהינתן  $q \in R$ , הפונקציה שולחת אליו. ובמילים פשוטות יותר - נאخد את כל הדריכים האפשריות למעבר הקלט שששייך לקבוצה, זו קבוצה שהיא וודאי חלק מ( $Q^N$ )  $P$  וכעת היא שם של מצב ב  $D$  לפי  $\sigma$ , וזה המצב אבל אחרי שטי קריואת אפשר להבini.  
**חשוב להתעכ卜 על חלק זה בהוכחה כי זה קצת מסורבל אבל אחרי שטי קריואת אפשר להבini.**  
**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

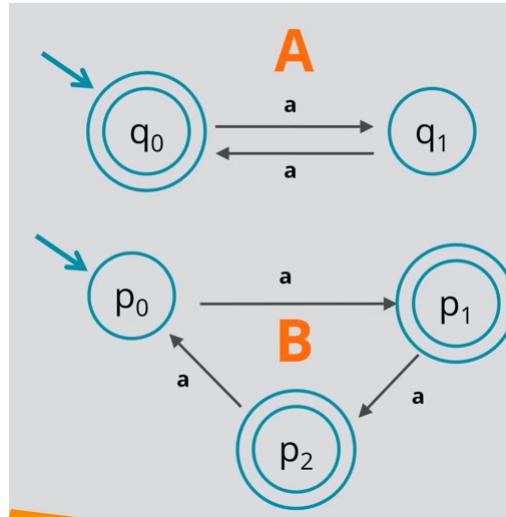
$$F^D = \{R \in Q^D : R \cap F^N \neq \emptyset\}$$

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 מזוע 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

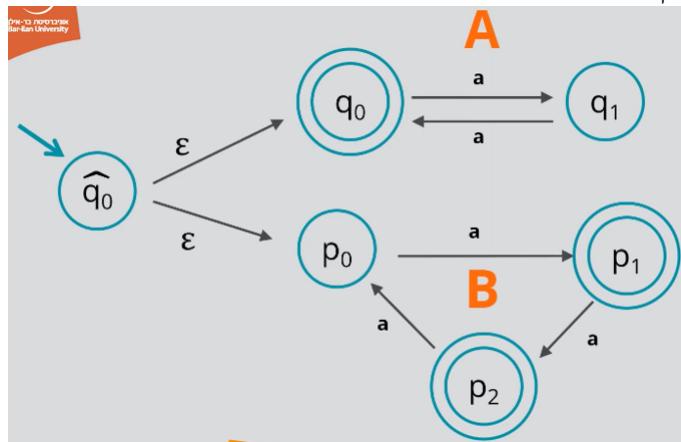
נרחיב את המושג של האוטומט הסופי. ובפרט, את המודל הלא דטרמיניסטי (שלפי טענה לעיל שקיים לאוטומט דטרמיניסטי).  
**הרעיון הבסיסי** הוא לאפשר מעבר באוטומט ללא קריית אף אותן קלט, כשנרצה לעשות זאת, נסמן זאת באמצעות האות  $\epsilon$ . על קשותות אלו ניתן לעبور ללא קריית קלט באוטומט. נדוגש כי לא **חייבים** לעبور על מעברי האפסילון, וכן **יתכן מצב שייהי לנו על מעבר  $\epsilon$** , למשל ונרצה לבחור האם לדלג 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:

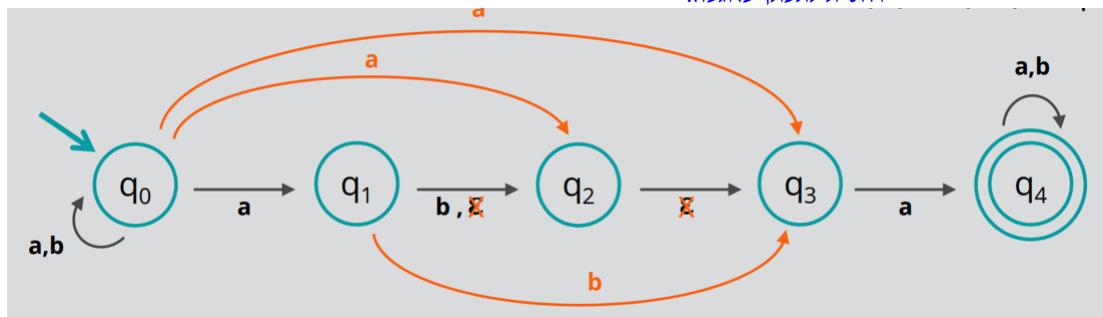


icut האוטומט מקבל את השפה  $C$ , ובחירה באופן לא דטרמיניסטי להיכן לכלכת. ביטלו את המ מצבים  $p_0, q_0$  ו  $q_0$  מכובדים התחלתיים והוספנו  $\hat{q}_0$  כמצב התחלתי חדש, עם שני מעברי אפסילון.icut, אם נרצה גם אוטומט דטרמיניסטי - נוכל להפוך אותו לכך באמצעות המעבר לעיל שתואר בהוכחה מעלה.

**טענה:** יהיו  $N^\epsilon$  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 אוטומטית מ $q_1$  ל $q_2$  תמי, בזכות מעבר האפסילון, וכן היא עוכרת קודסanco מ $q_1$  ל $q_0$ . לכן הרעיון והוא לאפשר מעבר  $q_2 \rightarrow q_0$  מראש של  $a$ , וכך גם ותרנו על מעבר האפסילון וס האוטומט נשאר כהו. באופו דומה, ניתן להעביר 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 לקרו 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  $\sigma$ ,

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

ובעברית: בהינתן אותן קלט ומצב, ממנו נוכל להגעת להרבה מצבים, אך זו תהיה קבוצה. נרצה לעבור על כל המצבים אליהם ניתן להגעה באמצעות  $N^\varepsilon$  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 האוטומט מצב נוכחי למקומות אחרים, ללא צורך 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  $A' = (Q', \Sigma', \Delta', q'_0, F')$  כדלהלן:  $A' = (Q', \Sigma', \Delta', q'_0, F') = Q \cup \{q_s\} = \Sigma' = \Sigma, q'_0 = q_0$  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  $\Delta$  איננה דטרמיניסטית, ולכן ממצב ואות עוברים לקבוצה של מฉบבים.

### 3.6.2 ערובוב שפות

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

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

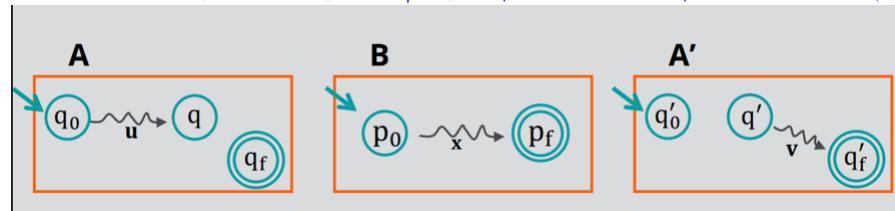
נראה כי ניתן לכתוב גם את  $L_3 = L_1 \circ L_2 \circ L_1$ . הרעיון יהיה פשוט:  $L_1, L_2$  רגולריות ולכן יש אוטומטים שמקבלים אותם. כל שנרצה, הוא לשגר את האוטומטים. נסמן בהתחמזה את האוטומטים המקבילים את השפות  $B, A$ . כמו כן, ניצור אוטומט  $A' = A$  (משם שכפול). הרעיון (לא פורמלי) יהיה כך -  
 א. מצב 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 קיבלנו - את השפה.

### 3.6.3 הכלאת שפות

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

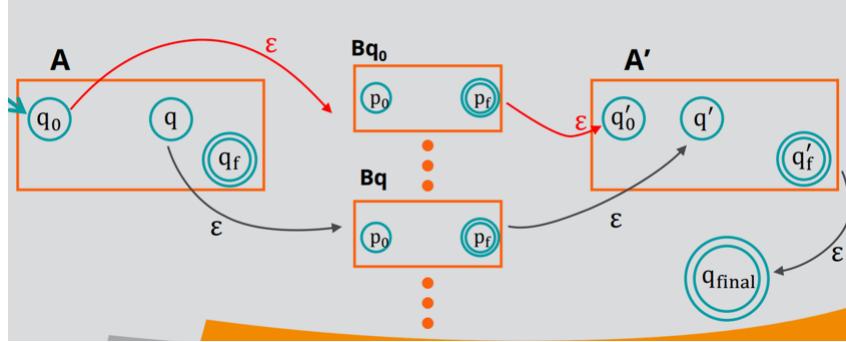
$$L = \{w | w = u \cdot x \cdot v : u, v \in L_1, x \in L_2\}$$

נכונה אוטומט לא דטרמיניסטי שיקבל את השפה. מטענה לעיל, אם נוכחהקיימים 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 הקודס: את  $u$  רצוחה לקרה על  $A$ , את  $x$  על  $B$  ואת  $v$  על  $A'$ .



כיצד תתכצע קריאות מיילס? כדלהלן:

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 היא כלית מז. הרו, מהו  $q$  הזה בתוך  $A$ ? זה סתם הווה ייחוש שאסור לנו לצעע. לנו יותרנו 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.  $\{q_{final}\} \cup Q' \cup (Q \times P) \cup Q^C = Q \cup (Q \times P) \cup Q^C$  (מודוע מכפלה קרטזית? נזכיר כי היה לנו שכפול של אוטומט  $B$ , ולכן המצביעים 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 בדרךנו 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 הטבעיים. לעומת זאת, " — " איןנה בהכרח תביא מ' טבעי, וכן הטבעיים 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"יליקית 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 שהקיים 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 נגוע 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 השל.  
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 המקבלים, לא נשנה דבר. כיצד נגיד פורמלית את קבוצת המצבים המקבלים?

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

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

$\sigma$  כאשר  $\Sigma \in \sigma$ .

$\emptyset$  – גם ה- $\emptyset$ , 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.  $(r_1)$  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  $bb$  פתרו: זו פשוט,  $a$  או  $b$  בהתחלה, ובסיום, באמצע  $bb$  חיגג, 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.

פתרו: 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:

ונכיה את הטענה שתראה לנו כיצד בהינתן אוטומט, ניתן להמירו לביטוי רגולרי.  
 $\Rightarrow$  נניח  $r = L$  ונכיה  $L$  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 נשים לב שבסופו שלם, והנחת האינדוקציה - הכל רגולרי זה ממש לא היה פורמלי, אני עצמן, סלחו לי, זה פשוט לא המטרה.  
 $\Leftarrow$  נניח  $L$  רגולרי, כלומר, 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 לעבור כלל כי אין אף מחרוזות שמתאימה לה. באוטומט כזה נדרש כמו דברים:  
 א. יש מצב התחלתי בודד יחיד 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).

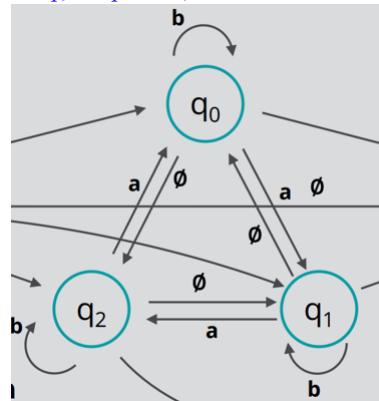
הקששות מסווגנות בביטויים רגולריים.

- המטרה תהיה לעבורי 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, אך עצת אי.

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

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

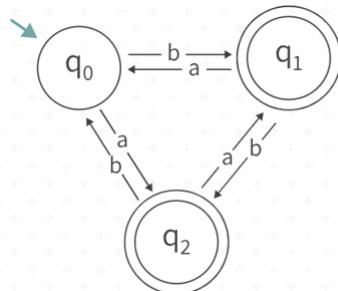
דוגמא לא תזקיק. נסתכל על האוטומט המקורי הבא. נראה כי אם נרצה למחוק את המצב  $q_1$  למשל, עליו לטפל בשתי קשיות.  $q_1 \rightarrow q_0$  עס "a", ולכן עצמאית של  $b$ -ים, וקשת נוספת  $q_1 \rightarrow q_2$  עס ".ab". ככלומר, אם נרצה להעלים את המצב  $q_1$ , נרצה להוסיף קשת מ $q_0$  לש  $q_2$  של הביטוי הרגולרי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



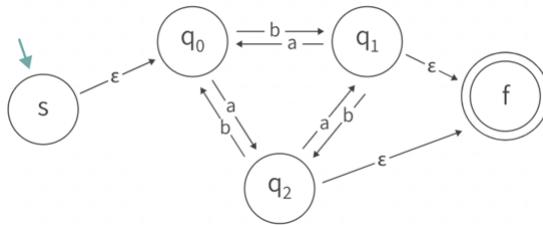
לאחר שביצענו דילול מצבים - נשארנו עם ביטוי רגולרי סופי, זהו הביטוי הרגולרי שמקיים  $L(r) = L$ .

**דוגמא:**

נרצה להמיר את האוטומט הבא, לביטוי רגולרי:

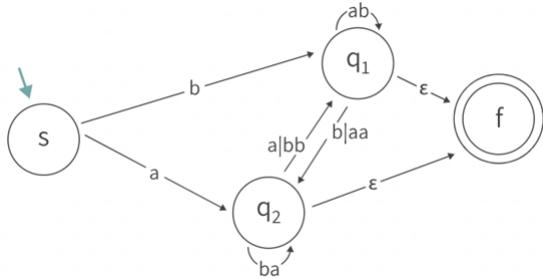


**שלב א.** נמירו לאוטומט מוכלל:



**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:



והמודרוב 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 $|yz| \leq N - j$  וכן  $xy^i z \in L$  כאשר  $i \geq 0$ . לפי תכונה 3, לכל  $i$  טבעי,  $xy^i z \in L$ , נסתכל על  $i = 0$  (מסתכר שכורס הזה אפס טכני? מוזר), אפשר לחתוך גס 2  $= i$ , נקבל כי  $xy^0 z = xz = a^{j+1} b^N$ , נשים לב כי  $j + l + k = N$  וכן  $0 < l + k < N$ . כלומר  $N \neq l + k$  וכן  $\#a_w \neq \#b_w$ , בסתיו לאמת הטענה. 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 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, ולאפשר אופציה של פירוק ייחיד).

**דוגמה נוספת.** נוכח כי שפת הפליאורוטיס 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 פלייאורוטיס, וכן  $n > 2N + 1$ . לכן, קיוס פירוק  $w = xyz$  כך  $|y| \geq N$  וכן  $|xy| \leq N$ . מכיוון, בהכרח  $x$  וכן  $y$  מוככבים מ"יס בלבך, ככלומר  $x = a^j$  וכן  $y = a^k$  וכך  $z = a^l b a^N$  וכן  $l > 0$ , לפי למת הניפוי לא ניתן למת  $xy^i z$  לכל  $i$  נקי  $i = 0$ ,  $xy^i z \in L$ , מתקיים  $k + l = N - j \neq N - j$ , וכך מילוי זו לא שייכת לשפה, כי  $i > 0$ .  
מכיוון 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  $|y| \geq N$  וכן  $|xy| \leq N$ . 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  $i = 0$  נקל  $xy^0 z = xz = a^j a^{p-j-k} = a^{p-k}$ , האם זה עוזר לנו? לא. זו לא שאלה טטודורית. נסה שוכ. נבחר  $i = p + 1$ . מה נקבל?

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

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  $k = n > n - 1 = r$  ו $i = n > n - 1 = r$  וכן  $j = \max\{i, k\} = n$ . אכו  $w \in L$ . וכן  $n > r = 3n - 1$ . לכן, לפי למת הניפוי, קיוס פירוק  $w = xyz$  כך  $|y| \geq N$  וכן  $|xy| \leq N$ . 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,  $l + r > 0$ , נקי  $i = 0$ , כלומר  $xy^i z \in L$ . בפרט עבור  $i = 1$  נקל

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

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 נוצר מעגל, מהסביר לעיל). לחلك לפני המעגל, נקרא  $x$  ולחחלק לאחר מכן 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 ונרצה להשתמש הוכחות אי רגולריות באמצעות דרך אחרת.  
למשל, נרצה להוכיח כי:

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

איןנה רגולרית. ובכן, נשים לב כי  $\bar{L} = \{w \in \{a, b\}^* | \#a_w = \#b_w\}$ , 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).

**דוגמה 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  $perm$  (ולמר  $perm(L)$  רגולרית.  
פתרון: האינטואיציה היא הרכה. פה, בהינתן כל שפה, כל הפרמוטציות על המילים היא גם שפה  
רגולרית? שמע שיטות. ואנו שיטות.  
נכ"ש כי אם  $L$  רגולרי אז גם  $perm(L)$  רגולרי. נבחר  $w = (ab)^*$   
ישים לב כי

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

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

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

נשים לב, אם  $L_1, L_2 = \bar{L}$  אינן רגולריות - יתכן כי  $L_1 \cup L_2$  רגולרית, למשל:  
נקבל  $L = \Sigma^* \cup \bar{L} = \Sigma^* \cup L_1 \cup L_2 = L$  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 בבלבד). ככלומר, 08 : 11 תקין, ואילו 94 : 28 לא תקין. יכולנו לייצג זאת באמצעות אוטומט סופי, נראה דרך אחרת:

$$T \rightarrow H : M$$

$$M = Minutes \text{ זה הזמן, } H = Hours \text{ וכן}$$

$$M \rightarrow LD$$

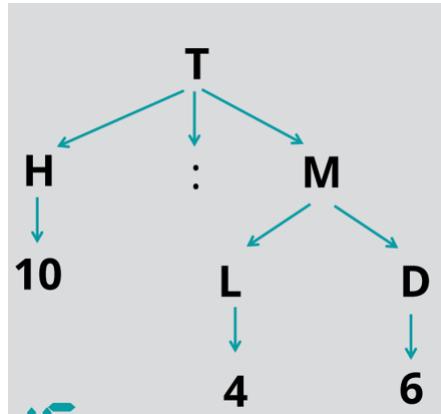
$$D \rightarrow 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$$

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

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.  $\Sigma$  הוא הא"ב הסופי, נקראים גם טרמינלים.
- ג.  $V$  הוא קבוצת המשתנים, 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 התחלתי,  $s \in V$ , 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  $uAv \Rightarrow_G u z v$  את הגיירה של המחרוזת  $u z v$  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  $\Sigma^*$  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 "אוונגו",  $\{a^n b^n | n \in \mathbb{N}\}$  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 שколоה לכתיבתה 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:

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

**פתרון:** 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  $L = \{w \mid w \neq w^r\}$ .  
**פתרון:** נרצה להתחיל כמו בפלינדרום, כלומר לאפשר  $aSa$  ו- $bSb$ . אבל, כמובן 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 וכן 3, 34, 02, 456 לא בשפה.

**פתרון:** 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 מושגנו  $A$  שהוא יכול להיות מה שנרצה מאחד הצדדים

$$S \rightarrow AabbaA$$

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

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

**פתרונות:** נרצה למשה 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:

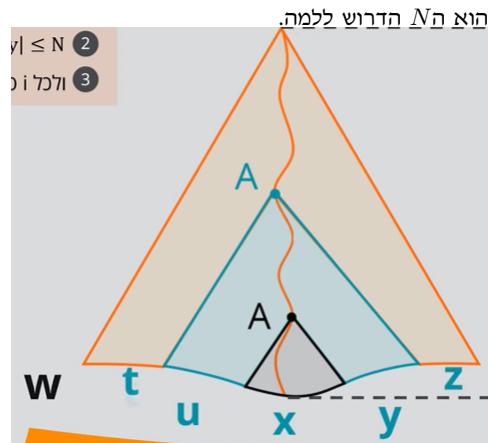
- א.  $|uy| > 0$
- ב.  $|uxy| \leq N$
- ג.  $\forall i \in \mathbb{N} : tu^i xy^i z \in L$ .

**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, ואז נחזר את השחרורה. וכן הלאה - נוכל להוסיף את החתיכה הכהולה כמו פעים 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  $n > |w| = 3n$ , לכן 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  $a$  ב- $uy$ . כמו כן, מהדרישה  $0 > |uy|$  בפרט יש שם לפחות 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 החדשת עם הנינפה  $\#a_w = n + 1$  או  $\#a_w = n + 2$ , כמו כן  $i > j$  יותר מ-2.  
ובכלנו סטירה לדרישת  $i > j$  יותר,  
פורמלית יותר,

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

כיוון ש- $|uy| > 0$ , בהכרח  $1 < n + 2 - |uy| \leq n + 1$ , בסטירה.  
ב.  $uy$  מורכב ממשיים בלבד: ננפח עס  $i = 2$ . בהכרח  $|uy| > 0$  ולכן קיימים שם  $b$  אחד לפחות.  
כיוון שנייפחנו עבור מס'  $a$ -ים חיובי ממש, מס'  $b$ -ים יגדל-cut, ככלור  $\#b_w > n + 1$  ועוד, סטירה  
לכך שמס'  $a$ -ים צריך להיות גדול יותר ממש.  
ג.  $uy$  מורכב ממשיים בלבד: ננפח עס  $i = 2$ . בהכרח  $|uy| > 0$ , לכן 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  $0 < |uy|$ , וכן יש גם  $a$ -ים וגם  $c$ -ים ולכן  $n$  נקבול  
ונפח עס  $i = 0$ . נקבע

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

כיוון ש- $|uy| \geq 2$ ,  $n + |uy| \geq n + 2$  ונקבל  $\#a_{w'} = n + 2 - |uy| \leq n = \#c_{w'}$ , כלומר מס'  $a$ -ים  
יהי קטן שווה במס'  $c$ -ים, בסטירה.  
ה.  $uy$  מורכב ממשיים ואז  $c$ -ים: בדומה, יודעים כי  $0 < |uy|$  וכן יש גם  $b$ -ים וגם  $c$ -ים ולכן  
 $n$  נפח 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 קיבלו 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  $\leq \varepsilon$ . אז, אם  $|S| \geq 1$ , מקיים 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 הלומות: בשתי הלומות 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 להכרייע 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.

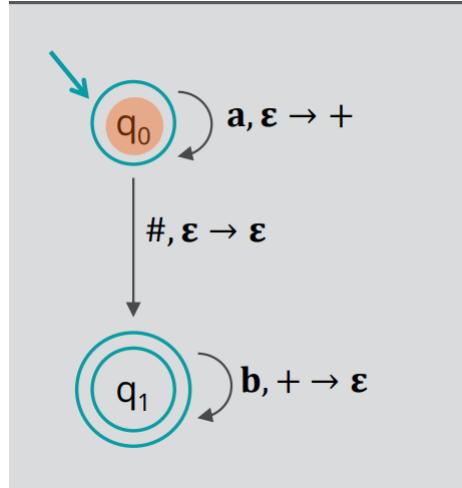
המחסנית היא רכיב זכרו שני תווים גם לכתוב בו, וגם לקרוא ממנו. המחסנית עובדת בשיטת *LIFO*. לאחר שנכנס, ראשוני שיצא. אם האוטומט רוצה, הוא כותב תוו בראש המחסנית ומכניס אותו, והוא יכול גם לקרוא תוו מראש המחסנית. אם המחסנית ברגע ריקה, יש שם  $\epsilon$  בראש. וכך:

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

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

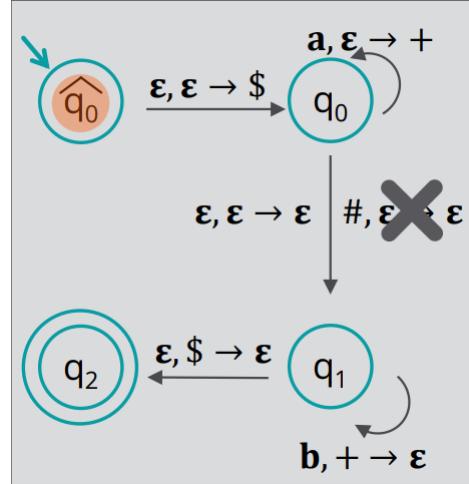
כאשר נרצה למחוק אותן מראש המחסנית, נבצע  $\epsilon \rightarrow \sigma$ .

**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  $\epsilon$ , וידרש להוציא  $b$  ממחסנית, אך המחסנית ריקה - ולכן נקבל שגיאה. כמובן: החישוב ית��ע.

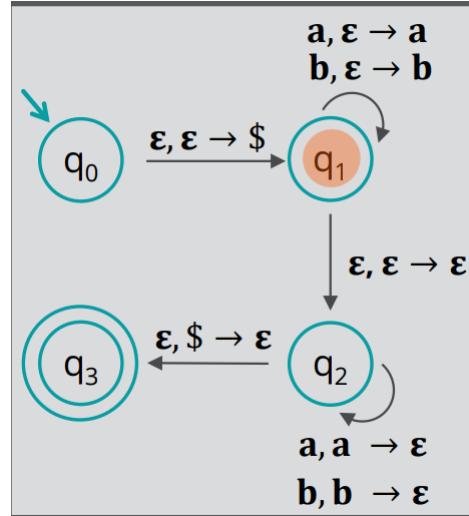


**דוגמא 2.** נרצה לבנות אוטומט מחסנית עבור השפה  $L = \{w \in \{a,b\}^* | w = a^n b^n | n \in \mathbb{N}\}$ . הרעיון יהיה כדלקמן: בדוגמה הקודמת - ידעו לדרוש מס'  $a$  גדול ממש מס'  $b$ . כתת נרצה לבדוק אותו מספר. הרעיון הוא זהה - אנחנו יודעים שкриיאת  $a$  דוחפת  $+$ . קרייאת  $b$  תוציא  $+$ . אם היה 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.



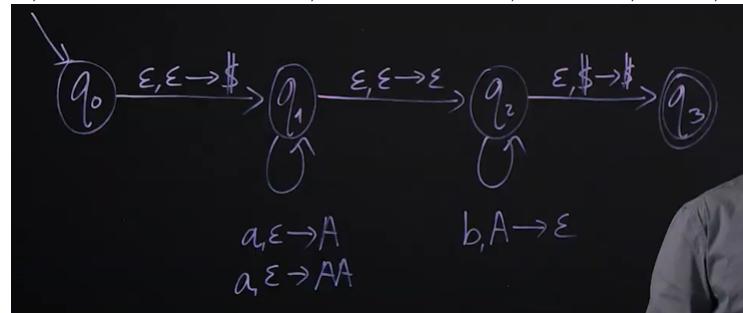
**דוגמה 3:** נרצה לבנות אוטומט מחסנית לשפה  $L = \{ww^R | w \in \{a, b\}^*\}$ . היא איננה רגולרית. הרעיון יהיה כמו בדוגמה הקודמת, נוציא  $\$$  כמו קודם. באשר למעברים: אם קראנו  $a$ , נדחוף  $a$  למחסנית. אם קראנו  $b$  נדחוף  $b$  למחסנית. כשנרצה, נוציא מעבר אפסילון. אם קראנו  $a$  נוציא  $a$  ממחסנית, ואם קראנו  $b$  נוציא  $b$  ממחסנית.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

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

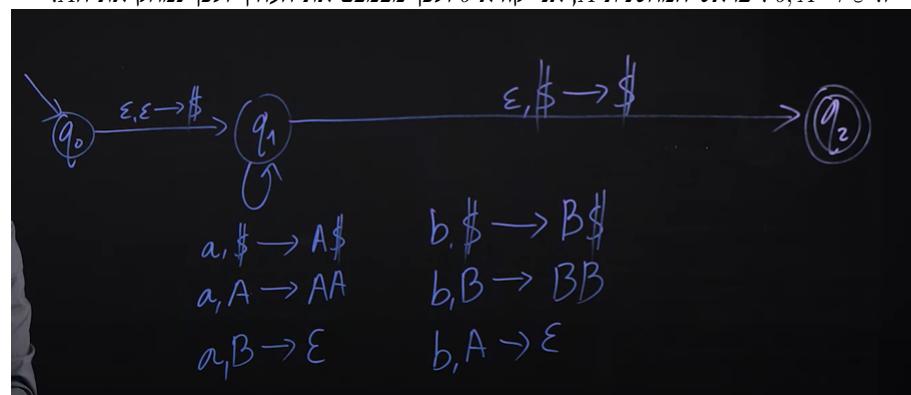
הרעין יהיה זה - מי בשפה? כל המילים שמתחלות ב $a$  ומס'  $b$  שלם הוא לפחות מס'  $a$ , וכן לכל יותר פעריים מס'  $a$ -ים. כמובן, נסיף  $\$$  בתחילת ובסוף, בשליל לדעת מתי התחתיות של המחסנית. כיוון שהאוטומט לא דטרמיניסטי: אפשר את ה"נירוש" או יותר נכון - הבחירה הלא דטרמיניסטיבית, לבין להכניס  $a$  לבין להכניס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



הוגמה 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.
- $\Sigma$  א"ב הקלט הסופי.
- $\Gamma$  א"ב המחסנית.
- $\Delta$  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 שנותר לי לקרוא, ומה יש בתחום המחסנית כרגע. ופורמלית -  $i(q, \Sigma, \Gamma, \Delta, q_0, F) = (Q, \Sigma, \Gamma, \Delta, q_0, F)$  אוטומט מחסנית.

קונפיגורציה הינה שלשה  $(x, u, q)$  באשר  $x \in \Gamma^*, u \in Q, q \in \Sigma^*$  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:** הינה  $C_1, C_2$  אוטומט מחסנית. ויהיו  $C_1, C_2$  קונפיגורציות של  $P$ . נסמן  $C_1 \mapsto_P C_2$  כלומר,  $C_1$  גורר את  $C_2$  אם כשנמצאים ב- $C_1$  ניתן לעبور ל- $C_2$  ע"י מעבר אחד.

**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 ל- $C_2$  ע"י 0 או יותר צעדים.

### 6.2.1 שפת האוטומט:

היא  $L(P) = \{w \mid (q_0, w, \varepsilon) \in F\}$  אוטומט מחסנית. השפה של  $P$  מוגדרת להיות

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

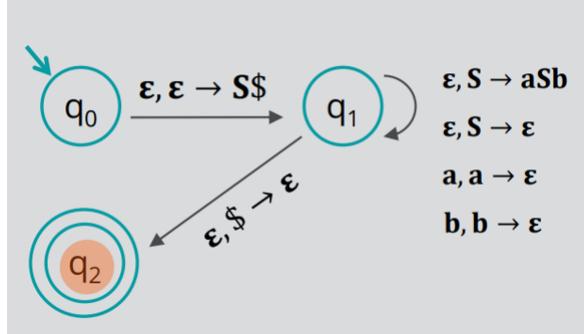
כלומר כל המחרוזות שניתן להתחיל מ $q_0$ , להגעה לpcb מכב, ככה שקרנו את כל הקטל, התחלו ממחסנית ריקה והסתמכו במחסנית כלשהי.

**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  $S\$$ . 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 כל היצירה, באמצעות דחיפת המשתנים אל המחסנית. כאשר בראש המחסנית משתנה, נפעיל כלל יצירה. כאשר בראש המחסנית אחרות קלט נקרה את הקלט מסרט הקלט. תמיד נתחיל מ  $q_1 \rightarrow q_0$  עם  $q_0$ . תמיד ב  $q_1$  יהיו כל הכללי היצירה + אפשרות לקריאה והשואה עם כל אותיות הקלט, לפי התנאי  $\varepsilon \rightarrow a, a$  וכו'.

#### 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 השפות זהה, אחרת ממילא מילים לא יוכל להיות בחיתוך. נסמן א"ב  $\Sigma$ .

יהי  $D = (Q^R, \Sigma, \delta^R, q_0^R, F^R)$  אוטומט סופי דטרמיניסטי מקבל את  $R$ .  
 ויהי  $P = (Q^C, \Sigma, \Gamma^C, \Delta^C, q_0^C, F^C)$  אוטומט מוחסנית מקבל את  $C$ .  
 נבנה אוטומט  $U = (Q^U, \Sigma, \Gamma^U, \Delta^U, q_0^U, F^U)$  שיברך את  $R \cap C$ .  
 נגדיר מרכיביו להלן:  

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

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

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

או משתמשים במוחסנית עברו השפה  $C$  בלבד, וכך:  
 נגדיר כעת 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 מכפלה. המעברים בין המ מצבים (שהם זוגות של מצבים מקוריים, ממש באוטומט מכפלה. הפעולות על המוחסנית תהיה של אוטומט  $P$  בלבד. בשורה הראשונה של הפונקציה, אלו מעברים עם קריאת קלט. זהו  $(q, \sigma) \delta^R (p', u)$  מעבר בשביל  $R$  והוא  $(p', u)$  הינו עבר  $C$ , לדעת היכן להתקדם, באשר  $\sigma$  הוא הדחיפה למוחסנית. הינו אוטומט דטרמיניסטי כמו אין בו מעברי אפסילון, ובשורה השנייה מותואר המצב הזה: ברכיב  $R$  הראשון האוטומט מתקדם כמו באוטומט של  $R$  ובשני כמו באוטומט של  $C$ . סה"כ, האוטומט מקבל את החיתוך.

#### 6.5 דקדוקים רגולריים ושפות רגולריות

כל שפה רגולרית ניתןcid�ו שפה חסרת הקשר, כולם קיימים דקדוק עבורה. נראה כעת כי לשפות רגולריות, יש מבנה מיוחד לדקדוק שיוצר אותה. דקדוק זה נקרא **דקדוק רגולרי** אותו אנחנו מחלקים לשני סוגים:

##### א. דקדוק רגולרי ימני

הכללים הם אחד מהצורות הבאות:  
 $\forall A, B \in V, \sigma \in \Sigma$   
 $A \rightarrow \sigma B$  .1  
 $A \rightarrow \sigma$  .2  
 $A \rightarrow \varepsilon$  .3

##### ב. דקדוק רגולרי שמאלי

הכללים הם אחד 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 ורק אם  $L = L(G)$

הוכחה: נוכיח צד אחד, אם  $L$  רגולרית ניתן למצוא מבנה עבורה דקdock רגולרי ימני.

$L(A) = L$  כ $\vdash A = (Q, \Sigma, \delta, q_0, F)$

מבנה דקdock רגולרי ימני  $G = (V, \Sigma^G, R, S)$

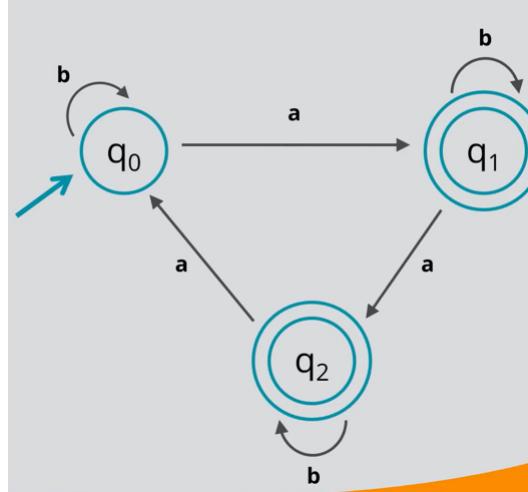
נגידר:  $S = \Sigma^G = Q$ . נשים לב, נתיחס לקבוצת המ מצבים המשותפים.

$q_0$  היא מצב נגידר כליל יצירה  $R$ .

1. לכל  $p = (\sigma, \delta)$  כלומר מ $q$  עברים עם  $\sigma$  למ' נגידר את הכלל  $\sigma p \rightarrow q$

2. לכל  $f \in F$  הכלל  $\epsilon \rightarrow f$  מודוע? אם הגענו למצב מקבל, נוכל להפסיק עם אפסיון את הדקdock כי זו מילה תקינה.

דוגמא. נתבונן באוטומט הבא המצווג בתמונה



נראה כי לכל מצב באוטומט יהו כל יצירה מתאימה. באוטומט ישנו מעבר מ $q_0$  אל  $q_1$  על  $a$  ולכוון נקבע  $aq_0 \rightarrow aq_1$ .  
בזקdock נקבע  $q_0 \rightarrow aq_1$ .  
בזקdock, סה"כ נקבע את הדקdock:

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

ובאשר למצבים המתקבלים, נקבע  $\epsilon \rightarrow q_1 \rightarrow q_2$ . סה"כ זה דקdock חסר ההקשר המתאים לאוטומט.

מס' הערות על היחידה:

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

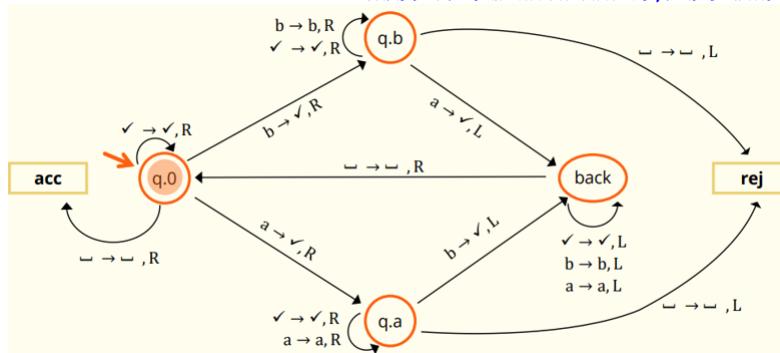
## 7 ייחודה 8: מכונת טיורינג

אנחנו ממשיכים במחקר שלנו בעקבות השאלה **לפתרון ע"י מחשב?**. ראיינו כבר כי יש שפות שהן אוטומט סופי והן אוטומט מחסנית, לא מסוגלים לפתרו. למשל השפה  $w = L$   $\{w^n c^n a^n\}$ . די ברור, כי מחשב מסוגל להוכיח את השפה. מכיוון המכונה היא שהבעה במודלים שחצינו. כתעת נציג את המודול החזק מכולם. אלן טיירינג הוא אבי תורת מדעי המחשב - הוא פיתח מודל מתמטי שבמהלך השנים התברר כמודול מרכזי שמתפרק כמחשב: **מכונת טיורינג**.

### 7.1 מהי מכונת טיורינג?

כמו במודלים קודמים, הקלט יהיה על גבי סרט הקלט. בשונה ממודלים קודמים, ניתן לכתוב על המודול שלנו. **תמיד** ניתן לכתב על המודול רק על הראש, בו אנו נמצאים כרגע. הרחבה נוספת ומשמעותית - במכונת טיורינג ניתן לזרז שמאללה. במכונת טיורינג אנו מניחים שסרט הקלט הוא אינסופי - בשני הצדדים. כאמור, גם אם התחלנו בנקודה כלשהיא על קו הסרט, נזוז שמאללה ויש שם תווים, נזוז שמאללה עוד 69,420 צדדים, וגם שם יהיה תווים. **לצורך הנוחות** - אנחנו מניחים כי במקומות בהם לא כתבנו עוד מופיע התו " ", ככלומר מופיע תחתון.

**דוגמה 1.** נתבונן בשפה  $\{w | \#a_w = \#b_w\} = L$ . זו שפה חסרת הקשר. נראה לראות כיצד המכונה מקבלת את המילוטים בשפה: הרעיון היה פשוט - נתחיל בקירiat המילה משמאלי, וקרוא אותה מהראשונה. נסמן ✓ על האות שקריאנו, ונטתקדס בקלט ימינה לחפש ✓ תואמת לה, נמצא אותה ונסמן בה ✓ גם כן. כתעת נזרז לחזרה הקלט - מילג על ✓ הפתאומיטים ונחפש שוב ✓ או ✓ תואם. נניח ומצאנו ✓, נזכיר ✓, ווישץ בסדרת הקלט לפעיאות  $a$  תואם וכן הלאה. כך המכונה תעבור, בערורה מילולית. תמייד בסיסים נחרז לחזרה הקלט משמאלי, וכך עד כל אותיות הקלט הוחלו ✓? אם ורק אם תהייה סדרקה של כל אזור הקלט בה הכל הווה ✓, אז נוכל לסייע. מתי נדע שהגענו ליעב לא טובי? כאשר במלצת סדרקה הגיעו לפיקוס של " ", בזווית יש אותן בקלט שאנו לה תואמת, ולכן הגיעו לאזור האסורה. ושולח למצב מות. כיצד נעכור זאת לתיאור גוף של מכונה?



ובכן, זוו מכונת הטיורינג שמקבלת את השפה. נסביר כיצד מבה קורה כאן:  $q.a$  הוא מצב בו ראיינו  $a$  ומחפשו לו  $b$  תואם. בזמנה,  $q.b$  הוא מצב בו ראיינו  $b$  ומחפשו לו  $a$  תואם. והה מצב בו משתמש בשבייה לחזרה לקצה השמאלי של הקלט. ישנו שני מצבים מיוחדים: מצב **accept**, **acc**, מצב זה יכול מילוי. ויש את המצב **reject** מהמילה **rej**. מצב זה זיהה מילה לגמרי. כאשר המכונה מגיעה לאחד המיצבים - הוא מפסיק לפעול. בכל מצב אחר המכונה בהכרח ממשיכה.

כמו כן, יש את המצב ההתחלתי  $q_0$ .  
בכל שלב במכונה: היא עושה שני דברים - זה לאושהו, וכותבת משחו על הסרטו. לנו עליו לתאר שלושה דברים: מה הקלטomial דבירים - זה לאושהו, מה נכתב על המכונה, ולאיזה כיוון איזה. לנו הסימנו  $a \rightarrow \checkmark, R$ ,  $a \rightarrow a, R$  ו-  $L$  שמאללה.  
משמעותו: בהינתן שקראיו  $a$ , שיס שס ✓ ותזוז ימיה.  $R$  ימיה ו-  $L$  שמאללה.

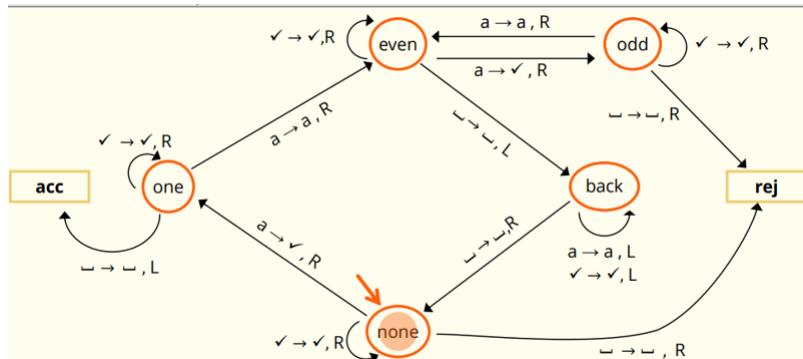
חשיבות - נשים לב, המכונה איננה יודעת שהגענו לקצה הקלט. נשים לב כי בהינתן קריאה של מילה, המכונה לא יודעת שבתא אחד משאללה נגמר הקלט. וכן מוגעות הולאות העצמיות *back* של *L*. ✓ → ✓. מה המשמעות? כאשר נרצה לחזור, אך שוב שפאללה. אל תנסה כולם בסיטוון. ואז אנחנו נהייה כבר במשבצת עם רוח - ודע כי הגענו לאזור לא טוב. וכשהגענו לאזור הלא טוק, אז יכתייך לך שתגידי להתחלה. עם הטיטומו *R*, → – שימושיות: ראות מתקף, תשאர אותו ולך ימינה. זה יכתייך לך שתגידי לאזור הקלט.

כל עוד נראה ✓ משאר ב*q0* עס לולאה עצמית כטובנו. כי לא נרצה להתקדים כי אין מה לזכור. אם אנחנו במשבצת *a*, אם נקרא *a* או ✓ ממשיך בקריאה ימינה. רק ✓ איננו מփשיט. בדיטה עכבר המשבצת *b* אם נראה *b* או ✓. כמו כן, במשבצת *back* אם נקרא *a* או *b*, נרצה שישארו שס כי אנחנו רואים רק לחזור חזרה ולא לגעת בהרבה.

מתי נדע להגיאו למשבצת מתקבל? נראה כי יוצא מ-*q0* מעבר *R*, → –. משמעותו: אם אנחנו במשבצת המקלט, וכל מה שעשינו היה לו זיהויו, אז שאר קלט, וכן הגענו לסתה: אולי שאנו כבר קלט – משמע הפעלה שלילו חייכת להתקבל, וכך נגיע למשבצת מתקבל. מתי נדע את סוכן לנו. כאשר אנחנו באם *a* או *b* בזוג תואם. במשבצת כזו: ישר לזריה. אין צורך להמשין. זו הייתה דוגמה זוגית פורכנת – זהה כל הרעיון. מזל וחושנו זה חזק פואז.

דוגמה 2. נרצה לבנות מכונת טירוגיג שתקבל את השפה:

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



השפה *L* איננה רגולרית, ואינה חסרת הקשר. נראה כי מכונת טירוגיג יודעת להכריע את השפה. הרעיון לבניית המכונה יהיה פשוט – מילה תתקבל אם מ"ט המ"ט שלה הוא חזקה של 2. הרעיון והזהה לעבור על קלט המילה ולמבחן בכל פעם *a* אחד, ואתה *a* אחריו להשאיר. כיצד נמבחן? אם ✓. כך נעבור בפעמים הראשונה על הקלט, למשל המילה *a/a*. ואז, נעבור פעמיים נוספת על הקלט: שוב, פעמיים נמבחן *a* ראשון, ואתה *a* אחריה שוארו: ✓✓✓✓. ומה עשוינו כאן בעצם? בכל פעם חלכנו את מ"ט ה-*a* פי 2, אם בסופו של דבר קובלנו משבצת *a* לשאר *a* אחד בלבד, אז המ"ט המקורי של מ"ט המילה היה חזקה של 2. מתי מילה לא בשפה? אם באיזשהו שלג, לפני שנגיע למ' האחרון, מס המ"ט היה או זוגן, דבר שיפורע לנו של אחד כו אחד לא.

נראה מה קורה במקרה: כרגע יישס שני משבצות, *acc* ו-*rej*.  
המשבצת *none* הוא המשבצת ההתחלתי – עדין לא קראונו *a* בסביב סיריקה זה. המשבצת *one* משמעותו שקראונו *a* בזוז. המשבצת *even*, *odd* בהתקדמות כאשר קראונו מ"ט או זוג/זוגי של *a*-ים. המשבצת *back* עכבר חזורה אחרת בקלט.

נראה את המכונה בתמונה, שיענה לבדוק לתיאור לעיל. נראה כי אם הגיעו למשבצת *one*, או זוג אחד בלבד וכן המילה כו בשפה, כי חזקה של 2, וכן אם הגיעו לסוף הקלט – נrz למשבצת *acc*. אם הגיעו למשבצת *odd* ולאחריו נגמר הקלט, נrz ל-*rej* כי המילה בהכרח לא בשפה אם מ"ט המ"ט או זוגי גדול

מאות, אפשרות נוספת להגעה ל-*rej* היא אם או *a*-ים בכלל, קלטר מה מצב *none*. סעיף 0 אינו טכני.

חשוב לציין - המכונה תמשיך לקרוא קלט כל עוד לא הגיעו לאחד משני מצבים העצירה: תמיד אנחנו זרים על הشرط, וכן תמיד כתובים עלייו. גם אם לא רוצים לשנות מה כתוב, עדין יש כתיבה חדשה של מה הייתה קודמת לנו.

## 7.2 הגדרה פורמלית

מכונת טיריניג הינה שביעיה  $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$  כך :

- $Q$  הינה קבוצת המצביעים,  $\Sigma$  הינו הא"ב,  $\Gamma$  הוא א"ב הشرط הסופי,  $q_0$  הוא המצביע ההתחלתי,
- הוא המצביע המקורי והוא המצביע הדוחה.
- $\delta$  הינה פונקציית המעברים המוגדרת:

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

באשר הפונקציה מקבלת מצב ואות על הشرط, ומזהירה מצב, אותן חדשה לשים בشرط ולאיזה כיוון להתקדם.

### 7.2.1 קונפיגורציה:

קונפיגורציה נתנת תיאור מותמי ממוצה של כל הדברים שיש לדעת בעת התקדמות המכונה. מה علينا לדעת בכל שלב? כיצד נראה הشرط בעת, באיזה מצב אנחנו במצבים, ובאיזה חלק בشرط אנחנו. כיצד נדע באיזה שרט בחלק אנחנו? נניח ונחנו עם הشرط *aab* במצב  $q_0$  באוט *a* השניה, אז הסימן *aq<sub>0</sub>ab* יתאר כי בעת אנחנו במצבים *aab* והשניה, כלומר אותן של אחר המצביע היא אותן בה אנו במצבים *qa*.

פורמלית, תהי  $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$  מכונת טיריניג. קונפיגורציה של  $M$  הינה מחרוזת  $wqsw$  באשר  $w, v \in \Gamma^*$  ו  $q \in Q$ . הכוונה היא כי  $w$  הוא תוכן הشرط ממשמאן בראש, שאחורי יש אנסוף רווחים  $v$  והוא תוכן הشرط מימין לאחר, שאחורי אנסוף רווחים. נשים לב כי  $wqsw$  קלומר הקונפיגורציות הללו, ניתן להוסיף כמה רווחים שנרצה לפני ואחרי, כל עוד מל' הרוחים סופי.

### 7.2.2 גיריה:

תהי  $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$  מכונת טיריניג ויהי  $C_1, C_2$  קונפיגורציות של  $M$ . נסמן  $C_1 \vdash_M C_2$  אם שנמצאים ב- $C_1$  ניתן לעبور ל- $C_2$  באמצעות צעד בודד.

הכללה - נסמן  $C_1 \vdash_M^* C_2$  אם שנמצאים ב- $C_1$  ניתן לעبور ל- $C_2$  באמצעות 0 או יותר צעדים.

### 7.2.3 קבלת ודוחייה של מחרוזות:

תהי  $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$  מכונת טיריניג. תהי  $\Sigma^* w$  מחרוזת. נאמר כי  $A$ .  $M$  מקבלת את  $w$  אם  $\vdash_M^* u(acc)sv$  באשר  $\Gamma \in \sigma \wedge \sigma \in \Gamma^* \wedge v \in \Gamma^*$  ( $u, v$ ) קלומר, רק מעניין שהגענו למצביע מקובל. לא מעניין שרט הקלטן  $B$ .  $M$  דוחה את  $w$  אם  $\vdash_M^* u(rej)sv$  באשר  $\Gamma \in \sigma \wedge \sigma \in \Gamma^* \wedge v \in \Gamma^*$  ( $u, v$ ) קלומר, רק מעניין שהגענו למצביע דוחיה. לא מעניין שרט הקלטן

נשים לב כי אכן, בנגדות למודלים קודמים, יש מצב דוחיה. ומדוע? המכונה תמשיך לפעול עד אנסוף אם לא נעצור אותה. אכן, חישוב שאיןו מקבל איינו בהכרח דוחה. וכך חיבורים להגדיר מהי דוחיה של מחרוזות, על מנת שמכונה לא תרוץ עד אנסוף.

#### 7.2.4 הכרעה של שפה

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

זה אס יש מילוי שהמכונה לא עוצרת עכורים לא במצב דחיה, ולא במצב קבלה? כמובן - לולאו  
 אינטואיטיבית על הקטלן, אז, המכונה לא מככירה שום שפה.  
 ישנו מכונות טיורינג שלא מככירות אף שפה.

#### 7.2.5 קבלת של שפה

תהי  $(Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$  מכונת טיורינג. ותהי  $\Sigma^* \subseteq L$  שפה. נאמר כי  $M$  מקבלת את  $L$  אם לכל  $w \in \Sigma^*$  מתקיים:  
 א.  $w \in L \iff M$  מקבלת את  $w$   
 ב.  $w \notin L \iff M$  לא מקבלת את  $w$   
 במקרה זה, נאמר כי  $L(M) = L$

כאן, ה"זיהوية" הפכה ל"אי-זיהوية". מושג חלש יותר. לא מקבלת = דוחה / לא מגיעה למצב קבלה.  
 כל מכונות טיורינג מקבלת שפה כלשהי, שהוא פשוט שפת כל המילויים שדרכו מגיעות  $acc$  במכונה.  
 זה אכן מוגדר richtig.

### 7.3 תיאור מבנית טיורינג באמצעות טבלת מעברים

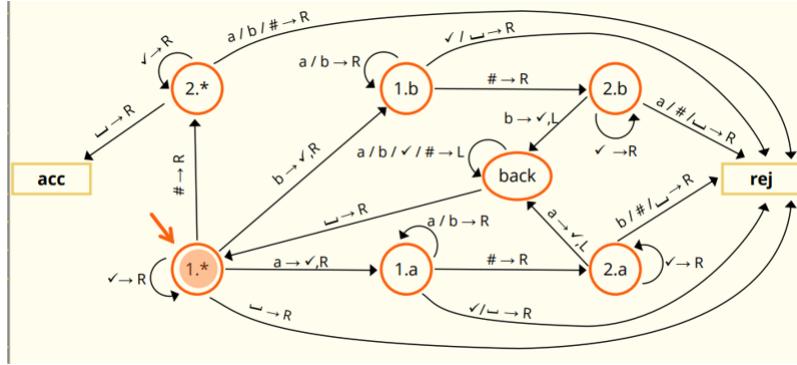
כעת, לאחר שלמדנו הגדרות פורמליות - נלמד מס' טכניקות וטריקים לבניית מכונות טיורינג.

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

נשים לב כי זו לא שפה גולרית, ולא שפה חסרת הקשר. בקלות, לפי למות הנינפוח, אם נבחר את המילה  $a^n \# a^n = w$  שאורך  $1, 2n+1$ , נקבל סטייה ע"י ניפוח עם  $0 = i$ . עם זאת, נוכל לבנות לה מכונות טיורינג.

הרעיון יהיה פשוט - נתחל בקורסית הסדרת משमאל, נזכיר אס ראיינו  $a$  או  $b$  ונמחקה עם  $\#$ . ממש נתקדם ונידרג על כל האותיות עד לאות הראשונה שפנואה לאחר  $\#$ . אם הן שוות ממשין, אם לא נלקח במצב דחיה. כך ממשיך עד שככל הסרט שלנו יהיה ריק, ואם אכן כך, המילה בשפה.

נשים לב כי אמינו ממשו גדול ומשמעותי - נזכיר שראיינו  $a$ , "כיצד נזכיר? נשותמש בתאי זכרון פנימי. בתא ראשון נזכיר באיזה צד של המחרוזת אנחנו, ובתא השני באיזה אוטומט זכרון צריכים לזכור. כיצד ממשים את תא הזיכרון הפנימי? במצבי המכונה. לכן מצביו האוטומט יהיו בזוגות - למשל: 1.b מציין שתא השמאלי של הזיכרון יש 1(קורי, אנחנו בצד השמאלי של המחרוזת  $1, w$ ), ואנחנו צריכים לזכור  $b$ . בדומה עבור שאר המצבים. המצב \* מציין שאנו בצד השמאלי של המחרוזת והתא ריק, כמובן \* מציין ריק. מכאן שהמצב ההתחלתי יהיה \*, 1, שהרי אנחנו בצד השמאלי ואין צורך לזכור דבר.



**טבלת מעברים**

ניתן לראות שמכונת הטירוגינג הופכת למסריהה. בדוגמה הקודמת היו יותר מדי מצבים ויתר מדי מעברים. נרצה להציג את מכונת הטירוגינג באמצעות טבלה, מה שיפשר את המודל הגרפי. בהינתן: מצב וסימון בסרט. הטבלה תגיד: מצב חדש, כתיבה, תזוזה. ככלומר ממש תיאור של פונקציית המעברים באמצעות טבלה. גם זו - דרך לתאר מכונת טירוגינג.

באשר לדוגמה הקודמת, זה הייצוג הטבלאי, המתמטי והמדויק יותר של מכונת הטירוגינג:

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

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

נראה כי אם נרצה לכתוב מודל גרפי שיפטור, יהיה לנו לפחות 40 מעברים. לכן נתאר בטבלה. נשים לב שהרעיון פשוט: בכל שלב נרצה לזכור מי האותיות שראינו עד כה ביריצה שמאל לيمין, כאשר נגעים לקובץ  $\{a, b, c\}$  או שיש לנו את כל אותיות הקלט - וכך נחזור שמאל. כאשר כל סרט הקלט מלא ברוחניים ויק מקלט, או הミילה בשפה. כאן בטבלה,  $S$ , הינה קבוצה שיכולה להיות אחת מ-8 הקובץות  $\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{c, b\}, \{a, b, c\}$

| מצב           | סימן בסרט  | מצב חדש                 | כתיבה  | תזוזה | תנאי              |
|---------------|------------|-------------------------|--------|-------|-------------------|
| $q.S$         | $\sigma$   | $q.(S \cup \{\sigma\})$ | ✓      | R     | $\sigma \notin S$ |
| $q.S$         | $\sigma$   | $q.S$                   | $\cap$ | R     | $\sigma \in S$    |
| $q.S$         | ✓          | $q.S$                   | $\cap$ | R     |                   |
| $q.\{a,b,c\}$ | a, b, c, ✓ | back                    | $\cap$ | L     |                   |
| $q.\emptyset$ | —          | acc                     | $\cap$ | R     |                   |
| back          | a, b, c, ✓ | back                    | $\cap$ | L     |                   |
| back          | —          | $q.\emptyset$           | $\cap$ | R     |                   |

$S \subset \{a, b, c\}$

$\sigma \in \{a, b, c\}$

**כל השאר  
עוברים  
- rej**

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

**דוגמה שלישית - השפה**  
 $L = \{x_1 \dots x_k \# y_1 \dots y_k \# z_1 \dots z_k | x_i, y_i, z_i \in \{0, \dots, 9\} \wedge \forall i : x_i \geq z_i \geq y_i\}$

בעת סריקת המחרוזות, علينا לזכור הפנימי מס' דברים: באיזה חלק של המחרוזות אני מבין החלוקה  $x, y, z$ . בתחילת הערך נקרא סטרוק שמאלי לימין את החלק הראשוני, ולאחר מכן, ונבדוק אם  $z$  המתאימה נמצאת בינהם. הרעיון יהיה לסרוק משמאל לימין את החלק הראשוני, לבחור את הספרה הראשונה ולזכור אותה בזיכרון פנימי. כך נמשיך עד שנתתקדם ל#. שם נמצא את ה $y$  המתאים ונזכור אותו בזיכרון פנימי. שם נמשיך עד ל# הבא ונמצא את ה $z$  המתאים וטוטו נזכור בזיכרון פנימי. נבדוק את שלושת הספרות שללו האם מקיימות את אי השווון, ואם כן נחזיר הלהה לתחילת הקלט. כמובן שככל יותר שקרהנו תשומנו ב✓. כאשר כל האותיות חוץ מהסמלויות יהוו ריקות, נדע שהמילה צריכה להתקבל ושאכן הייתה התאימה. נראה כי יוצר גרפּי של מכונה זו כולל המון מצבים. המון. ובטבלה: נעיר כי כל מצב יהיה עם שלושה>Status. יתאר את החלק בו נמצאים, האיבר השני בשלשה הוא תוקן תא האיקס, והשלישי הוא תא הוויא. \* מסמל שעוזר לא נקרה ספרה. נשים לב כי השורה החשובה ביותר בטבלה היא שורת התנאי, אם ורק אם מתקיימים התנאי  $\tau_2 \geq \tau_1$  ואז המילה מתתקבל. אחרת, במצב rej. איך התנאי היה בא לידי ביטוי במצבים עי' אוטומט גוף? אם היינו בז.4.2, ורצינו לקרוא 3, היינו חזרים back כי אות חוקית, אם היינו מקבלים 9 היינו הולכים לדוחיה כי היא לא בין ארבע לשתיים.

| תנאי                    | תזוזה | כתיבה | מצב חדש                           | סימון ברט'    | מצב                               |
|-------------------------|-------|-------|-----------------------------------|---------------|-----------------------------------|
|                         | R     | ✓     | X.σ.*                             | σ             | X.*.*                             |
|                         | R     | ⊑     | X.*.*                             | ✓             | X.*.*                             |
|                         | R     | ⊑     | X.σ.*                             | 0,1,...,9,✓   | X.σ.*                             |
|                         | R     | ⊑     | Y.τ.*                             | #             | X.τ.*                             |
|                         | R     | ✓     | Y.τ.*                             | σ             | Y.τ.*                             |
|                         | R     | ⊑     | Y.τ.*                             | ✓             | Y.τ.*                             |
|                         | R     | ⊑     | Y.τ. σ                            | 0,1,...,9,✓   | Y.τ. σ                            |
|                         | R     | ⊑     | Z.τ <sub>1</sub> . τ <sub>2</sub> | #             | Y.τ <sub>1</sub> . τ <sub>2</sub> |
|                         | R     | ⊑     | Z.τ <sub>1</sub> . τ <sub>2</sub> | ✓             | Z.τ <sub>1</sub> . τ <sub>2</sub> |
|                         | L     | ✓     | back                              | σ             | Z.τ <sub>1</sub> . τ <sub>2</sub> |
| $\tau_1, \tau_2 \neq *$ | R     | ⊑     | acc                               | ⊑             | Z.*.*                             |
|                         | L     | ⊑     | back                              | 0,1,...,9,✓,* | back                              |
|                         | R     | ⊑     | X.*.*                             | ⊑             | back                              |

**כל השאר עוברים ל- rej**

### 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 בטלטlat מערבים, או שרטוט, פורמלים. נראה מ"ס דוגמאות:

**דוגמה ראשונה.** תאר מכונת טירינג שמכריעה את השפה  $L = \{a^n b^n c^n | n \geq 1\}$  כמו שעשינו בעבר, מעבר איטרטיבי משמאלי לימין ובכל שלב מורידים בהתאם באמציאות ו-

*a, b, c* מקבלים מילה א"מ'ם כל הקלט שנותר הוא ו. ובפסודו -

א. כל עוד התו הווה ✓ - 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 *a* ✓ את הראש ימינה

1. אם התו הנקרא הוא *c* או - דחלה.

2. אם התו הנקרא הוא *b* - כתוב במקומו ✓ והוא את הראש ימינה.

ג. כל עוד התו הווה *b* או ✓ הוא את הראש ימינה.

1. אם התו הנקרא הוא *a* או - דחלה.

2. אם התו הנקרא הוא *c* - כתוב במקומו ✓ והוא את הראש שמאליה.

ד. ✓ את הראש שמאליה עד לתו הראשון שמיין לרצף הרוחחים בתחילת הסרט.

ה. חזר לשלב '.

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

נצרך לחזור להגדרת הכפל - מהי ההגדירה של פועלות הכפל?  $j \times i$  אומר - בצע חיבור של *i*, *j*

פעמים. בהינתן מילה  $a^2 b^3 c^6$  נרצה על כל אות *a*, למוחוק (לטמן) ון *c* כמספר פעמים שמופיע *b*. כיצד

נעשה זאת? על כל אות *a*, נלק' לכל אות *b* ונמחק עבורא אחת מתאיימה *b*, עד שלא ישארו אותיות

*b*. ואז נצורך להציג את אותיות *b*, למוחוק את *a* האחת שבה השתמשנו, ולהתקדם לאות הבאה

*b*, מתי נסיים? ככליא ישארו אותיות *a*, אנחנו יודעים שלא נותר בהתחלה ולכן רק עלינו לודוא שלא

נותרו אותיות *c* בסוף הקלט. אם אכן לא נותרו ✓ המילה בשפה. נuire כי אותיות *b* יוחקו עם סימנו

*X* למשל, בשביל שנדע להציגן לאחר מכן. וה邏輯ית של אותיות *c* יהיה עם *V* למשל. ובפסודו:

א. מחק אות *a* וכותב אות רווח במקומות. אם האות הראשונה אינה *a*, דחלה.

ב. מסרוק ימינה עד לאותיות *b*. בשלב זה, אם נתקל באות אחרת שאינה *a* או *b* דחלה.

- ג. כל עוד נותרו אותיות  $b$  שאין מוחוקות:
1. מחק את  $b$  ראשונה וכותב  $X$  במקום.
  2. סרוק עד לאות  $c$  הראשונה, אם יש אותה מחק אותה וכותב  $V$  במקום.
  3. אם הגעת ל\_ \_ או  $a$ , דחה.
  - ד. התקדם שמאלה עד תחילת הקלט והחלף בדרך את כל אותיות  $X$  ב\_.
  - ה. חוזר לתחילת הקלט, אם נותרו אותיות  $a$  חוזר לשלב א'.
- אחרת, התקדם לסוף הקלט. אם לא נותרו אותיות  $c$  קבל. אחרת: דחפה.

## 7.4 שימוש במכונת טיורינג לחישוב פונקציות

הבהרה - הקורס עוסק בבעיות הכרעה. חישוב פונקציה היא בעיית חישוב. נחרוג מנהל הקורס, ונראה כיצד אפשר לחשב פונקציות באמצעות מכונת טיורינג.

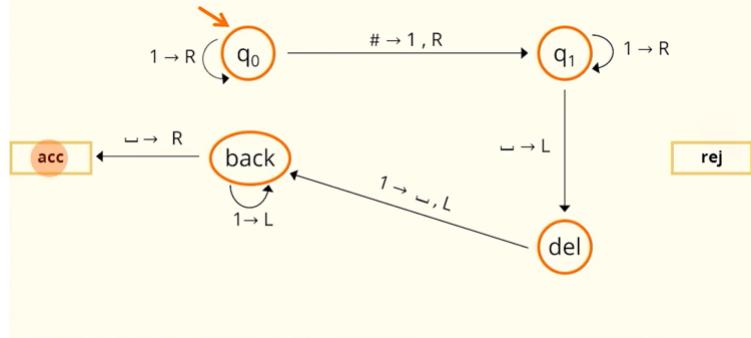
הגדרה: תהי  $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$  מכונת טיורינג. ותהי  $f : \Sigma_1^* \rightarrow \Sigma_2^*$ . נאמר כי  $M$  מחשבת את  $f$  אם:

- א.  $\Sigma = \Sigma_1 \wedge \Sigma_2 \subset \Gamma$ .
- ב. לכל  $w \in \Sigma_1^*$  מותקיים  $q_0 w \vdash_M^* accf(w)$ .

(הבהרה - כל פונקציה שאנו מכירים תהיה חייבת להיות מקודדת לצורת מחושצת).

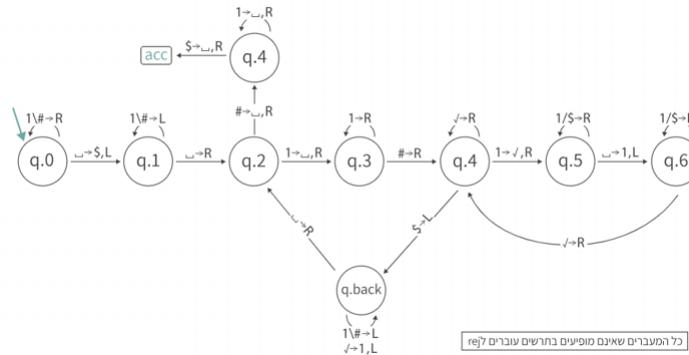
נרצה לחשב פונקציות  $M : \mathbb{N}^k \rightarrow \mathbb{N}^l$ , אם כן נרצה ליצגם באמצעות "ב" בשילוב  $\Sigma_2^* \rightarrow \Sigma_1^*$ .  
 כיצד ניציג את הפונקציה באמצעות "ב"? ביצוג אונרי. כך למשל: המספר 3 יוצג ע"י 111, המספר 7 ע"י 1111111 וכן הלאה. כמו כן  $0 = \epsilon$ . כיצד נפרק פרמטרים? באמצעות#. למשל  $(2, 3) = 11\#111$ .

**דוגמא:** יהיו מספרים  $\mathbb{N} \in y, x$ . כתוב מכונת טיורינג שמחשבת את הפונקציה  $f(x, y) = x \cdot y$ . הרעיון יהיה פשוט: בכלל היצוג האונרי, כל האחדות כתובות על הסרט. כל שיש לעשות הוא לחת את#. שמספריד, לשים שם 1, ולהוריד 1 מהסוף וככ' נקבל את התוצאה על הסרט שלנו.



**דוגמא שנייה:** יהיו מספרים  $\mathbb{N} \in y, x$ . כתוב מכונת טיורינג שמחשבת את הפונקציה  $y \cdot x = x \cdot y$ . הרעיון יהיה דומה לרעיון שביצעו בכתיבת המכונה לשפה  $a^i b^j c^k$  באשר אגחנו בעת ריצויים לחשב את הפונקציה  $f(1^i, 1^j) = 1^{i+j}$ . עם זאת - זה חישוב ולא הכרעה ולא כזה הבדלים. מה יהיה הרעיון? נתקדם על הסרט עד לסוף וכותבו תמייד \$. וcutת באופן איטרטיבי - נלק על האחדות של המספר, משמאלי, בכל פעם ונוריד אחת ממשם, ובהתאם געבור על כל האחדות במס' השני שנמצאה מימין יותר, ובזיאג נעתיק אותן לאחר ה\$. כל אחד שি�մנו, נשים עליו  $X$  למשל. כך נדע מיהם האותיות בחילוק השני של המספר. לאחר שכל החלק השני מלא באיקסים, נחזרו לחילוק הראשון.

של המספר, שם נמשיך באופן איטרטיבי עד שלא יותרו ספרות מצד השמאלי של המספר, והפלט ייכה לנו לאחר סימן ה $\$$ . ובתיאור מכונה זה יראה כך –



מס' הבהרות על היחידה:

- א. מכונת טיריניג יכולה להעביר מילה במצב מכבול, גם אם לא קראו את כל הקלט. למשל – כל המילים שמתחלילות בה, אין צורך להמשיך לקרוא פרט לתו הראשון.
- ב. השפה שמכונת הטיריניג מקבלת הינה יחידה.
- ג. יתכוו שני קומפיגורציות  $C_1 \neq C_2$  במחזור ריצת הקלט.
- ד. בהינתן מכונת טיריניג שמכריעה את  $L$ , ניתן שיש מכונת טיריניג אחרת שמקבלת את  $L$  ולא מכיריה אותה. למשל:  $\{a, b\}^* = \{aw | w \in \{a, b\}^*\}$ . בדור הראשון ניתנת להכרעה פשוטות. מצד שני, אפשר לבנות מכונה שבהינתן שהאות הראשונה  $a$  הולכים במצב מכבול ואחרת מתקדמים ימינה ללא הפסקה. היא תקבל את השפה, אך לא תכיר אותה כי יש מילים שעוברן היא לא עוצרת.
- ה. במכונת טיריניג שמחשבת פונקציית המכונה עצרת בחלק השמאלי של הקלט.

## 8. ייחידה 9: וריאציות של מכונות טיריניג

### 8.1 מודל חישובי - הגדרה פורמלית

**מודל חישובי:** אוסף של מכונות שעבורם מוגדרים המושגים הכרעה וקבלת של שפות.

- הגדרה:** יהיו  $A, B$  מודלים חישוביים. נאמר כי  $A$  ו-  $B$  שקולים, אם לכל שפה  $L$  :
- א. קיימות מכונה במודל  $A$  שמכריעה את השפה  $L$  אם ומ"מ קיימת מכונה במודל  $B$  שמכריעה את השפה  $L$ .
  - ב. קיימות 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

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

**טענה:** נסמן את המודל עם שני הכוונים באות  $(wo, T, O)$ , ועם הסרט ימינה ב-  $O(ne)$ . אי, ו-  $T$  ו-  $O$  שקולים.

הוכחה:

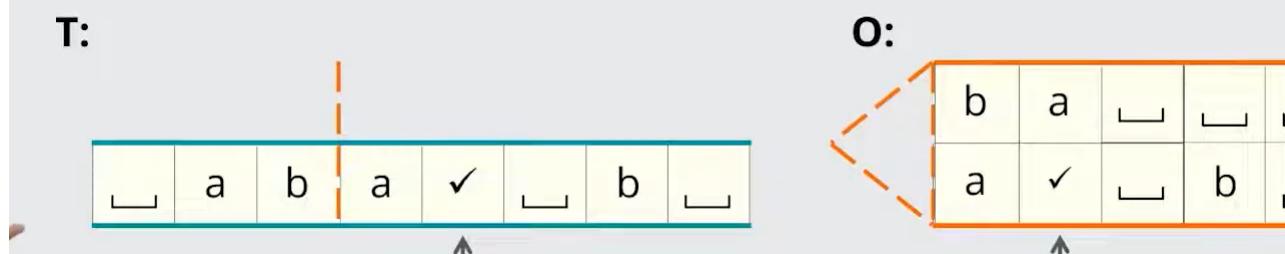
א. נראה שלכל מכונה במודל  $O$  יש מכונה במודל  $T$  -  $M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, acc^T, rej^T)$ . נבנה  $(Q^o, \Sigma^o, \Gamma^o, \delta^o, q_0^o, acc^o, rej^o)$  מכונת טיריניג למודל  $O$ . נבנה  $(Q^o, \Sigma^o, \Gamma^o, \delta^o, q_0^o, acc^o, rej^o)$  מכונת טיריניג למודל  $T$ .

על פניו, נראה שהמכונות יהיו זהות רק שלא נשמש בצד השמאלי. ובכן - כמעט. נזכיר כי אם אנחנו במצב התחלתי זים שמאלה, אנחנו נשארים באותו מקום ואילו במודל הבסיסי זים שמאלה - יש לנו רצף על הפער הזה.

הרעיון יהיה כדלקמן - כל הרכיבים יהיו זים למחר. נוסיף לטבלת המעברים את הדבר הבא: נסמן ב-\$ את האות הראשונה משמאלו לקט, בשביל לדעתה היכן מתחילה הסרט השמאלי האינסופי. כל שנדרש לשנות הוא אם אנחנו נמצאים במעבר הראשוני, זים שמאלה: בצע שני פעולות - אז שמאלה, אל \$, וכשתראהدولר: תלק ימינה ואל תנסה דבר. ככלומר - שתי פעולות אלו יבטיחו שנשאר באותו מקום.

ב. נראה שלכל מכונה במודל  $T$  יש מכונה במודל  $O$  -  $M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, acc^T, rej^T)$ . נבנה  $M_o = (Q^o, \Sigma^o, \Gamma^o, \delta^o, q_0^o, acc^o, rej^o)$  מכונת טיריניג למודל  $O$ .

הຽנו והיה לצע סימולציה" כאילו אנחנו בקורס דו כורוי. ככלומר - נבחר מקום לתחילה פמנו את הטרט השמאלי, כל מה שהוא משמאלה, נקלף, וכך שטראה כך:



נשאלת השאלה, כיצד זה אפשרי? ובכן - אנחנו נקבע את הא'ג להיות זוגות סדרות. את נקודות הקיפול נסמן ב-\$.

מעבר לקו הקיפול: הזוג ימינה בקורס המקורי תתרגם לתזואה ימינה בקורס החדש, ו הזוג שמאלה לתזואה שמאלה בקורס החדש. (אנו כיוון

לאחר קו הקיפול: הזוג ימינה בקורס  $T$  תתרגם לתזואה שמאלה בקורס  $O$ , ו הזוג שמאלה בקורס  $T$  תתרגם לתזואה ימינה בקורס  $O$ . (ההפקן

כמו כן - על המכונה לזכור באיזה שורה היה נמצאת מכונה  $O$ : העולגה או התחתונה - ולכן נשמש בתא גזרו פלמי: יסמן לחלק התחתון, ו  $U$  לעליון. כאשר המכונה  $T$  תצא מעבר לקו הקיפול נגזר  $U$ , ולהפוך עבורה  $D$ .

כמו כן, נשמר בתא גזרו פלמי את המעל בו היוו ב- $T$ . כל שיינו במכונה החדשה בעלת שתי השורות תלוות לאחר שתי השורות, ככלומר - אם במכונה החדשה היה לנו זוג  $(a, -)$  והם יהולן ל-  $b$  כאשר אותו עדרון מופיע ליקוי והפרקנה, נקלף מקומות את הזוג  $(b, -)$  - ככלומר רך מה השתונה. כמובן, פיו משתנה מהזוג זה בהתאם  $D/U$ .

באשר המכונה הכתומה,  $O$  תראה דולר: הוא צריך לצע לו לאו שתחריז אוותה ימינה, ולשנות את הכוון - ככלומר אס הרוה  $D$  או עכשו  $U$  ולהפוך.

cutet למימוש הפורמל.

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

הערה - החלק  $\{ - \}$  הוא בשביל האתחול של המכונה בעלת הזוגות.

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

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

וכן טבלת המכਬים:

| $\tau, \sigma, \pi \in \Gamma^T$ | מצב      | סימן             | מצב חדש | כטיבה | תווות | תנאי                               |                     |
|----------------------------------|----------|------------------|---------|-------|-------|------------------------------------|---------------------|
| q.D                              | $\pi$    | p.D              | $\pi$   | L     |       | תווות שמאלה<br>(q,σ) $M^T(p,τ, L)$ |                     |
| q.U                              | $\sigma$ | p.U              | $\tau$  | R     |       | תווות שמאלה<br>(q,τ) $M^T(p,τ, L)$ |                     |
| q.D                              | —        | p.D              | —       | L     |       | תווות ימינה<br>(q,—) $M^T(p,τ, R)$ |                     |
| q.U                              | —        | p.U              | —       | R     |       | תווות ימינה<br>(q,—) $M^T(p,τ, R)$ |                     |
| q.D                              | $\pi$    | p.D              | $\pi$   | R     |       | תווות ימינה<br>(q,σ) $M^T(p,τ, R)$ |                     |
| q.U                              | $\sigma$ | p.U              | $\pi$   | L     |       | תווות ימינה<br>(q,—) $M^T(p,τ, R)$ |                     |
| q.D                              | —        | p.D              | —       | R     |       | פגעה בקצה                          |                     |
| q.U                              | —        | p.U              | —       | L     |       |                                    |                     |
| q.D                              | \$       | q.U              | —       | R     |       |                                    |                     |
| q.U                              | \$       | q.D              | —       | R     |       |                                    |                     |
| $q_0^0$                          | τ        | q.τ              | \$      | R     |       | $\tau \in \Sigma \cup \{—\}$       |                     |
| q.σ                              | τ        | q.τ              | —       | R     |       | $\sigma \in \Sigma$                |                     |
| q.—                              | —        | back             | —       | L     |       |                                    |                     |
| back                             | —        | back             | —       | L     |       |                                    |                     |
| Back                             | \$       | q.D              | —       | R     |       |                                    |                     |
| acc <sup>c</sup> .D              | המ'      | acc <sup>0</sup> |         |       |       |                                    |                     |
| acc <sup>c</sup> .U              | המ'      | acc <sup>0</sup> |         |       |       |                                    |                     |
| rej <sup>c</sup> .D              | המ'      | rej <sup>0</sup> |         |       |       |                                    |                     |
| rej <sup>c</sup> .U              | המ'      | rej <sup>0</sup> |         |       |       |                                    |                     |
|                                  |          |                  |         |       |       |                                    | סימן                |
|                                  |          |                  |         |       |       |                                    | אתחול               |
|                                  |          |                  |         |       |       |                                    | פגעה בקצה           |
|                                  |          |                  |         |       |       |                                    | תווות ימינה         |
|                                  |          |                  |         |       |       |                                    | תווות שמאלה         |
|                                  |          |                  |         |       |       |                                    | תווות מקורית        |
|                                  |          |                  |         |       |       |                                    | וכן טבלת המכובים:   |
|                                  |          |                  |         |       |       |                                    | כל השאר עוברם - rej |

נשים לב כי בטבלה מופיע מצב האתחול - שכן הקלט ניתן לנו כקלט אינסובי בשני הצדדים, ואנו, בהינתן אוטיות הקלט צריכים לתרגםו לאוגות על פני סרטם עם ימינה בלבד. כמו כן, אם בסיום המכונה המקורית הגיעו גם אנחנו, ובודמה על rej. הערה אחורונה - נשים לב כי במקרה הסרט ימני, יש אנסוף רוחים. לא נוכל לטפל בכלם ולהפוך אותם לאוגות: ולכן נקבע מצב בו מצאנו רוחה חדש, ונקבע שנטיעחס אליו אליו היה זוג רוחים.

### 8.3 מודל TS

ורייציה נוספת היא מכונת טיריג'ינג שיכולה להשאר במקום, במודל זה הראש יכול להשאר במקום: ככלומר ניתן להשאיר את הראש באותו מקום על הסרט גם כאשר עוברים בין מצבים. נזכיר כי במודל T המקורי הגדרנו את פונקציית המעברים כך:

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

icut נגדיר את הפונקציה כך:

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

כאשר  $S$  - השאות במקומות.

כך למשל: אם אנו עובדים עם הרטט  $abba$ , המשמעות של  $\delta(q_0, a) = (q_1, b, S)$  היא - תעביר לנצח  $q_1$ , תחליף את האות בה כרגע אתה נמצאת  $a$ , הרי אנו בתחלת הרטט לאות  $b$ , אבל ברטט - השאר במקומות.

**טענה:** המודל  $T$  והמודל  $TS$  הינם שקולים.

**הוכחה (ריעוית):** ברור כי כל מוכנה  $T$  הינה גם מוכנה  $TS$  כי אפשר שלא להשתמש במצב  $S$ . בכיוון השwi, הרעיון היה גם כן לא מסובך יותר מדי - לכל מצב,  $q_1$  למשל, נצור מצב ביןיהם  $q_1^L$ . מה הרעיון? אנחנו רצחים לבנות מוכנת  $TS$  מוכנת  $T$  ולכך - אם יש לנו את האות  $S$ , למשל  $\delta^T(q_1^L, b) = (q_1, b, R)$ , הוא יתורגם ל-  $\delta^T(q_0, a) = (q_1, b, L)$  ולאחר מכן  $(q_1, b, L) = (q_1, b, S)$  ככלומר - זו ל族自治州 הומני של  $S$  ימינה, אה"כ כמו שהוא שפה שאחתה צריכה שמאליה על הרטט. ואז אכן השגנו מה שריצינו: לא אזנו ברטט אבל אזנו במצב.  $Q^T = Q^{TS} \cup \{q^L | q \in Q^{TS}\}$  נשים לב שפורמלית,

## 8.4 מודל OR

נדיר מודל מוכנות טיריניג חדש בשם  $OR$ . במודל זה, הראש יכול לבצע בכל מעבר רק פעולה אחת: או אז על הרטט (ימינה או שמאליה) או לכתוב במיקום 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  $TS$ , ומטוראנטייביות השקילות (הרוי זה יחס שקילות) נקבע כי מודל  $OR$  שקול למודל  $T$ .  
כיוון ראשון: לכל מודל  $OR$  קיימת מוכנה שcolaה מודל  $TS$ .

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

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

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

כיוון שאנו במכונה המקורית אזו לא כתיבת קלט על המכונה, וכך נדמה זאת ע"י כך שנכתוב את אותן האות ונטקדים באותו הכיוון.

ב. מקרה שני: אם  $\delta^{OR}(q, \sigma) = (p, \tau)$  באשר  $\delta^{TS}(q, \sigma) = (p, \tau)$   $\forall p, q \in Q^{OR}$  והמעבר המקביל ב-  $\delta^{TS}$  יהיה:

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

כיוון שאנו לא זים (נארים במקומות) וכן כותבים אותן קלט.

**כיוון שני:** לכל מכונה במודל  $TS$  קיימת מכונה במודל  $OR$  שקופה לה. תהי  $M_{TS}(Q^{TS}, \Sigma^{TS}, \Gamma^{TS}, \delta^{TS}, q_0^{TS}, acc^{TS}, rej^{TS})$  מכונה מודל  $TS$ . בונה מכונה מודל  $OR$   $M_{OR} = (Q^{OR}, \Sigma^{OR}, \Gamma^{OR}, \delta^{OR}, q_0^{OR}, acc^{OR}, rej^{OR})$ . ביוון זה יהיה מרכיב יותר כיון שבמעברים בהן המכונה  $M_{TS}$  כותבת אותן ימינה או שמאליה, אלא יתכן מעבר שכול יחיד במכונה  $M_{OR}$ .

לכן, נמיר חלק מהמעברים במכונה  $M_{TS}$  לשני מעברים עוקבים במכונה  $OR$ : במעבר ראשון כתוב אותן ובעבר השני נבע את התזוזה. לשם כך, נזדקק למצבים חדשים שיחברו בין המעברים. לכל מצב  $q$  נגדיר שני מצבים יייחודיים לו:  $q^L, q^R$ . וכך נקבל:

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

כעת נגדיר את פונקציית המעברים.  
מעבר הביניים תמיד יבצע תזוזה ימינה או שמאליה בלבד, לכל אותן שבסרט ולכל:

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

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

בහינתן מעבר  $(p, \tau, R)$  באשר  $\delta^{TS}(q, \sigma) = (p, \tau, R)$  ונק:

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

בහינתן מעבר  $(p, \tau, L)$  באשר  $\delta^{TS}(q, \sigma) = (p, \tau, L)$  ונק:

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

כמו כן ברור כי אם  $bS$  נשאים במקומות זה טוב לנו כאן וזה טריואלי.

## 8.5 מכונות טיוריינג מרובת סרטים

וריאציה נוספת למודל, כאשר לכל מכונה יש מט' סרטים כלשהו גדול שווה ל-1, **קבוע מראש**. לכל סרט ראש כתוב מישל, אך ככל מוחברים לבקר המרכזי של המכונה. בכל צעד המכונה קוראת וכותבת בכל אחד מהסרטים, וגם זה בכל אחד מהסרטים. הראשונים הסרטים השונים יכולים לזרז ב敞开 ב"ת". במכונה עם מט' סרטים, הקלט תמיד בתחילת הסרט הראשון, ושאר הסרטים ריקים.

דוגמה. כתוב מכונת טיוריינג עם מט' סרטים שתכרייע את השפה  $L = \{w \in \{a, b\}^* | w = w^r\}$ . בתחילת הסרט הראשון, דיב פשוט. בתחילת הסרט השני, נעתיק את הקלט מהסרט הראשון לסרט השני. הסרט הראשון הרראש יהיה על האות השמאלית ביותר של המילה, ובסרט השני הרראש יהיה על האות הימנית ביותר של

המייה. בכל שלב, נבדוק האם שני הזרים עם אותה אות - ואם כן נתקדם. אם יש שוויון לאורך כל הדריך, אז המחרוזת היא פלינדרום. בשביל הפשיות, נניח שניתן להשאר במקום במקומו. ראיינו כבר שמודל שקול למודול  $T$ . זו המכונה -

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

טענה: לכל  $\mathbb{N} \in k$ , המודול של מכונת הטיורינג עם  $k$  סרטים שקול למודול מכונת הטיורינג המקורי. הוכחה (לא פורמלית בכלל אבל חשובה): לכל  $k$  ולכל מכונת טיורינג  $(Q^k, \Sigma, \Gamma^k, \delta^k, q_0^k, acc^k, rej^k)$  עם  $k$  סרטים, וכייה יי' קיימות מכונות  $M^k = (Q^1, \Sigma, \Gamma^1, \delta^1, q_0^1, acc^1, rej^1)$  עם סרט אחד, שקולות. נרצה לעשות סימולציה של המכונה עם  $k$  הסרטים על המכונה עם הסרט אחד. מה צריך בכל סימולציה?

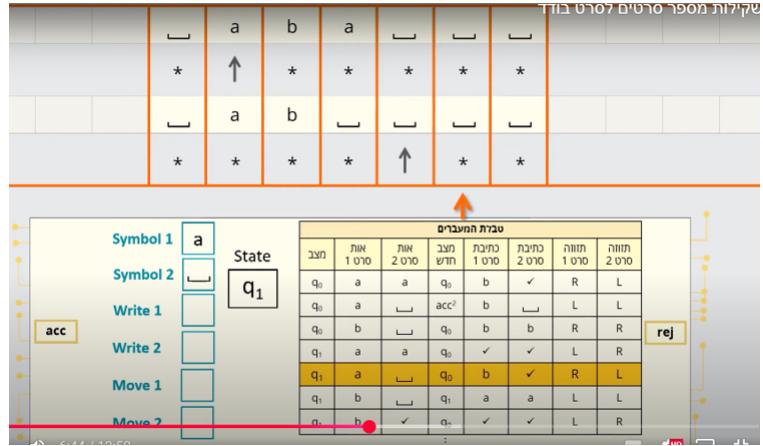
1. ייצוג הקונפיגורציה של המכונה המסומלת ע"י או בתוך המכונה המסומלת ( $M^1$ )
2. סמלולץ המיעברים בין הקונפיגורציות

הרעין יהיה כזה - נדגים אותו עם  $k=2$ : יש לנו שני סרטים, הנפוך אותם ל-4 כך שבסיסודרים בשורות אחת אחרי השניה. הסרט הראשון, ומתחתיו סרט שמוסמן כך - איפה שהיה הראשי יסומן חז, ובכל שאר מקומות יסומן \*. השורה השלישי, תהיה הסרט השני, והשורה הרביעית תהיה באופן זהה שורה של חז וכוכבויות. באופן דומה על  $k$  סרטים נשכפל למכונה עם  $2k$  סרטים כשמתחתת לכל סרט סרט של \* וחצים.

מדוע עשינו זאת? כיון שעליינו לדעת היכן נמצא הראש.

סימולציה של צעד בודד של  $M_2$  במעבר ל- $M_1$  עולה המון צעדים -

שם כך נשתמש בתאי זכרון  $symbol1, symbol2, write1, write2, Move1, Move2$ . הרעיון הוא ש- $symbol1$  נשומר את הסימן שמויע בacz העליון ובהתאמה  $symbol2$  השימן שמויע בacz התחתון.  $write1$  נשומר מה צריך לכתוב בכל אחד מהחיצים,  $Move$  נשומר לאיזה כיוון צריך להזיא את החז. בנוספ',  $M_1$  תשמור את כל טבלת המיעברים של  $M_2$  בכל סימולציה, הראש של  $M_1$  ) המכונה עם סרט אחד שבנינו, יהיה במשבצת שמוביל למשבצת עם החzman השמאלי ביותר. נשומר תא זכרו נספ' בשם  $state$  שיישמר את המצביע הנוכחי של  $M_1$ . המכונה מבצעת ריקפה פעם אחת משמאלי לימין - וממלאת את ששת התאים לעיל לפי מה שטמאצא בסרט (כאשר היא תראה את החיצים). כתע - המכונה מחזיקה במאה שהיא רואה בסרט הראשון ובסרט השני, אנחנו יודעים מה המצביע בו אנו נמצאים - ולכן כתע ניתן לחפש את השורה המתאימה בטבלת המיעברים (כאן מטה, בכתום) ולדעת מה יהיה החזע הבא של המכונה. במקרה שלו יכתב ✓, בסרטים וזו  $L$ , בהתאמה.



כasher ha'mekoneh siyeha s'reika rashiyyah, meshimal li'min, ha'a u'verat la's'reika no'sefet mi'min l'sh'maal - casher ha'a t'kalev bi' ha'mata'im, ha'a t'cavot v'tzoz b'hata'ot l'ma sh'marano b'z'keron ha'fnemi b'sh'shat ha'ta'ot, v'tshene a'otot b'hata'ot. n'sim leb ci' am y'sh tzozah L b'move2 move2 (l'meshil, ai' ha'ch' sh'iz'oh y'hiha ha'ch' b'sh'orah ha'rbe'uyit. ha'ch' shel  $M_1$  az' rk v'rok meshimal li'min v'oz' mi'min l'sh'maal br'atz'. ne'ir ci' casher co'tb'is", meshenim at' col ha'rbe'uyah. sh'eri ha'a'b ha'a'b ha'rbe'uyah.

ne'ir ci' b'cel sh'el s'reika - ha'ch' cr'ik la'tchill m'sabat'ach achot meshimal ha'ch' ha'sh'maal bi'ot. b'cel sh'el - z'va ch'sob. mit' ha'mekoneh ma'psika le'ubor? k'she'ia meshimal li'min - c'she'ia m'sabat'ach achot meshimal ha'ch' ha'sh'maal bi'ot. ach'ri ha'ch' ha'imini bi'ot, v'k'she'ia mi'min l'sh'maal - m'sabat'ach achot meshimal ha'ch' ha'sh'maal bi'ot. b'cel si'om sh'el s'reika - an'chenu ha'olkim le'korao b'tbat' ha'me'urim, ma' u'sim ba'hitnun m'zv, v'shi'i at'otiot k'lat.

am ha'matzb' hofek l'state ai'  $M_1$  u'verat la'matzb' m'kbel. am ha'matzb' hofek l'rej ai'  $M_1$  u'verat la'matzb' d'chiya. ma' am ha'a la' magu' la'dakhya v'la' k'vella? ai'  $M_1$  cm'oo  $M_2$  t'mash'ik n'atz' le'ro'z. b'shelb ha'tchillati - tamid y'hiha at'chol v'ho'ficht ha'mekoneh la'k'pi' sh'tavora le'ul. v'tamid n'tchill m'q. at' at'ot ha'tchillik sh'hadg'mnu ca'n, ni'tun le'b'atz ul' k s'retim - ba'indok'zahia ao' b'cel dr'k ach'ra.

## 8.6 טగ'irot la'ayichud v'chit'oz sh'fotot br'iu'ot/k'vilot

cut'at n'shamsh b'mo'del ri'bi'i s'retim" ba'matzu'ot t'cavka shel ri'zha b'mekbel la'ho'chot ha'te'nu'ot ha'bo'ot ha'ba'a

te'una: y'hiyo  $L_1, L_2$  sh'fotot br'iu'ot, ai'  $L_1 \cap L_2$  cr'iyah.

ha'vachah: y'hiyo  $A, B$  sh'fotot ha'cr'iu'ot v'ha'mekone'ot sh'mek'ru'ot otton b'hata'ah  $M^A, M^B$ . n'vna ha'mekone'ah  $M^C$  sh'tcar'ui at'  $B \cap A$ . ru'yon ha'vachah y'hiha la'shamsh b'shni s'retim, r'ashit n'ut'ik at' ha'tocn' shel ha'seret ha'ra'oshon la'seret ha'shni v'oz' at' ha'ras'ha shel shni ha'seretim l'sh'maal ha'k'lat. ha'seret ha'ra'oshon y'sm'al at'  $M^A$  v'ha'shni at'  $M^B$ . n'p'ul ck:

a. n'ut'ik at' ha'k'lat  $w$  la'seret ha'shni v'v'haz'ar at' ha'ras'ha la'tchillat ha'seretim.

b. n'r'z at'  $M^A$  ul' g'bi ha'seret ha'ra'oshon, am d'ch'ha - d'ch'ha.

c. am  $M^A$  k'v'lah,  $M^C$  t'r'z ul' ha'seret ha'shni at'  $M^B$  - am k'v'lah: k'v'lah. am d'ch'ha - d'ch'ha. n'sim leb ci' ha'mekone'ah zo tamid u'z'chra, ci'yon sh'  $M^A, M^B$  m'cr'iu'ot sh'fotot v'lc'nu tamid u'z'chra.

te'una: y'hiyo  $L_1, L_2$  sh'fotot k'vilot, ai'  $L_1 \cap L_2$  k'v'lah.

ha'vachah: y'hiyo  $A, B$  sh'fotot k'vilot v'ha'mekone'ot sh'mek'ru'ot otton b'hata'ah  $M^A, M^B$ . n'vna ha'mekone'ah sh'tk'bel at'  $B \cap A$ . n'p'ul ck:

a. n'ut'ik at' ha'k'lat  $w$  la'seret ha'shni v'v'haz'ar at' ha'ras'ha la'tchillat ha'seretim.

ב. נרץ את  $M^A$  על גבי הסרט הראשון, אם דחתה - דחה.  
 ג. אם  $M^A$  קיבלה,  $M^C$  על הסרט השני את  $M^B$  - אם קיבלה: קבל. אם דחתה - דחה.  
 נשים לב כי מכונה זו חייבת לקבל את כל המיללים בשפה, ולכן לכל מילה בשפה נקבע נקבל שהיא קיבלה בשתי המכונות אותן. וכך גם בחיתוך. כלומר, כל מילה בשפה מתקבלת וכל מילה שאינה בשפה לא מתקבלת.

טענה: יהיו  $L_1, L_2$  שפות כרייעות, איי  $L_1 \cup L_2$  כריעה.  
 טענה: יהיו  $L_1, L_2$  שפות קבילות, איי  $L_1 \cup L_2$  קבילה.  
 רעיון הוכחה לשני סעיפים אלו - בדיקת כמה בשתי הטענות הקודמות. נרץ במקביל, אם נקבע באחת המכונות שהamille התקבלה איי גם המילה שלנו תתקבל. אם שתי המכונות ידרו - המילה לא בשפה.).

## 8.7 אוטומט עם שתי מחסניות P2

נדיר מודל חדש, אוטומט מחסנית עם שתי מחסניות. במודל זה, בכל מעבר ניתן להכנסיה/להוציא אותה בשתי מחסניות שונות, במקביל. פונקציית המעברים תוגדר להיות:

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

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

טענה: מודל P2 שקול למוכנות טיריניג במודל T.  
 הוכחה:  
 צד אחד פשוט, בהינתן מכונת טיריניג, ניתן לבנות ממנה אוטומט עם שני סրטים.  
 נכיח כי בהינתן מכונת טיריניג, ניתן לבנות ממנה אוטומט עם שתי מחסניות.  
 רעיון הבניה הוא שנדמה כל קונפיגורציה של הסרט במכונה  $M$  ע"י שתי המחסניות יחד ב- $P$ : במחסנית הראשונה נשמר את האותיות (למעט אינסוף הרוחים) בצד שמאל של הסרט עד מקום הראש (לא כולל), כאשר האות השמאלית ביותר בסרט תהיה בתחתית המחסנית. במחסנית השנייה נשמר את האותיות מהראש וימינה בסרט (למעט אינסוף הרוחים), כאשר האות העליונה במחסנית תהיה האות עלייה מצעע הראש והאות הימנית ביותר ב היתר הסרט תהיה בתחתית המחסנית. בתחתית של כל מחסנית יהיה הסימן  $\$$ .  
 האוטומט  $P$  מתחילה את הריצה כאשר שתי המחסניות ריקות. כדי לדמות את הקונפיגורציה ההתחלתית של  $M$  דרוש שלב אתחול שבו נמלא את המחסניות מותך תוכן הקלט שבסרט של  $Q$  (זכור שהאוטומט  $Q$ , ע"פ ההגדרה, קורא את הקלט באופן סדרתי, ואינו יכול לחזור ולקרוא שוב את האותיות שנקראו)  
 נגיד מצב אחד שיקרה את אותיות הקלט אחת אחר, ולכל אות שנקראת מהקלט - האוטומט יכניס אותה למחסנית 1. אחר כך נעבור למצב אחר, שיעביר את כל האותיות ממחסנית 1 למחסנית 2.  
 המשך הוכחה - בקמפוס מפורט.

## 8.8 סרט דו ממדוי - מודל 2D

המודל הדו ממדי הוא מעין מטריצה עם אנסוף עמודות ואנסוף שורות, אך מודל זה חסום בסרט מצד שמאל ומלמטה. כמתואר בתמונה:

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

הראש הקורא יכול לזרז ימינה, מטה, שמאליה ומעלה.  
פונקציית המעברים תוגדר כך:

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

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

**דוגמה.** נגידר מושג מתמטי, מס'  $x$  יקרא משולשים אם, ורק אם קיים  $\mathbb{N} \in n \in \text{cz}$  ש-  $i = \sum_{i=0}^n L$  כעת, נסתכל על השפה  $\{x\}$  הוא מספר משולשי  $|x| = 2D$ .

נרצה להכריע את  $L$  באמצעות  $2D$ . מה הרעיון? תחילה, נגידר שהקלט שלנו יהיה כמספר אונארי. כל הקלט בהתחלה יהיה בשורה התחthonה של המטריצה. בללאה זוורת: בכל שלב  $i$  נתיק את כל האחדות בשורה הקודמת, פרט ל-? השמאליות ביותר. ככלומר בהתחלה נתיק 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  $L$  ל- $T$ . נתחיל במספור הטבלה, כך שהتا השמאלי ביותר יהיה  $(0, 0)$  וכן הלאה נתקדם. לאחר מכן מיפוי חד-חד ערכי מהמטריצה לטבלה ע"י צירמת הטבלה עם התאים  $(0, 2), (1, 0), (0, 1), (2, 0), (1, 1), (0, 0)$ , וכן הלאה.... ככלומר - מעתיקים בכל פעם את האלכסון הבא אל המטריצה, כאשר מתחילה להעתיק אותו ממנה של כל אלכסון. כמו כן נמספר את התאים בהתחאם, ככה שהتا הראשוν יהיה 0, בהתאם לאלכסון השני תאים 1, 2 וכן הלאה.

נרצה להתאים את התוצאות.

בහינתן ראש שנמצא בסרט הדו ממדי בתא מס'  $i$ , בשורה  $y$  ובעומודה  $x$  - איזה

התא מימין לתא  $?$  הוא  $(x + y + 1)$

התא משמאלי לתא  $?$  הוא  $(x + y)$

התא מעל התא  $?$  הוא  $(x + y + 2)$

התא מתחת לתא  $?$  הוא  $(x + y + 1)$

נבחן כי אם נctrיך לזרז שמאליה בעומודה השנייה ביותר מאשר מקום כי הסרט חסום משמאלי, ואם נctrיך לזרז למטה בשורה התחthonה ביותר נשאר במקום כי הסרט חסום מלמטה. ככלומר, אם  $x = 0$  אז  $i$  והוא  $y = 0$  גם כן  $i$ .

הרענון יהיה ליצור מכונה עם 4 סרטים, שלפי שיקוליות קודמות שcolaה ל- $T$ .

בסרט העליון: יהיה המיפוי החד ערכי שייצרנו הסרט בודד מהטבלה הדו ממדי. הסרט השני ישמור את מקומות עמודה  $x$ , הסרט השני את מקום שורה  $y$  הסרט הדו ממדי והסרט האחרון את חישוב פונקציית המיפוי. ככלומר, הוא יחשב לאיזה תא הראש צריך להציבו לאחר המעבר. נבחן כי

פונקציית המיפוי היא ע"י ביצוע פעולות חיבור וחיסור ולכון ניתנת לחישוב ע"י מכונת טירינג.

**דוגמה.** נניח ויש לנו במקומות 0, 1 את האות  $a$  ונקלט  $\{q_0, b, U\} = \{q_1, b, \delta_{2D}\}$  אזי בסרט הראשון ישמר  $a$  בסרט השני ישרם 0 ובסרט השלישי תוחשב פונקציית המיפוי למשהו  $2 + 0 + 2 + 0 + 0 = 2$ .

## 8.9 מכונת טירינג שאינה דטרמיניסטית

כידוע, במודל זה יהיה מס' מצבים אליהם ניתן יהה עבור קלט נתון. נגידר היטב את מושג הדחיה והקבלת מודל מסווג זה:

**הגדרה:** תהי  $M$  מכונת טירינג לא דטרמיניסטית וכן  $w \in \Sigma^*$ .  
 $M$  מקבלת את  $w$ , אם קיים חישוב של  $M$  על  $w$  ש兆ע 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:  
 $M$  מכירעה את  $L$  אם לכל  $w \in \Sigma^*$ :  
אם  $w \in L$   $M$  מקבלת את  $w$ .  
אם  $w \notin L$   $M$  דוחה את  $w$ .

**הגדרה:**  $M$  מקבלת את  $L$  אם לכל  $w \in \Sigma^*$  מתקיים  $w \in L$  אם ומן  $M$  מקבלת את  $w$ .

דוגמא. נגידר מילים על מהירות  $\bar{I} = 0, \bar{I} = 1, \bar{0} = 001101$  ו $\bar{0} = \overline{110010}$ . נגידר את השפה

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

כלומר, אוסף המילים בהם קיימת טיפה של המילה, שכן רעליל עליה את המשלים נכלל פלינדרוס. למשל,  $110001$ , אס' נסתכל על  $00 = \bar{u}$  מתקיים  $11 = \bar{v}$  ואז אנו מתקיים השוויון. וצ'ה לבנות אוטומט לא דטרמיניסטי שיקבל שפה זו.

פתרון - יהוו להשתמש בטבלת המעברים שיצרוו למכונית הטירינג של שפת הפליינדרוס + התוספת כאו מטה:

| מזהה  | כתב             | כתב חדש | סימון           | כתב | מזהה |
|-------|-----------------|---------|-----------------|-----|------|
| PAL:  | $q_0$           | —       | acc             | —   | R    |
| .     |                 |         |                 |     |      |
|       | back            | —       | $q_0$           | —   | R    |
| FLIP: | $\widehat{q_0}$ | 0,1     | $\widehat{q_0}$ | —   | R    |
|       | $\widehat{q_0}$ | 0,1,—   | flip            | —   | S    |
|       | flip            | 0       | flip            | 1   | R    |
|       | flip            | 1       | flip            | 0   | R    |
|       | flip            | —       | back            | —   | L    |
|       | back            | 0,1     | back            | —   | L    |
|       | back            | —       | $q_0$           | —   | R    |

הרוינו - נוסר מכך חדש שיסורק את כל הקלט משמאלו לימיון, וקרו לו  $\phi$ , וכן מכך *flip* שישנה כל 0 לאחד וכל אחד לאפס. הרוינו והוה להפעיל כל פעע את *flip* על חלק אחר בקלט, באפוא סימטריו בו הנקודות, ואז לקרו *PAL*, שתבזוק אם המחרוזת שהתקבל היא פילומרים - ואם, ורק אם זה יקרה אז נך לנצח מוקבל.

כעת, המכואה תעורר בזאות על כל מילה ששייכת לשפה ולא תעורר על מילים שלא שייכות לשפה, ולכן המכואה מקבלת את השפה  $L$ .

**השימוש במכונה טיירינג לא דטרמיניסטי שימושי במיוחד עבור קבלה של שפות.**

### 8.9.1 הכרעה וקבלת של שפות

עבור מכונה לא דטרמיניסטית  $N$  ושפה  $L$ :

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

מכירעה את  $N$  אם  $N$  מקבלת את כל המילים בשפה ולא מקבלת את כל המילים שלא בשפה.

נשים לב להבדל - במכונה דטרמיניסטי ישנו היישוב ייחידת אם הוא עוצר במצב מקבל המילה מותקבלת. במכונה לא דטרמיניסטי יש הרבה חישובים אפשריים - מילה מתקבלת אם באחד המסלולים היא התקבלה. ונדחית אם בכל החישובים נחתה.

**טענה:** מודל מכונת טיירינג הלא דטרמיניסטי שקול למודל הדטרמיניסטי (הערה, זה לא נכון בזמן ריציה אך לא נדוע בכך).

**הוכחה:** הרוינו יהיה שהמכונה הדטרמיניסטי שנבנה  $M$  תעבור על כל הקלטים האפשריים, אם אחד התקבל נך במצב מקבל.

בידיינו טבלת מעברים. נשים לב שעבור אותן קלט  $s$  ומצב  $q$  יתכן כי לא דטרמיניסטי ( $s, q \in \Sigma^*$ ) מיעברים שונים, א' למשל, נסמן אותם במספרים  $1, \dots, k$ . בעת - ניצור מכונה עם שני סרטים. על הסרט הראשון נרץ את הקלט. בסרט השני נכתוב תחילת מס' אחדות ממש' מס' מס' מס' מספר המעברים באוטומט. בעת, נבצע סימולציה על המכונה באמצעות המספרים: תחילת, כל המספרים יהיו 1, וכן נפעל לפי אפשרות אחת. אח"כ המצב השני יהיה 2 נניח, ונריץ את הסדרה  $1, 1211111, \dots, 1, 112111111$ , וכן הלאה - וכך העבור על כל המסלולים האפשריים. נקבע מילה ונעצור אמ"מ באחד המסלולים הגענו במצב מקבל. סה"כ מכונה עם שני סרטים כפי שראינו בעבר שcolaה למcona עם סרט אחד, וכן ישנה שקידות. נuir כי פורמלית יש לעשות סרט נוסף, להעתיק אליו את הקלט, ושם לבצע את ה"עבודה" ובדיקת האפשרויות השונות בשביל לשומר על הקלט המקורי בשלמותו.

#### פורמליות -

1. כתוב 1 בסדר בחריות.
2. העתק קלט מסרט כספת הקלט" לסרט עבודה.
3. הרץ את המכונה  $N$  על סרט עבודה" לפי הסדרה שבסרט בחריות". בכל צעד בדוק אם המכונה  $N$  הגיעו במצב  $acc^N$  עבור במצב  $acc^M$ .
4. מחק תוכן סרט עבודה.
5. בסריט בחריות כנוב את המחרוזת הבאה לפי סדר מניה (שאלה, למה לפי סדר מניה ולא לקסיקרפי? יתכן חישובים שלא ידחו או יתקבעו אלא יתקעו אז לא נעצור לעולם, لكن אסור לנו להמשיך כל הדרך בחישוב אחד אלא צריך לשנות בכל פעם את המסלול לאחד חדש, שלא נתקע לעולם בתוך חישוב אחד).
6. חוזר לשלב 2.

### 8.10 סגירות באטעןויות איד-דטרמיניזם

**טענה:** תהי  $L$  שפה שמתقبلת ע"י מכונת טיירינג לא דטרמיניסטי, איז גם  $(prefix(L))$  (תחיליות) גם היא מותקבלת ע"י מכונת טיירינג לא דטרמיניסטי.

**טענה:** יהיו  $L_1, L_2$  כריעות. איז  $L_1 \circ L_2$  כריעה.

רעיון הוכחה: נחלק את המחרוזת לרישא וסיפא, ובכל פעם את הרישא נרץ על גבי המוכונה  $A$  שמקיימות  $L(A) = L_1$  ואת הסיפא על  $B$  המקיים  $L(B) = L_2$ . כיוון ש  $A, B \in A$  מכוורות את השפות בהתאם, בהכרח עברו כל מחרוזות נקלט או דחיה או קבלה. קבלה של מחרוזות תהיה אם המחרוזת של הסיפא ושל הרישא התקבלה, ודחיה אם אחת מהן נדחתה. כך נרץ על כל הרישות והסיפות האפשריות. כיצד? בכל שלב נעתיק את האות השמאלית ביותר אחר, ונרץ את המוכונה. אם נקלט סימנו, אחרת נמשיך להעתיק אותן הבאה. כך נעבור על כל רישא וסיפא אפשרית. בכל שלב שזכה נרץ על הסרטן השני את  $M_A$ , אם קיבל נרץ על הראשון ( $\text{prefix}(L)$ ) או  $M_B$  אם קיבלה נקבול, אחרת נדחה. נשים לב שכיוון שהשפות כrüיעות, בכל אחת מהמוכנות תמיד נדחה/נקבל וכך לא נגיע למצב של אי עצירה.

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

**הוכחה:** אם  $L$  כrüיע, אז  $L^*$  כrüיע, ואמם  $L$  קבילה אז  $L^*$  קבילה.  
אם  $M$  מכונת טיריניג דטרמיניסטי שמכוורה את  $L$ . נבנה מכונת  $N$  לא דטרמיניסטי שמכוורה את  $L^*$ . המכונה  $N$  שני סרטנים, שליהם נקראו: "קלט" ו"עובדיה". הרעיון הוא ש  $N$  תפרק את הקלט שלו באופן לא דטרמיניסטי לחלקים. היא תעתק כל "חלק" מסרטן ה"קלט" לסרטן "עובדיה", ואז תפעיל את המכונה  $M$  על סרטן ה"עובדיה". רק אם המכונה  $M$  קיבלה את כל החלקים, אז המכונה  $N$  גם תקבל. אחרת – היא תדחה.

ספקטיבית, המכונה  $N$  תפעל בקורס הבא:  
אם בסרטן "קלט" רואים רווח – קבל כל עוד הסרטן קלט לא רואים רווח, חזר על התהליך הבא בחר באופן לא דטרמיניסטי בין האפשרויות הבאות: א. העתק אות מסרטן "קלט" לסרטן "עובדיה", וזה את שני הראשונים ימינה ב. היז את הראש הסרטן "עובדיה" לטו השמאלי ביותר שאינו רווח ועבור לשלב 3, אם אין צזה – בצע את שלב א" הרץ את המכונה  $M$  על "סרטן עובדה", עד לעצירה אם החישוב הסתתיים במצב דחיה – דחיה. מחק תוכן סרטן "עובדיה" וחזור לשלב 1.  
נשים לב שכיוון ש  $M$  מכונת להכרעה, שלב 3 בהכרח מסתיים, ולכן כל התהליך בהכרח מסתיים. ולכן, זהה מכונת להכרעה. אם קלט  $w$  ב  $L^*$  אז יש לו פיצול  $w_k \dots w_1 = w$  כך שכל  $i$  מותקים  $L \in w_i$ . לכן, החישוב של  $N$  שבשלב 2 בדוק מעתיק בכל פעם את ה  $w_i$  המתאיםibia את המכונה למצב מתקבל.  
ומצד שני, אם  $w$  לא ב  $L^*$ , אז לכל ריצה, בהכרח יהיה חלק שמוועתק הסרטן "עובדיה" שהוא לא בשפה, ולכן המכונה תדחה בשלב 3.

**טענה:** תהי  $L$  כrüיע. אז  $\text{reverse}(L)$  כrüיע.  
הוכחה: אם  $L$  כrüיע קיימת מכונת טיריניג  $M_L$  שמכוורה אותה. המכונה  $M_R$  ראשית תהפוך את סרטן הקלט, ואז תרץ את המכונה  $M_L$  שתקבל או תדחה בהתאם. (שכן אם הפכנו את סדר המילים, המילה תהיה ב  $L$ ).

## 9. ייחידה 10: התזה של צראץ'-טיריניג

האם מכונות טיריניג היא המודל חזק ביותר שnochol למוצוא, שמותאים למחשב מודרני? ביחידה זו נגלה.

### 9.1 סגירות

- יהיו  $L_1, L_2$  שפות כrüיעות האם יש למוגבלותה הבאה סגירות?
- איחוד  $L_1 \cup L_2$  – כן
  - חיתוך  $L_1 \cap L_2$  – כן
  - שרשור  $L_2 \circ L_1$  – כן
  - סגור קלין  $L_1^*$  – כן
  - שפה התחילית  $\text{prefix}(L_1)$  – כן
  - רוורס  $-\text{Reverse}(L_1)$  – כן

ג. משלים:  $\overline{L_1} - \text{cn}$

- יהיו  $L_1, L_2$  שפות קבילות האם יש למניפולציה הבאה סגירות?
- איחוד  $L_2 \cup L_1$  - cn
  - חיתוך  $L_1 \cap L_2$  - cn
  - שרשור  $L_1 \circ L_2$  - cn
  - סגור קלין  $L_1^*$  - cn
  - ( $\text{prefix}(L_1)$  - cn) אין סגירות למשלים ולא להורס.

## 9.2 היחס בין הכרעה לקבלה

טענה: אם שפה כריעה, היא בהכרח קבילה.  
הוכחה: כריעה, יש מכונה שמקரיעה אותה. בהכרח, היא גם מקבלת אותה. כנדרש.

טענה: אם  $L$  ומס' קבילות, אז  $\overline{L}$  כרעה.  
הוכחה: יש מכונות שמקבלות את  $\overline{L}$  ו-  $L$ . נוצר מכונה חדשה  $D$ , ובמצע סימולציה של הריצה על המכונות. בהכרח, כל מילה שייכת או  $\overline{L}$  או  $\overline{L}$ , ולכן אחת 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 מהמילה. מכ"כ כמובן, אם המילה התקבלה במכונה של  $L$  אז המילה בשפה ואם התקבלה במכונה של  $\overline{L}$  המילה לא בשפה, וכך נדחה וסה"כ אכן  $\overline{L}$  כרעה.

## 9.3 מכונות טיריניג שוקלה לתוכנית מחשב

סוף סוף הגענו לטענה שלנו - מכונת טיריניג חזקה לפחות כמו תוכניות מחשב.  
נចטריך ראשית להגדיר מהו מחשב. במקומות לדבר עליון, נגיד תוכניות מחשב. מה שנעשה יהיה להראות כי ככל בעיית הכרעה שניתן לפטור באמצעות תוכניות מחשב, ניתן לפטור באמצעות מכונות טיריניג.

הרעיון הוא להראות שכל דבר שניינו לעשוות בשפת תכנות, ניתן לעשוות באמצעות מכונות טיריניג. באיזה שפת תכנות נבחר? אם נבחר את לאוואה או פיאטון לא נסימן בקרוב כי יש בהן המון דברים. כמו כן נשים לב לטענה: ככל דבר שניינו לעשוות בשפת תכנות  $A$ , ניתן לעשוות בשפת תכנות  $B$  (פרט לשפות ספציפיות מאוד). לשם כך נגיד שפת תכנות משלנו בשם SIMPLE שתכלול את כל הדברים החינוניים לשפת תכנות. ככל תוכנית בכל שפה תוכל להיות מומרת לשפה SIMPLE. השפה תכלול:

- משתנים טבעיים  $i, j, k$  שמיישמו משתנים
- מערכות טבעיות  $A[], B[], C[]$ ... בכל תא ערך מותך א'ב ג'. המרכיבים הם אין סופיים.
- אתחולל - הקלט נמצא בתאים הראשוניים של המערכות, המשתנים מאותחלים לאפס.
- פעולות:
  - השמה - בקבוע, למשל  $A[2] = \# i$ . וכן השמה בין שני משתנים  $j = i$  או  $A[2] = B[7]$ .
  - פעולות חשבון  $a = x + y, b = x - y, c = xy$
- תנאים:
  - שווין בין תנאים בערך  $A[i] == B[j]$  או לבדוק אם שווין  $j \geq i$ . כל משתנה מופיע רק פעם אחת בשורה, כלומר לא ניתן לכתוב  $j = j + k$ .
- זרימה

הפעולות ממושפרות, ומtbodyות אחר השניה. פרט לפקודה goto שהולכת לשורה ספציפית שיכולה לבוא במהלך הריצה. כמו כן יש פעולה stop(a) שמחזירה ערך  $a$  ועיצרת.

**דחיה וקבלה:** עבור קלט  $w$  ותוכנית  $P$  בשפה SIMPLE נאמר כי:

$P$  מקבלת את  $w$  אם הריצה של  $P$  על  $w$  עצרת עם ערך חזרה 1.  
דוחה את  $w$  אם הריצה של  $P$  על  $w$  עצרת עם ערך חזרה 0.

**הכראה וקבלת של שפות:** עבור שפה  $L$  ותוכנית SIMPLE  $P$  מכירעה את  $L$  אם היא מקבלת את כל המילים שבל- $L$  ודוחה את כל המילים שאינן ב- $L$ .  
מקבלת את  $L$  אם היא מקבלת את כל וرك המילים ב- $L$ .

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

**הטענה:** המודל של מכונת טיריניג ומודל SIMPLE הינם שקולים.  
**הוכחה:**

כיוון ראשון: לכל מכונת טיריניג יש תוכנית  $P$  שcola. לא תעכב על כיון זה כי די ברור שככל שfat יูลית יכולה לדמות באמצעות מבנה נתונים כלשהו את מודול מכונת הטיריניג.  
כיוון שני: לכל תוכנית בשפה SIMPLE יש מכונת טיריניג שcola.  
נראה כיצד למשם את השפה במכונת טיריניג.  
משתנים - לכל משתנה יהיה סרט נפרד. במערכות: לכל תא במערך יהיה תא בסרט הקטל. במספריים הטבעיים: יהיה יציג אונרי של אחדות. כל הסרטים מאוחדים לרווח שיתאר לנו ערךAPS.

פעולות: השמה בין משתנים טבעיים היא העתקה מסרט אחד לשני. השמה בין ערכאים במערך  $[j][i] = B[i]$ , ראשית הולכים ל- $i$ , כיוון שהוא יציג אונרי זה קל: הולכים למערך  $A$  ולסרט של משתנה  $i$  ומתקדמיים בהתאם על הסרט של המשתנה כל פעם צד ובמקביל במערך, עד שנגיע למיקום הרלוונטי. בדומה על  $j$  במערך  $B$  ומשנים בהתאם את הערכאים בסרטים. השמה בקבוע נעשית באופן דומה. השמה בקבוע למשתנה טبعי - ראשית מוחקים את כל הקטлы הקויים הסרט המשנה, אז מוסיפים אחדות בהתאם ליצוג האונרי המתאים. באשר לפועלות חשבון - הן כפל הן חיבור והן חיסור ראיינו בעבר כיצד למשם במכונת טיריניג (מופיע כאן מעלה).  
תנאים: כיצד בודקים אי שוויון? ריצה במכונות מהתחלה, ובבודקים היכן מגיעים בראשונה לרוחות. מס' גדול יותר אם הגיענו לאחר יותר - כי יש בו יותר אחדות. ואיך בודקים אם הגיענו לרוחות באותו הזמן. השוואה בין תאים במערכות - מגיעים אליהם פשוט ובודקים אם הערך בפנים זהה במערכות שווין שתזאור לעיל.  
זרימה: במכונת טיריניג כל פקודה ממושפרת. מצבים המכונה הם הפקודות השונות. לכל מצב גם טבלת המעברים קבועה לאיזה מצב עוברים - וכן קל למשם goto. וכן בהתחאם stop(1) = acc וcn stop(0) = rej  
סה"כ, לכל תוכנית מחשב יש מכונת טיריניג עם מס' סרטים שcola, שלא כפי שראינו יש מכונת טיריניג עם סרט אחד שcola. כנדרש.

■

**מעתה ואילך, מחשב=מכונת טיריניג. כל שפה שכריעה/קבילה ע"י מחשב, קרייה/קבילה ע"י מכונת טיריניג.**

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

**פסוזו קוד:** כיוון שכל מכונת טיריניג שcola לשפה simple, מעתה נוכל לכתוב את מכונת הטיריניג בשפת simple או בכל שפת תכנות שהיא. ויתרה מזאת - ניתן וכך נעשה: נכתב מעתה פסוזו קוד לתרגילים. לצורך הדוגמה: נרצה להכريع את השפה

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

לשם כך נכתוב את 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 דטרמיניסטית 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  $[a] \rightarrow aa[a]$ . המשמעות היא שבמהלך כל היצירה ניתן להחליף את המחרוזות השמאלית, בימנית. כך למשל, בהינתן המחרוזות  $[aa]$  לפי כלל היצירה  $\rightarrow [aaa]$ .

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

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

בעזרת חזרה על הכללי השמאלי ניתן ליצור מחרוזות עם מס' זהה של  $a$  ו- $b$ , שני הכללים האחרונים אפשריים סידור חדש של אותיות המחרוזת.

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

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

הweeney יהיה שנגזר מילה עם מס'  $a$ -ים שווה למס'  $b$ -ים שווה למס'  $c$ -ים, ולאחר מכן נדרש בסדר את המחרוזות בסדר הרצוי של קודם  $a$ -ים אחר  $b$ -ים ואחר  $c$ -ים. מחרוזות תתקבל רק אם סיימנו אותה ללא סוגרים. כל עוד המס מופיעות, לא סיימנו את תהליך הגזירה. די להשתכנע שנגזרה זו עבדת - מס' דוגמאות יוששו זאת.

**טענה:** שפה הינה קבילה, אם ו רק אם קיימים דקדוק כללי שיוצר אותה.  
**הוכחה:** כיוון ראשון - בהינתן דקדוק כללי נבנה מכונת טיריניג שמקבלת את הדקדוק, אך בשל השקילות של מכונות טיריניג לתוכנית מחשב, ניתן לבנות תוכנית מחשב שיוצרת את הדקדוק. נבנה תוכנית לא דטרמיניסטיבית:

קלט:  $w$   
 $u = S . 1$   
 $repeat : .2$   
 פצל באופן לא דטרמיניסטי את  $u$  ל- $z$

בחר באופן לא דטרמיניסטי גירה  $v \rightarrow t$  של  $G$   
 אם  $y \neq t$  דחה  
 $u = xz$   
 אם  $u == w$  קיבל

**כיוון שני** - נתונה מכונת טיריניג  $M$ , נבנה ממנה דקדוק כללי  $G$  כך ש  $L(M) = L(G)$ .  
 נסביר רעיון כללי - לצורך הבנה בלבד: במכונת טיריניג קונפיגורציה היא מהצורה זו  
 $a_0 baa$  למשל. נניח ונסתכל בביטול המעברים ונראה כי

$$a_0 baa \vdash_M a a q_1 a a$$

אזי, הרעיון יהיה להגדיר את הדקדוק לפי הקונפיגורציות, כלומר

$$a_0 baa \Longrightarrow_G a a q_1 a a$$

מה השתנה כאן בין המחרוזות?  $a q_1 \rightarrow a_0 b$ , ולכן זה יהיה כלל יצירה ב- $G$ . וכך, באופן דומה,  
 נגדיר את כל הדקדוק לפי הקונפיגורציות.  
 ומה באשר לתזוזה שמאליה? למשל -  $a a q_1 a b \vdash_M a q_0 a b b \rightarrow q_0 a b$ , זה יתורגם לכל  $a q_1 b \rightarrow a_0 a b$ .  
 באופן כללי: אם  $\delta(q, \sigma) = (p, \pi, R)$  אז  $\delta(q, \sigma) = (p, \pi, L)$ . אם  $\delta(q, \sigma) = (p, \pi, R)$  אז  $\delta(q, \sigma) = (p, \pi, L)$  לכל  $\Gamma \in \tau$ :  
 $\tau q \sigma \rightarrow p \tau \pi$ .

## 9.5 היררכיה של חומוסקי

לפי היררכיה מתקיים:  
 בתחתיות הפרמידה, **השפות הרגולריות** שמתקבלות ע"י **דקדוק רגולרי**, וע"י מודל האוטומט  
**הסوفي**.  
 מעלייהם, השפות **חסירות ההקשר** שמתקבלות ע"י **דקדוקים חסרי הקשר** וע"י מודל אוטומט  
**מחסנית**.  
 מעלייהם, **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'טיריניג

מכונת הטיריניג המכונה ע"י אלן טיריניג במאיה הקודמת.  
 במאיה הקודמת, בסביבות שנות העשרים - נכנס עולם המתמטיקה למשבר קיומי סביב הפורמליות של  
 המתמטיקה. לכן הגיע דיוקן הילברט להניח מסמך עקרונות הבא:  
 1. **שלמות:** כל טענה שנכונה מתמטית, ניתנת גם להוכחה.  
 2. **נאוטות:** רק טענות נכונות ניתנות להוכחה.  
 3. **בריעות:** לפתח שיטה בעזרתה יהיה ניתן לקבוע האם טענה נכונה או לא.

בהמשך, הוכח שיעד מס' 1 אינו ניתן להשגה. ויעד מס' 2 - כן ניתן להגשה. באשר לעיד מס' 3,  
 אלן טיריניג פיתח את מכונת הטיריניג כדרך לתאר אלגוריתם. המרצה שלו, צרצ'טיריניג: כל אלגוריתם שניין לתיאור  
 לאחר מכן ניסחו את התזה שנקראת **התזה של צרצ'טיריניג**:

כלשהן, ניתן גם לתיאור מכוכנת טיורינג. ובפרט, אין מודל חישובי חזק יותר מכוכנת טיורינג. **ניסיונות** לב שזו תזה ולא משפט שהוכח מתמטי.

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

## 10. יחידה 11: אי כריעות

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

### 10.1 אimotoת תוכנה

ناس"א שלחה ב-1998 Challie challiet לحلל. אמהה אבד הקשר אליה בכניסה למאים. כשבדקנו מידע, גילו שעבדו על החלטית שני צוותים שונים שהתייחסו ליחידות המידה באופן שונה: צוות אחד התייחס במטרים ואחד ב-yards. והחותמות ברורות. נאס"א רצתה למנוע זאת. כמובן בהינתן תוכנית של החלטית ומפרט - האם התוכנית עומדת בדרישות המפרט? זו בעיה הכרעה. בהינתן תוכנית  $P$ , ומפרט  $S$ , נגידיר את השפה  $\{P \text{ תוכנית, } S \text{ מפרט וכן } P \text{ עומדת בתנאי המפרט} | S\}$ .  
 $PS = \{(P, S) | P \text{ תוכנית, } S \text{ מפרט, } P \text{ עומדת בתנאי } PS\}$

נניח ונרצה לפתח תוכנית להכרעת  $PS$ . נקרא לה  $D - PS$ . שמקבלת תוכנית  $P$  ומפרט  $S$ . כיצד נקבל תוכנית? תוכנית היא רצף של תווים ולבן הקלט לתוכנית מחשב יכול להיות תוכנית. (למשל, הקומפיילר מקבל תוכנית, ממיר אותה לשפת מכונה, ומהזיר 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 חשושת שהתוכנית לא עובדת טוב עבור קלט מסוים,  $w$ . במקרה זה, בעיית ההכרעה היא:

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

כמובן, האם בהינתן תוכנית ומחרוזת מתקיים  $1 = (w)(P)$ ? כמובן, שנريץ את  $w$  על התוכנית, נקבל 1. זו גרסה מנוגנת מזו של הבעיה המקורית: כיון שמתיחס רק לדבר אחד במפרט, ודורשת לעבור על כל קלט  $w$  אפשרי. אבל - זה אבל גודול, היא תנאי הכרחי (ולא מספק) לכל נסיוון לענות על הבעיה המקורית. נשים לב שדרישה היא גם שהתוכנית تستטיים, כי אם לא تستטיים ותרוץ באופן אינסופי בפרט היא לא תחזיר אחד. האם  $ATM$  כרעה? על כךណוןCut.

### 10.2 קבילה ATM

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

הוכחה: כיצד נבנה תוכנית שתתקבל תוכנית אחרת? זה בדיק מה שעשויה מערכת ההפעלה של המחשב: היא בעצמה תוכנית, תוכנה, שמקבלת תוכניות ומוציא אותן. נסמן תוכנה זו ש谋יצה תוכניות שהיא מקבלת  $U$ . כמובן -

$U(P, w)$  מוציא את  $P$  על  $w$  ומחזירה את ערך החזרה של החזירה.  
**נשים לב כי:**

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

$U$  מחזירה את ערך החזירה שהתקבל מהריצה של  $P$  על  $w$ . מריצה את התוכנה  $P$  על הקלט  $w$  (במקרה שבו  $P$  אינה תוכנית מחשב תקינה אז  $U$  מחזירה ערך 10). נשים לב שאם  $P$  לא עוצרת על  $w$  אז גם  $U$  לא עוצרת על הזוג  $(P, w)$ . התוכנה  $U$  פועלת באופן דומה לאופן 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 טירוגינ אוניברסלית - מכונת טירוגינ שהקלט שלה הוא תיאור מכונת טירוגינ + 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  $D - ATM$  מקבלת  $(P, w)$ . נבנה תוכנית אחרת - בשם  $z$  בהינה מהירות הקלט של  $z$  נתארה כך:

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

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

עת, נרצה להריץ את  $Stupid(Stupid)$ . מה יחזיר? נשים לב שבפרט תמיד יוחזר משחזרו, כיון שהוא תמיד מוגדר כי  $D - ATM$  הינה תוכנית להריצה. לכן תמיד נחזיר ערך כלשהו. נשים לב כי  $Stupid(Stupid) = 1/0$ .

نب"ש  $D - ATM(Stupid, Stupid) = 1$  איי.  $Stupid(stupid) = 1$  אבל  $Stupid(Stupid, Stupid) \in ATM$ . ולכן  $a = D - ATM(z, z); z = Stupid$  הנ"ל כאשר  $a = 1$ . ולכן, יוחזר סה"כ  $1$ .  $Stupid(Stupid) = 0$  סתרה.

نب"ש  $D - ATM(Stupid, Stupid) = 0$  מכאן,  $Stupid(Stupid) \notin ATM$ . כלומר  $Stupid(Stupid) \neq ATM$ . ולכן בשרותה לעיל כאשר  $a = 0$ . ולכן יוחזר  $0$ !.  $Stupid(Stupid) = 0$  נקבע  $z = stupid$ . כלומר  $z = 0$ . נשים לב כי  $Stupid(Stupid) = 0$  סתרה.

סה"כ -  $1 \neq 0, 0 \neq 1$  בסתרה כי אלו שתי אפשרויות היחידות שיכלולות להיות, ולכן  $ATM$  לא כריעה. ■

**קיבלונו לראשונה שפה לא כריעה - בעיה שלא ניתן לפתרון ע"י מחשב: בעיית אימות תוכנה לא פתירה ע"י מחשב.** לא ניתן לקבוע אלגוריתם כללי שקבע 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. ולכן  $\overline{ATM}$  לא כריעה. עם זאת,  $ATM$  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) נקבע  $ATM$  כרעה. בסתרה.

**מסקנה:** כיצד נוכיח ששפה אינה קבילה? נוכיח שהשפה לא כריעה, והמשלים שלה כן קבילה. אז אם סקנה כמו בדוגמה כאן, השפה שלנו לא קבילה (אחרת בסתרה למשפט לעיל).

**הערה:** נסמן תוכניות להכרעה בקידומת  $D$ , ותוכניות ל渴渴ה בקידומת  $A$ .

## 10.5 בעיית העצירה

נגיד את השפה הבאה:  $\{(P, w) | P(w) \downarrow\} = HALT = \{(P, w) | P(w) \text{ הפסיקת } w \text{ עוצרת}\}$  (כלומר - התוכנית לא נתקעת). החץ מסמל עצרה.

**טענה:**  $HALT$  אינה כריעה.

**הוכחה:** נב"ש כי  $HALT$  כריעה. תהי  $D - HALT$  התוכנית שמכריעת את  $HALT$ . (לא ידוע מה הקוד.)

نبנה תוכנית  $D - ATM$  שמכריעת את  $ATM$ , וזה תהייה הסתירה:

$D-ATM(P, w)$ :

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

מה קורה כאן? אם אין עצרה על  $w$ , אז התוכנית  $p$  לא מקבלת את  $w$  ולכן מחזירים אפס. אחרת, התוכנית לא עוצרת, מובטח לנו שיש עצרה של  $p$  על  $w$ . ואז - פשוט נristol את המכונה היררכו-אליטית  $U$  על התוכנית עם הערך  $w$ , ונחזיר את הערך המתקבל. נראה כי  $U(P, w) = 1 \iff P(w) = 1$  ולכן  $P(w) = 1 \iff U(P, w) = 1$ . סה"כ - ייצרנו תוכנית שמכריעת את  $ATM$ , ולכן יתקבל 1 אם ו רק אם העצרה של  $P$  על  $w$  החזירה 1. סה"כ - ייצרנו תוכנית שמכריעת את  $ATM$ , ולכן  $ATM$  כרעה, בסתרה.

■

**טענה:**  $HALT$  קבילה

**הוכחה:** נבנה תוכנית שמקבלת את  $HALT$

$A-HALT(P, w)$ :

```
U(p, w)
return(1);
```

אם  $P$  עוצרת על  $w$ , אז התחליק בשורה השנייה יסתתיים וועברים לשורה השלישי. אם  $P$  לא עוצרת על  $w$ , אז נשאר תקווים בשורה 2 ולא נגע לשורה השלישי - אך לא נחזיר 1. ככלומר סה"כ התוכנית מחזירה 1 אם ו רק אם התוכנית עוצרת ולכן  $HALT$  קבילה.

**טענה:**  $\overline{HALT}$  לא כריעה, ולא קבילה. (ההוכחה בדיקת כמו ב- $ATM$ .)

| קבילה | כריעת |                   |
|-------|-------|-------------------|
| +     | ✓     | $ATM$             |
| +     | ✗     | $\overline{ATM}$  |
| ✓     | ✗     | $HALT$            |
| ✗     | ✗     | $\overline{HALT}$ |

## 10.6 שפות לא פתירות

שפה שאינה כריעת קבילה נקראת שפה לא פתירה.

## E השפה 10.6.1

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

כלומר, שפת כל התוכניות כך שהשפה שלחן ריקה. למשל,  $Q(x) : while(1) : E$  כי התוכנית לא עוצרת לאף קלט, ובפרט לא מקבלת אף קלט.  
**טענה:** לא קריאה.

**הוכחה:** נב"ש כי  $E$  קרואה. תהי  $D - \overline{ATM}$  התוכנית שמכריעה את  $E$ . נבנה  $D - \overline{ATM}$  שמכריעה את  $\overline{ATM}$  שהיא אינה קרואה, אז מקבל סתייה(. 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  $w$  מחזיר 1, אז  $U(P, w)$  תחזיר 1 לכל קלט. ואם  $P$  על  $w$  לא מחזיר 1, אז  $Q$  לא מקבל שום קלט. ולכן השפה של  $Q$  ריקה אם ו רק 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  $P$  מחזיר 1 על הקלט  $w$ , אז השפה של  $Q$  היא של הא"ב. אחרת, השפה ריקה. ובמילים אחרות: לא חזר בדיק החפץ. אם  $P(w) = 1$  אז יוחזר מ-  $D - E(Q)$ , כי השפה אינה ריקה. המטרה בשורת  $Q$  היא לתרגם את הזוג  $(P, w)$  לשורת קוד אחות אותה נשלח לתוכנית  $D - E$ .  
לכן,  $Q \in E \iff Q \in \overline{ATM} \iff (P, Q) \in \overline{ATM}$ , בסתירה כי  $\overline{ATM}$  לא קרואה.

למבנה הוכחשה יהיה כאן - קוראים **ודקציה** שתכח נטו על כך. מניחים בשלילה, מקבלים קופסה שחורה (לא יודעים מה קוראה אליה בפנים, כאן  $- E$ ) וממנה יוצרים קופסה שחורה גדולה יותר שתוביל לסתירה.

**טענה:** לא קבילה.  
**הוכחה:** נוכיח ש  $\overline{E}$  קבילה, אז נב"ש כי  $E$  קבילה, ונקבל כי לפי משפט  $L \leq \overline{L}$  קבילות  $L \iff$  קרואה( בסתייה לכך ש  $E$  אינה קרואה).

כעת נוכיח  $\overline{E}$  קבילה:  $\{Q | L(Q) \neq \emptyset\} \neq \emptyset$ . 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 הנ"ל אינה דטרמיניסטית. תחילתה היא מנחשת מחירות  $w$  כלשהי, ואז היא שולחת אותו למכוונה הורטואלית. אם השפה לא ריקה - לבסוף היא תצליח לנחש מילה כלשהי. אם המילה ריקה: לעולם לא יוחזר 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 ששפטן זהה.  
טענה: נב"ש  $\overline{EQ}$  אינה קבילה.

הוכחה: נב"ש  $\overline{EQ}$  כ"כ קבילה. תהי  $A - \overline{EQ}$  שמקבלת את  $EQ$ . נבנה  $A - EQ$  שמקבלת את  $E$ , וזו נקבל סטירה - כיון  $Sh$  לא קבילה. מה הרעיון? בידינו קופסה שחורה,  $A - EQ$ . נרצה להכניס לה שתי תוכניות  $Q_1, Q_2$  ונשים לב שהיא תחזיר  $cn/la/takku$  ולא תעצור. מכך קופסה הדולה יותר  $A - E(P)$  שמקבלת את  $E$ . גדריה כך:

A-E(P):

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

נשים לב כי  $Q_1$  היא תוכנית עם שפה ריקה, כי תמיד מחזירה אפס. השפה של  $P$  שווה לשפה של  $Q_2$  אם השפה של  $P$  ריקה. לעומת זאת, מתקיים  $L(Q_1) = L(Q_2) = \emptyset$  ולכן  $(Q_1, Q_2) \in EQ$ . אם  $AE$  מחרירה על  $P$ . כלומר סח'  $Q_2$  תוכנית  $Q_1, Q_2$  ששמורה ריקה, השפה שלנו תהיה ריקה אם היא שווה לשפה  $Q_1, Q_2$  זו בעיה שנייה לפחות כי קבילה מהשלילה, סח' כ' קיבלנו כי הראודקציה מקבלת את  $E$  בסטריה.

טענה:  $EQ$  לא כריעה כי אם שפה היא כריעה, היא בהכרח קבילה. וכך גם הכוון ההפוך נכון - לא קבילה גורר לא כריעה.

### $\overline{EQ}$ השפה 10.6.3

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

טענה: לא קבילה.  
הוכחה: נב"ש  $\overline{EQ}$  קבילה, ותהי  $A - \overline{EQ}$  שמקבלת את  $\overline{EQ}$ .  
נבנה  $A - \overline{ATM}$  שמקבלת את  $\overline{ATM}$  (או סטירה, 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 הדולה יותר, שבහינתן תוכנית וקלט תמים נכון. כתוב:

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

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

מה הרעיון מאחרוי ההוכחה? להתחילה כבסיס  $cn/la$ , ואז להמיר נוניה התוכניות. הרעיון התהיליך הוא מקבלים זוג  $(P, w) \rightarrow (Q_1, Q_2)$  משמשים בהם לקבל את  $\overline{ATM}$  בסטריה. נרצה זוג ככה  $L(Q_1) \neq L(Q_2) \iff P(w) \neq 1$ .

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

$Q_1 = "Q_1(x) : return(U(" + P + ", " + w + "));$   
 $Q_2 = \{Q_2(x) : return(1);$   
 $return A - \overline{EQ}(Q_1, Q_2);$

כאשר  $Q_2$  תחזיר תמיד 1, ככל שפה שלה היא כל  $\Sigma^*$ . ולעומת זאת,  $Q_1$  תחזיר את תוצאת המוכנה האוניברסלית על הקלט  $U(P, w)$ . וכך

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

מדוע? אם  $P(w) = 1$ , זה אומר לכל קלט, כי הרצנו תוכנית. וכך השפה תהיה  $\Sigma^*$ . נשים לב כי  $L(Q_1) \neq L(Q_2) \iff (P, w) \in \overline{ATM}$ , מתקיים  $L(Q_2) \neq \emptyset$ , ולכן  $(P, w) \in \overline{ATM} \iff$  השפה שפונה שמהותה בשרה; הינה 1 אם  $P(w) \neq 1$ , ולכן  $P(w) \in \overline{ATM}$ , כלומר  $L(Q_1) \neq L(Q_2)$ . מכאן מקבלת את  $\overline{ATM}$ , בסתרה.

הערה חשובה: **כיצד מוגדרת**  $\text{באו}(x)Q_1$ ? **לכל קלט**  $x$ , **תמיד** יוחזר תוצאה המבונה הווירטואלית - **ולבן** השפה  $\text{asm}\text{ התוצאה} \text{ יצאה אחד, ככלומר } (P, w) \in ATM$ , **אזי כל מילה בשפת הא"ב**  $Q_1$  **שלאו** תתקבל **ולבן** השפה  $\Sigma^*$ , **מנגד** - **אם לא** יצאה אחד אזי אף מילה לא תתקבל.

**טענה:**  $\overline{EQ}$  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 עי' אף מכונת טיריניג (כלומר, אף מחשב).

**דוגמא.** נתבונן בפונקציה הבאה:  
 $\Sigma_1 = \Sigma$  הא"ב של האותיות שדרושות לכתיבה בשפה SIMPLE. יהי  $\Sigma_2 = \{0, 1\}$   
 $X_E : \Sigma_2^* \rightarrow \Sigma_1^*$  עי'

$$X_E(x) := \begin{cases} 1 & x \in E \\ 0 & x \notin E \end{cases}$$

פונקציה זו תקרא הפונקציה המczyinit של  $E$  או אותה  $E$  לעילן. פונקציה זו אינה ניתנת לחישוב עי' אף מכונת טיריניג. שחררי, אם קיימות מכונות טיריניג  $M$  שמחשבת אותה, ניתן לבנות ממנה מכונה  $M'$  שמכריעת את  $E$ , בסתרה כי  $E$  לא כרעה.

הערה. האם קיימות פונקציות מהטבעיים לטבעיות גם שלא ניתנות לחישוב? ודאי. הפונקציה לעיל היא של מהרוזות, ניתן לקודם לערך מספרי לפי טבלת אסקי למשל.  
**מינוח.** לפונקציה שנייתנת לחישוב עי' מכונת טיריניג נקראה פונקציה חשיבה.

## 10.8 רזוקציות והוכחות ברזוקציות

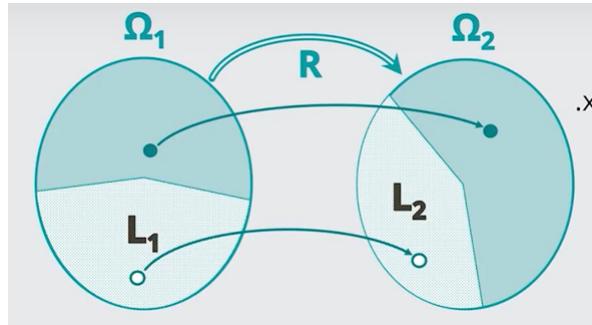
כל הדוגמאות שהשתמשנו להוכחה של אי כריעות/אי קבילות בדוגמאות הקודמות נקבעו רזוקציה.  
הרעין של רזוקציה הוא פתרון בעיה אחת, באמצעות פתרון מוכר של בעיה אחרת. כתבת נואר באופן פורמלי את מושג הרזוקציה:

הגדרה: רזוקציית התאמה היא - יהיו שני מרחבים,  $\Omega_1, \Omega_2$ .  
 $\Omega_1$  הוא מרחב הביעות אני מעוניין לפתור,  $\Omega_2$  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 שלכל  $x \in \Omega_1$  מתקיים  $x \in L_1 \iff R(x) \in L_2$ . ככלומר - המקור ב-  $L_1$  אם' המתמונה שלו ב-  $L_2$ .

נשים לב - הפונקציה שומרת על השיקות לקבוצות המותאמות. נשים לב כי רזוקציה מ-  $L_1$  אל  $L_2$  אינה פונקציה מ-  $L_1$  ל-  $L_2$ . אלא מן המרחב ש-  $L_1$  נמצא בו מרחב ש-  $L_2$  נמצא בו.



**סימן:** אם יש רדוקציית התאמה ניתנת לחישוב  $L_2 \leq_m L_1 \rightarrow L_1 \text{ אי נסמן}$

**טענה:** תהיינה  $L_1, L_2$  שפות. אם:

כריעה  $L_2^*$

$L_1 \leq_m L_2^*$

איי כריעה.

**הוכחה:** תהי  $D = L_2 - L_1$  התוכנית שמכריעת את  $L_2$ . בהינתן  $x \in \Omega_1$  נרצה לדעת האם  $x \in L_2$ .  
1. נחשב  $y = R(x) \in L_2$  כיוון שהוא רדוקציה,  $y = R(x) \in D$ .  
2. נוכיח שזאת מושגתו שנקשר תהיה אותה התשובה על האם  $x$ . **נדרש.**

**טענה:** תהיינה  $L_1, L_2$  שפות. אם:

קבילה  $L_2^*$

$L_1 \leq_m L_2^*$

איי קבילה.

**מסקנות:** אנו משתמשים ברדוקציה בדרך השיליה וכך גם יהיה במחזור. נתרגם את הטענות לעיל:  
תהיינה  $L_1, L_2$  שפות. אם:  $L_1$  לא כריעה וכן  $L_1 \leq_m L_2$  איי  $L_2$  לא כריעה.  
תהיינה  $L_1, L_2$  שפות. אם:  $L_1$  לא קבילה וכן  $L_1 \leq_m L_2$  איי  $L_2$  לא קבילה.

**כיצד נוכח שפה לא כריעה/קבילה? נמצאה שפה  $L_1$  לא כריעה/קבילה, ונבנה רדוקציית התאמה ניתנת לחישוב  $L_1 \leq_m L_2$ .**

### 10.8.1 רדוקצייה משפה בריעה

**טענה:** תהי  $A$  שפה בריעה. אז, קיימת רדוקציית התאמה ניתנת לחישוב  $A \leq_m ATM$

**הוכחה:** תהי  $A$  כריעה. אזי קיימת  $D^A$  המכריעת אותה. נבנה רדוקציה  $R$  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 הרדוקציה מקבלת מילה, ומוריצה את המcona על המילה. אם  $D^A$  אלי  $w \in A$  קבילה, אז  $R(w)$  מקבלת תוקף. אחרת,  $w \in A$  לא קבילה, ולכן  $D^A(w) == 0$ . הרדוקציה מקבלת כל מילה ולכן בשביל שהרדוקציה תתקיים נרצה להחזיר איבר כשלשו ב- $ATM$ , החזרנו תוכנה שמקבלת כל מילה ואיבר - שבירט בתוכנה כי היא מקבלת כל מילה. אחרת,  $D^A$  לא קבילה וכן נרצה להחזיר זוג של א' מותאים ל- $ATM$  - מכונה שלא מקבלת דבר, ומילה, שודאי לא שייכת אליה כי היא לא מקבלת דבר. נשים לב כי הרדוקציה חישובית ומודדרת היטב - והיא מכרעה, היא תמיד תעצור. לכן הרדוקציה  $R$

תמיד תחזיר קלט. כמו כן, ניתן לראות כי  $x \in A$ ,  $R(x) \in ATM \iff x \in ATM$  כיון שאחרת לא נקבל זוג שמתאים ל- $ATM$ . סה"כ בנו רדוקציה שתמיד עובדת וקיים.

**מסקנה:** באופן דומה, ניתן לבנות רדוקציה התאמה ניתנת לחישוב מכל שפה כריעה לכל שפה לא כריעה (פרט לשפה הריקה ול- $\Sigma^*$ ). וכך, מכל שפה לא קבילה לכל שפה לא קבילה.

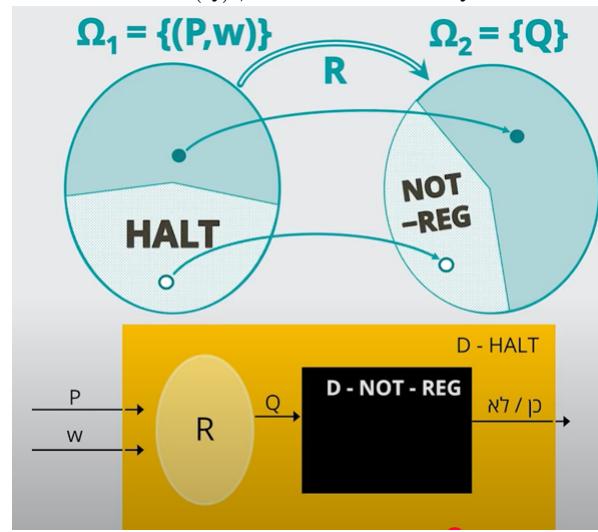
נשים לב כי משפה כריעה, לא ניתן למצוא רדוקציה ניתנת לחישוב אל השפה הריקה. שכן הרדוקציה צריכה להחזיר לכל פלט  $\in A$ , פלט בשפה שנשחנו אליו. בשפה הריקה אין פלטים כלל, ולכן לא קיימת רדוקציה כזו. באופן דומה - לא קיימת רדוקציה משפה כריעה אל  $\Sigma^*$ .

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

### NOT – REG השפה 10.8.2

נגדיר  $\{Q\}$  לא רגולרית |  $Q \in NOT - REG$ .  
למשל,  $.palindrome \in NOT - REG$  שפת הפלינדרום, כיון שלא רגולרית. עם זאת,  
 $startA \notin NOT - REG$  כי שפת המחרוזות שפותחות בה כן רגולרית.

טענה:  $NOT - REG$  לא כריעה.  
נראה כי  $HALT \not\leq_m NOT - REG$   
נשים לב כי  $HALT \in \Omega_1 = \{Q\}$  וכן  $HALT \in \Omega_2 = \{P, w\}$  וכ"ז  $Q = R(P, w)$  רצתה כי הפונקציה תחזיר  $(P, w)$  כצ'ש כלומר, בחינתן זוג  $(P, w)$  נקבעה על ידי  $R$  לא רגולרית אם  $w \downarrow P(w)$  עצרת. ובתייאור -



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 הירטואלית, ואז מרים את התוכנית פליינדרום על הקלט איקס ומחרוזה תשובה מותקבלת. זו רדוקציה שנייה לחישוב, אך מקבלים זוג מחרוזות, ומחרוזים מחרוזת בודדת. סה"כ קיבלו מה שרצינו -  $L(Q)$  לא רגולרית אם  $w \neq P(w)$ . נשים לב כי

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

השפה של  $Q$  תליה במקרה יקרה בשורה  $U(P, w)$ . אם החישוב מסתיים, נגע לשורה האחורונה ונחזיר את שפת הפליינדרומים. אחרת, לעומת געוע וגעוע לשורת ההחזרה והשפה תהיה ריקה. סה"כ קיבלו מה שרצינו -  $L(Q)$  לא רגולרית אם  $w \neq P(w)$ . מודיעו בחורנו בשפת הפליינדרומים כי היא לא רגולרית, יכולנו לבחור כל אחת אחרת שאינה רגולרית וניתנת לחישוב - וזה היה עוזב.

**טענה:**  $\overline{NOT - REG}$  לא קבילה.  
הוכחה: נראה כי  $\overline{ATM} \subset_m NOT - REG$ .

נשים לב כי  $\overline{ATM} \in \Omega_1 = \{(P, w)\}$  וכן  $NOT - REG \in \Omega_2 = \{Q\}$ . נרצה כי  $\overline{ATM} \in NOT - REG$ .  
כלומר, בהינתן זוג  $(P, w)$  נרצה כי הפונקציה  $R$  תחזיר  $Q$  כך ש  $Q \in NOT - REG$ .  
סה"כ  $L(Q) \iff Q \in NOT - REG$ .  
נשים לב כי אכן  $\overline{ATM} \subseteq NOT - REG$ . ולכן אם נוכחות קיימות כזו לפי טענה מההרצאה - אכן  $NOT - REG$  לא קבילה גם היא.  
אנחנו מחפשים ליצור תוכנית  $Q$ , שהשפה שלה אינה רגולרית, אם  $w \neq P(w)$ . 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 המחרוזות פליינדרום, אם כן היא מחרוזה אחד. אחרת - לא פליינדרום:  
מרים מכונה וירטואלית עם התוכנית והקלט. נשים לב כי

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

כיון שגם  $P(w) = 1$ , אז יוחזר 1 עבור פליינדרומים ו-1 עבור שאר הקלטים - סה"כ כל  $\Sigma^*$ .  
אם  $P(w) \neq 1$ , אז השפה שתתקבל היא שפת הפליינדרומים בלבד. סה"כ  $L(Q)$  לא רגולרית (ושווה לשפת הפליינדרומים אם  $w \neq P(w)$ ) כי  $\Sigma^*$  כן רגולרית, ולכן הרדוקציה מתאימה ונכונה.

**עוד דוגמאות לרדוקציות - הייתה עצמן בשבייל בכתב CAN. אבל בשיהיה כוח  
להעתיק CAN עוד דוגמאות מהקמפייס!!!!!!**

!!!!!!

!!!!!!

!!!!!!  
כולל תרגולים והוכחות ברדוקציה. יש שם הרבה בקמפוס לתרגל כולל תרגול עצמאי

!!!!!!

## 10.9 סיכום - השפות הלא כריעות/קבילות

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

## 10.10 בעיית הנחש

הຮושם שנוצר עד כה – הוא שכל הבעיות הלא כריעות קשור לתוכניות. ובכן, זה לא נכון. דוגמה לכך היא בעיית הנחש. נתונם לנו ריבועים כנ"ל:



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



השאלה היא, האם ניתן לשבץ שני אריחים בשתי המשבצות, כך שיתאימו זו לאו. כשהחכוונה ביתהaimo – אפשר לחבר שני אריחים זה לזה רק אם הצבעים בפאות הנוגעתו, זהים. למשל, בדוגמה מעלה, אפשר לחבר את הימני והאמצעי, כי הפאות הצדדיות שלן הקרובותצבע צהוב. כמו כן, אפשר גם בין המשבצות הללו ליצור צירוף של יותר משתי אריחים – כמה שרצים, כל עוד הכלל נשמר ונתנו מתחילה במשבצת הראשונה ומס'ים בשנייה. נשים לב כי אסור לסובב את האריחים ומותר להשתמש כמו שרוצים באותו סוג אריה. כמו כן – גם מסלולי נחש כאלו חוקיים:



יש מסלולים - ללא אפשרות ליצור מסלול.

**פורמלית, הבעיה תואר כך:**

\* קבוצת ארכיחס  $T = (t_1, \dots, t_n)$

כאשר כל ארכיח  $t_i \in \mathbb{N}^4$  (כל ארכיח מיוצג ע"י 4 מס' טבעיים שמייצגים את הצלבים)

\* זוג נקודות כזה  $\mathbb{N} \times \mathbb{Z}$

הבעיה היא: { יש נחש תקין של ארכיח  $T$  בחצי המישור העליון שמחבר בין  $p$  ל $q$  } |  $(T, p, q)$

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

נשים לב כי אם נגדיר { יש נחש תקין של ארכיח  $T$  בכל המישור שמחבר בין  $p$  ל $q$  } |  $(T, p, q)$  או בעיה כריעה.

## 10.11 התוכנית של הילברט

זכור בתוכנית הילברט מהיחידה הקודמת: לבנות יסודות פורמליים למתמטיקה. ראיינו כי היעד השלישי-כrüוות: למצוא אלגוריטם שקובע האם טענה היא נכון או שלא, חיכה ליחידה הנוכחית. ובכן, טירינגן לא מצא את האלגוריתם שהילברט חיפש - טירינגן הוכיח שאין אלגוריתם כזה, ולא יכול להיות. ניתן להסביר את הדברים מתוך מה שראינו ביחידה זו: ראיינו כי למשל *ATM* לא כריעה - ולכן אין אלגוריתם מותמטי או תכני, שמודע להכריע את הבעיה. (():