

סיכום הרצאות לבחן - מודלים חישוביים

13 בנובמבר 2025

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

תוכן עניינים

3	יחידה 2 - שפות	1
4	הא"ב	1.1
4	מחירות	1.2
4	פעולות על מחירות	1.3
4	שפות	1.4
5	פעולות על שפות:	1.4.1
6	הכליה ושוון בין שפות	1.4.2
7	פעולות	1.5
7	רישא, סיפא ותתי מילימ	1.6
8	יצוג בעיית הכרעה בשפה	1.7
8	יחידה 3 - אס"ד אוטומט סופי דטרמיניסטי	2
10	רכיבי האוטומט	2.1
10	הגדרה פורמלית של אוטומט סופי דטרמיניסטי	2.2
11	שפת האוטומט	2.3
11	בנייה אוטומטים	2.4
15	בנייה אבסטרקטית	2.5
15	אוטומט משלים	2.5.1
15	שפה רגולרית	2.5.2
15	אוטומט המכפלת	2.5.3
16	אוטומט מכפלת לאיחוד והפרש	2.5.4
17	אוטומט אתחול	2.5.5
18	שפת היזנג	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	
43	סגורות לסגור קלין	5.7.3	
44	אי סגורות לחיתוך	5.7.4	
44	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: מוכנות טיריניג	7	
52	מהי מוכנות 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 מוכנות טיריניג באמצעות טבלת מעברים	7.3	
58	תיאור מוכנות טיריניג באמצעות פסודו קוד	7.3.1	
59	שימוש במוכנות טיריניג לחישוב פונקציות	7.4	
60	יחידה 9: וריאציות של מוכנות טיריניג	8	
60	מודל חישובי - הגדרה פורמלית	8.1	
60	סרט ימינה בלבד	8.2	
62	מודל <i>TS</i>	8.3	
63	מודל <i>OR</i>	8.4	
63	מוכנות טיריניג מרובת סרטים	8.5	
65	סגורות לאחד וחיתוך שפות כrüיות/קבילות	8.6	

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

1. יחידה 2 - שפות

ישן כמה סוגים 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 קשייר. בעיות אלו הן בעיות שהתשובה אליהן היא כן או לא. זו בעיות חישוב ספציפיות יותר, כיון שיש לה שתי תשובות בלבד. בקורס נתמקד בעיות אלו. מדוע? הן פשوطות להגדירה ונויותו, השאלה שתענין אותנו בהמשך - האם יש בעיות שלא ניתן לפתורן באמצעות מחשב. סיבה נוספת להתusalem, היא שככל בעית חישוב ניתן להמיר לבאית הכרעה.

כיצד נמיר בעית 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 בעית החישוב.

1.1 הא"ב

תמיד הקלט לבעיות הכרעה ייצג ע"י סדרה של תווים. התווים שמהם נוצרת סדרה קרו[א האלפבית](#). או פשוט א"ב. נסמן באות Σ . בקורס זה הא"ב יהיה תמיד סופי. את אותיות הקלט נסמן לעיתים באותיות יווניות - σ, τ.

למשל - עבור הבעיה האם x ראשוני?", הא"ב יהיה $\{0, 1, 2, \dots, 9\} = \Sigma$. עבור הבעיה, האם הגף קשור?" נזדקק לא"ב $\{1, \dots, 9, (,), \{, \}, ", "\} = \Sigma$ 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 "סססססססססס" תקרא מחרוזת/[מילה](#)/סדרה. מילים יסומנו באותיות קטנות כגון y, u, v, x , וכו'. w | **אורץ של מילה**. מס' התווים ב-w. אורץ המילה יהיה סופי, ובקורס הזה כל המילים יהיו סופיות.

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

סימונו: עבור מילה w ואות s נסמן s^w את מס' הפעמים שהאות s מופיעה w.

קבוצת כל המחרוזות מעלה Σ - תסומן Σ^* . למשל, $ab \in \{a, b\}^*$ בעוד $ac \notin \Sigma^*$ וכן $\epsilon \in \Sigma^*$. שכנן הקבוצה משמאלה אינה ריקה, יש שם את קבוצת כל המחרוזות הלא ריקות מעלה Σ תסומן ותוגדר:

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

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

1.3 פעולות על מחרוזות

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

b. **חזקת - 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, $ababab = (ab)^3$. מתקיים כי $\epsilon^0 = \epsilon$ 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 Σ^* .
- ב. שרשור - יהיו שפות L_1, L_2 . נגיד שרשור עליה:

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

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

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

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

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

לדוגמה:

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

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

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

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

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

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

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

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

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

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

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

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

1.4.2 הכללה ושוויון בין שפות

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

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

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

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

מהגדרת האיחוד,

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

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

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

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

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

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

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

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

1.5 פועלות reverse

נדיר פורמלית את פועלות *reverse* על מילה w :

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

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

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

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

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

1.6 רישא, סיפא ותתי מילים

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

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

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

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

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

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

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

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

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

1.7 ייצוג בעיית הכרעה בשפה

נחוור להתחלה. נראה כעת, כיצד נוכל להעזר בהגדרות שאספנו על מנת ליעזג בעיית הכרעה. נסתכל על בעיית ההכרעה: האם מס' w הוא ראשוני? 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 טבוי n מוצג ע"י רצף של n אחדות.**

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

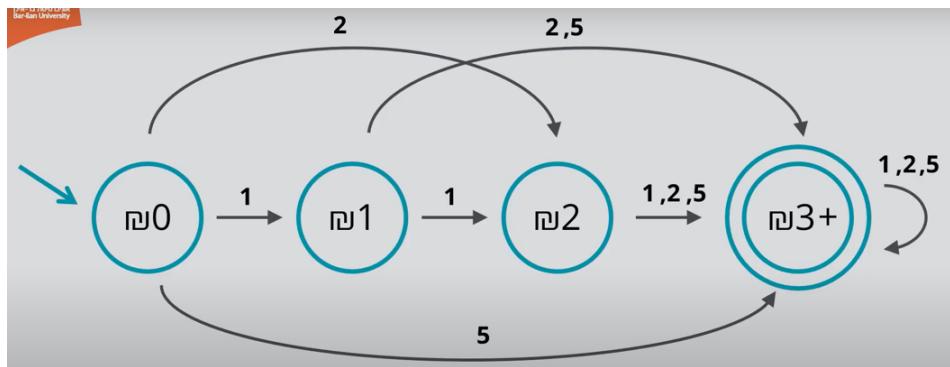
ביחידה זו נדון במושג המחשב. קודם לכן, דנו במושג בעיה=שפה. כעת נדבר על מודול חישובי כלשהו, לא מחשב ספציפי. היכולות החישוביות של האוטומט הסופי, די מוגבלות. אז מדוע נלמדו? יש לו שימושים מרכזיים בעולם האמיתי, והוא ישמש אותנו בחימום והכוה למודלים המורכבים בהמשך.

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 למונית, 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 - שנדע 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:

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

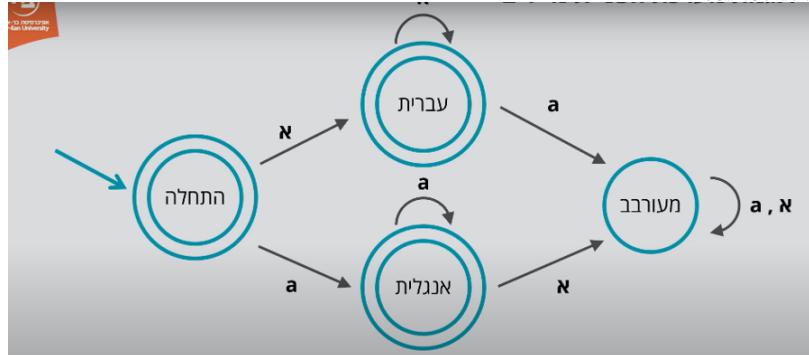
$$L = \{\sigma_1\sigma_2\dots\sigma_n \mid \sum_{i=1}^n \sigma_i \geq 3\}$$



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 מביחינו 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 הסטיים במצב מקבל.

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

2.2 הגדרה פורמלית של אוטומט סופי דטרמיניסטי

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

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

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

$q_0 = \text{ מצב תחيلي}$
 $F = \text{ קבוצת המצבים המקבלים } \{q_i, q_j, \dots, q_k\}.$
 בעת, בהינתן אוטומט שיטואר בצורה מתמטית, נוכל לעבור לאוטומט בצורת ציור. שניהם כמפורט אוטומטיים. אחד פורמלי יותר ואחד פחוס.

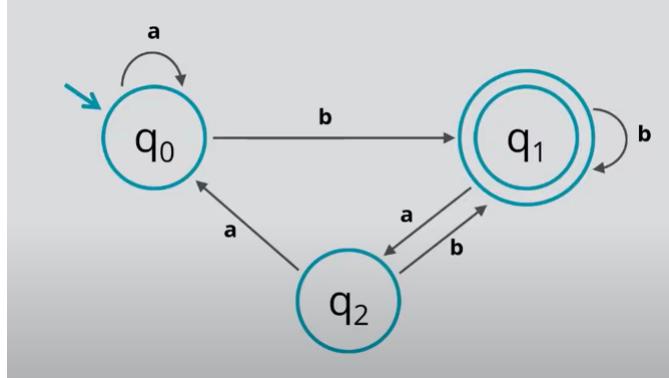
מה זה דטרמיניסטי? בכל שלב, ובכל מצב, ישנה אפשרות אחת בלבד עבור אות קלט لأن להתקדם. לא צריכה להיות בחרה של המשמש או של המחשב עצמו באיזה כיוון כדי לו להתקדם.

2.3 שפת האוטומט

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

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

כלומר עת נשתכל על Σ^* , פונקציה שתקבע מחרוזת, לאתו בודד. כמובן, בהינתן למשל האוטומט 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 לראותות כאן שאכן נגע ל q_2 . כמובן שנסכים לב שערכי δ נקבעים באופן מלא בהתאם לערכי δ .

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

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

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

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

2.4 בניית אוטומטיים

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

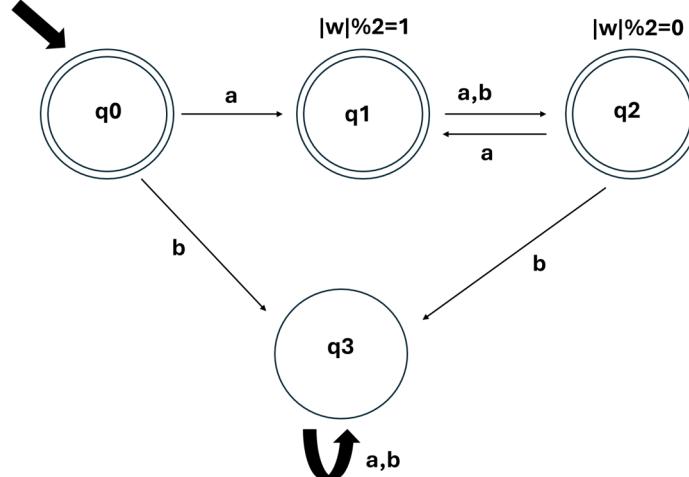
- א. חשוב להבין את השפה, לתרגם את המתמטיקה להבנה. נודא שברורו איזה מילים בשפה ואיזה לא.
- ב. בונה וצייר את האוטומט עצמו. לצורך זאת - זה סוג של מחשב, ולכן יתכונו כמה פתרונות לאותו התרגיל.
- ג. נבצע בדיקות הריצה של מילים בשפה ומילים שאינם בשפה. נודא שגם האוטומט עובד.

אוטומט שלד: נניח ונרצה לבנות אוטומט לשפה מעל $\{a, b\} = \Sigma = \{w | w = aa\sigma_1\sigma_2\dots\dots\}$. כלומר, כל המילים שנפתחות ברצף aa . מהי המילה שבودאי תופיע שם? aa . אנחנו רוצה לבנות אוטומט שלד, שיביא את המילה aa , והוא 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 הטריוואליות:**
- השפה הריקה - אוטומט לשפה שלא מכילה אף מילה, כלומר \emptyset . בצד יראה אוטומט זה? מצב יחיד, q_0 , שאינו מקבל, עם עצמי $loop$ על a .
 - המילה הריקה - אוטומט עבור השפה $\{\}$. בצד יראה אוטומט זה? נזכיר q_0 מקבל, אשר קיבל a, b כאוטיות קלט, נ עבר למצב מלבדת. ושם יהיה עצמי עבור אוטיות הקלט. לעומת זאת, ברגע שנקלט a, b , תס היפור - הלהנו למצב 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:

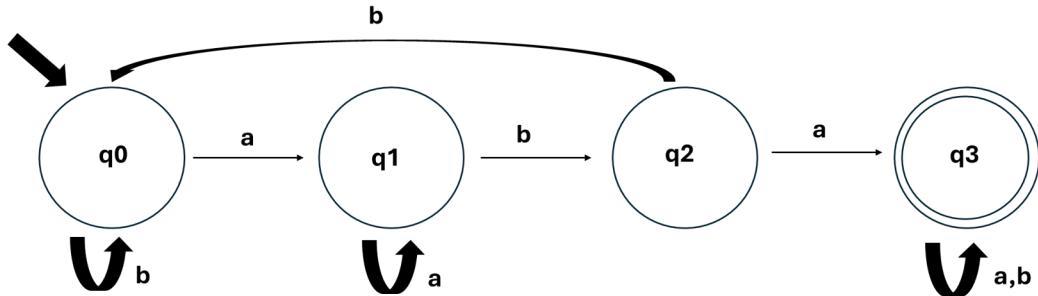
יהי $\Sigma = \{a, b\} = \Sigma$ שלנו. נגיד את L להיות השפה של כל המילים מעל Σ בהן האות b אינה במקומות הזוגי. נראה כי האוטומט שלנו יהיה:



עלים רמה, ובכן נשים לב שנרצה להגדיר מצבים שונים עבור אורך מילה זוגי ואי זוגי, וכן נרצה להפריד את האות הראשונה. אם החילנו באות b , איי אנחנו כבר רוצים להקלע למצב 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 -

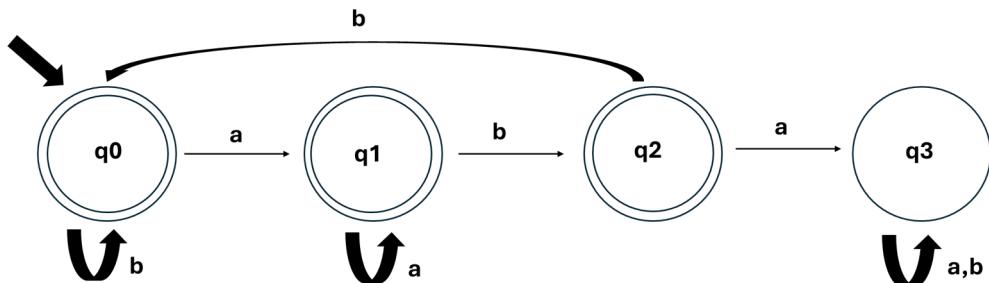
$$L = \{w = u_1aba u_2 : u_1, u_2 \in \Sigma^*\}$$



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:

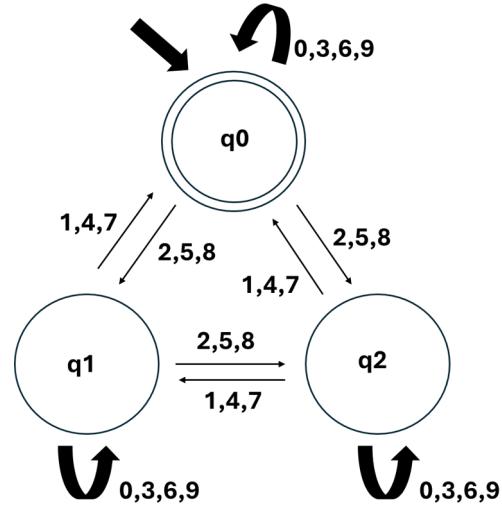


בשיטתה זו נוכל להשתמש **תמיד** כאשר נרצה לבנות אוטומט לשפה המשילימה, בהינתן שאנחנו ידעים את האוטומט לשפה המקורית. וכך, כמו בהסתברות למשל, נוכל לבנות אוטומט לשפה וואז להפעיל עליה שלילה", רק שכן במקום בצע $p - 1$ נשנה את המצבים המתקבלים.

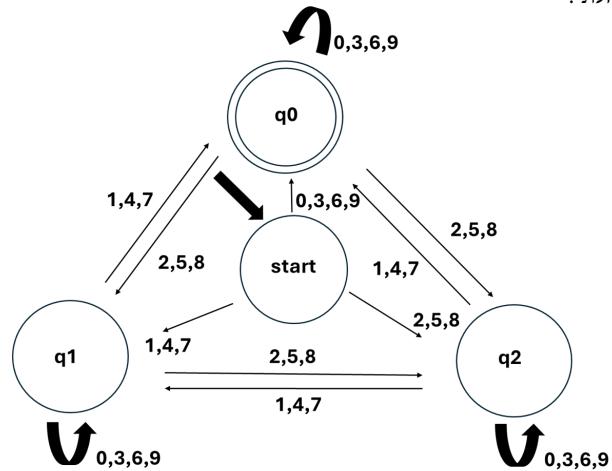
דוגמה 4. נרצה לבנות אוטומט A שיקבל את השפה הבאה מעל אל"ב $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

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

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 השארית של סכום ספרות המספר עד כה הוא בדיק. i . مكان נקבע אוטומט הבא -

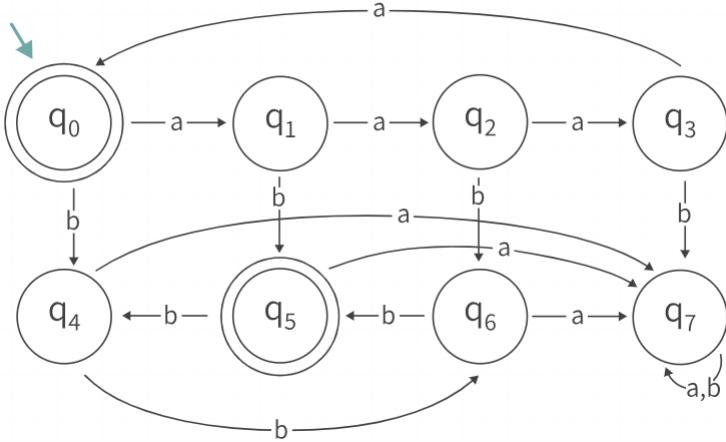


נשים לב, האם זה אוטומט מספק עבורנו? כמובן. מה עם המילה הריקה? מדוע היא במצב מקבל? הרי מילה ריקה איננה מתחילה בשלוש. ולכן, נצטרך להוסיף מצב נוסף להוות מצב התחלה:



עכשו האוטומט שלנו, מוכן לקבל את הקלט.

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



מדובר בדוגמה המורכבת ביותר עד כה ולכן נתעכט. בראשית, נרצה שהמצבים q_0, q_1, q_2, q_3 יישמרו את מס' מופעי a . כמובן כל אחד מהם שומר את תוצאת המודולו של החלוקה ב-4. לאחר רצף מופעי a , האוטומט הৎכם זוכר למשעה את ההפרש בין שארית החלוקה ב-4 של מופעי a לבין שארית החלוקה ב-3 ש策ריכה להיות. כך q_4 הוא $z - 1$, q_5 הוא $z + 1$, q_6 הוא $z + 3$ בלבד הוא מצב מקביל - שכן ההפרש זהה, וב q_6 ההפרש הוא 1, כמובן של b גודלה יותר באחד. לכן q_5 בלבד הוא מצב מקביל - שכן ההפרש יצא בבדיקה אפס. כמו כן נראה כי q_7 יהיה מצב מלבד, אם במקרה ירצו לשולח אליו עוד a לאחר רצף של b או אם שארית החלוקה של המילה $b^m a$ היא שלוש, לא יוכל שהミילה בשפה, שכן נרצה שייהיה שוויון בין השאריות וכן שארית 3 חלוקה לשולש היא אפס. לכן גם מצב זה ישלח למלאות.

2.5 בניית אבסטרקטית

2.5.1 אוטומט משלים

טענה: هي A אוטומט, אז קיימים אוטומט B כך ש $\overline{L(A)} = \overline{L(B)}$

את האוטומט המשלים, מקבלים כתוצאה מהפיכת המצביעים המקוריים של A , ללא מקבלים, ואת הלא מקבלים, למקבלים.

לשיטת זו קוראים בנית אבסטרקטית של אוטומטים, אנו משתמשים בנית אוטומט אחד לשימוש עבור אחד אחר.

2.5.2 שפה רגולרית

הגדרה: תהיו L שפה. נאמר כי L רגולרית, אם קיים אוטומט סופי A שמקבל את השפה כך $L = L(A)$.

טענה: אם L רגולרית, אז גם \overline{L} רגולרית.

2.5.3 אוטומט המכפלת

טענה: תהיינה L_1, L_2 שפות רגולריות. אז $L_1 \cap L_2$ רגולרית.

הוכחה: $L_1 \cap L_2$ רגולרית ולכן קיימים אוטומטים A, B כך ש $L_1(A) = L_2(B) = L_1 \cap L_2$ וכן $C = L_1 \cap L_2$.

בננה אוטומט סופי C , כך ש $C = L_1 \cap L_2$.

נשים לב - באוטומט סופי אין לנו שמי' של מושג מה תוצאת הקלט, פרט לרגע בו אנחנו מקבלים אותו. ולכן בשבייל לדעת האט מילה נמצאת בשני אוטומטים (ובפרט בחיתוך), יש להרים במקביל, ואז בסוף להשוות האם סיימנו בשני האוטומטים במצב מקביל.

נסמן את האוטומטיים:

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

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

כעת נרצה להגדיר את C פורמלית:

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

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

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

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

2.5.4 אוטומט מכפלה לאיחוד והפרש

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

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

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

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

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

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

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

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

כלומר מכפלה קרוטית של האיחוד של ההפרשים.

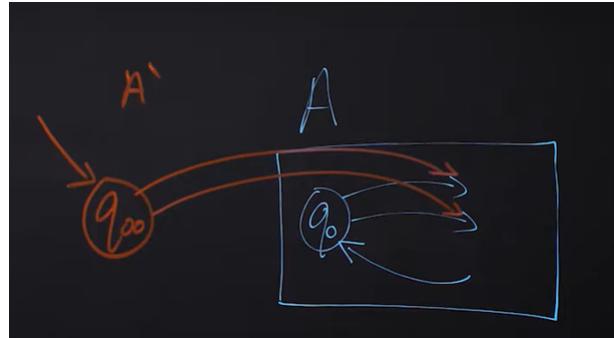
מסקנה: בהינתן שתי שפות רגולריות, Σ כל המניפולציות הבאות על השפות קיימים אוטומט שמקבל את המניפולציה: איחוד, חיתוך, משלים, הפרש סימטרי.

2.5.5 אוטומט אתחול

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

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

2.5.6 שפת הזוג

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

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

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

תרגיל. יהיו L_1, L_2 רגולריות מעל אותו א"ב Σ . הוכיח כי $ZigZag(L_1, L_2)$ רגולרית מעל Σ .
הוכחה: L_1 רגולרית לכן קיים אוטומט סופי דטרמיניסטי שמקבל אותה - נסמןו A , בדומה עבור L_2 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 .
 שיבahir ל- i - אתה חולץ A .
באשר למכפלה התחלו -

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

כיוון שכל מצב הוא שלישיה, ונרצה להיות כרגע בתחום A ולכן הסימון של A כיוון שכל מילה בשפה פותחת באות A .
פונקציית המכפלה: היא אמורה לקבל מצב ואות ולהחזיר מצב. כפי שאמרנו מצב אצלונו הוא שלישיה סדרה, לכן הפונקציה מקבל שלישיה ואות ותחזיר שלישיה.

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

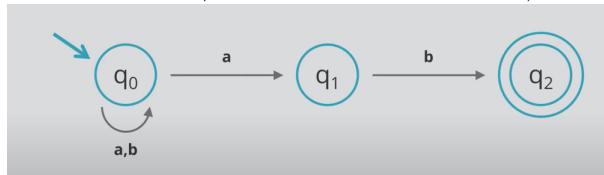
הטבר: כאשר אנחנו נמצאים בתוך A , נרצה לעcor בקורסיה הקלט הראה אל A וכן נפעיל את דלתא של A , נשאר בפ' כי לא צו B וכן ה $sign$ ישתנה ל B כי אין כרגע ב A וכקורסיה הקלט הראה נרצה לעcor אל B . **בזומה, עכו המקרה השני.**
cut נגע אל קבוצת המצביעים המקבילים:

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

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

3 ייחידה 4: אוטומט סופי לא דטרמיניסטי

איך שלא, נתחילה בדוגמה. נסתכל על הדוגמה מטה, על פניו - נראה ליטימי. מה ההבדל? נשים לב, מה מצב q_0 ישנו שני>Statusים שונים של a . או להשאר במקום, המותbeta ע"י ללא 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 - בפ' אין טיפול באות הקלט a . כפי שהיא במודל הדטרמיניסטי. כמו כן, ישנו מילים שלולות לא יסתימו במצב מקובל: למשל, $aa = aa$ לא הגיעו למצב מקובל.



באוטומט הסופי הלא דטרמיניסטי, תתכן הרצה זהה שתוביל לתוצאות קלט שונות. **האוטומט הלא דטרמיניסטי לא קובע תוצאהճ-בקרה חד-** **ערבית.**
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 ישאר ב q_2 אלא החישוב יתקע למחרוזת w (אפשר להסתכל על זה כמו קriseה של התוכנית".).

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

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

ההבדל המרכזי הוא בפונקציית המעברים. 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:

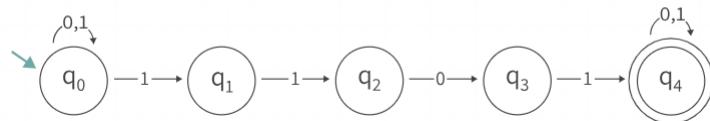
$$L(N) = \{w \mid \exists f \in F : f \in \Delta^*(q_0, w)\} = \{w \mid \Delta^*(q_0, w) \cap F \neq \emptyset\}$$

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".

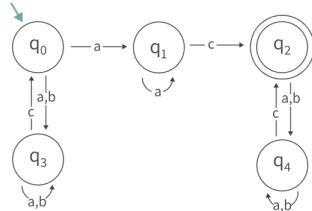


דוגמא 2. בניית אוטומט לא דטרמיניסטי שיקבל את השפה הבאה:

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

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 חיללה. 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 טיפלנו בהז במאובט q_4 . סה"כ היה נראה מפהיד - אבל המודל הלא דטרמיניסטי עזר לנו לגשת לזה בצורה טובה.



3.4 שיקולות

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

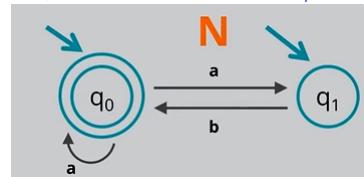
כל שפה שניתן לקבל באמצעות אוטומט לא דטרמיניסטי, ניתן לקבל באמצעות אוטומט דטרמיניסטי.
נראה להוכיח טענה זו. לפני כן, נזכיר את המושג של אוטומט לא דטרמיניסטי.
נדיר כעט אוטומט לא דטרמיניסטי $N = \{Q, \Sigma, \Delta, Q_0, F\}$ כאשר Q_0 היא קבוצת המ מצבים ההתחלתיים. ככלומר - אפשר יותר מ מצב ההתחלתי אחד. לשם כך, המודל יבחר בצורה לא דטרמיניסטיבית בהתחלה, 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 .

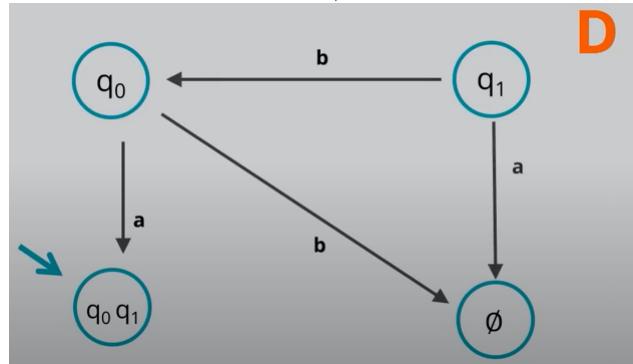
ראשית, נתחל בדוגמה שתעזר לנו במבנה ההוכחה.



מה ההבדל בין אוטומט לא דטרמיניסטי לבין דטרמיניסטי? בהינתן קלט כלשהו, ללא דטרמיניסטי ניתן להגעה לקבוצת מצבים. אנחנו נרצה להתחיל בקבוצות הבנויות אוטומט שմצביעו בסידוק כל הקבוצות האפשרות. למשל, בהינתן קבוצת מצבים שווה להגעה אליה $\{q_0, q_1\}$ בקריאה קלט ניתן להגעה לקבוצות הבאות $\{q_0, q_1, \emptyset\}$. כעט יוצר אוטומט ששמתו ויהו בדוק - שמות הקבוצות הללו. נראה כי לאוטומט בדוגמה שלו, ישנו 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 המילים שככל מקרה - האוטומט המקורי היה נתבע בחישוב בדוק 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 המ מצב q_0q_1 . מה באשר לקרויה של b מ- q_0 ? נראה כי אין אפשרות להמשיך - וכן b יועכר באוטומט הדטרמיניסטי, אל הקבוצה הריקה.



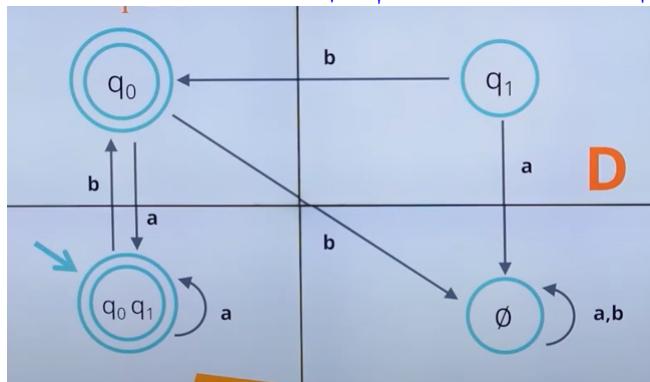
ובכן, זה האוטומט שלנו. נשים לב כי הוא עדין איננו דטרמיניסטי. עדיין, לא כל מילך מטופל בכל אפשרויות הקלט.

נניח ואנונו במאובן 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 באובי הקלט b שתצא ממאובן q_0q_1 . נשים לב שהגענו למאובן q_0 או למאובן q_1 . מהמאובן q_0 או זו דڑ להמשיך עם b , ואילו מהמאובן q_1 ניתן להמשיך אל q_0 בפזול הלא דטרמיניסטי, וכן סה"כ אם אנחנו הגיעו ל- q_0q_1 נוכל להגיע רק למאובן q_0 וכן נבער מעבר של אותן הקלטים b מ- $q_0 \rightarrow q_0q_1 \rightarrow q_0$. עבור a מ- q_0q_1 אפשר להמשיך אל q_0 או q_1 ולא ניתן לעשות כלום - וכן את הקלט a תשאor בלוואה עפ"י במאובן q_0q_1 .

נעיר, כי תheid כשנגיון למאובן של הקבוצה הריקה - נבער לוואה עפ"י בסיסי במאובן q_0q_1 . כיוון שבאוטומט המקיים נתקענו מפילה שאנו לשכז זה, וכן ייתן 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, נראה כי bN ורק q_0 הוא מצב מקבל - וכן גם אצלנו הוא והוא מצב מקבל. כמו כן, אם הגיעו למאובן q_0q_1 משמע שכאותומט המקורי היה דורך למליה להתקבל שהסתימה ב- q_0 וגם ב- q_1 בפרט ב- q_0 , וכן זה גם מעצם.

כך נראה האוטומט הדטרמיניסטי, שמקבל את השפה:



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

- קבוצת המ מצבים -

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

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

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

כלומר, פונקציית המעברים תוגדר להיות: בהינתן קבוצה R (זה הרו' שם המיצב) ואות σ , נרצה לעבור לקבוצה שהיא איחוד כל המיצבים, כך שבהינתן $q \in R$, הפונקציה שולחת אליו. ובמילים פשוטות יותר - נאخد את כל הדריכים האפשריות למעבר הקלט שששייך לקבוצה, זו קבוצה שהיא וודאי חלק מ(Q^N) P וכעת היא שם של מצב ב D לפי σ , וזה המצב אבל אחרי שטי קריואות אפשר להבini.
חשוב להתעכ卜 על חלק זה בהוכחה כי זה קצת מסורבל אבל אחרי שטי קריואות 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

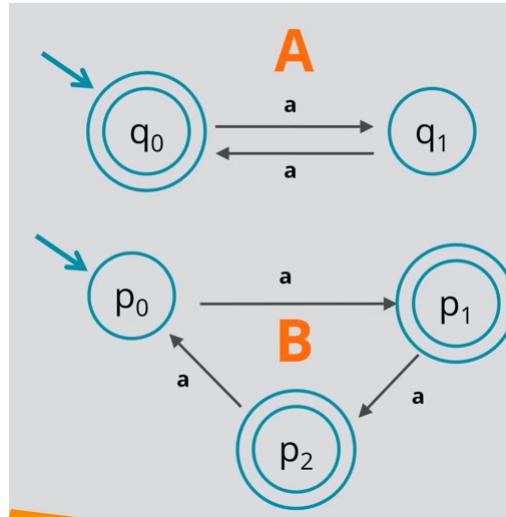
$$F^D = \{R \in Q^D : R \cap F^N \neq \emptyset\}$$

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 מזווע אכו האוטומט D מתקבל את אותה השפה, אלא רק הגדרנו 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

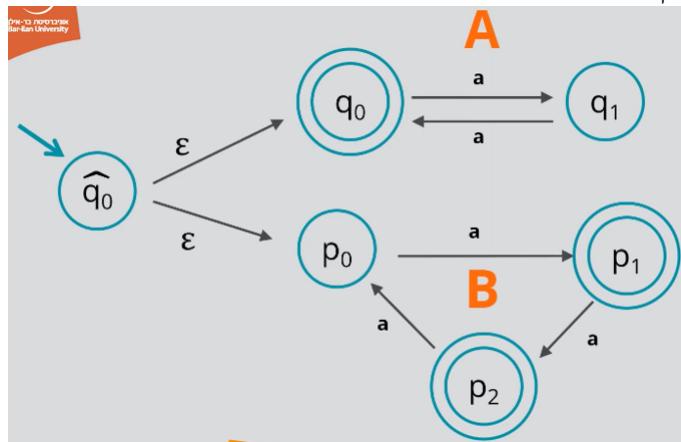
נרחיב את המושג של האוטומט הסופי. ובפרט, את המודל הלא דטרמיניסטי (שלפי טענה לעיל שקיים לאוטומט דטרמיניסטי).
הרעיון הבסיסי הוא לאפשר מעבר באוטומט ללא קריית אף אותן קלט, כשנרצה לעשות זאת, נסמן זאת באמצעות האות ϵ . על קשותות אלו ניתן לעبور ללא קריית קלט באוטומט. נדוגש כי לא **חייבים** לעبور על מעברי האפסילון,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

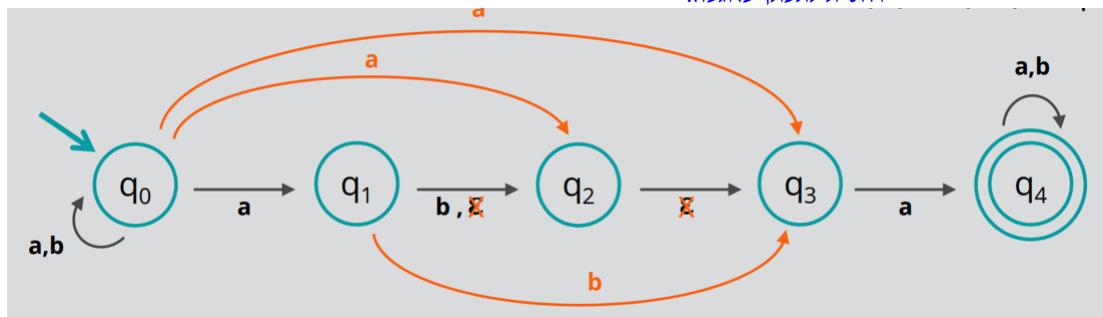


icut האוטומט מקבל את השפה C , ובחירה באופן לא דטרמיניסטי להיכן לכלכת. ביטלו את המ מצבים p_0, q_0 ו q_0 מכובדים התחלתיים והוספנו \hat{q}_0 כמצב התחלתי חדש, עם שני מעברי אפסילון.icut, אם נרצה גם אוטומט דטרמיניסטי - נוכל להפוך אותו לכך באמצעות המעבר לעיל שתואר בהוכחה מעלה.

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

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

ראשית נקבעו בדוגמה.



בתמונה מתואר אוטומט הכלול מעבריו אפסילון. נשים לב כי ישנו מעבר אפסילון $q_3 \rightarrow q_2$. ומה זה אומר? כל קלט יכול לעבור שס. כמו כן, ישו מעבר אפסילון $q_2 \rightarrow q_1$. מה הוניבע בהיפוך האוטומט ללא מעברי אפסילון? נראה כי האות a למשל, יכולה לעבור 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 $q_3 \rightarrow q_1$. כתוב 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 σ ,

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

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

ד. $Q_0^N = \bigcup_{p \in Q_0^\varepsilon} CL(p)$ - ככלומר קבוצת המצביעים הראשוניים של האוטומט N בהיקף המצביעים הראשוניים, ולאוסף את כל המצביעים שמשמו ניתן להגעה אליהם ללא קריאת קלט. אלו יהיו - המצביעים 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 מערבי אפסילון, $\Delta'(q, \sigma) = \{\delta(q, \sigma)\}$

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

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

3.6.2 ערובות שפות

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

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

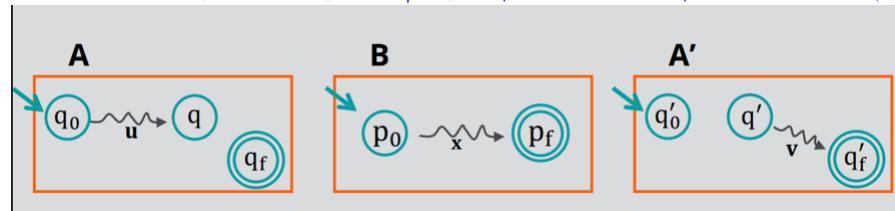
נראה כי ניתן לכתוב גם את $L_3 = L_1 \circ L_2 \circ L_1$. הרעיון יהיה פשוט: L_1, L_2 רגולריות ולכן יש אוטומטים שמקבלים אותם. כל שנרצה, הוא לשגר את האוטומטים. נסמן בהאותה את האוטומטים המקבילים את השפות B, A . כמו כן, ניצור אוטומט $A' = A$ (משם שכפול). הרעיון (לא פורמלי) יהיה כך -
 א. מצב ההתחלתי יהיה המצב ההתחלתי של A . ממשנוначילה. שם נקרא את w_1 .
 ב. מכל מצב מקובל של A , נעביר מערבי אפסילון אל המצב ההתחלתי של B . שם נקרא את w_2 .
 ג. מכל מצב מקובל של B , נעביר מערבי 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:

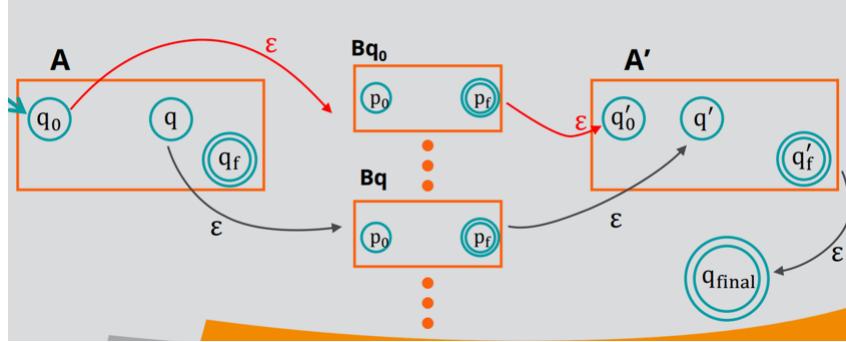
$$L = \{w | w = u \cdot x \cdot v : u, v \in L_1, x \in L_2\}$$

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



כיצד תתכצע קריאות מיילס? כדלהלן:

כיצד נוכל לנוע בתוך האוטומטיות? למשל, לעבר מ- q_0 אל q_f ? באמצעות מעבר אפסילון מ- q_0 אל q' . ואז קירiat המילה תהיה לולקה. שיש לנו שזוגמא זו היה כלית מז. הרו, מהו q' הזה בתוך A ? זה סתם הינה ייחוש שאסור לנו לכך. לנו יותר שקיימות מעבר אפסילון אחרים. הכוונה שבודינו: כאשר גניע למק' p_0 , נטרץ לאזכור ולשומו, אותה מק' ב- A הוביל אותנו אליו. לנו הפטרון היה:



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' המקבל 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. $\{q_{final}\}$ מודוע מכפלת קרטזית? נזכיר כי היה לנו שכפול של אוטומט B , וכן המצביע בו תלויים במצבים השונים של Q
ב. $q_0^C = q_0$
ג. $F^C = \{q_{final}\}$
ד. פונקציית המעברים:

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

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

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

4 יחידה 5: שפות וגולריות

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

4.1 סגירות

קובוצת ניתנת 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 לולאה עצמית עם כל הא"ב.

4.1.1 סגירות לשרשור

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 קודם נסיף מעבר אפסילון, ונטול את המצבים המקוריים הקודמים. כתע, נסיף מעבר אפסילון מהמצב המקביל של A למצב התחלתי היחיד של B , וסיימנו. כל המצבים המקוריים יתבטלו באוטומט החדש, המצב המקביל היחיד יהיה המצב המקורי של B שוב, נניח שקיים אחד כי אם לא נסיף מעבר אפסילון, והמצב התחלתי יהיה המצב התחלתי של A .

4.1.2 סגירות לסגור קלין

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

הוכחה לא פורמלית: רגולרית لكن קיימים אוטומט A שמקבל אותו. נבנה אוטומט לא דטרמיניסטי שקיבל את L . בדומה, נניח שקיימים מצב התחלתי יחיד ומצב מקבל יחיד. אם לא נגוע לשם. בשבייל לקבל את L^* , כל שנעשה הוא להוסיף 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 שבוצע 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 לרוורס

תהי L רגולרית. אז גם L^R רגולרית.

הוכחה לא פורמלית: סמן את האוטומט שמקבל את L כ- A (קיים כי רגולרית). 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 יצא ממנו, וכל ח' שיוצא ממנו יכנס אליו. בדומה, עבור כל מצב אנחנו נחפץ את סדר החצים. וכן, המצב q_0 יחפץ כתולען מקבל. כתעת לединו אוטומט A' שמקבל את שפת הרוורס. 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 נשנה מצבים מקבלים, כך כל תחילית של מילה תתקבל בשפה. פרט לקבוצת המצבים מקבלים, לא נשנה דבר. כיצד נגיד פורמלית את קבוצת המצבים מקבלים?

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

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

4.2 ביטויים רגולריים

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

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

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

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

א. בסיס

σ כאשר $\Sigma \in \sigma$.

\emptyset – גם ה- \emptyset , ביטויים רגולריים.

ב. פעולות שניינן לביצוע:

1. $r_1 r_2$, 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.

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

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

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 של $b|a$. כאשר אנחנו רוצים סיטים רק מהאפשרויות $\{aa, bb, ab\}$. נשים לב כי אם נדרש רצף של $a|b$ ובסופו אחד מהשווים aa או b בלבד, נעה על הדורש. מזען aa יכנס ממש, bb יכנס עס b אחת מ- $(a|b)$ ואחת נוספת לסתום, ab בדומה עם a מה $(a|b)$ ו- b בסופו. כלומר הבינו היה -

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

4.3 המרת אוטומט לביטוי רגולרי

טענה: תהי L שפה. אז

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

הוכחה:

ונכיח את הטענה שתראה לנו כיצד בהינתן אוטומט, ניתן להמירו לביטוי רגולרי.
 \Rightarrow נניח $r = L$ ונוכיח L רגולרי. הוכחת צד זה לא קשה יותר מדי, מכיון באינדוקציה על $|r|$ כשהבסיס הוא על ההגדלה לעיל, ובצעד מנוחים שבבקרה קיבלונו זודל שגדל מחדך שהתבצע ע"י אחת מהמניפולציות: איחוד, ששורר, סגוררים וכובב. וכך 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).

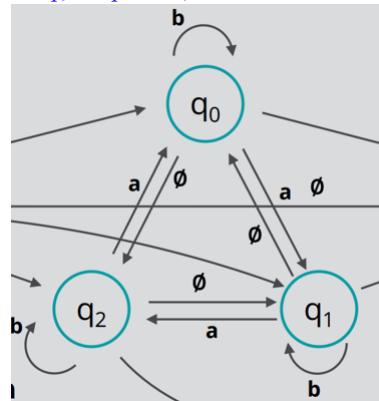
הקששות מסווגנות בביטויים רגולריים.

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

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

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

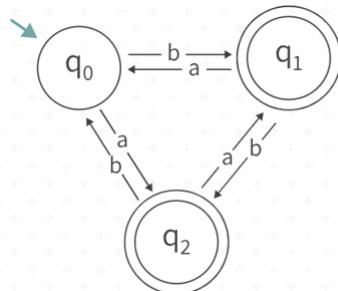
דוגמא לא תזקיק. נסתכל על האוטומט המקורי הבא. נראה כי אם נרצה למחוק את המצב q_1 למשל, עליו לטפל בשתי קשיות. $q_1 \rightarrow q_0$ עס "a", ולכן עצמאית של b -ים, וקשת נוספת $q_1 \rightarrow q_2$ עס ".ab". ככלומר, אם נרצה להעלים את המצב q_1 , נרצה להוסיף קשת מ- q_0 לש q_2 של הביטוי הרגולרי a^*a . האם זה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



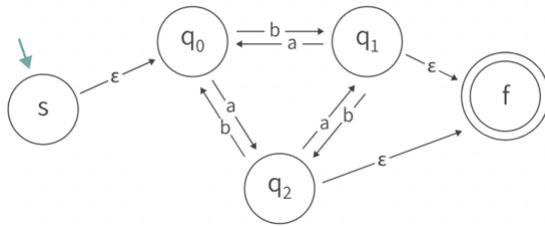
לאחר שביצענו דילול מצבים - נשארנו עם ביטוי רגולרי סופי, זהו הביטוי הרגולרי שמקיים $L(r) = L$.

דוגמא:

נרצה להמיר את האוטומט הבא, לביטוי רגולרי:

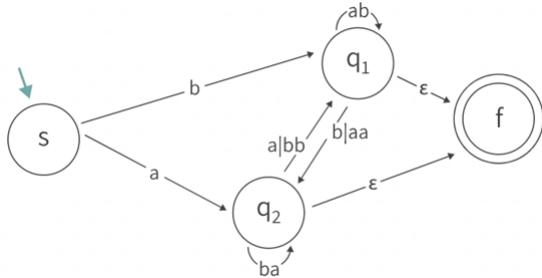


שלב א. נמירו לאוטומט מוכלל:



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:



והסדרוב הזה זה x .

4.4 למת הניפהו

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

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

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

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

דוגמה. נסתכל על השפה

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

זו שפה שאינה גולרית. נב"ש שהוא כו וגולריות. לפי הلمה, קיוס N שפקיות את כל הזרוש. תהז $w = a^N b^N$ שהוא חלק מהשפה והוא $> N = |w|$. עבורה קיוס פירוק xyz פורק $x = a^k$, $y = a^j$ ו $z = a^l$ כך $|xy| \leq N$ ו $|y| < N$ ו $|yz| \leq N - j$ וכן $xy^i z \in L$ כאשר $i \geq 0$. לפי תכונה 3, לכל i טבעי, $xy^i z \in L$, נסתכל על $i = 0$ (מסתכל שכקורת זהה אפס טכני? מורות, אפשר לחתוך גס 2) 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 לא ניתן L לאינה 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 אינה 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, ולאחר מכן פירוק ייחיד).

דוגמה נוספת. נוכח כי שפת הפליאורוטיס מעלה א"ג $\{a, b\}$ אינה גולרית.
פתרון: נניח בשיליה כי L כו וגולריות. אז לפי למת הניפוי קיוס N שיענה על הזרוש. נבחר את המילה $w = a^n b a^n$, אכן, w היא פליינטס, 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 $y = a^k$ $z = a^l b a^N$ וכן $l > 0$, לפי למת הניפוי לפחות $i = 0$, $xy^i z \in L$, נקבע $k + l = N - j \neq N$, $xy^0 z = xz = a^k a^l b a = a^{k+l} b a^N$ וכן מילוי זו לא שייכת לשפה, כי $i > 0$.
מכיוון שקיים סתיו והשפה L אינה גולרית.

דוגמה 2. נוכח כי השפה $L = \{a^p : p \text{ prime}\}$ אינה גולרית.
פתרון: נב"ש כי L כו וגולריות. אז, לפי למת הניפוי קיוס N שיענה 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 $i = n - j - k \geq 0$. בפרט, לפי הлемה, לפחות $i > 0$, $xy^0 z = xz = a^j a^{p-j-k} = a^{p-k}$, האם זה עוזר לנו? לא, זו לא שאלה טטודורית. נסה שוכ. נבחר $i = 1$. 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 ראשוני, מכפלה של ראשוני במשהו אחר, תתו לא ראשוני. בסתיו.

דוגמה 3. נוכח כי השפה $L = \{a^i b^j c^k : i, j, k \geq 0 \wedge j = \max(i, k)\}$ אינה גולרית.
פתרון: נב"ש כי L כו וגולריות. אז, לפי למת הניפוי קיוס N שיענה על כל הזרוש של הлемה. נבחר את המילה $w = a^n b^n c^{n-1}$, מתקיים לפחות $k = n - 1 = n - n - 1 = 0$. לכן, $j = \max\{i, k\} = n$ וכן $i = n > n - 1 = 0$. וכן $n > 3n - 1 = 2n > N$. 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, לפחות $i = l + r > 0$, נקבע $l + r = n$. כלומר, $x = a^l$, $y = a^r$, $z = a^{n-l-r} b^n c^{n-1}$.

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

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.
מלמת הניפוי. 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).

זוגמה 1. נגדיר את הפעולה $perm$ (פרמוטציה) על השפות הרגולריות כשפה של כל המילים w , כך
שקיים $y \in L$ פרמוטציה של w .
הוכח הף: אם L רגולרית, היא סגורה תחת פעולה $perm$ (לטיר $perm(L)$ 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 אוטומט עם מצב יחיד - ללא עצמיה עם כל
הא"ב.

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

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

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

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

5 יחידה 6: שפות חסרות הקשר

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

5.1 דоказך חסר הקשר

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

$$T \rightarrow H : M$$

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

$$M \rightarrow LD$$

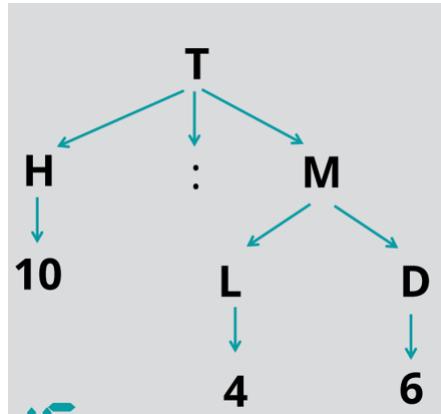
$$D \rightarrow 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$$

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

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 היצירה
- ב. Σ הוא הא"ב הסופי, נקראים גם טרמינלים.
- ג. V הוא קבוצת המשתנים, אוסף השמות שנתנו באשר נתנו את כללי היצירה.
- 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 $uzv \Rightarrow_G uAv$ את הגיירה של המחרוזת uAv בפי G .
עבור מחרוזות $(V \cup \Sigma)^*$ נסמן $u \Rightarrow_G^* v$ אם ניתן לעבור מ- u על ידי 0 או יותר גיירות של G .

5.4 שפת הדקדוק

השפה של G הינה

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

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

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

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

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

5.5 יצירות דקדוקים חסרי הקשר

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

דוגמה 1. כתוב דקדוק חסר הקשור לשפה הבאה:

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

נשים לב שהשפה שוקלה כתיבתה כך - $L = \{a^x b^y a^z b^z b^y b^x | x, y, z \geq 0\}$. מדובר נעדייף 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 $L = \{w \in \{a, b\}^* | \#a_w = 2\}$

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

$$S \rightarrow AaAaA$$

$$A \rightarrow bA|\varepsilon$$

5.6 למת הניפוח לשפות חסروفות הקשר

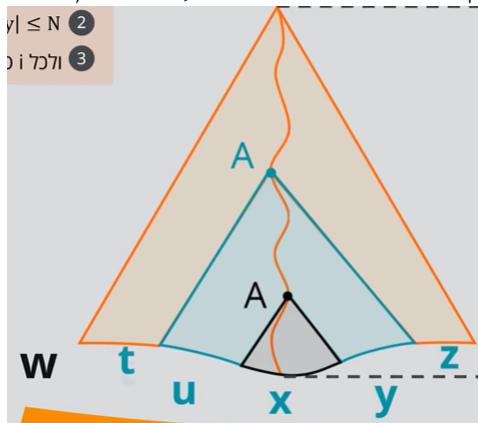
בדומה למת הניפוח עבור שפות רגולריות, משתמש בematת הניפוח להוכחת אי חסروفות-הקשר של שפה.

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 לתת העץ הגדל של A יקרא y ומשמאלו u . החלקים מעל A הראשון מימין ומשמאלו בתאמה יהיו t ו- z . וכך - קיבלנו את הפירוק (בתמונה). כעת, כיצד נוכל $\forall i \in \mathbb{N}$? באינדוקציה. נראה כי עבור $i = 0$ כל שציריך הוא לחתות את החלק הכהיל, להוציאו החוצה, ולהעלית מעלה את הכהילו, ואז נזיר את השורה. וכן הלאה - נוכל להוסיף את החתיכיה הכהולות כמו פעמיים שרצחה, ומתחתיו את השורה.



הוגמה 1. הוכח הפרך - השפה $\{w \in \{a,b,c\}^* : w = a^k b^k c^k | k \in \mathbb{N}\}$ היא חסروفת הקשר.
הפרכה - נב"ש כי L חסروفת הקשר. אזי לפי הлемה לעיל, קיים n שעונה על הדרוש. נגדיר $w = a^n b^n c^n$.

מתקיים $n > |w| = 3n$, לכן קיים פירוק $w = tuxyz$ כך ש- $|uy| > 0$ וכן $|uy| \leq n$. נשים לב כי מהדרישה השנייה, בפרט לא ניתן כי uy יכיל a או c (בעה u ו- c יחידיו, ולכן חסורה לפחות אחת אחת uy). כמו כן, מהדרישה $0 < |uy| < n$ יש שם לפחות אחת אחת. נגידר $i = 2$ וונפץ $z^2 xy^2 tu$. זו מילה שניפחנו אותה מסויימות, ולא את כולם, ולכן התנאי לא יישמר, כלומר מס' a, b, c במחוזות יהיה שונה. $\exists i \in \mathbb{N}$ וכן זו סתייה למת הניפוח. מכאן, ש- L אינה חסروفת הקשר.

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

הפרכה - נב"ש כי השפה L חסروفת הקשר. אזי, לפי הлемה לעיל, קיים n שעונה על הדרוש. נגדיר $w = a^{n+2} b^{n+1} c^n$, אכן $w \in L$ וכן $|w| = 3n + 3 > n$. מכאן שקיים פירוק $w = tuxyz$ כך $|uy| > 0$ ו- $|uy| \leq n$. נפצל למקרים -
א. uy מורכב ממשיים בלבד: נבחר $i = 0$, בהכרח מס' הא-ים ירד, ויהיה כעת קטן יותר מאשר n .

icut, במליה החדש עם הניפוי $\#a_w = n + 1$, $\#a_w \leq n + 1$, $\#a_w < n + 2$, $i = 0$, כמו כן $j > i$ או $j > i$ וקיים סטירה לדרישה $j > i$.
פומילית יותר,

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

כיוון ש $0 < |uy| < n + 2 - |uy| \leq n + 1$, בהכרח $i < j$, בסטירה.
ב. uy מרכיב בלבד: ננפח עם $= 2$. i . בהכרח $0 < |uy| < |uy|$ ולכן קיים שם b אחד לפחות.
כיוון שניפחנו עבור מס a -ים חיובי ממש, מס' ה- b -ים יידל עט, ככלומר $\#b_w > n + 1$ ושוב, סטירה
לכך שמס' ha -ים צריך להיות גדול יותר ממש.
ג. uy מרכיב בלבד: ננפח עם $= 2$. i . בהכרח $0 < |uy| < |uy|$, שכן קיים שם c אחד לפחות.
נראה כי

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

כיוון ש $1 \geq |uy| \geq n + |uy| \geq n + 1$ ומתקיים גם $j > k$ בסטירה.
ד. uy מרכיב ab -ים ואז b -ים: אנו יודעים כי $0 < |uy| < |uy|$, וכן יש גם a -ים וגם b -ים ולכן ≥ 2
ננפח עם $= 0$. נקבל $i = 0$.

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

כיוון ש $2 \geq |uy| \leq n = \#c_{w'} = n + 2 - |uy| \leq -2$, $|uy| \leq -2$ ונקבל $\#c_{w'} = \#a_w$, ככלומר מס ha -ים
יהה קטן שווה במס' hc -ים, בסטירה.
ה. uy מרכיב ab -ים ואז c -ים: בדומה, יודעים כי $0 < |uy| < |uy|$ וכן יש גם b -ים וגם c -ים ולכן
 ≥ 2 , ננפח עם $= 2$ ונקבל $i = 0$,

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

בדומה, נראה כי $2 \geq |uy| \geq n + |uy| \geq n + 2 = \#c_{w'} = n = \#a_w$ בסטירה.
נשים לב שאלו כל האפשרויות, לא ניתן רצף של שלושת האותיות כיוון $shn \leq |uxy|$.
שכל אחת מהאפשרויות קיבלו סטירה, סה"כ לכל פירוק אפשרי נקבל סטירה. מכאן - סטירה למלת
הניפוי. השפה אינה חסרת הקשר.

3. הוכחה הפרק: השפה הבאה חסרת הקשר $L = \{a^{i!} \mid i \in \mathbb{N}\}$ מעל $\Sigma = \{a\}$.
הפרבה: נב"ש חופשית הקשר. ככלומר, לפי למת הניפוי, קיים n שיוננה על תנאי הלמה. נגיד
את המילה $w = a^{n!}$, מתקיים $n! \geq |w|$, וכך קיים פירוק $w = tuxyz$ כך ש $|uxy| \leq n$ ו $|uy| > 0$.
נסמן $k = |uy|$. מתקיים כי $n \leq k \leq n!$. נסתכ על ניפוי עם $= 2$. ככלומר,
נסתכ על ניפוי עם $= 1$.

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

נרצה להראות כי $n! + k \leq n! + n < n! + (n+1) \leq n!(n+1) = (n+1)!$
קיבלו סתירה.

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

וסה"כ הניפה עם $i=2$ הוא $L \notin$, וזה סתירה לлемת הניפה.

5.7 סגירות לשפנות חסימות הקשר

- טענה: השפנות חסימות הקשר שגורות לאיחוד, שרשור וסגור קלין.
- אין סגירות לחיתוך ומשלים.
- יש סגירות ל $prefix$ וכן $reverse$ לפי תרגיל מהקמפוס.

5.7.1 סגירות לאיחוד

תהינה L_1 ו L_2 שפנות חסימות הקשר. אז, לפי ההגדרה, קיימים דקדוקים חסרי הקשר $G_1 = L_1 = L(G_1)$ ו $G_2 = L_2 = L(G_2)$ כך ש $(V_1, \Sigma_1, S_1, R_1), G_1 = (V_2, \Sigma_2, S_2, R_2), G_2 = (V_U, \Sigma_U, S_U, R_U)$ נבנה $V_1 \cap V_2 = \emptyset$. בה"כ נניח כי $L(G_U) = L_1 \cup L_2$. מזוז $S_U \notin V_1 \cup V_2$. מדוע נוכל להגיד זאת? אחרת נוכל להתחילה גירה עם כללי יצירה של דקדוק אחד ולהמשיך עם כלו של הדקדוק השני, וכך נוכל לודא שהקבוצות זרות, ואם יש חפיפה בשמות המשתנים, נשנה את שמות המשתנים. נגיד "

$$\Sigma_U = \Sigma_1 \cup \Sigma_2.$$

$$V_U = V_1 \cup V_2 \cup \{S_U\}.$$

ג. S_U - התחלתי.

ד. { $S_U \rightarrow S_1 | S_2$ } קלומר איחוד כללי יצירה של שתי הקבוצות פלוס כל כללי יצירה מהמצב התחלתי לאחד מהמצבים התחלתיים.

5.7.2 סגירות לשרשור

תהינה L_1 ו L_2 שפנות חסימות הקשר. אז, לפי ההגדרה, קיימים דקדוקים חסרי הקשר $G_1 = L_1 = L(G_1)$ ו $G_2 = L_2 = L(G_2)$ כך ש $(V_1, \Sigma_1, S_1, R_1), G_1 = (V_2, \Sigma_2, S_2, R_2), G_2 = (V_o, \Sigma_o, S_o, R_o)$ נבנה $L(G_o) = L_1 \circ L_2$. נגיד "

$$\Sigma_o = \Sigma_1 \cup \Sigma_2.$$

$$V_o = V_1 \cup V_2 \cup \{S_o\}.$$

ג. S_o - התחלתי.

ד. { $S_o \rightarrow S_1 S_2$ } קלומר איחוד כללי יצירה של שתי הקבוצות פלוס כל כללי יצירה מהמצב התחלתי לשרשור הממצבים התחלתיים.

5.7.3 סגירות לסגור קלין

תהינה L_1 שפה חסמת הקשר. אז, לפי ההגדרה, קיים דקדוק חסר הקשר $G_1 = (V_1, \Sigma_1, S_1, R_1)$ כך ש $L_1 = L(G_1)$.

נבנה $L(G_*) = L_1^*$ שקיים נגיד "

$$\Sigma_* = \Sigma_1.$$

$$V_* = V_1 \cup \{S_*\}.$$

ג. S_* - התחלתי.

ד. { $S_* \rightarrow S_1 S_* | \varepsilon$ } מה קורה כאן? שרשור שהולך מ $S_* S_*$ אל S_* , בסוף ה S_* . מסתומים 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 לחיוך

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 הקשורות הקשר, גם L_2 היא כזו, ולכן שתיהן הקשורות הקשר. נראה כי $\{0\} \subseteq L_1 \cap L_2 = \{a^n b^n c^n | n \geq 0\}$ שאינה חסרת הקשר, כפי שהראינו לפני למות הניפוח.

מסקנה: אין חיותך תחת מושלים, כיון $\overline{L_1 \cap L_2} = \overline{L_1} \cap \overline{L_2}$, יש סגירות לאיחוד, נב"ש שיש סגירות למשלים, אז השפה מימין חסרת הקשר, וזו שווה לשפת החיותך, וגם היא חסרת הקשר, בסתייה.

מסקנות לסיום היחידה:

- א. אם שפה היא סופית, היא רגולרית, אז היא בפרט חסרת הקשר.
- ב. אם שפה היא רגולרית, אז היא חסרת הקשר.
- ג. לא מתקיים שכל שפה רגולרית מוכלת ממש בתוך שפה חסרת הקשר, למשל $*^*$ רגולרית, ולא מוכלת ממש בשפה חסרת הקשר.
- ד. אם אחד המשתנים גוזר את המילה הריקה לא בהכרח בדקדוק. למשל: $\varepsilon \rightarrow aA, A \rightarrow a$. בכל מקרה, המילה הקצרה ביותר בשפה והיחידה בה היא a ולא המילה הריקה.
- ה. ידי G דקדוק חסר הקשר, תהי s מילה בשפה של G . יתכנו שני עצי גיררה שונים עבור s .
- ו. תהי L מעל א"ב $C = |\Sigma|$. איז, אם L מקיימת את תנאי למota הניפוח לשפות רגולריות, היא לא מקיימת את תנאי למota הניפוח לשפות הקשר. כיון שאם יש רק אות אחת בא"ב אז האפשרויות לחילוקה של המילה הם למעשה שקולים בשתי הלמאות: בשתי הלמאות אנחנו מנפחים רצף של אות כלשהיא (האות היחידה בא"ב) שאורכו גדול שווה ל-1 וקטן שווה לא (הקבוע של הלמה).

6 יחידה 7: אוטומט מחסנית

ביחידה זו נזכיר את המודל החישובי, אוטומט מחסנית, שמקבל את השפות הקשורות הקשר. המודל הינו הרחבה של מודל האוטומט הסופי.

עד היום - התיחסנו לקלט שמניגע כאוטומט שמובן מאליו. כתעת נרחביב - הקלט שאנתנו קוראים על גבי האוטומט, נמצא על מה שנקרה: **סרט הקלט**. סרט הקלט מוחובר לאוטומט. האוטומט קורא מהסרט את האותיות אחת אחרersh מאין. באוטומט סופי, סרט הקלט הינו *read-only*. כמובן, אפשר לקרוא ממנו וכי אפשר לכתוב עליו. העבודה שלא ניתן לכתוב על האוטומט - היא החולשה של האוטומט סופי.

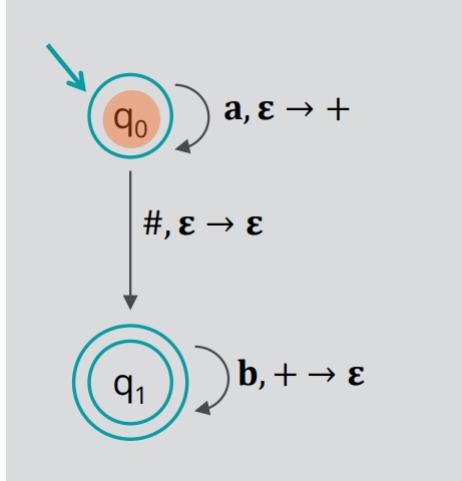
לכן, כאשר רצינו להזכיר שפה כמו $\{a^n b^n | w = \{a, b\}^*\}$ לא יוכלנו לעשות זאת עם אוטומט סופי, כיון שנדרשו לזרכו. לשם כך - קיבל את המחסנית. המחסנית היא רכיב זכרו שנitin גם לכתוב בו, וגם לקרוא ממנו. המחסנית עובדת בשיטת *LIFO*. לאחר שנכנס, ראשון שיצא. אם האוטומט רוצה, הוא כותבתו בראש המחסנית ומכניס אותו, והוא יכול גם לקרואתו מראש המחסנית. אם המחסנית כרגע ריקה, יש שם ε בראש. וכך:

כאשר רצתה לדוחף אותן קלט, נבצע $\sigma \rightarrow \varepsilon$.

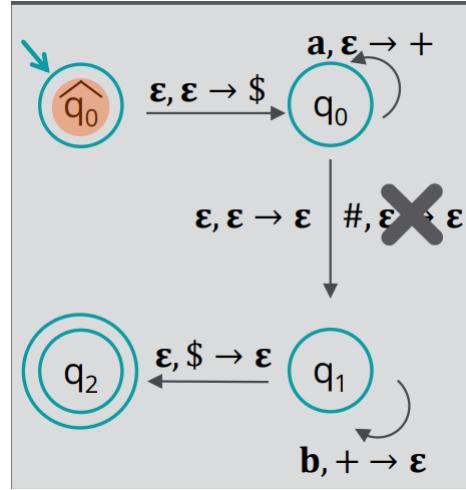
כאשר רצתה לא לגעת במחסנית, נבצע $\varepsilon \rightarrow \varepsilon$.

כאשר נרצה למחוק אות מראש המחסנית, נבצע $\varepsilon \rightarrow \sigma$.

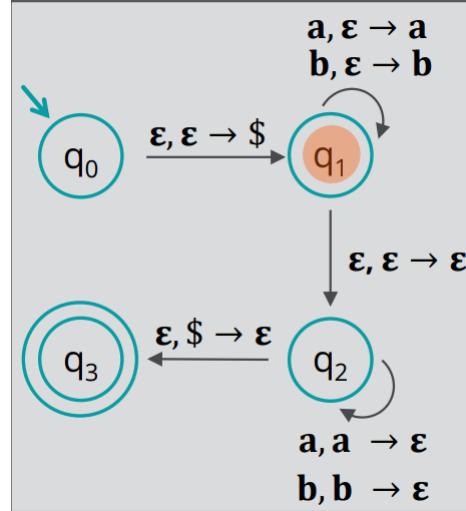
דוגמה 1. נרצה לבנות אוטומט מחסנית עבור השפה: $L = \{w \in \{a,b\}^* : w = a^n \# b^k | n \geq k\}$. איןנו רגולרית. נשים לב שאנו צריכים לצרוף, בשביל לאזכור כמה a -ים היו לנו עד כה. כיצד נסמן את המחסנית על גבי האוטומט? הסימן $+ \rightarrow \varepsilon$ אומר: קח את האפסיילון שיש בראש המחסנית, שים שם פלוס. כמובן, תכניס פלוס בראש המחסנית. מה הסימן $\varepsilon \rightarrow \varepsilon$ אומר? אל תעשה כלום עם הקלט בראש המחסנית. והסימן $\varepsilon \rightarrow +$? קח את הפלוס בראש המחסנית, תחליף אותו באפסיילון $-$ כלומר, תוציא את הפלוס. זה הכל. מה יקרה כאשר נכს עם המילה $a \# bb$? הקלט יגיע למצב q_1 עם האות b , וידרש להוציא b מהחסנית, אך המחסנית ריקה - ולכן נקבל شيئاו. כלומר: החישוב ית��ע.



דוגמה 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 odal מס' b .icut נרצה לבדוק אותו מספר. הרעיון הוא כזה - אנחנו יודעים שקריאת a דוחפת $+$. קריאת b תוצאה $+$. אם היה מס' שווה - המחסנית תהיה ריקה. כיצד נדע שהיא ויקה? התו הראשון שנדרוף יהיה §. כך נתחיל את הקלט שלנו, כנסים את הקלט, אם בראש המחסנית יש §, אז במס' a היה שווה במס' b , ולכןicut בראש המחסנית § ונציא אותו ונקבל את המילה. למעשה § יעזר לנו להתגבר על חוסר הידעיה של מצב המחסנית. המעבר בין q_0 ל q_1 יהיה באמצעות מעבר אפסיילון $-$ בין רצף המ-ים לרצף ה-ם.



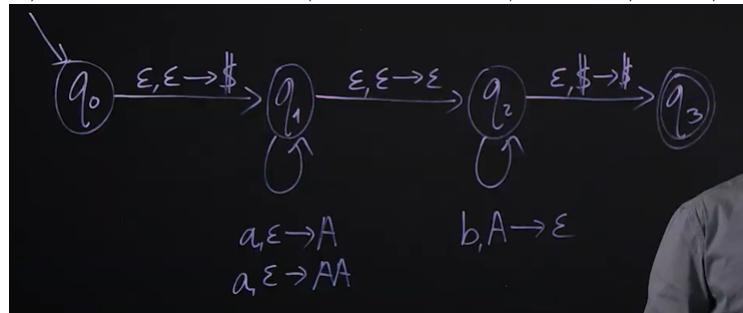
דוגמה 3. נרצה לבנות אוטומט ממחסנית לשפה $L = \{ww^R | w \in \{a,b\}^*\}$. היא איננה רגולרית. הרעיון יהיה ככזה בדוגמה הקודמת, וסיף $\$$ כמו קודם. באשר למעברים: אם קראנו a , נדחוף a למחסנית. אם קראנו b נדחוף b למחסנית. כשנרצה, נספיק מעבר אפסי lone. אם קראנו a נוציה a ממחסנית, ואם קראנו b נוציה b ממחסנית.שוב, אם קיבלנו ממחסנית ריקה - המחרוזות קייזו זאת זה, ולכן המילה בשפה, ככלומר קיבל $\$$ ומשם במצב מקבל. (הרי, נזכיר שאם הכנסנו מילה למחסנית, אם נקרא אותה מראש המחסנית לתחתית זה בדיק אבל בדיק *reverse* שליה).



דוגמה 4. נרצה לבנות אוטומט ממחסנית עבור השפה

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

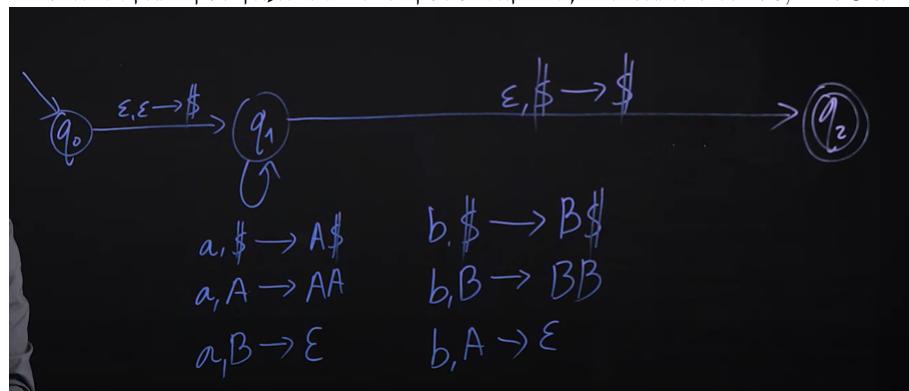
הרעיון יהיה זה - מי בשפה? כל המילים שמתחלות ב a ומס' ה b שלם הוא לפחות מס' a , וכן לכל יותר פעריים מס' a -ים. כמובן, נסיף $\$$ בתחילת ובסיום, שביל לדעת מתי התחתיות של המחסנית. כיון שהאוטומט לא דטרמיניסטי: אפשר את ה"ניחוש" או יותר נכון - הבחירה הלא דטרמיניסטית, בין להכניס a לבין להכניס aa למחסנית. כאשר מקבל b נוצרת a מהמחסנית. אם לאחר קריית הקלט הגענו ל $\$$, כלומר סיימנו את הקלט כנדרש - נ עבור מצב המקלט.



דוגמה 5. נרצה לבנות אוטומט מחסנית עבור השפה

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

הרעיון יהיה זה: נתחיל מס' $\$$ וגם סימן איתו, שביל לדעת מתי אנחנו בתחתית המחסנית. נרצה להוסיף סימן שייעזר לנו להבין מה המצב העודף שלנו כרגע. שיש לנו כיו יש 6 מצבים:
 א. $\$ \rightarrow A\$$. בראש המחסנית $\$$, ואני קורא a , יש לי עודף של a כרגע ולכן נרצה זכור זאת A בראש המחסנית.
 ב. $a, A \rightarrow AA$. בראש המחסנית A , אני קורא a , הייתי בעודף של a , אני ממשיך בעודף של A ולכן נקרא שב פעם a .
 ג. $\epsilon, B \rightarrow B$. בראש המחסנית B , אני קורא a ואני בעודף של B , لكن נמחק את הסימון של העודף B , ונמשך ממש הלהה כי a קיזז את B שציין את העודף על b ספציפי כלשהו
 ד. $b, \$ \rightarrow B\$$. בראש המחסנית $\$$, אני קורא b ונכנס לעודף של B ולכן נכנס B .
 ה. $b, B \rightarrow BB$. בדומה לב'
 ג. $b, A \rightarrow \epsilon$. בראש המחסנית A , אני קורא b ולכן מצטצם את העודף ולכן נמחק את A .



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

6.1 הגדרה פורמלית

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

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

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

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

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

6.2 תהליך החישוב של האוטומט

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

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

גרירה: הינו $C_1, C_2 \in F$ אוטומט מחסנית. והואו קונפיגורציות של P . נסמן $C_1 \mapsto_P C_2$ כלומר, C_1 גורר את C_2 אם כשנמצאים ב- C_1 ניתן לעבור C_2 לע"י מעבר יחיד. **נרחיב את המושג:** נסמן $C_1 \mapsto_P^* C_2$ כלומר, C_1 גורר כוכבית את C_2 אם כשנמצאים ב- C_1 ניתן לעبور C_2 לע"י 0 או יותר צעדים.

6.2.1 שפת האוטומט:

הינו $L(P) = \{ w \mid P \text{ מוגדרת להיות}$

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

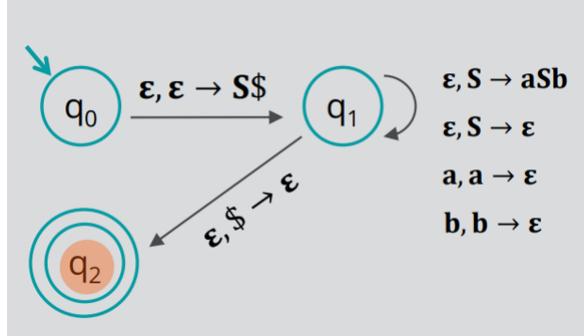
כלומר כל המחרוזות שניתן להתחיל מ q_0 , להגעה לpcb מכב, ככה שקרנו את כל הקטל, התחלו ממחסנית ריקה והסתמכו במחסנית כלשהי.

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 לו וויאו $\$$ למחסנית. 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 נוכל 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 זה Σ .

$$\begin{aligned} \text{יהי } D = (Q^R, \Sigma, \delta^R, q_0^R, F^R) \text{ אוטומט סופי דטרמיניסטי מקבל את } R \\ \text{והי } P = (Q^C, \Sigma, \Gamma^C, \Delta^C, q_0^C, F^C) \text{ אוטומט מוחסנית מקבל את } C \\ \text{בנייה אוטומט } U = (Q^U, \Sigma, \Gamma^U, \Delta^U, q_0^U, F^U) \text{ שיקבל את } C \cap R \\ \text{נגדיר מרכיביו להלן:} \\ Q^U = Q^R \times Q^C \\ q_0^U = (q_0^R, q_0^C) \\ F^U = F^R \times F^C \\ \text{או משתמשים במוחסנית עבור השפה } C \text{ בלבד, וכך:} \\ \Gamma^U = \Gamma^C \end{aligned}$$

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 עבור C , לדעת היכן להתקדם, באשר σ הוא הדחיפה למוחסנית. הינו דטרמיניסטי לנו אין בו מעברי אפסילון, ובשורה השנייה מותאמת המצב הזה: ברכיב 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 ניתן כידוע שהיא חסרת הקשר, כולם קיימים דקדוק עבורה. נראה כעת כי לשפות רגולריות, יש מבנה מיוחד לדקדוק שיוצר אותה. דקדוק זה נקרא **דקדוק רגולרי** והוא אותו אנחנו מחלקים לשני סוגים:

א. דקדוק רגולרי ימני

$$\begin{gathered} \text{הכללים הם אחת מהצורות הבאות:} \\ \forall A, B \in V, \sigma \in \Sigma \\ \begin{aligned} A &\rightarrow \sigma B & .1 \\ A &\rightarrow \sigma & .2 \\ A &\rightarrow \varepsilon & .3 \end{aligned} \end{gathered}$$

ב. דקדוק רגולרי שמאלי

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

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

טענה: שפה L רגולרית אם ורק אם קיימת DFA $A = L(G)$

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

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

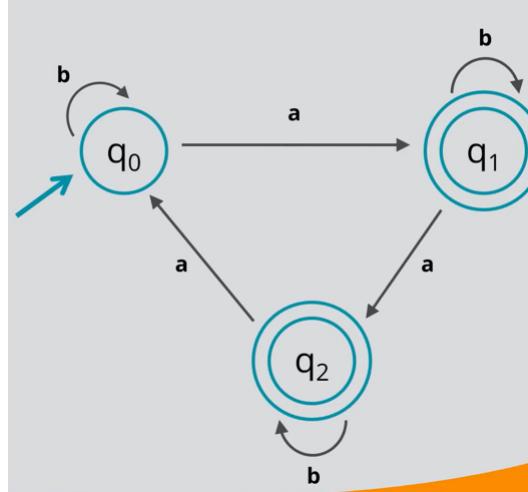
מבנה DFA רגולרי ימני $G = (V, \Sigma^G, R, S)$

נגידר: $\Sigma = \Sigma^G = Q$. נשים לב, נתיחס לקבוצת המ מצבים המשותפים. $S = q_0$

cut גדריר כללי יצירה R :

1. לכל $p = p(q, \sigma)$ כלומר מ q עברים עם σ למ' גדריר את הכלל $\sigma p \rightarrow q$
2. לכל $f \in F$ הכלל $\sigma \rightarrow f$ ידוע? אם הגענו למצ表 מקובל, וכל להפסיק עם אפסילון את DFA יי' זיהילה 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 $aq_0 \rightarrow aq_1$.

בדוק נקבע $aq_1 \rightarrow aq_1$.

בז'וניה, מה' נקבע את DFA:

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

ובאשר למצבים המתקבלים, נקבע $\varepsilon \rightarrow q_1 \rightarrow \varepsilon, q_2 \rightarrow \varepsilon$. מה' DFA זהו DFA חסר ההקשר המתאים לאוטומט.

מס' העורות על היחידה:

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

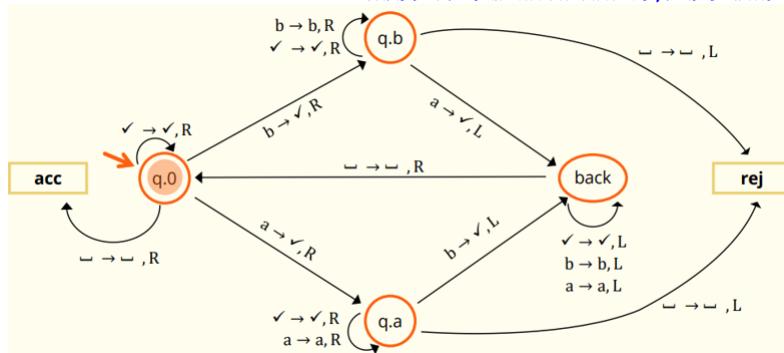
7 ייחודה 8: מכונת טיורינג

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 $w = L$ היא שפה שהיא אוטומט סופי והוא אוטומט מחסנית, לא מסוגלים לפתרו. דהיינו $L = \{w^n c^n a^n\}$. די ברור, כי מחשב מסוגל להוכיח את השפה. מכיוון המכונה היא שהבעה במודלים שחצינו. כתעת ציע את המודול החזק מכולם. אלן טיירינג הוא אבי תורת מדעי המחשב - הוא פיתח מודל מתמטי שבמהלך השנים התברר כמודול מרכזי שמתפרק כמחשב: **מכונת טיירינג**.

7.1 מהי מכונת טיירינג?

כמו במודלים קודמים, הקטל יהיה על גבי סרט הקלט. בשונה ממודלים קודמים, ניתן לכתוב על המודול שלנו. **תמיד** ניקן לכנות על המודול רק על הראש, בו אנו נמצאים כרגע. הרחבה נוספת ומשמעותית - במכונת טיירינג ניתן לזרז שמאללה. במכונת טיירינג אנו מניחים שסרט הקטל הוא אינסופי - בשני הצדדים. כאמור, גם אם התחנו בנקודה כלשהי 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 מראשונה. נסמן \checkmark על האות שקריאנו, ונתזדקם בקלט ימינה לחפש b תואמת לה, נמצא אותה ונסמן בה \checkmark גם כן. כתען נחזר לתחילת הסרט - מציג על \checkmark הפתאומיטים וחפש שוב a או b תואם. נניח ומצאו a , נזכיר \checkmark , ונישקן ב巡视ת הסרט לפעיאות a תואם וכן הלאה. כך המכונה תעבור, בURA מילוטית. תמייד בסיסים נחזר לתחילת הסרט משמאלי, וכך עד כל אותיות הסרט יציג נזען של תואמת הולופה ב? אם ורק אם תהייה סריקה של כל אזור הסרט בה הכל הינה \checkmark , אז נוכל לסיים. מתי נזען שהגענו למשך לא טובי? כאשר בURA סריקה הגענו לפקס של " ", בזווית יש אותן בקלט שאנו לה תואמת, ולכן הגענו לאזור האסורה. ושולח למכב מות. כיצד ניעביר זאת לתיאור גוף של מכונה?



ובכן, זוו מכונת הטיירינג שמקבלת את השפה. נסביר כיצד מבה קורה כאן: $q.a$ הוא מצב בו ראיינו a ומיחסים לו b תואם. בזמנה, $q.b$ הוא מצב בו ראיינו b ומחשימים לו a תואם. $q.b$ והה מצב בו משתמש כשבוי לחזר לקופה השמאלי של הסרט. ישנו שני מצבים מיוחדים: מהייתה **accept**, מעכז זה יכול מילוי. ויש את המצב **reject** מהייתה **rej**. מעכז 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: הוא עושה שני דברים - זה לאושהו, וכותבת משחו על הסרט. לכן עלינו לתאר שלושה דברים: מה הסרטomial רואה, מה האותה, ומה כוון איזו. לכן הסימנו $a \rightarrow \checkmark, R$ מסמל: בהיותו שקראיי a , שיש שם \checkmark ותזוז ימינה. R ומייה ושמאללה.

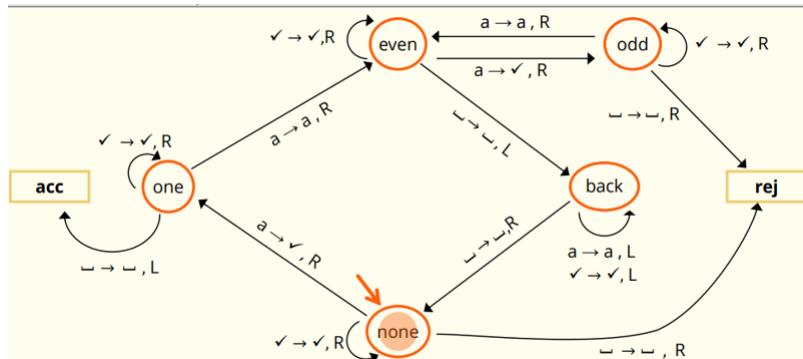
חשיבות - נשים לב, המכונה איננה יודעת שהגענו לקצה הקלט. נשים לב כי בהינתן קריאה של מילה, המכונה לא יודעת שבתא אחד משאללה נגמר הקלט. וכן מוגעות הולאות העסיות *back* של *L*. ✓ → ✓. מה המשמעות? כאשר נרצה לחזור, אך שוב שאללה. אל תנסה כולם בסיטוון. ואז אנחנו נהייה כבר במשבצת עם רוח - ודע כי הגענו לאזור לא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

כל עוד נראה ✓ משאר ב**q0** עס לולה עצמים כטובנו. כי לא נרצה להתקדים כי אין מה לזכור. אם אנחנו בפעץ *a*, אם נקרו *a* או ✓ ממשיך בקריאה ימינה. רק ✓ איננו מփשיט. בדיטה עכבר המיצג *b* אם נראה *b* או ✓. כמו כן, בפעץ *back* אם נקרו *a* או *b*, נרצה שישארו שס כי אנחנו רועים רק לחזור חזרה ולא לגעת בהירה.

מתי נדע להגיאו לפעץ מקלט? נראה כי יוציא מ**q0** מעבר *R*, → –. משמעותו: אם אנחנו בפעץ המקלט, וכל מה שעשינו היה לזרז מילאה, אז שאר קלט, וכן הגענו לסתה: אולי שאנו כבר קלט – משמע הפעלה שליו חייכת להתקבל, וכך נגיע לפעץ מקלט. מתי נראה איננו *b*, או ב*b*, או ב*a*? או ב*a* בין זוג תואם למי ששמרנו שאנו צורכים לפעטו לו בין זוג תואם. בפעץ זהה: ישן לדחיה. אין צורך להמשין. זו הייתה דוגמה פורנכת – זהה כל הרעיון. מזל וחושני זה חזק פאה.

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 והוה לעכבר על קלט הפעלה ולמבחן בכל פעם *a* אחד, ואתה *a* אחריו להשאיר. כיצד נמבחן? אם ✓. כך נעכבר בפעם הראושונה על הקלט, למשל המילה *a/a*. ואז, נעכבר פעמי'ן על הקלט: שוב, פעם את נמבחן *a* ראשון, ואת *a* אחריה שוארו: ✓✓✓✓. ומה עשוינו כאן בעצם? בכל פעם חלכנו את מ"ט ה"ט פי 2, אם בסופו של דבר קובלנו מיצג בו נשאר *a* אחד בלבד, אז המ"ט המקורי של המ"ט 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 יש לנו מיצגים *acc* ו*rej*. המיצג *one* משמעותו ש愧ב *none* הוא המיצג ההתחלתי – עדין לא קראונו *a* בסביב סיריקה זה. המיצג *one* משמעותו שקראונו *a* בזוז. המיצג *even*, *odd* בההתאמה כאשר קראונו מ"ט או זוג/זוגי של *a*-ים. המיצג *back* עכבר חזרה אחרת בקלט.

נראה את המכונה בתמונה, שעה בדוק לתיאור לעיל. נראה כי אם הגיעו למיצג *one*, או זוג אחד בלבד וכן המילה כו בשפה, כי חזקה של 2, וכן אם הגיעו לסוף הקלט – אך למיצג *acc*. אם הגיעו למיצג *odd* ולאחריו נגמר הקלט, כלומר *rej* כי המילה בהכרח לא בשפה אם מס 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, Σ הינו הא"ב, Γ הוא א"ב הشرط הסופי, q_0 הוא המצביע ההתחלתי,
- הוא המצביע המקורי והוא המצביע הדוחה.
- δ 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 *aq₀ab* 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

מה אם יש מילוי שהמכונה לא עוצרת עכורים לא במצב דחיה, ולא במצב קבלה? כמובן - לולאה אינסופית על הקלט, 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-קבלה". מושג חלש יותר. לא מקבלת = דוחה / לא מגיעה למצב קבלה.
 כל מכונות טיורינג מקבלת שפה כלשהי, שהוא פשוט שפת כל המילויים שדרכו מגיעים ל.acc במכונה.
 זה אכן מוגדר richtig.

7.3 תיאור מבנית טיורינג באמצעות טבלת מעברים

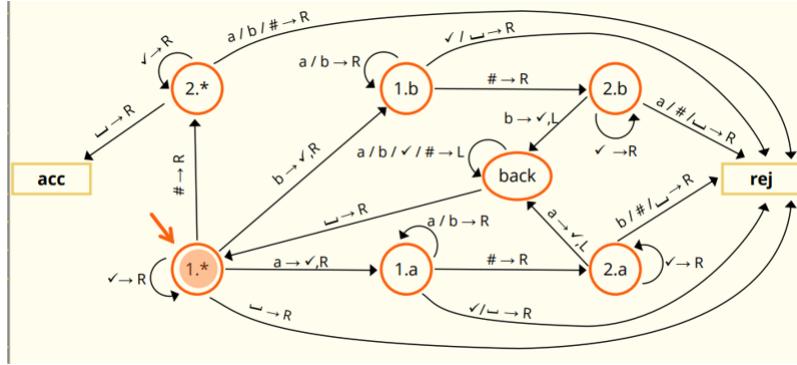
כעת, לאחר שלמדנו הגדרות פורמליות - נלמד מס' טכניקות וטריקים לבניית מכונות טיורינג.

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

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

הרעיון יהיה פשוט - נתחיל בקירiat הסרט משמאלי, נזכיר אם ראיינו a או b ונמחקה עם $\#$. ממש נתקדם ונידרג על כל האותיות עד לאוות הראשונה שפוניה 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 אם ראיינו a , "כיצד נזכיר? נשותמש בתאי זכרון פנימי. בתא ראשון נזכיר באיזה צד של המחרוזת אנחנו, ובתא השני באיזה אותן אנחנו צריכים לזכור. כיצד ממשים את תא זכרון הפנימי? במצבי המכונה. לכן מצביו האוטומט יהיו בזוגות - למשל: 1.b מציין שתא השמאלי של הזכרון 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.*	σ		1. σ	✓	R	$\sigma \in \{a, b\}$
1. σ	π		1. σ	↑	R	$\pi \in \{a, b\}$
1.*	✓		1.*	↑	R	
1. τ	#		2. τ	↑	R	$\tau \in \{a, b, *\}$
2. τ	✓		2. τ	↑	R	
2. σ	σ		back	✓	L	
2.*	—		acc	—	R	
back	a, b, ✓, #		back	↑	L	
back	—		1.*	↑	R	

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

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	σ	q.(S ∪ {σ})	✓	R	σ ∈ S
q.S	σ	q.S	⊆	R	σ ∈ S
q.S	✓	q.S	⊆	R	
q.{a,b,c}	a, b, c, ✓	back	⊆	L	
q.∅	—	acc	⊆	R	
back	a ,b, c, ✓	back	⊆	L	
back	—	q.∅	⊆	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 נזכרן ה- X . במהלך הסריקה נפרק משלב ימינו את החלק הראשון, לבחו את הספרה הראשונה ולזכור אותה בזיכרון פנימי. כך נמשיך עד שנתקיים $\#$. שם נמצא את ה- y המתאים ונזכור אותו בזיכרון פנימי. שם נמשיך עד $\#$ הבא ונמצא את ה- z המתאים וטנו זכרו בזיכרון פנימי. נבדוק את שלושת הספרות שללו האם מקיימות את אי השווון, ואם כן נחזיר הלהה לתחלת הקטל. כמובן שככל יותר שקרהנו תשומנו ב✓. כאשר כל האותיות חוץ מהסמלויות יהוו ריקות, נדע שהמילה צריכה להתקבל ושאכן הייתה התאימה. נראה כי יוצר גרפי של מכונה זו כולל המון מצבים. המון. ובטבלה: ניר כי כל מצב יהיה עם שלושה>Status. יתאר את החלק בו נמצאים, האיבר השני בשלשה הוא תוקן תא האיקס, והשלישי הוא תא הווי. * מסמל שעוזר לא נקרה ספרה. נשים לב כי השורה החשובה ביותר בטבלה היא שורת התנאי, אם ורק אם מתקיים התנאי $\tau_2 \geq \tau_1$ אז $\sigma \geq \tau_1$ המילה מתתקבל. אחרת, במצב rej. איך התנאי היה בא לידי ביטוי במצבים עי' אוטומט גוף? אם הינו ב- τ_2 ורצינו לקרוא τ_1 , הינו harusים ל- $back$ כי אותן חוקיות, אם הינו מקבלים 9 הינו 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 ₁ . τ ₂	#	Y.τ ₁ . τ ₂
	R	⊑	Z.τ ₁ . τ ₂	✓	Z.τ ₁ . τ ₂
	L	✓	back	σ	Z.τ ₁ . τ ₂
$\tau_1, \tau_2 \neq *$	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 הינו הוא ✓ - הוא את הראש ימינה.

1. אם הינו הנקרא ✓ - קבל. (אין קלט)

2. אם הינו הנקרא ✓ *a* או *c* - דחה.

3. אם הינו הנקרא *a* ✓ כתוב במקומו - והוא את הראש ימינה.

ב. כל עוד הינו הנקרא *a* ✓ או *b* או *c* או *d* את הראש ימינה

1. אם הינו הנקרא *c* או - דחה.

2. אם הינו הנקרא *b* ✓ כתוב במקומו ✓ והוא את הראש ימינה.

ג. כל עוד הינו הוא ✓ או ✓ הוא את הראש ימינה.

1. אם הינו הנקרא *a* או - דחה.

2. אם הינו הנקרא *c* ✓ כתוב במקומו ✓ והוא את הראש שמאליה.

ד. הוא 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 פעולה הכפל? $j \times i$ אומר - בצע חיבור של *i*, *j*

פעמים. בהינתן מילה $a^2 b^3 c^6$ נרצה על כל אות *a*, למוחק (לטמן) ון *c* כס"ס הפעמים שמופיע *b*. *b*.

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

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

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

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

- 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. חזר לתחילת הקלט, אם נותרו אותיות a חזר לשלב א'.
- אחרת, התקדם לסוף הקלט. אם לא נותרו אותיות c קבל. אחרת: דחפה.

7.4 שימוש במכונות טיורינג לחישוב פונקציות

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

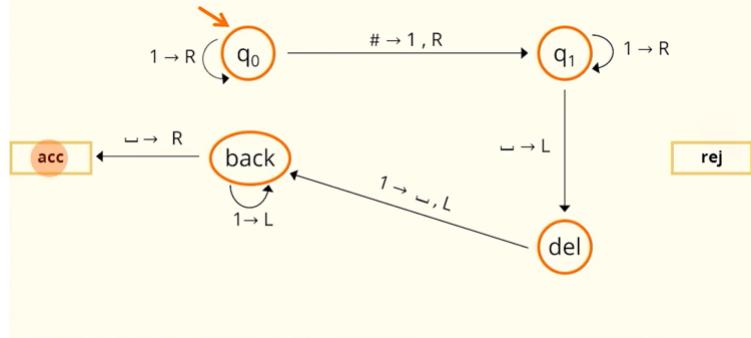
הגדרה: תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, rej)$ מכונת טיורינג. ותהי $f : \Sigma_1^* \rightarrow \Sigma_2^*$. נאמר כי M מחשבת את f אם:

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

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

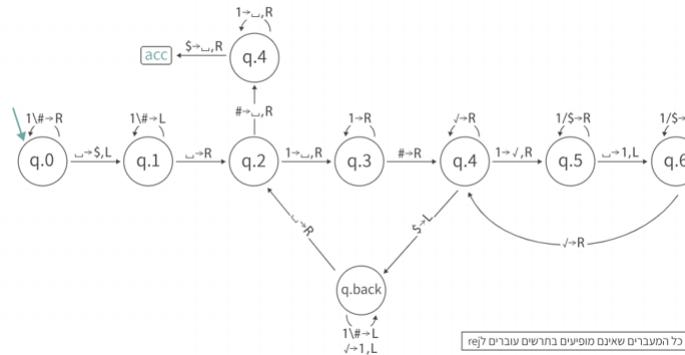
נרצה לחשב פונקציות $M : \mathbb{N}^k \rightarrow \mathbb{N}^l$, אם כן נרצה ליצגם באמצעות "ב" בשילוב $\Sigma_2^* \rightarrow \Sigma_1^*$.
 כיצד ניציג את הפונקציה באמצעות "ב"? ביצוג אונרי. כך למשל: המספר 3 יוצג ע"י 111, המספר 7 ע"י 1111111 וכן הלאה. כמו כן $0 = \epsilon$. כיצד נפרק פרמטרים? באמצעות#. למשל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

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 למצב מכבול ואחרת מתקדמים ימינה ללא הפסקה. היא תקבל את השפה, אך לא תכريع אותה כי יש מילים שעוברן היא לא עוצרת.
- ה. במכונת טיריניג שמחשבת פונקציית המכונה עצרת בחלק השמאלי של הקלט.

8 יחידה 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:

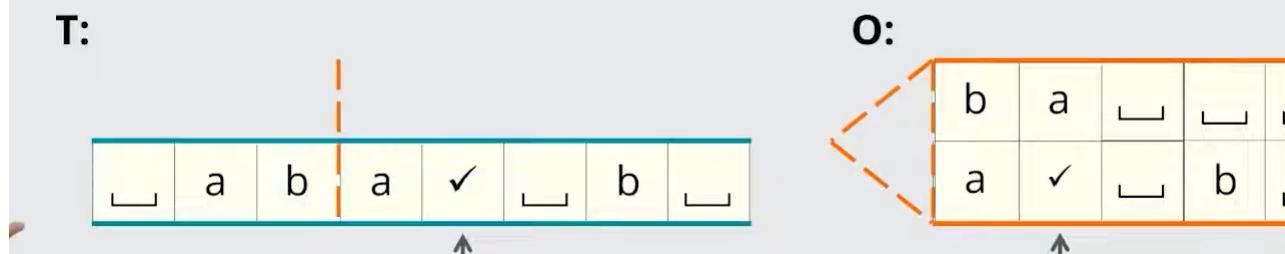
א. נראה שלכל מכונה במודל O יש מכונה במודל T - $M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, acc^T, rej^T)$. נבנה $(Q^o, \Sigma^o, \Gamma^o, \delta^o, q_0^o, acc^o, rej^o)$ מכונת טיריניג למודל O . נבנה $(Q^o, \Sigma^o, \Gamma^o, \delta^o, q_0^o, acc^o, rej^o)$ מכונת טיריניג למודל T .

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 רעד על הפער הזה.

הרעיון יהיה כדלקמן - כל הריבים יהיו זים למחר. נסיף לטבלת המעברים את הדבר הבא: נסמן ב-\$ את האות הראשונה משמאלי לקט, בשביל לדעתה היכן מתחילה הסרט השמאלי האינסופי. כל שנדרש לשנות הוא אם אנחנו נמצאים במעבר הראשון, 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 לבעס סימולציה" כאילו אנחנו בקורס דו כורוי. ככלומר - נבחר מקום לתחילה פמנו את הטרט השמאלי, כל מה שהוא משמאלי, נקלף, בזורה שתראה כך:



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

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

לאחר קו הקיפול: הזוג ימינה בקורס T תתרגם לתזואה שמאלה בקורס O , ותזואה שמאלה בקורס T תתרגם לתזואה ימינה בקורס O). (ההפק

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 לעז הקיפול נגזר U , ולהפוך עבורי D .

כמו כן, נשמר בזאת גזרו פלמי את המעכט בו היוינו ב- T . כל שיינו במכונה החדשה בעלת שתי השורות יהלך לאחר שתי השורות, ככלומר - אם במכונה החדשה היה לנו זוג $(a, -)$ והם יהלך ל- b כאשר אותו עדין מופיע לעז והפרקנה, נקלף ממש מקום את הזוג $(b, -)$ - ככלומר רך מה השתונה. כמובן, פוי משתנו מהזוג זה בהתאם U/D .

באשר המכונה הכתומה, O תראה דולר: הוא צריך לבעס לולאה שתחריז אורה ימינה, ולשנות את הכוון - ככלומר אס הרה D או עכשו U ולהפוך.

cutet למימוש הפורמל.

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

הערה - החלק $\{\tau | \tau \in \Sigma \cup \{-\}\}$ 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	π	p.D	π	L		תווות שמאלה (q,σ) $M^T(p,τ, L)$	
q.U	σ	p.U	τ	R		תווות שמאלה (q,τ) $M^T(p,τ, L)$	
q.D	—	p.D	—	L		תווות ימינה (q,—) $M^T(p,τ, R)$	
q.U	—	p.U	—	R		תווות ימינה (q,—) $M^T(p,τ, R)$	
q.D	π	p.D	π	R		תווות ימינה (q,σ) $M^T(p,τ, R)$	
q.U	σ	p.U	π	L		תווות ימינה (q,—) $M^T(p,τ, R)$	
q.D	—	p.D	—	R		פגעה בקצתה	
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 ^l .D	המ'	acc ⁰					
acc ^r .U	המ'	acc ⁰					
rej ^l .D	המ'	rej ⁰					
rej ^r .U	המ'	rej ⁰					
						כל השאר עוברם - rej	

נשים לב כי בטבלה מופיע מצב האתחול - שכן הקלט ניתן לנו כקלט אינסובי בשני הצדדים, ואנחנו, בהינתן אוטיות הקלט צריכים לתרגםו לאוגות על פני סרט עם ימינה בלבד. כמו כן, אם בסיום המכונה המקורית הגיעו גם אנחנו, ובודמה על *rej*.

הערה נוספת - נשים לב כי במקרה הסרט ימני, יש אנסוף רוחחים. לא נוכל לטפל בכלם ולהפוך אותם לאוגות: ולכן נקבע מצב בו מצאנו רוחח חדש, ונקבע שנתייחס אליו כאילו היה זוג רוחחים.

8.3 מודל TS

וריאציה נוספת היא מכונת טיריג'ינג שיכולה להשאר במקום, במודל זה הראש יכול להשאר במקום: כלומר 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 נגידר 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 $(q_0, a) = (q_1, b, S)$ הינה - תעבור למאובן 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 במאובן S . בכיוון השwi, הרעיון היה גם כן לא מסובך יותר מדי - לכל מאובן q_1 למשול, נצור מצב ביןיהם q_1^L . מה הרעיון? אנחנו נרצה לבנות מוכנת TS מוכנת T ולכן - אם יש לנו את האות S , למשול (q_1^L, b, L) , הוא יתורגם ל- $\delta^T(q_0, a) = (q_1, b, L)$ ולאחר מכן $\delta^T(q_1^L, b) = (q_1, b, S)$ ככלומר - זו למאובן הומני של S ימינה, אה"כ דמננו תזוז למצב שאחתה צריכה שמאלה על הרטט. ואז אכן השגנו מה שרצינו: לא אזנו ברטט אבל אזנו במאובן.

$$. Q^T = Q^{TS} \cup \{q^L | q \in Q^{TS}\}$$

8.4 מודל OR

היהי עצמן בשביל לראות. יש להשלים זה ולהעתיק לכואן לסייעו!!!

8.5 מוכנת טיריניג מרובת סרטים

וריאציה נוספת למודל, כאשר לכל מוכנה יש מס' סרטים כלשהו גדול שווה מ-1, **קבוע מראש**. לכל סרט ראש כתוב מסויל, אך כולם מחוברים לבקר המרכז של המוכנה. בכל צעד המוכנה קוראת וכותבת בכל אחד מהסרטים, וגם זה בכל אחד מהסרטים. הראשים הסרטים השונים יכולים להיות באופן ב"ת". במכונה עם מס' סרטים, הקלט תמיד בתחילת המוכנה נושא על הסרט הראשון, ולאחר הסרטים ריקים.

דוגמא. כתוב מוכנת טיריניג עם מס' סרטים שתכרייע את השפה $L = \{w \in \{a, b\}^* | w = w^r\}$ הרעיון, די פשוט. בתחילת: נעתיק את הקלט מהסרט הראשון לסרט השני. הסרט הראשון יהיה על האות השמאלית ביותר של המילה, ובסרט השני הראש יהיה על האות הימנית ביותר של המילה. בכל שלב, נבדוק האם שני הראשים עם אותה אות - ואם כן נתקדם. אם יש שווין לאורץ כל הדריך, אז המחרוזת היא פלינדרום. בשבייל הפשטות, נניח שניתן לחשאר במקומות במכונה. ראיינו כבר שמודל TS שקול למודל T . זו המוכנה -

מצב	1	אות סרט 1	אות סרט 2	מצב חדש	1	כתבת סרט 1	כתבת סרט 2	תיזוה סרט 1	תיזוה סרט 2
copy	a	—	copy	—	a	R	R		
copy	b	—	copy	—	b	R	R		
copy	—	—	back	—	—	S	L		
back	—	a/b	back	—	—	S	L		
back	—	—	check	—	—	L	R		
check	a	a	check	—	—	L	R		
check	b	b	check	—	—	L	R		
check	—	—	acc	—	—	S	S		

טענה: לכל $\mathbb{N} \in k$, המודל של מוכנת הטיריניג עם k סרטים שקול למודל מוכנת הטיריניג המקורי. $M^k = (Q^k, \Sigma, \Gamma^k, \delta^k, q_0^k, acc^k, rej^k)$ הוכחה (לא פורמלית בכלל אבל חשובה): לכל k ו לכל מוכנת טיריניג $M^1 = (Q^1, \Sigma, \Gamma^1, \delta^1, q_0^1, acc^1, rej^1)$ עם k סרטים, נוכח כי קיימת מכונה עם k סרטים, נוכח כי קיימת מכונה $M^1 = (Q^1, \Sigma, \Gamma^1, \delta^1, q_0^1, acc^1, rej^1)$.

נרצה לעשות סימולציה של המוכנה עם k הטריטים על המוכנה עם הטרט האחד. מה צריך בכל סימולציה?

1. ייצוג הקונפיגורציה של המוכנה המסומלצת ע"י או בתוך המוכנה המסומלצת (M^1)

הרעין יהיה כזה - נדגים אותו עם $k = 2$: יש לנו שני סטריטים, הנפק אוטום ל 4^k שטודרים בשורות אחת אחרי השניה. הטרט הראשון, ומתוךתו סטרט שמשומן כך - איפא שהיה הראש יסומן חז, ובכל שאר המיקומות יסומן *. השורה השלישי הטרט השלישי, והשורה הרביעית תהיה באופן זהה שורה של חז וכוכביות. באופן דומה על k סטריטים נשכפל למוכנה עם $2k$ סטריטים כשותחת לכל סטרט של * וחצים.

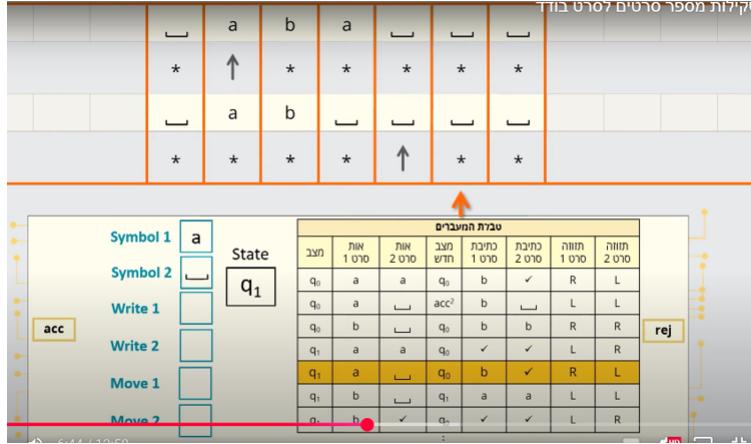
מדוע עשינו זאת? כיון שעליינו לדעת היכן נמצא הראש.

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

לשם כך נשתמש בתאי זכרון נוספים: $.symbol1, symbol2, write1, write2, Move1, Move2$. הרעיון הוא שב- $symbol1$ נשמר את הסימן שופיע בחץ העליון ובהתאם להסימן שופיע בחץ התיכון. $write2$ נשמר מה שצרכיך לכתוב בכל אחד מהחצים, $move$ נשמר לאיזה כיוון צריך להזיא את החץ. בנוסף, M_1 תשמור את כל טבלת המעברים של M_2 .

בכל סימולציה, הראש של M_1 (המוכנה עם סטרט אחד שנבנו), יהיה במצב שמשומאל למשבצת עם החץ השמאלי ביותר. נשמר תא זכרון נוסף $state$ בשם $state$ שיישמר את המצב הנוכחי של M_1 . המוכנה מבצעת סריקה פעם אחת משמאלי לימין - וממלאת את ששת התאים לעיל לפני שהיא שטמצא בטרט (כאשר היא תראה את החצים).

cut - המוכנה מחזיקה בהמה שהיא רואה בטרט הראשון ובטרט השני, אנחנו ידעים מה המצב בו אנו נמצאים - וכך cut לחשוף את השורה המתאימה בטבלת המעברים (כאן מטה, בכתום) ולדעת מה יהיה הצעד הבא של המוכנה. במקרה שלנו יכתב ✓, בטריטים ונזוז R, L בהתאם.



כאשר המוכנה סיימה סריקה ראשונה, משמאלי לימין, היא עוברת לסריקה נוספת נספחת מימין לשמאלי - כאשר היא תתקל בי המותאים, היא תכתוב ותזוז בהתאם למה ששמרנו בזיכרון הפנימי בששת התאים, ותשנה אותם בהתאם. נשים לב כי אם שיש תזוזה L למשול, אין החץ שייזו היה החץ בשורה הרביעית. החץ של M_1 זו רוק ורוק משמאלי לימין ואיז מימין לשמאלי ברצף. נuir כי כאשר כתובים" משנים את כל הרביעייה. שחרי הא"ב הוא רביעיות.

נuir כי בכל שלב בסריקה - החץ צריך להתחיל משבצת אחת משמאלי לחץ השמאלי ביותר. בכל שלב - זה חשוב. מתי המוכנה מפסיק לעבור? כשהיא משמאלי לימין - כשהיא משבצת אחת מימין אחרי החץ ימני ביותר, וכשהיא מימין לשמאלי - משבצת אחת משמאלי אחריו החץ השמאלי ביותר. בכל סיום שלב סריקה - אנחנו הולכים לקרוא בטבלת המעברים, מה עושים בהינתן מצב, ושתי אותיות קלות.

אם המצב הופך $state$ או M_1 עוברת למצב מכב. אם המצב הופך rej או M_1 עוברת למצב דחייה. מה אם היא לא מגע לא לדחיה ולא לקבלת? או M_2 כמו M_1 תמשיךנצח לנצח.

בשלב התחלתי - תמיד יהיה אתחול והפיכת המcona לכפי שתואר לעיל. ותמיד נתחל מ q_0 . את אותו התחליק שהדגמנו כאן, ניתן לבצע על k סרטים - באינדוקציה או בכל דרך אחרת.

8.6 סגירות לאיחוד וחיתוך שפות כריעות/קבילות

cut לשימוש במודל ריבוי סרטים" באמצעות טכנית של ריצה במקביל להוכחת הטענות הבאות הבא

- טענה:** יהיו L_1, L_2 שפות כריעות, אז $L_1 \cap L_2$ כריעה.
- הוכחה:** יהיו A, B שפות הcriuot והאוטומטים שמקבילים אותן בהתאם M^A, M^B . נבנה מcona M^C שתכרייע את $A \cap B$. רעיון ההוכחה יהיה להשתמש בשני סרטים, ראשית נעתק את התוכן של הסרט הראשון לסרט השני ונווי את הראש של שני הסרטים לשמאלה הקלט. הסרט הראשון יסמל את M^A והשני את M^B . נפעיל כך:
- נעתק את הקלט w לסרט השני ווחזר את הראש לתחילת הסרטים.
 - נרים את M^A על גבי הסרט הראשון, אם דחתה - דחה.
 - אם M^A קיבלה, M^C תרץ על הסרט השני את M^B - אם קיבלה: קבל. אם דחתה - דחה.
- נשים לב כי מכונה זו תמיד עוצרת, כיון ש- M^A, M^B מכrieut שפות ולכן תמיד עוצרות.

- טענה:** יהיו L_1, L_2 שפות קבילות, אז $L_1 \cap L_2$ קבילה.
- הוכחה:** יהיו A, B שפות קבילות והאוטומטים שמקבילים אותן בהתאם M^A, M^B . נבנה מכונה M^C שתקבל את B $\cap A$.
- נפעיל כך:
- נעתק את הקלט w לסרט השני ווחזר את הראש לתחילת הסרטים.
 - נרים את M^A על גבי הסרט הראשון, אם דחתה - דחה.
 - אם M^A קיבלה, M^C תרץ על הסרט השני את M^B - אם קיבלה: קבל. אם דחתה - דחה.
- נשים לב כי מכונה זו חיבת לקלט את כל המילים שבספה, וכך לכל מילה שבספה נקלט שהיא קבילה בשתי המכונות אותן. וכך גם בחיתוך. כלומר, כל מילה בשפה מתקבלת וכל מילה שאינה בשפה לא מתקבלת.

- טענה:** יהיו L_1, L_2 שפות כריעות, אז $L_1 \cup L_2$ כריעה.
- טענה:** יהיו L_1, L_2 שפות קבילות, אז $L_1 \cup L_2$ קבילה.
- (רעיון ההוכחה לשני טעפים אלו - בדיק כמוה בשתי הטענות הקודמות. נרים במקביל, אם נקלט באחת המכונות שהמילה התקבלה אז גם המילה שלנו התקבלה. אם שתי המכונות ידחו - המילה לא בשפה.).

P2 8.7 אוטומט עם שתי מחסניות

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

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

כלומר, המעבר $\Delta(q, a, \$, \varepsilon, \#) = (p, b, \$, \varepsilon, \#)$ אומר כאשר אנו במצב q וקוראים a , אם בראש המחסנית הראשונה $\$$ אפשר לעבור למצב p ובו נוציא את התו $\$$ ממהחסנית הראשונה (האפסילון מוחק אותה), ובמחסנית השנייה שרים, כתוב $\#$.

מודל זה נראה מtbody ראשון כשולחן למודל האוטומט המקורי - אך לא.

טענה: מודל P2 שקול למcona טירינג במודל T .

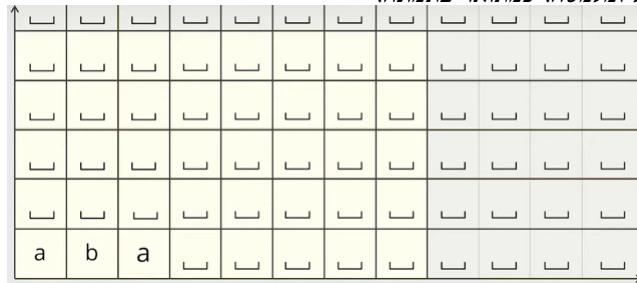
הוכחה:

צד אחד פשוט, בהינתן אוטומט מחסנית שכזה קל לדמותו אותו למcona טירינג עם שני סרטים.

נכיה כ' בהינתן מכונה 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 בתתית המחסנית. במחסנית השנייה נשמר את האותיות מהראש וימינה בסרט (למעט אינסוף הרוחים), אשר האות העליונה במחסנית השנייה עליה מצביע הראש והאות הימנית ביותר בסרט תהיה האות בתתית המחסנית. בתתית של כל מחסנית יהיה הסיכון \$. האוטומט P מתחילה את הריצה כאשר שתי המחסניות ריקות. כדי לדמות את הקונפיגורציה ההתחלתית של M דרוש שלב אתחול שבו נמלא את המחסניות מותך תוך הקלט שבסרט הקלט של P (נזכר שהאוטומט P , ע"פ ההגדרה, קורא את הקלט באופן סדרתי, ואין יכול לחזור ולקרוא שוב את האותיות שנקראו). נגידר מכך אחד שיקרא את אותיות הקלט אחת אחת, ולכל אות שנקראת מהקלט - האוטומט יכנס אותה למחסנית 1. אחר כך נעבור למצב אחר, שיביר את כל האותיות ממחסנית 1 למחסנית 2. המשך הוכחה - בקמפוס. (אני בספק שצרכי לדעת זאת בע"פ.)

8.8 סרט דו ממדי - מודל 2D

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



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

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

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

דוגמה. נגידר כוושג מתמטי, מס' x קורא משולשים אם, ורק אם קיים $\mathbb{N} \in n$ כך ש- $i = \sum_{i=0}^n i$ הוא מספר משולשי $|x|$. $L = 2D$ נרצה להכריע את L באמצעות $2D$ מהריעו? תחילה, נגידר שהקלט שלו יהיה כמספר אונארי. כל הקלט בתתילה יהיה בשורה התחרזונה של המטריצה. בולאה הוראות: בכל שלב n נעתיק את כל האחדות בשורה הקדמת, פרט להן השמאליות ביותר. ככלומר בתתילה נעתיק את כל המספרים פרט לאחד בודד, אח"כ את colum פרט לשני אחדות, שלוש אחדות וכן הלאה.... מס' x יהיה שייך לשפה אם ו רק תהיליך זה, קיבל צורת משולש במטריצה.

טענה: מודל 2D שקול למודל T המקורי.

הוכחה: נתחיל במספר הטבלה, כך שהטה השמאלי ביותר יהיה $(0, 0)$ וכן הלאה נתקדם. לאחר מכן נזכיר מייפוי חד-חד ערכי מהמטריצה לטבלה ע"י יצירת הטבלה עם התאים $(0, 0), (1, 0), (0, 1), (2, 0), (1, 1), (0, 2)$.

וכן הלאה.... כלומר - מעתיקים בכל פעם את האלכסון הבא אל המטריצה, כאשר מתחילה להעתיק אותו ממטה של כל אלכסון. כמו כן נמספר את התאים בהתאם, ככה שהתא הראשון יהיה 0, בהתאם באלכסון השני ותאים 1,2 וכן הלאה.

נרצה להתאים את התוצאות.

בהתאם ראש שמנצא בסרט הדו מmedi בתא מס' i , בשורה y ובעמודה x - איזי

$i + (x + y + 1)$ הינו

$i - (x + y)$

התא מעל התא i

$i + (x + y + 2)$

התא מתחת לתא i

$i - (x + y + 1)$

המשך החוכחה כשייה לי כוח!!!!!!

!!!!!!

!!!!!!

8.9 מבנות טיריניג שאינה דטרמיניסטיות

כידוע, במקרים מסוימים ניתן יהי להגעה עבור קלט נתון. נגידר היטב את מושג הדחיה והקבלה במודול מסווג זה:

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

הגדרה: עבור שפה $\Sigma \subseteq L$, נאמר כי:
 $w \in \Sigma$ מכריעה את L אם לכל Σ^* $w \in \Sigma$ מתקבלת את w .
 אם $w \in L$ איזי M מקבלת את w .
 אם $w \notin L$ איזי M דוחה את w .

הגדרה: M מקבלת את L אם לכל Σ^* $w \in L$ אם ומ"מ M מקבלת את w

דוגמא. נגידר משלים על מהירות $0 = \bar{1}$, $1 = \bar{0}$. וכן למשל $001101 = \overline{110010} = \overline{11}$. נגידר את השפה

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

כלומר, אוסף היטולים בהם קיימת סיפא של הטוילון, שכןור נפער לעליה את המשלים נקבע פוליאדרום.
 למשל 11000 , אס נסטכל על $00 = \bar{u}$ מתקיים $11 = \bar{v}$ ואז אנו מתקיים השוויון. נזאת לבנות אוטומט לא דטרמיניסטי שיקבל שפה זו.
 הפתרון - יהו להשתמש בטכנית המעכרים שיצרנו למכונית הטיריניג של שפת הפוליאדרום + התוספת
 כאו מטה:

PAL:	מצב	סימן	מצב חדש	כתב	תוויה	תוויה
	q_0	-	acc	ת	ת	R
	.					.
	.					.
	back	-	q_0	-	-	R
FLIP:	מצב	סימן	מצב חדש	כתב	תוויה	תוויה
	\widehat{q}_0	0,1	\widehat{q}_0	ת	ת	R
	\widehat{q}_0	0,1,-	flip	ת	ת	S
	flip	0	flip	1	1	R
	flip	1	flip	0	0	R
	flip	-	back	-	-	L
	back	0,1	back	ת	ת	L
	back	-	q_0	ת	ת	R

הຽיון - נוסף מצב חדש שיסורק את כל הקלט משמאלי לימי, נקרא לו \widehat{q}_0 , וכן מצב *flip* שি�שנה כל 0 לאחד וכל אחד לאפס. הירויו והוה להפעיל כל פעע את *flip* על חלק אחר בקלט, באופו סימטרי בו הקצאות, ואז לקרוא *LAL*, שתבזוק אס המחרוזות שהתקבל היא פילינדרום - ואם, ורק אם זה יקרה אז נלך למסב מקל. בעת, המכונה תעצור בוזאות על כל מילה ששיוכת לשפה ולא תעוצר על מילים שלא שיוכות לשפה, וכך המכונה מקבלת את השפה *L*.
השימוש במוגנות טירינג לא דטרמיניסטי שימושי במיוחד עבור קבלה של שפות.

8.9.1 הכרעה ו渴לה של שפות

עבור מכונה לא דטרמיניסטיבית *N* ושפה *L*:

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

נשים לב להבדל - במוגנות דטרמיניסטיבית ישנו היישוב יחידת אם הוא עוצר במצב מקבלת המילה מתקבלת. במוגנות לא דטרמיניסטיבית יש הרבה היישובים אפשריים - מילה מתקבלת אם באחד המסלולים היא התקבלה. ונדחית אם בכל החישובים נדחתה.

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

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

בידיינו טבלת מעברים. נשים לב שעבור אות קלט *s* ומצב *q* ייתכנו (כי לא דטרמיניסטיבית) מס' מעברים שונים, א' למשל. נסמן אותם במספרים *k*....1. בעת - ניצור מכונה עם שני סרטים. על הסרט הראשון נריץ את הקלט. בסרט השני נכתב תחילתה מס' אחדות כמס' מס' ממעברים באוטומט. בעת, נבצע סימולציה על המכונה באמצעות המספרים: תחילת, כל המספרים יהיו 1, וכן נפעל לפי אפשרויות אחד. אח"כ המצב השני יהיה 2 נניח, ונריץ את הסדרה12111111, ואח"כ 112111111 והלאה - כך נעבור על כל המסלולים האפשריים. נקבל מילה ונעצור אמ"מ באחד המסלולים הגענו במצב מקבל. סה"כ מכונה עם שני סרטים כפי שראינו בעבר שcolaה למוגנה עם סרט אחד, וכך ישנה שיקולות. נעיר כי פורמלית יש לעשות סרט נוספת, להעתיק אליו את הקלט, ושם לבצע את ה"עבודה" ובדיקת האפשרויות השונות בשביל לשומר על הקלט המקורי בslashesות.

פורמלית -

1. כתוב 1 בסדר בחירות.

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

8.10 סגירות באמצעות אידטרמיניזם

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

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

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

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

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

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

המילה, המילה תהיה ב L).

9. ייחידה 10: התזה של צראץ'-טיירינג

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

9.1 סגירות

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

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

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

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

אין סגירות למשלים ולא לrorrs.

9.2 היחס בין הכרעה לקבלה

טענה: אם שפה כrüoot, היא בהכרח קבילה.

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

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

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

9.3 מכונות טיירינג שcoleה לתוכנית מחשב

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

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

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

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

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

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

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

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

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

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

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

■

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

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

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

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

לשם כך נכתבת התוכנית הבאה:

Double(w):

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

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

9.4 דקדוקים כלליים

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

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

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

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

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

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

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

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

$$\begin{aligned} \text{כלט: } w \\ u = S . 1 \\ \text{repeat : } . 2 \\ \end{aligned}$$

פצל באופן לא דטרמיניסטי את u ל xyz
 בחר באופן לא דטרמיניסטי גזרה v $\rightarrow t$ של G

אם $y \neq t$ אם דוחה

$$u = xvz$$

אם $u == w$ קיבל

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

$$aq_0baa \vdash_M aaq_1aa$$

אזי, הרעיון יהיה להגיד את הדקדוק לפי הקוניגורציות, כלומר

$$aq_0baa \Rightarrow_G aaq_1aa$$

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

9.5 ההרכבה של חומוסקי

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

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

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

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

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

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

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

10 יחידה 11: אי כרייעות

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

10.1 אimotoות תוכנה

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

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

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

האם PS כרייעת?

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

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

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

כלומר, האם בהינתן תוכנית ומחרוזת מתקיים $1 = (w)P$ (כלומר, שנרץ את w על התוכנית, קיבל 1). זו גרסה מנוגנת מאוד של הבעיה המקורית: כיון שמתיחס רק לדבר אחד במפרט, ודורשת לעבור על כל קלט w אפשרי. אבל - זה אבל גדול, והוא תנאי הכרחי (ולא מספיק) 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 כריעה? על כך נדון-cut.

10.2 ATM קבילה

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

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

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

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

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

10.3 לא כריעה ATM

טענה: *ATM* לא כריעה.

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

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

נשים לב כי הקלט של $D - ATM$ הוא מחרוזות ותוכנית. אך שלחנו שתי מחרוזות. על פניו - זה חוקי כיון שלא שלחנו תוכנית, ולכן על הזוג הזה היא תחזיר תשובה כלשהי (אפס או אחד). הקוד של $D - ATM \subseteq Stupid$ icut, נרצה להריץ את *Stupid(Stupid)*. מה יוחזר? נשים לב שבפרט תמיד יוחזר ממשו, כיון שא תמיד מוגדר כי $D - ATM$ הינה תוכנית להכרעה. לכן תמיד נחזיר ערך כלשהו. נשים לב כי $stupid(stupid) = 1/0$ בבדק.

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

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

סה"כ - $Stupid(Stupid) \neq 0, 1$ בסתייה כי אלו שתי אפשרויות היחידות שיכولات להיות, ולכן *ATM* לא כריעה. מש"ל.

■

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

10.4 שפה שאינה קבילה

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

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

10.5 בעיית העצירה

נדיר את השפה הבאה: $\{(P, w) | P(w) \downarrow\} = HALT = \{(P, w) | P(w) \text{ halts}\}$ - אוסף כל הזוגות, כך שהריצה של P על

המחזרות w עצרת (כלומר - התוכנית לא נתקעuta). החץ מסמל עצירה.

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

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

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

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

■

טענה: $HALT$ קבילה

הוכחה: בניית תוכנית שמקבלת את $HALT$:

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

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

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

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

10.6 שפות לא פתרות

שפה שאינה כריעה וקבילה נקראת שפה לא פתרה.

E השפה 10.6.1

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

כלומר, שפת כל התוכניות כך שהשפה שלחן ריקה. למשל, E : $Q(x) : \text{while}(1) : E$ כי התוכנית לא עצרת אף קלט, ובפרט לא מקבלת אף קלט.

טענה: E לא כריעה.

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

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

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

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

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

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

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

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

טענה: E לא קבילה.

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

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

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

```
if q is not a program return 1
guess w ∈ Σ*
return (U(Q,w));
```

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

EQ השפה 10.6.2

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

אוסף כל התוכניות, כך ששפנתן זהה.

טענה: נב"ש כי EQ אינה קבילה.

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

A-E(P):

```
Q1 = P
Q2 = "Q(x){return0}";
return (A-EQ(Q1, Q2))
```

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

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

- לא קבילה גורר לא כריעה(ה).

\overline{EQ} השפה 10.6.3

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

טענה: לא קבילה. \overline{EQ}

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

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

```
Q1 =
Q2 =
return A - \overline{EQ}(Q1, Q2);
```

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

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

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

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

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

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

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

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

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

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

- לא קבילה גורר לא כריעה.

10.7 פונקציות לא חשיבות

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

דוגמא. נתבון בפונקציה הבאה:

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

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

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

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

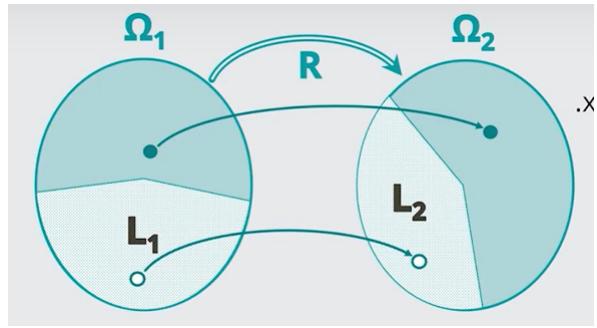
10.8 רדוקציות והוכחות ברדוקציות

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

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

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

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



סימן: אם יש רדוקציית התאמה ניתן לחישוב $L_2 \preceq_m L_1 \rightarrow$ אז נסמן $L_2 \preceq_m L_1$

טענה: תהיינה L_1, L_2 שפות. אם:

$$\text{כריעה } L_2^*$$

$$L_1 \preceq_m L_2^*$$

אז כריעה.

הוכחה: תהי $D = L_2 - L_1$ התוכנית שמכריעה את L_2 . בהינתן $x \in \Omega_1$ נרצה לדעת האם $x \in L_1$ נחישב $L_2(y) = R(x) \in L_2$. כיון שגם $R(x) \in L_2$ $\iff R(x) \in L_1$ נחישב $y = R(x) \in L_2$. נחישב $x \in L_1$.
התשובה שנחיזיר תהיה אותה התשובה על האם $x \in L_2$. **בנדרש.**

טענה: תהיינה L_1, L_2 שפות. אם:

$$\text{קבילה } L_2^*$$

$$L_1 \preceq_m L_2^*$$

אז קבילה.

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

בציד נוכחה שפה לא כריעה/קבילה? נמצאה שפה L_1 לא כריעה/קבילה, ונבנה רדוקציית התאמה ניתנת לחישוב L_1 מ- L_2 .

10.8.1 רזוקצייה משפה בריעה

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

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

$R(w)$:

if $D^A(w) == 1$:

return " $Q(x)\{return(1);\};"aba";$ ";

else

return " $Q(x)\{return(0);\};"aba";$ ";

נשים לב כי הרזוקציה מקבלת מילה, ומוריצה את המכונה על המילה. אם $w \in A$ קיבלה, אזי $D^A(w)$ מקבלת כל מילה, ולכן בשביל שהרזוקציה תתקיים נרצה להחזיר איבר כלשהו ב- ATM , החיזרנו תוכנה שמקבלת כל מילה וアイיר - שבסרט בתוכנה כי היא מקבלת כל מילה. אחרת, D^A לא קיבלה ולפניהם נרצה להחזיר זוג שלא מתאים ל- ATM - מכונה שלא מקבלת דבר, ומילה, שודאי לא שייכת אליה כי היא לא מקבלת דבר. נשים לב כי הרזוקציה חישובית ומוגדרת היטב - והיא מכירה, היא תמיד תעזר. לכן הרזוקציה R תמיד תחזיר קלט. כמו כן, ניתן לראות כי $x \in A \iff R(x) \in ATM$, כיון שאחרת לא נקלט זוג שמתאים ל- ATM . סה"כ בנו רזוקציה שתמיד עבדת וקיים.

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

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

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

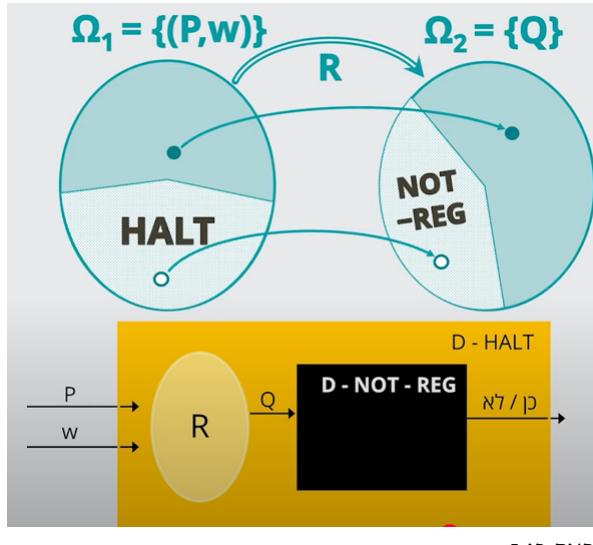
NOT – REG השפה 10.8.2

נדיר $\{Q\}$ לא רגולרית $.NOT – REG := \{Q |$ לשפה הפלינדרום, $.palindrome \in NOT – REG$ ולמשל, $startA \notin NOT – REG$ עם זאת,

טענה: $NOT – REG$ לא בריעה.

$.HALT \preccurlyeq_m NOT – REG$

נשים לב כי $(P, w) \in NOT – REG \in \Omega_2 = \{Q\}$ וכן $HALT \in \Omega_1 = (P, w)$ ונרצה כי הפונקציה $R(P, w)$ תחזיר $R(Q, w)$ כלומר, בהינתן זוג (P, w) רגולרית, מחרוזות שפותחות בו $P(w)$ רגולרית. ובתיאור -



נראה כי -

$R(P, w)$:

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

עונה על הדרוש. ראשית היא מקבלת זוג (P, w) , והיא קוראת למוכנה הווירטואלית, ואז מרים את התוכנית פלינדרום על הקלט איקס ומחרירה תשובה מתקבלת. זו רדוקציה שניתנת לחישוב, שכן מקבלים זוג מחרוזות, ומחרוזים מחרוזות בודדות.
חשוב לציין ש R לא מרצה את המחרוזת (x) . 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 תלויה במקרה יקרה בשורה $U(P, w)$. אם החישוב מסתיים, נגיע לשורה האחורונה ונחיזיר את שפת הפליינדרומים. אחרת, נעלם לא נעצור ונגע לשורת ההחזרה והשפה תהיה ריקה.
סה"כ קיבלנו מה שרצינו - $L(Q) \subseteq L(P(w))$ לא רגולרית אם $\downarrow P(w) \neq 1$ עצרת. מדוע בחרנו בשפת הפליינדרומים?
כי היא לא רגולרית, יכולנו לבחור כל אחת אחרת שאינה רגולרית וניתנת לחישוב - וזה היה עובד.

טענה: $NOT - REG$ לא קבילה.

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

נשים לב כי $(P, w) \in \overline{ATM} \in \Omega_1 = (P, w)$ וכן $NOT - REG \in \Omega_2 = \{Q\}$
כלומר, בהינתן זוג (P, w) נרצה כי הפעונקציה R תחזיר $Q = R(P, w)$ כך ש -
 $P(w) \neq 1 \iff L(Q) \subseteq NOT - REG$.
נשים לב כי אכן $\overline{ATM} \not\leq_m NOT - REG$ לא קבילה, ולכן אם נוכחים שקיימות צו לפि טענה מההרצאה - אכן $P(w) \neq 1$ לא קבילה גם היא.

אנחנו מוחשים ליצור תוכנית Q , שהשפה שלה אינה רגולרית, אם $1 \neq P(w)$. 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 המחרוזת פלינדרום, אם כן היא מוחזירה אחד. אחרת - לא פלינדרום:
מראיצה מכונה וירטואלית עם התוכנית והקלט. נשים לב כי

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

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

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

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

10.9 סיכום - השפות הלא כרייעות/קבילות

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

10.10 בעיית הנחש

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



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



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



יש מסלולים - ללא אפשרות ליצור מסלול.

פורמלית, הבעה תתואר כך:

* קבוצת אריחים (t_1, \dots, t_n)

כasher כל אריח $t_i \in \mathbb{N}^4$ כל אריח מיוצג ע"י 4 מס' טביעים שמייצגים את הצבעים

* זוג נקודות קצה $\mathbb{N} \times \mathbb{N}$ בין $p, q \in \mathbb{Z} \times \mathbb{Z}$

הבעיה היא: יש נחש תקין של אריחי T בחצי המישור העליון שמחבר בין p ל- q

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

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

נשים לב כי אם נגדיר $\{(T, p, q)|$ יש נחש תקין של אריחי T בכל המישור שמחבר בין p ל- q .

10.11 התוכנית של הילברט

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

השלישי - כריעות: למצוא אלגוריתם שקובע האם טענה היא נכונה או שלא, חיכה ליחידה הנוכחית.
ובכן, טירינג לא מצא את האלגוריתם שהילברט חיפש - טירינג הוכיח שאין אלגוריתם כזה, ולא
יכול להיות. ניתן להסביר את הדברים מנקודת מבטו של שריאנו ביחידה זו: ראיינו כי למשל *ATM* לא כריעה
- וכן אין אלגוריתם מתמטי או תכני, שידעו להכריע את הבעיה. (}):