

חישוביות וסיבוכיות

תוכן העניינים

4	1 מכונות טיורינג
4	הגדרה של מכונת טיורינג
17	טבלת המעברים
21	חישוב פונקציות
24	2 מודלים חישוביים שקולית
28	3 מכונות טיורינג מרובת סרטים
28	מכונת טיורינג מרובת סרטים: הגדרה היוריסטית
28	מכונת טיורינג מרובת סרטים: הגדרה פורמלית
29	קונפיגורציה של מכונת טיורינג מרובת סרטים
31	שקילות בין מטמ"ס למ"ט עם סרט יחיד
36	4 מכונת טיורינג אי דטרמיניסטית
36	הגדרה של מכונת טיורינג אי-דטרמיניסטית
38	עץ החישוב של מ"ט א"ד
39	שקילות בין מכונת טיורינג אי-דטרמיניסטית למכונת טיורינג דטרמיניסטית
43	5 התזה של צרף טיורינג ודקדוקים כלליים
43	היחס בין הכרעה וקבלה
44	שקילות של מכונת טיורינג ותוכנית מחשב
44	SIMPLE
50	דקדוקים כלליים
56	דקדוקים כלליים ומכונת טיורינג
57	ההיררכיה של חומסקי
57	כל שפה חסרת הקשר הינה כריעה
59	6 תכונות סגירות של R ו- RE
59	הגדרה של השפות R ו- RE
59	היחס בין הכרעה וקבלה
60	סגירות של שפות כריעות ושפות קבילות
65	סגירות של שפות כריעות תחת $DROPOUT$
66	קידוד של מ"ט דטרמיניסטית
66	מ"ט אוניברסלית U
69	7 אי-כריעות
69	השפות L_d, L_{halt}, L_{acc} לא כריעות
73	השפה L_E לא כריעה
75	השפה L_{EQ} לא כריעה
78	סיכום: כריעות וקבילות של שפות

79	8 רדוקציה
79	טבלה של רדוקציות
79	מכונת טיורינג המחשבת את פונקציה
81	רדוקציות
88	דוגמאות בשימוש של משפט הרדוקציה בין שפות משלימות (משפט 8.2)
88	דוגמאות בשימוש של משפט הרדוקציה (משפט 8.1)
95	9 מבוא לסיבוכיות זמן
95	הגדרה של סיבוכיות זמן
102	יחס בין הסיבוכיות של מכונת טיורינג עם סרט יחיד ומכונת טיורינג מרובת סרטים
	103 דטרמיניסטית אי טיורינג ומכונת דטרמיניסטית טיורינג מכונת של הסיבוכיות בין יחס
	104 P המחלקה
	107 $PATH$ בעיית
	108 $RELPRIME$ הבעיית
	110 שימושיים משפטים של *הוכחות
10	10 NP והמחלקה P המחלקה
	113 P המחלקה
	114 P ב- לבעיות דוגמאות
	114 $HAMPATH$ ההמילטוני המסלול בעיית
	115 אימות אלגוריתם
	115 NP המחלקה
	118 אי-דטרמיניסטית טיורינג למכונת NP בין הקשר
119	הקשר בין המחלקה P ו- NP
122	11 NP שלמות
122	המחלקות NPC ו- NPH
123	בעיית הספיקות
123	בעיית SAT
124	משפט קוק לויין
124	גרסאות של $kSAT$
124	בעיית $3SAT$
126	הוכחת משפט קוק לויין*
132	12 רדוקציות פולינומיאליות
132	$CLIQUE$ היא NP - שלמה
134	בעיית הקבוצה הבלתי תלויה
136	בעיית הכיסוי בקודקודים
137	הבעיית VC
138	$PARTITION$
138	רדוקציות פולינומיאליות
139	שפות NP שלמות
140	13 סיבוכיות מקום ושלמות ב PSPACE
140	הגדרה של סיבוכיות מקום
143	משפט סביץ'
145	המחלקה $PSPACE$
146	שלמות ב- $PSPACE$
147	נוסחאות בוליאניות עם כמתים
152	המחלקה L
152	המחלקה NL

152	שלמות ב- NL
152	שיויון NL ו- coNL

שיעור 1

מכונות טיורינג

1.1 הגדרה של מכונת טיורינג

הגדרה 1.1 מכונת טיורינג (הגדרה היוריסטית)

הקלט והסרט

- מכונת טיורינג (מ"ט) קורא קלט.
- הקלט עצמו נמצא על סרט אינסופי מחולק למשבצות.
- כל תו של הקלט כתוב במשבצת אחת של הסרט.
- במכונת טיורינג אנחנו מניחים שהסרט אינסופי לשני הכיוונים.
- * משמאל לתחילת הקלט יש רצף אינסופי של תווי רווח " ".
- * מימין לסוף הקלט יש רצף אינסופי של תווי רווח " ".

...	␣	␣	a	b	b	b	a	a	␣	␣	␣	...
-----	---	---	---	---	---	---	---	---	---	---	---	-----

הראש

- במצב ההתחלתי הראש בקצה השמאלי של הקלט.

...	␣	␣	a	b	b	b	a	a	␣	␣	␣	...
			↑									

- הראש יכול לזוז ימינה על הסרט וגם שמאלה על הסרט.
- הראש קורא את התוכן של המשבצת שבה הוא נמצא.
- הראש יכול לכתוב על משבצת, אבל רק על המשבצת שבה הראש נמצא.

תאור העבודה של המכונה

- בתחילת הריצה, הקלט כתוב התחילת הסרט כאשר מימינו נמצא רצף אינסופי של תווי ␣ -ים.
- הראש מצביע על התא הראשון בסרט והמכונה נמצאת במצב התחלתי q_0 .

q_0	...	␣	␣	a	b	b	b	a	a	␣	␣	␣	...
				↑									

- בכל צעד חישוב, בהתאם למצב הנוכחי ולאות שמתחת לראש (התו הנקרא), המכונה מחליטה:
 - * לאיזה מצב לעבור
 - * מה לכתוב מתחת לראש (התו הנכתב)
 - * לאן להזיז את הראש (תא אחד ימינה, או תא אחד שמאלה, או להישאר במקום).
- למכונה ישנם שני מצבים מיוחדים:
 - * q_{acc} : אם במשך הריצה המכונה מגיעה ל- q_{acc} היא עוברת ומקבלת.
 - * q_{rej} : אם במשך הריצה המכונה מגיעה ל- q_{rej} היא עוברת ודוחה.
 - * אם המכונה לא מגיעה ל- q_{acc} או q_{rej} היא תמשיך לרוץ לנצח.

הגדרה 1.2 מכונת טיורינג

מכונת טיורינג (מ"ט) היא שביעה

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$$

כאשר:

Q	קבוצת מצבים סופית ולא ריקה
Σ	אלפבית הקלט
Γ	אלפבית הסרט
δ	פונקציית המעברים
q_0	מצב התחלתי
q_{acc}	מצב מקבל יחיד
q_{rej}	מצב דוחה יחיד

$$_ \notin \Sigma$$

$$\Sigma \subseteq \Gamma, _ \in \Gamma$$

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

1.1 דוגמה

נבנה מכונת טיורינג אשר מקבלת מילה אם היא בשפה

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

ז"א השפת כל המילים עם מספר שווה אותיות a ו b . הפאודו-קוד של המכונה, כדלקמן.פסאודו-קוד

- (1) סורקים את הקלט משמאל לימין.
 - אם לא מצאנו a וגם לא מצאנו $b \Leftarrow$ מקבלת.
 - אם האות הראשונה שהראש מצא היא a , כותבים עליו \checkmark , ועוברים לשלב (2).
 - אם האות הראשונה שהראש מצא היא b , כותבים עליו \checkmark , ועוברים לשלב (3).
- (2) ממשיכים לזוז ימינה עד שנמצא b תואם.
 - אם לא מצאנו $b \Leftarrow$ דוחה.
 - אם מצאנו b כותבים עליו \checkmark , חוזרים לתחילת הקלט וחוזרים לשלב (1).
- (3) ממשיכים לזוז ימינה עד שנמצא a תואם.
 - אם לא מצאנו $a \Leftarrow$ דוחה.
 - אם מצאנו a כותבים עליו \checkmark , חוזרים לתחילת הקלט וחוזרים לשלב (1).

כעת נתן הגדרה פורמלית של המכונת טיורינג שמבצעת את האלגוריתם הזה.

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$$

כאשר Q הקבוצת המצבנים הבאה:

$$Q = \{q_0, q_a, q_b, q_{back}, q_{rej}, q_{acc}\}.$$

המשמעותם של כל המצבים נרשמים בטבלה למטה:

q_0	המצב ההתחלתי. אליו נחזור אחרי כל סבב התאמה של זוג אותיות.
q_a	מצב שבו ראינו a ומחפשים b תואם.
q_b	מצב שבו ראינו b ומחפשים a תואם.
q_{back}	מצב שנשתמש בו כדי לחזור לקצה השמאלי של הקלט ולהתחיל את הסריקה הבאה (סבב ההתאמה הבא).
q_{acc}	מצב מקבל.
q_{rej}	מצב דוחה.

האלפבית של הקלט, Σ , והלפבית של הסרט, Γ , הינן:

$$\Sigma = \{a, b\}, \quad \Gamma = \{a, b, \sqcup, \checkmark\}.$$

הפונקציית המעברים $\delta : Q \times \Sigma \rightarrow Q \times \Gamma \times \{L, R\}$ היא מוגדרת כדלקמן.

$$\delta(q_0, a) = (q_a, \checkmark, R),$$

$$\delta(q_0, b) = (q_b, \checkmark, R),$$

$$\delta(q_0, \sqcup) = (q_{acc}, \sqcup, R),$$

$$\delta(q_a, \checkmark) = (q_a, \checkmark, R),$$

$$\delta(q_a, a) = (q_a, a, R),$$

$$\delta(q_a, b) = (q_{back}, \checkmark, L),$$

$$\delta(q_b, \checkmark) = (q_b, \checkmark, R),$$

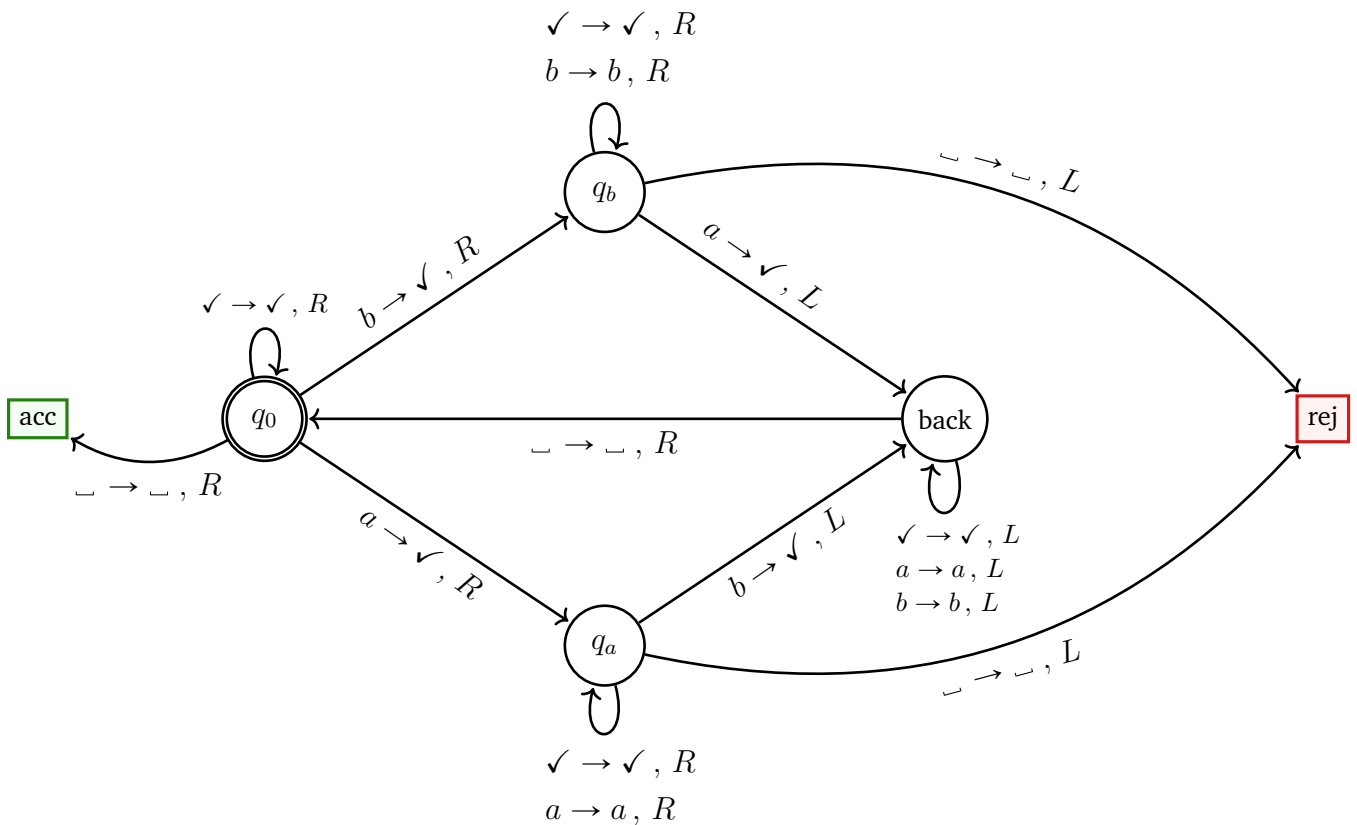
$$\delta(q_b, b) = (q_a, b, R),$$

$$\delta(q_b, a) = (q_{back}, \checkmark, L).$$

לעתים קל יותר לרשום את פונקציית המעברים δ כטבלה:

$Q \backslash \Gamma$	a	b	\sqcup	\checkmark
q_0	(q_a, \checkmark, R)	(q_b, \checkmark, R)	(q_{acc}, \sqcup, R)	(q_0, \checkmark, R)
q_a	(q_a, a, R)	$(q_{back}, \checkmark, L)$	(q_{rej}, \sqcup, L)	(q_a, \checkmark, R)
q_b	$(q_{back}, \checkmark, L)$	(q_b, b, R)	(q_{rej}, \sqcup, L)	(q_b, \checkmark, R)
q_{back}	(q_{back}, a, L)	(q_{back}, b, L)	(q_0, \sqcup, R)	$(q_{back}, \checkmark, L)$

תרשים מצבים



1.2 דוגמה

בדקו אם המכונת טיורינג של הדוגמה 1.1 מקבלת את המילה .aab

פתרון:

⌊	q_0	a	a	b	⌊
⌊	✓	q_a	a	b	⌊
⌊	✓	a	q_a	b	⌊
⌊	✓	q_{back}	a	✓	⌊
⌊	q_{back}	✓	a	✓	⌊
q_{back}	⌊	✓	a	✓	⌊
⌊	q_0	✓	a	✓	⌊
⌊	✓	q_0	a	✓	⌊
⌊	✓	✓	q_a	✓	⌊
⌊	✓	✓	✓	q_a	⌊
⌊	✓	✓	rej	✓	⌊

1.3 דוגמה

בדקו אם המכונת טיורינג של הדוגמה 1.1 מקבלת את המילה .abbbbaa

פתרון:

⌊	q_0	a	b	b	b	a	a	⌊
⌊	✓	q_a	b	b	b	a	a	⌊
⌊	q_{back}	✓	✓	b	b	a	a	⌊
q_{back}	⌊	✓	✓	b	b	a	a	⌊
⌊	q_0	✓	✓	b	b	a	a	⌊
⌊	✓	q_0	✓	b	b	a	a	⌊
⌊	✓	✓	q_0	b	b	a	a	⌊
⌊	✓	✓	✓	q_b	b	a	a	⌊
⌊	✓	✓	✓	b	q_b	a	a	⌊
⌊	✓	✓	✓	q_{back}	b	✓	a	⌊
⌊	✓	✓	q_{back}	✓	b	✓	a	⌊
⌊	✓	q_{back}	✓	✓	b	✓	a	⌊
q_{back}	⌊	✓	✓	✓	b	✓	a	⌊
⌊	q_0	✓	✓	✓	b	✓	a	⌊
⌊	✓	q_0	✓	✓	b	✓	a	⌊
⌊	✓	✓	q_0	✓	b	✓	a	⌊
⌊	✓	✓	✓	q_0	b	✓	a	⌊
⌊	✓	✓	✓	✓	q_b	✓	a	⌊
⌊	✓	✓	✓	✓	✓	q_b	a	⌊
⌊	✓	✓	✓	✓	q_{back}	✓	✓	⌊
⌊	✓	✓	✓	q_{back}	✓	✓	✓	⌊

⊥	✓	✓	q_{back}	✓	✓	✓	✓	⊥
⊥	✓	q_{back}	✓	✓	✓	✓	✓	⊥
⊥	q_{back}	✓	✓	✓	✓	✓	✓	⊥
q_{back}	⊥	✓	✓	✓	✓	✓	✓	⊥
⊥	q_0	✓	✓	✓	✓	✓	✓	⊥
⊥	✓	q_0	✓	✓	✓	✓	✓	⊥
⊥	✓	✓	q_0	✓	✓	✓	✓	⊥
⊥	✓	✓	✓	q_0	✓	✓	✓	⊥
⊥	✓	✓	✓	✓	q_0	✓	✓	⊥
⊥	✓	✓	✓	✓	✓	q_0	✓	⊥
⊥	✓	✓	✓	✓	✓	✓	q_0	⊥
⊥	✓	✓	✓	✓	✓	✓	⊥	q_{acc}

הגדרה 1.3 קונפיגורציה

תהי $M = (Q, q_0, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ מכונת טיורינג. קונפיגורציה של M הינה מחרוזת

$$uq\sigma v$$

כאשר משמעות:

$$u, v \in \Gamma^*, \quad \sigma \in \Gamma, \quad q \in Q.$$

q מצב המכונה,
 σ הסימון במיקום הראש
 u תוכן הסרט משמאל לראש,
 v תוכן הסרט מימין לראש.

דוגמה 1.4 (המשך של דוגמה 1.2)

u	q	σ	v
⊥	q_0	a	a b ⊥
⊥ ✓	q_a	a	b ⊥
⊥ ✓ a	q_a	b	⊥
⊥ ✓	q_{back}	a	✓ ⊥
⊥	q_{back}	✓	a ✓ ⊥
⊥	q_{back}	⊥	✓ a ✓ ⊥
⊥	q_0	✓	a ✓ ⊥
⊥ ✓	q_0	a	✓ ⊥
⊥ ✓ ✓	q_a	✓	⊥
⊥ ✓ ✓ ✓	q_a	⊥	⊥
⊥ ✓ ✓	q_{rej}	✓	⊥

דוגמה 1.5

בנו מכונת טיורינג אשר מקבלת כל מילה בשפה

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

ז"א מילים בעלי מספר אותיות a אשר חזקה של 2.

פתרון:

ראשית נשים לב למשפט הבא:

משפט 1.1

מספר שלם n שווה לחזקה אי-שלילית של 2, כלומר $n = 2^k$ ($k \geq 0$) אם ורק אם קיים שלם m עבורו חילוק של n ב-2 בדיוק m פעמים נותן 1.

הוכחה:

כיוון \Leftarrow

אם $n = 2^k$ ($k \geq 0$) אז $\frac{n}{2^k} = 1$.

כיוון \Rightarrow

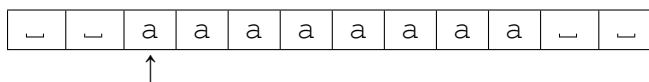
■ אם קיים $m \geq 0$ עבורו $\frac{n}{2^m} = 1$ אז $n = 2^m$ ולכן n שווה לחזקה אי-שלילית של 2.

לאור המשפט הזה נבנה אלגוריתם אשר מחלק את מספר האותיות במילה ב-2 שוב ושוב בצורה איטרטיבית.

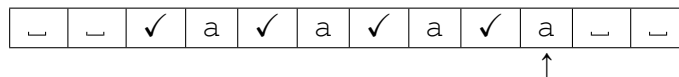
- אם אחרי סיבוב מסויים נקבל מספר אי-זוגי שונה מ-1, אז אין מצב שמספר האותיות a הוא חזקה של 2.
- בצד שני אם אחרי סיבוב כלשהו נקבל בדיוק a אחת הנשארת, ז"א אחרי מספר מסוים של חילוקים של המספר אותיות a קיבלנו 1, אזי מובטח לנו שהמספר של אותיות a הוא שווה לחזקה של 2.

כעת נסביר כיצד המכונת טיורינג מבצעת את השיטה הזאת בפועל כדלקמן.

(1) במצב ההתחלתי יש מחרוזת של רצף אותיות a כתובה על הסרט והראש נמצא מתחת האות הראשונה.



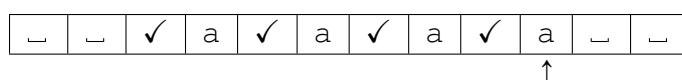
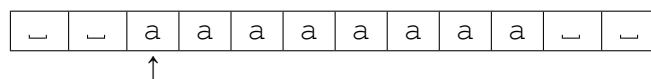
(2) עוברים על הקלט משמאל לימין ומבצעים מחיקה לסירוגין של האות a . כלומר, אות אחת נמחק ואות אחת נשאיר וכן הלאה, עד שמגיעים לקצה הימין של המילה.



(3) אחרי שהראש הגיע לסוף המילה:

- אם מצאנו אות a אחת בדיוק \Leftarrow המכונה תקבל.
- אם כתוב ✓ בתו האחרון \Leftarrow המכונה תדחה.
- אחרת, אם כתוב a בתו האחרון הראש חוזר לתחילת המחרוזת וחוזרים לשלב (2).

כדוגמה של מילה המתקבלת על ידי האלגוריתם, למטה רשומות האיטרציות של האלגוריתם הזה על המילה $w = aaaaaaaaaa$ (8 אותיות a). במצב ההתחלתי הסרט נראה כדלקמן.



(איטרציה 1) לבסוף האיטרציה $i = 1$ הסרט נראה כך:

התו האחרון a אז ממשיכים לאיטרציה הבאה.



איטרציה 2 בסוף האיטרציה $i = 2$ הסרט נראה כך:

התו הראשון הוא a אז ממשיכים לאיטרציה הבאה.

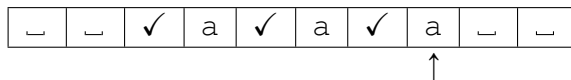
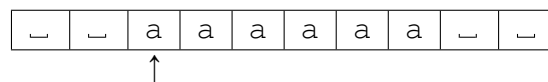


איטרציה 3 לאחר האיטרציה $i = 3$ הסרט נראה כך:

התו האחרון הוא a אז ממשיכים לאיטרציה הבאה.

איטרציה 4 באיטרציה $i = 4$ יש אות a אחת בדיוק אז המכונה מקבלת.

כדוגמה של מילה הלא המתקבלת על ידי האלגוריתם, למטה רשומות האיטרציות של האלגוריתם הזה על המילה $w = aaaaaa$ (6 אותיות a).
במצב ההתחלתי הסרט נראה כדלקמן.



איטרציה 1 לבסוף האיטרציה $i = 1$ הסרט נראה כך:

התו האחרון a אז ממשיכים לאיטרציה הבאה.



איטרציה 2 בסוף האיטרציה $i = 2$ הסרט נראה כך:

התו הראשון הוא ✓ אז דוחה.

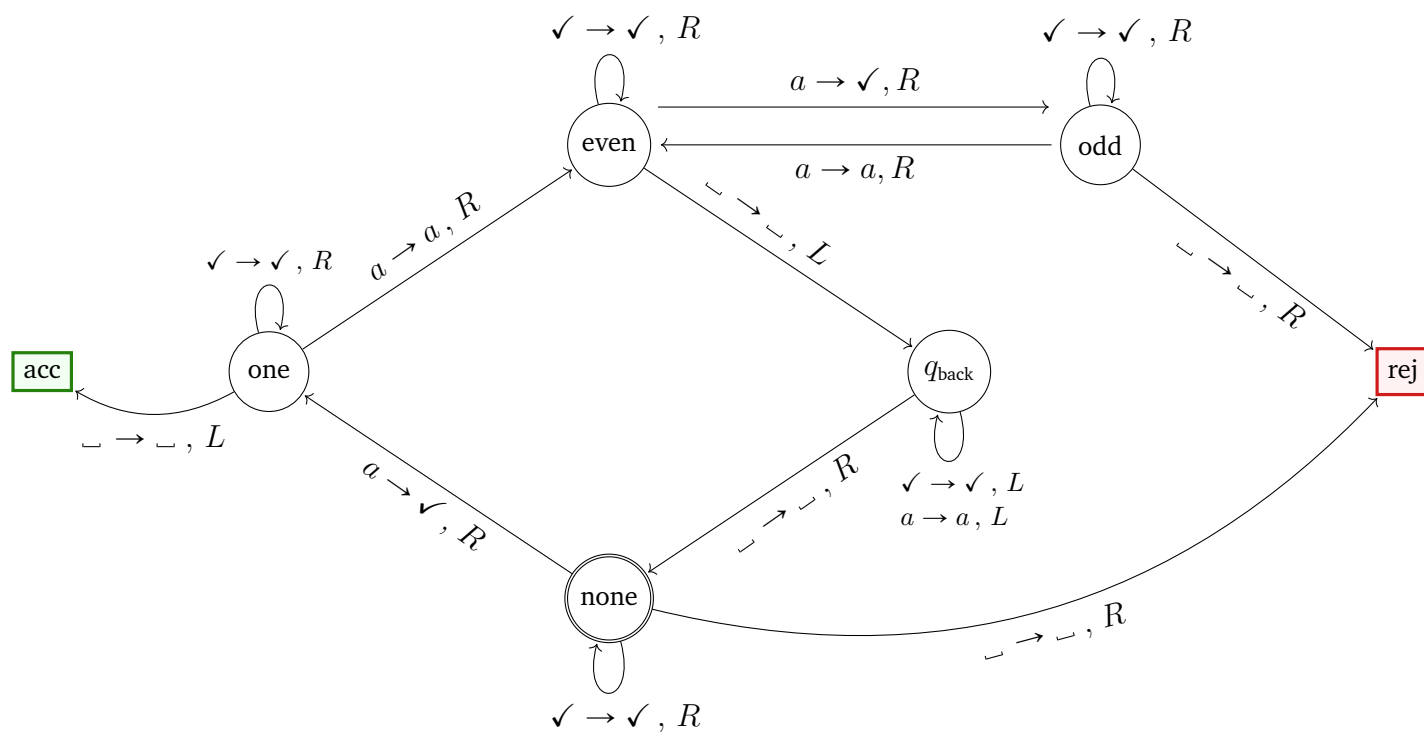
כעת נתן הגדרה פורמלית של המכונת טיורנג שמקבלת השפה הזאת:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej}) ,$$

כאשר $\Sigma = \{a\}$, $\Gamma = \{a, _, \checkmark\}$, והקבוצת המצבים היא $Q = \{q_0, one, even, odd, q_{acc}, q_{rej}\}$ כאשר המשמעותם הם מפורטים למטה:

מצב none:	מצב התחלתי. עדיין לא קראנו a בסבב סריקה זה.
מצב one:	קראנו a בודד.
מצב even:	קראנו מספר זוגי של a .
מצב odd:	קראנו מספר אי-זוגי של a .
מצב q_{back} :	חזרה שלמאלה.
מצבים למטה:	

הפונקציות המעברים מתוארות על ידי התרשים



דוגמה 1.6

בדקו אם המילה aaaa מתקבלת על ידי המכונת טיורינג בדוגמה 1.5.

פתרון:

$_$	none	a	a	a	a	$_$
$_$	\checkmark	one	a	a	a	$_$
$_$	\checkmark	a	even	a	a	$_$
$_$	\checkmark	a	\checkmark	odd	a	$_$
$_$	\checkmark	a	\checkmark	a	even	$_$
$_$	\checkmark	a	\checkmark	back	a	$_$
$_$	\checkmark	a	back	\checkmark	a	$_$
$_$	\checkmark	back	a	\checkmark	a	$_$
$_$	back	\checkmark	a	\checkmark	a	$_$
back	$_$	\checkmark	a	\checkmark	a	$_$
$_$	none	\checkmark	a	\checkmark	a	$_$
$_$	\checkmark	none	a	\checkmark	a	$_$
$_$	\checkmark	\checkmark	one	\checkmark	a	$_$
$_$	\checkmark	\checkmark	\checkmark	one	a	$_$
$_$	\checkmark	\checkmark	\checkmark	a	even	$_$
$_$	\checkmark	\checkmark	\checkmark	back	a	$_$
$_$	\checkmark	\checkmark	back	\checkmark	a	$_$
$_$	\checkmark	back	\checkmark	\checkmark	a	$_$
$_$	back	\checkmark	\checkmark	\checkmark	a	$_$
back	$_$	\checkmark	\checkmark	\checkmark	a	$_$
$_$	none	\checkmark	\checkmark	\checkmark	a	$_$
$_$	\checkmark	none	\checkmark	\checkmark	a	$_$
$_$	\checkmark	\checkmark	none	\checkmark	a	$_$

⌊	✓	✓	✓	none	a	⌊
⌊	✓	✓	✓	✓	one	⌊
⌊	✓	✓	✓	acc	✓	⌊

u	q	σ	v
⌊	none	a	aaa ⌊
⌊ ✓	one	a	aa ⌊
⌊ ✓ a	even	a	a ⌊
⌊ ✓ a ✓	odd	a	⌊
⌊ ✓ a ✓ a	even	⌊	⌊
⌊ ✓ a ✓	back	a	⌊
⌊ ✓ a	back	✓	a ⌊
⌊ ✓	back	a	✓ a ⌊
⌊	back	✓	a ✓ a ⌊
⌊	back	⌊	✓ a ✓ a ⌊
⌊	none	✓	a ✓ a ⌊
⌊ ✓	none	a	✓ a ⌊
⌊ ✓ ✓	one	✓	a ⌊
⌊ ✓ ✓ ✓	one	a	⌊
⌊ ✓ ✓ ✓ a	even	⌊	⌊
⌊ ✓ ✓ ✓	back	a	⌊
⌊ ✓ ✓	back	✓ a	⌊
⌊ ✓	back	✓	✓ a ⌊
⌊	back	✓	✓ ✓ a ⌊
⌊	back	⌊	✓ ✓ ✓ a ⌊
⌊	none	✓	✓ ✓ a ⌊
⌊ ✓	none	✓	✓ a ⌊
⌊ ✓ ✓	none	✓	a ⌊
⌊ ✓ ✓ ✓	none	a	⌊
⌊ ✓ ✓ ✓ ✓	one	⌊	⌊
⌊ ✓ ✓ ✓	acc	✓	⌊

דוגמה 1.7

בדקו אם המילה aaa מתקבלת על ידי המכונת טיורינג בדוגמה 1.5.

פתרון:

⌊	none	a	a	a	⌊
⌊	✓	one	a	a	⌊
⌊	✓	a	even	a	⌊
⌊	✓	a	✓	odd	⌊
⌊	✓	a	✓	⌊	rej

u	q	σ	v
⌊	none	a	aa ⌊

␣ ✓	one	a	a ␣
␣ ✓ a	even	a	␣
␣ ✓ a ✓	odd	␣	␣
␣ ✓ a ✓ ␣	rej	␣	␣

דוגמה 1.8

מהי השפה של המכונה למטה:



פתרון:

(1) סורקים את הקלט משמאל לימין.

- אם התו הנקרא a או b עוברים לתו ימינה הבא וחוזרים לשלב (1).
- אם התו הנקרא $_$ אז הגענו לסוף הקלט, ועוברים לשלב (2).

(2) עוברים שמאלה לתו הארון של המילה.

- אם התו הנקרא $a \Leftarrow$ מקבל.
- אחרת דוחה.

לכן המכונה מקבלת שפת המילים המסתיימות באות a .

דוגמה 1.9

מהי השפה של המכונה למטה:



פתרון:

(1) במצב ההתחלתי:

- אם התו הנקרא $_$ \Leftarrow מקבל.
- אם התו הנקרא a מורידים אותו על ידי $_$ ועוברים לשלב (2).
- אחרת \Leftarrow דוחה.

(2) עוברים ימינה עד שמגיעים לסוף המילה.

- אם התו האחרון הוא b , מורידים אותו על ידי $_$, חוזרים לתחילת המילה וחוזרים לשלב (1).
- אחרת דוחה.

בכל איטרציה המכונה מורידה תו a בתחילת המילה וחוזרת ומורידה תו b תואם בסוף המילה. בכל איטרציה אם המכונה לא מוצאת b תואם בסוף המילה היא דוחה המילה. אחרת אם המכונה לא דחתה המילה וכל האותיות נמחקות אז המילה מתקבלת. לכן המכונה מקבלת שפת המילים

$$\{a^n b^n \mid n \geq 0\}.$$

הגדרה 1.4 גרירה בצעד אחד

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ מכונת טיורינג, ותהייה c_1 ו- c_2 קונפיגורציות של M .
נסמן

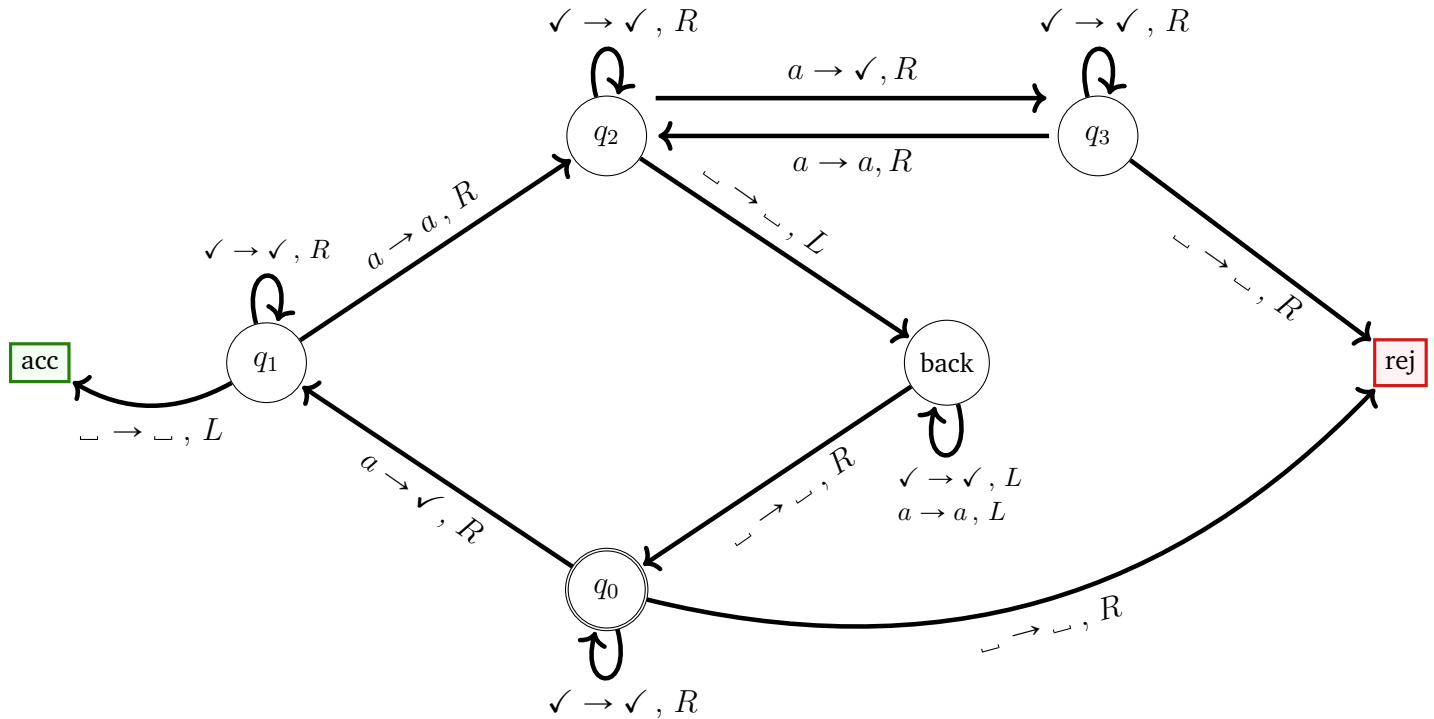
$$c_1 \vdash_M c_2$$

(במילים, c_1 גורר את c_2) אם כשנמצאים ב- c_1 עוברים ל- c_2 בצעד בודד.

דוגמה 1.10 (המשך של דוגמה 1.5)

עבור המכונת טיורינג שמתוארת בתרשים למטה מתקיים

$$\checkmark q_0 a \checkmark a \vdash_M \checkmark \checkmark q_1 \checkmark a$$



הגדרה 1.5 גרירה בכללי

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ מכונת טיורינג, ותהי c_1 ו- c_2 קונפיגורציות של M .
נסמן

$$c_1 \vdash_M^* c_2$$

אם ניתן לעבור מ- c_1 ל- c_2 ב-0 או יותר צעדים.

דוגמה 1.11 (המשך של דוגמה 1.5)

עבור המכונת טיורינג שמתוארת בתרשים למטה מתקיים

$$\checkmark q_0 a \checkmark a \vdash_M^* \checkmark \checkmark \checkmark q_4 a$$

בגלל ש:

$$\checkmark q_0 a \checkmark a \vdash_M \checkmark \checkmark q_1 \checkmark a$$

$$\vdash_M \checkmark \checkmark \checkmark q_1 a$$

$$\vdash_M \checkmark \checkmark \checkmark a q_2 _$$

$$\vdash_M \checkmark \checkmark \checkmark q_4 a .$$



הגדרה 1.6 קבלה ודחייה של מחרוזת

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ מכונת טיורינג, ו- $w \in \Sigma^*$ מחרוזת. אומרים כי:

• M מקבלת את w אם

$$q_0 w \vdash_M^* u q_{acc} \sigma v$$

כאשר $u, v \in \Gamma^*$, $\sigma \in \Gamma$ כלשהם.

• M דוחה את w אם

$$q_0 w \vdash_M^* u q_{rej} \sigma v$$

כאשר $u, v \in \Gamma^*$, $\sigma \in \Gamma$ כלשהם.

הגדרה 1.7 הכרעה של שפה

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, acc, q_{rej})$ מכונת טיורינג, ו- $L \subseteq \Sigma^*$ שפה. אומרים כי M מכריעה את L אם לכל $w \in \Sigma^*$ מתקיים:

• $M \Leftarrow w \in L$ מקבלת את w .

• $M \Leftarrow w \notin L$ דוחה את w .

הגדרה 1.8 קבלה של שפה

תהי $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ מכונת טיורינג, ו- $L \subseteq \Sigma^*$ שפה. אומרים כי M מקבלת את L אם לכל $w \in \Sigma^*$ מתקיים:

• אם $w \in L$ אז M מקבלת את w .

• אם $w \notin L$ אז M לא מקבלת את w .

במקרה כזה כאשר M מקבלת את השפה L , נכתוב ש-

$$L(M) = L.$$

1.2 טבלת המעברים

1.12 דוגמה

בנו מכונת טיורינג שמכריעה את השפה

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

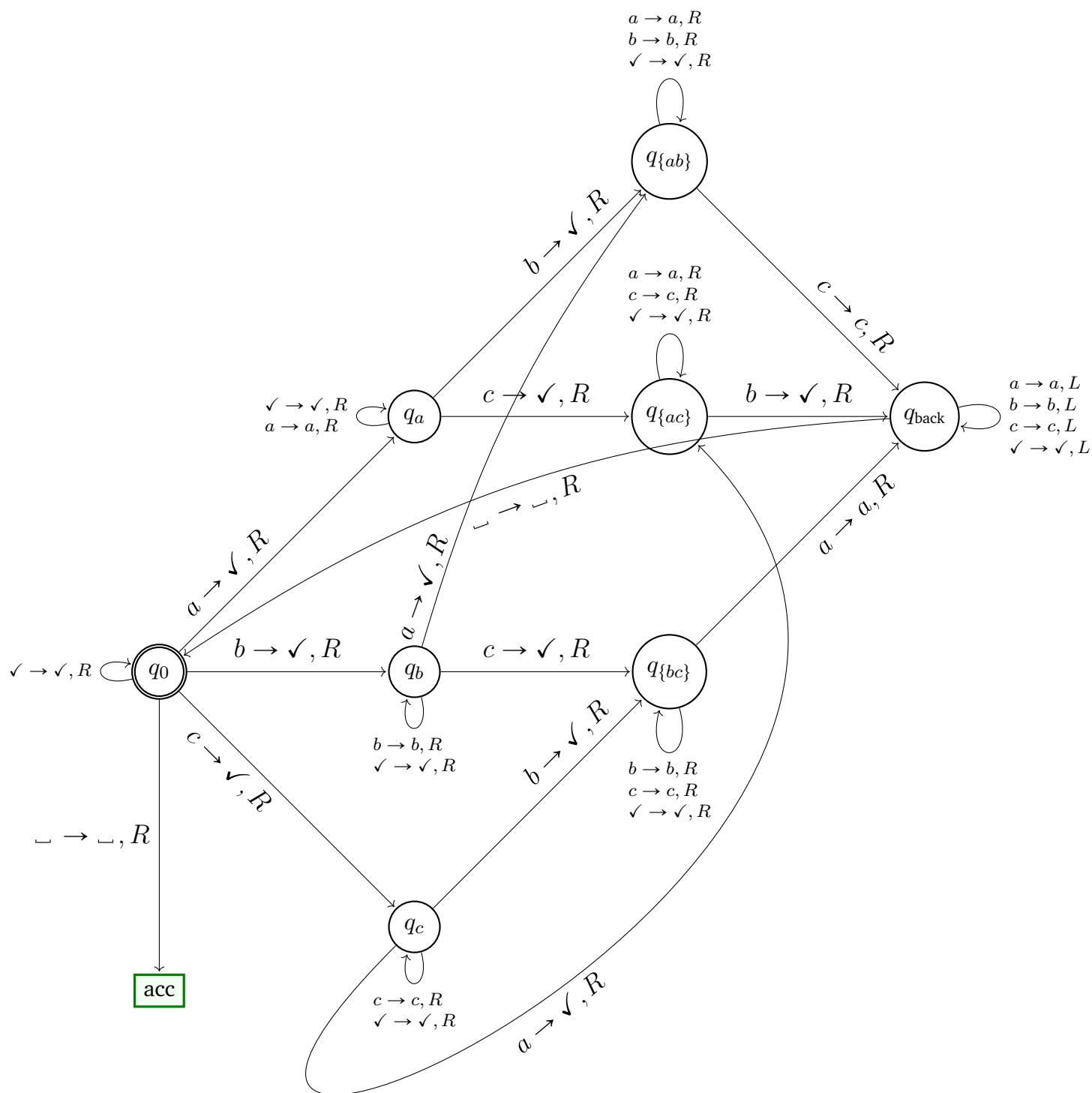
פתרון:

נתאר את המכונה על ידי הטבלת המעברים של המכונה. הסימן S מסמן כל זוג אותיות שונות מהקבוצה $\{a, b, c\}$ ללא חשיבות לסדר. כלומר:

$$S = \{a, b\}, \quad S = \{b, c\}, \quad S = \{a, c\}.$$

מצי	סימון בסרט	מצב חדש	כתיבה	תזוזה	תנאי
q_0	σ	$q.\sigma$	✓	R	$\sigma \in \{a, b, c\}$
$q.\sigma$	σ	$q.\sigma$	↻	R	$\sigma \in \{a, b, c\}$
$q.\sigma$	τ	$q.\{\sigma\tau\}$	✓	R	$\sigma, \tau \in \{a, b, c\} \wedge \sigma \neq \tau$
$q.S$	σ	$q.S$	σ	R	$\sigma \in S$
qS	σ	q_{back}	✓	L	$\sigma \notin S$
q_{back}	a, b, c, \checkmark	q_{back}	↻	L	
q_0	␣	q_{acc}	↻	R	
q_{back}	a, b, c, \checkmark	q_{back}	↻	L	
q_{back}	␣	q_0	↻	R	

כעת נתאר את המכונה על ידי תרשים המצבים של המכונה:



דוגמה 1.13

בנו מכונת טיורינג שמכריעה את השפה

$$\{x_1 \dots x_k \# y_1 \dots y_k \# z_1 \dots z_k \mid x_i, y_i, z_i \in \{0, 1, 2\}, \forall i, x_i \geq z_i \geq y_i\}$$

פתרון:

מצב	סימון בסרט	מצב חדש	כתיבה	תזוזה	תנאי
$X **$	σ	$X\sigma*$	\checkmark	R	
$X **$	\checkmark	$X **$	\checkmark	R	
$X\sigma*$	$0, 1, 2, \checkmark$	$X\sigma*$	\cap	R	
$X\tau*$	$\#$	$Y\tau*$	\cap	R	
$Y\tau*$	σ	$Y\tau\sigma$	\cap	R	
$Y\tau*$	\checkmark	$Y\tau*$	\cap	R	
$Y\tau\sigma$	$0, 1, 2, \checkmark$	$Y\tau\sigma$	\cap	R	
$Y\tau_1\tau_2$	$\#$	$Z\tau_1\tau_2$	\cap	R	
$Z\tau_1\tau_2$	\checkmark	$Z\tau_1\tau_2$	\cap	R	
$Z\tau_1\tau_2$	σ	q_{back}	\checkmark	L	
$Z **$	\sqsubset	q_{acc}	\cap	R	$\tau_1 \geq \sigma \geq \tau_2$
q_{back}	$0, 1, 2, \checkmark$	q_{back}	\cap	L	
q_{back}	\sqsubset	$X **$	\cap	R	



1.3 חישוב פונקציות

הגדרה 1.9 מכונת טיורינג שמחשבת פונקציה f

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

- $\Sigma_2 \subset \Gamma$ ו- $\Sigma = \Sigma_1$.
- לכל $w \in \Sigma_1^*$ מתקיים $q_0 w \vdash q_{acc} f(w)$.

דוגמה 1.14 חיבור אונרי

בנו מכונת טיורינג אשר מקבלת את הקלט

$$1^i \# 1^j$$

ומחזירה את פלט

$$1^{i+j}.$$

פתרון:



דוגמה 1.15 כפל אונרי

בנו מכונת טיורינג אשר מקבלת את הקלט

$$1^i \# 1^j$$

ומחזירה את פלט

$$1^{i \cdot j}.$$

פתרון:

- לדוגמה, נניח שהקלט הוא 2 כפול 2. הקלט הוא 11#11.

- נרצה להבדיל בין הקלט לבין הפלט.
לכן בתחילת הריצה, נתקדם ימינה עד סוף הקלט ונוסיף שם את התו \$.
לאחר מכן נחזור לתחילת הקלט.
- על כל אות 1 במילה השמאלית נעתיק את המילה הימינית לאחר סימן ה- \$.
- לאחר מכן נשאיר רק את התווים שלאחר סימן ה- \$. כלומר, נמחק את כל מה שאינו פלט.



μ	q	σ	ν
$_$	q_0	1	1#11 $_$
$_11\#11$	q_1	$_$	$_$
$_11\#11$	q_1	\$	$_$
$_$	q_1	$_$	11#11\$
$_$	q_2	1	1#11\$
$_ _$	q_3	1	#11\$
$_ _1\#$	q_4	1	1\$
$_ _1\#\checkmark$	q_5	1	\$
$_ _1\#\checkmark1\$$	q_5	$_$	$_$
$_ _1\#\checkmark1\$1$	q_6	$_$	$_$
$_ _1\#$	q_6	\checkmark	1\$1 $_$
$_ _1\#\checkmark$	q_4	1	\$1 $_$
$_ _1\#\checkmark\checkmark$	q_5	\$	1 $_$
$_ _1\#\checkmark\checkmark\1	q_5	$_$	$_$
$_ _1\#\checkmark\checkmark\11	q_6	$_$	$_$

_ _1#✓	q_6	✓	\$11_
_ _1#✓✓	q_4	\$	11_
_ _1#✓	back	✓	\$11_
_	back	_	1#11\$11_
_ _	q_2	1	#11\$11_
_ _ _	q_3	#	11\$11_
_ _ _#	q_4	1	1\$11_
_ _ _#✓	q_5	1	\$11_
_ _ _#✓1\$11	q_5	_	_
_ _ _#✓1\$111	q_6	_	_
_ _ _#	q_6	✓	1\$111_
_ _ _#✓	q_4	1	\$111_
_ _ _#✓✓	q_5	\$	111_
_ _ _#✓✓\$111	q_5	_	_
_ _ _#✓✓\$1111	q_6	_	_
_ _ _#✓	q_4	✓	\$1111
_ _ _#✓✓	q_4	\$	1111
_ _ _#✓	back	✓\$	1111
_ _ _	back	_	#11\$1111
_ _ _ _	q_2	#	11\$1111
_ _ _ _ _	q_7	1	1\$1111
_ _ _ _ _ _ _	q_7	\$	1111
_ _ _ _ _ _ _ _ _	acc	1	111

שיעור 2

מודלים חשובים שקולית

הגדרה 2.1 מודל חשובי

מודל חשובי הוא אוסף של מכונות טיורינג שעבורן מוגדרים המושגים של הכרעה וקבלה של שפות.

הגדרה 2.2 מודלים שקולים חשובית

יהיו A ו- B מודלים חשוביים. אומרים כי A ו- B שקולים אם לכל שפה L התנאים הבאים מתקיימים:

- (1) קיימת מ"ט במודל A שמכריעה את L אם"ם קיימת מ"ט במודל B שמכריעה את L .
- (2) קיימת מ"ט במודל A שמקבלת את L אם"ם קיימת מ"ט במודל B שמקבלת את L .

דוגמה 2.1

נסמן ב- T את מודל מכונת הטיורינג הבסיסי.

במודל זה בכל צעד ניתן לזוז ימינה או שמאלה. אך לא ניתן להישאר במקום, באותה המשבצת בסרט. במודל זה, הסרט הוא אינסופי לשני הכיוונים. בתחילת החישוב הראש נמצא בתחילת הקלט.

נסמן ב- O את מודל מכונת הטיורינג עם סרט ימינה בלבד.

במודל זה בכל צעד ניתן לזוז ימינה או שמאלה. אך לא ניתן להישאר במקום, באותה המשבצת בסרט. במודל זה, הסרט הוא אינסופי לכיוון אחד בלבד - ימינה. בתחילת החישוב, הקלט ממוקם בקצה השמאלי של הסרט והראש נמצא בתחילת הקלט. החישוב מתנהל כמו במכונה במודל T , למעט כאשר הראש נמצא במשבצת השמאלית ביותר בסרט וצריך לזוז שמאלה - במקרה כזה הראש נשאר במקום ולא זז.

הוכיחו כי המודל T והמודל O שקולים חשובית.

פתרון:

יש להוכיח ש:

- לכל מ"ט במודל O קיימת מ"ט שקולה במודל T .
- לכל מ"ט במודל T קיימת מ"ט שקולה במודל O .

כיוון ראשון

נוכיח כי לכל מ"ט במודל O קיימת מ"ט שקולה במודל T . כלומר:

נתונה $M^O = (Q^O, \Sigma^O, \Gamma^O, \delta^O, q_0^O, acc^O, rej^O)$ במודל O .

נבנה $M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, acc^T, rej^T)$ שקולה במודל T .

- רכיבי המ"ט M^T זהים לאלו של המ"ט M^O , מלבד מהתכונה שהראש של M^O לעולם לא זז מעבר לקצה השמאל של הקלט.
- נעבוד רק עם צד ימין של הסרט האינסופי של M^T ואז M^T תהיה שקולה ל- M^O .

- כדי לדאוג שהראש של המכונה הדו-כיוונית M^T לא זז מעבר לקצה השמאל של הקלט, נוסיף מצבים חדשים וגם מעברים חדשים לפונקציית המעברים של M^T , שמבטיחים שהראש של M^T לא זז מעבר לקצה השמול של הקלט, באופן הבא.
- בתחילת כל חישוב, המכונה M^T מסמנת את המשבצת מצד שמאל וליד המשבצת הראשונה של הקלט בסימן מיוחד \$.
- נגדיר את הפונקציית המעברים של M^T כך שכל פעם שהראש מגיע למשבצת המסומנת \$, הראש חוזר ימינה למשבצת הראשונה של הקלט, כמפורט בטבלת המעברים למטה.

לכן, המכונה M^T השקולה למכונה M^O היא

$$M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, q_{acc}^T, q_{rej}^T),$$

כאשר

$$Q^T = Q^O \cup \{q_0^T, q_\$ \}, \quad \Sigma^T = \Sigma^O, \quad \Gamma^T = \Gamma^O \cup \{\$ \}, \quad q_{acc}^T = q_{acc}^O, \quad q_{rej}^T = q_{rej}^O$$

והפונקציית המעברים מתוארת בטבלה למטה.

תנאי	תזוזה	כתיבה	מצב חדש	סימון	מצב
	L	Ω	$q_\$$	σ	q_0^T
	R	$\$$	q_0^O	$_$	$q_\$$
$\forall q \in Q^O$	R	$\$$	q	$\$$	q

הוכחנו את הכיוון הראשון:

ראינו מכונה דו-כיוונית השקולה למכונה חד-כיוונית.

כעת נוכיח את הטענה בכיוון הראשון:

נראה מכונה חד-כיוונית השקולה למכונה דו-כיוונית.

כיוון שני

נוכיח כי לכל מ"ט במודל T קיימת מ"ט שקולה במודל O . כלומר:

נתונה $M^T = (Q^T, \Sigma^T, \Gamma^T, \delta^T, q_0^T, acc^T, rej^T)$ במודל T .

נבנה $M^O = (Q^O, \Sigma^O, \Gamma^O, \delta^O, q_0^O, acc^O, rej^O)$ שקולה במודל O .

- נסמן "קו המפריד" על הסרט של המכונה הדו-כיוונית M^T .

...	_	a	b	b		b	c	c	a	b	_	...
-----	---	---	---	---	--	---	---	---	---	---	---	-----

- נסמן את המשבצת הראשונה של הסרט של המכונה החד-כיוונית M^O עם \$.

- כל שאר המשבצות של הסרט של M^O נחלק לשני חצאים: חצי העליון U וחצי התחתון D .

- תוכן הסרט של המכונה הדו-כיוונית M^T נכתב על סרטה של המכונה החד-כיוונית M^O כך:

* החלק של הסרט שמצד שמאל של קו המפריד נכתב בשורה העליונה של סרט M^O בכיוון הפוך (מימין לשמאל).

* החלק של הסרט שמצד ימין של קו המפריד נכתב בשורה התחתונה של סרט M^O בכיוון הרגיל (משמאל לימין).

\$	b	b	a	␣	␣	␣	␣	...
	b	c	c	a	b	␣	␣	...

- * תזוזה ימינה של M^T מצד ימין של קו המפריד \Leftarrow תזוזה ימינה בשורה התחתונה של M^O .
 - * תזוזה ימינה של M^T מצד שמאל של קו המפריד \Leftarrow תזוזה שמאלה בשורה העליונה של M^O .
- תזוזה ימינה ב- M^T :

␣	a →	b →		b →	b →	c →	c →	a →	␣ →
---	-----	-----	--	-----	-----	-----	-----	-----	-----

תזוזה שקולה ב- M^O :

\$	← b	← b	← a	␣	␣	␣	␣
	b →	c →	c →	a →	b →	␣ →	␣

- * תזוזה שמאלה של M^T מצד ימין של קו המפריד \Leftarrow תזוזה שמאלה בשורה התחתונה של M^O .
- * תזוזה שמאלה של M^T מצד שמאל של קו המפריד \Leftarrow תזוזה ימינה בשורה העליונה של M^O .

תזוזה שמאלה ב- M^T :

␣	a ←	b ←		b ←	b ←	c ←	c ←	a ←	␣ ←
---	-----	-----	--	-----	-----	-----	-----	-----	-----

תזוזה שקולה ב- M^O :

\$	→ b	→ b	→ a	␣	␣	␣	␣
	b ←	c ←	c ←	a ←	b ←	␣ ←	␣

לכן, המכונה M^O השקולה למכונה M^T היא

$$M^O = (Q^O, \Sigma^O, \Gamma^O, \delta^O, q_0^O, q_{acc}^O, q_{rej}^O),$$

נסביר את כל הרכיבים של M^O :

- לכל מצב $q \in Q^T$ נגדיר q_U ו- q_D של Q^O , כדי להבחין בין המצבים שבהם הראש נמצא בחלק העליון לבין המצבים שבהם הראש נמצא בחלק התחתון של הסרט.

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

$$\Gamma^O \subseteq (\Gamma^T \times \Gamma^T) \cup \{\$ \}.$$

$$q_{acc}^O = q_{acc}^T.$$

$$q_{rej}^O = q_{rej}^T.$$

- הפונקציות המעבריים δ^O מתוארת בטבלת המעברים למטה. בטבלה, הסימנים $\tau, \sigma, \pi \in \Gamma^T$ מסמנים כל תו שבאלפבית Γ^T :

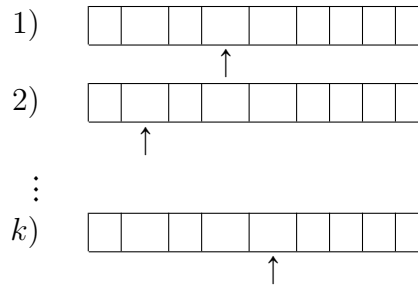
תנאי	תזוזה	כתיבה	מצב חדש	סימון	מצב
אתחול					
$\tau \in \Sigma \cup \{_ \}$ $\sigma \in \Sigma$	R	$\$$	q_τ	τ	q_0^O
	R	$_ \sigma$	q_τ	τ	q_σ
	L	$_ _$	back	$_$	$q._$
	L	\curvearrowright	q_{back}	$_ \tau$	q_{back}
	R	\curvearrowright	$q_0^T.D$	$\$$	q_{back}
תזוזה מקורית שמאלה					
תזוזה שמאלה: $(q, \sigma) \xrightarrow{M^T} (p, \tau, L)$	L	$\pi \tau$	p_D	π	q_D
	R	$\tau \pi$	p_U	σ	q_U
תזוזה שמאלה: $(q, _) \xrightarrow{M^T} (p, \tau, L)$	L	$_ \tau$	p_D	$_$	q_D
	R	$\tau _$	p_U	$_$	q_U
תזוזה מקורית ימינה					
תזוזה ימינה: $(q, \sigma) \xrightarrow{M^T} (p, \tau, R)$	R	$\pi \tau$	p_D	π	q_D
	L	$\tau \pi$	p_U	σ	q_U
תזוזה ימינה: $(q, _) \xrightarrow{M^T} (p, \tau, R)$	R	$_ \tau$	p_D	$_$	q_D
	L	$\tau _$	p_U	$_$	q_U
פגיעה בקצה					
	R	\curvearrowright	q_U	$\$$	q_D
	R	\curvearrowright	q_D	$\$$	q_U
כל השאר עוברים ל-rej					

שיעור 3

מכונות טיורינג מרובות סרטים

3.1 מכונת טיורינג מרובת סרטים: הגדרה היוריסטית

מכונת טיורינג מרובת סרטים (מטמ"ס) היא הכללה של מ"ט עם סרט יחיד. ההבדל הוא שלמטמ"ס ישנו מספר סופי של סרטים, נניח $k > 1$ סרטים.



- לכל סרט יש ראש שלו.
- בתחילת העבודה הקלט w כתוב בתחילת הסרט הראשון וכל שאר הסרטים ריקים. הראשים בכל סרט מצביעים על התא הראשון בסרט, והמכונה נמצאת במצב התחלתי q_0 .
- בכל צעד חישוב, לפי המצב הנוכחי ול- k התווים שמתחת ל- k הראשים, המכונה מחליטה לאיזה מצב לעבור, מה לכתוב מתחת לכל אחד מ- k הראשים ולכן להזיז את הראש בכל אחד מ- k סרטים.
- הראשים של הסרטים יכולים לזוז באופן בלתי-תלוי בהתאם לפונקצית המעברים של המטמ"ס.

3.2 מכונת טיורינג מרובת סרטים: הגדרה פורמלית

3.1 מכונת טיורינג מרובת סרטים

מכונת טיורינג מרובת סרטים היא שביעייה:

$$M = (Q, \Sigma, \Gamma, \delta_k, q_0, q_{acc}, q_{rej})$$

כאשר $Q, \Sigma, \Gamma, q_0, q_{acc}, q_{rej}$ מוגדרים כמו מ"ט עם סרט יחיד (ראו הגדרה 1.2). ההבדל היחיד בין מ"ט עם סרט יחיד לבין מטמ"ס הוא הפונקציות המעברים. עבור מטמ"ס הפונקציות המעברים היא מצורה הבאה:

$$\delta_k : (Q \setminus \{q_{acc}, q_{rej}\}) \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, S\}^k$$

3.1 דוגמה



$$\delta_k \left(q, \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \right) = \left(p, \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} R \\ R \\ L \end{pmatrix} \right).$$

3.3 קונפיגורציה של מכונת טיורינג מרובת סרטים

הכללה של קונפיגורציה של מ"ט עם סרט יחיד:

$$\begin{pmatrix} u_1 q \ v_1 \\ u_2 q \ v_2 \\ \vdots \\ u_k q \ v_k \end{pmatrix}$$

3.2 דוגמה

בנו מטמ"ס שמכריעה את השפה:

$$L_{w^R} = \{ w = \{a, b\}^* \mid w = w^R \}.$$

כלומר שפת הפלינדרומים.

פתרון:

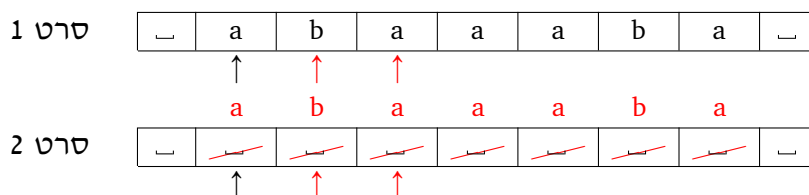
נבנה מ"ט עם שני סרטים:

תאור המכונה:

נסמן M_2 המ"ט עם 2 סרטים שמכריעה את השפה L_{w^R} .

M_2 = על הקלט w :

(1) מעתיקה את w לסרט 2.



(2) מזיזה את הראש בסרט 1 לתו הראשון ב- w ואת הראש בסרט 2 לתו האחרון ב- w .

(3) משווה בין התווים שמתחת לראשים:

- אם התו שמתחת לראש בסרט 1 הוא $_$
 $\text{acc} \leftarrow$
- אם התווים שמתחת לראשים שונים $\text{rej} \leftarrow$
- אחרת מזיזה את הראש בסרט 1 ימינה ואת הראש בסרט 2 שמאלה, וחוזרת לשלב (3).

הפונקציה המעברים של M_2 היא:

$$\begin{aligned}\delta\left(q_0, \begin{pmatrix} a \\ _ \end{pmatrix}\right) &= \left(q_0, \begin{pmatrix} a \\ a \end{pmatrix}, \begin{pmatrix} R \\ R \end{pmatrix}\right), \\ \delta\left(q_0, \begin{pmatrix} b \\ _ \end{pmatrix}\right) &= \left(q_0, \begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} R \\ R \end{pmatrix}\right), \\ \delta\left(q_0, \begin{pmatrix} _ \\ _ \end{pmatrix}\right) &= \left(q_{\text{back}}, \begin{pmatrix} _ \\ _ \end{pmatrix}, \begin{pmatrix} L \\ S \end{pmatrix}\right), \\ \delta\left(q_0, \begin{pmatrix} _ \\ _ \end{pmatrix}\right) &= \left(q_{\text{back}}, \begin{pmatrix} _ \\ _ \end{pmatrix}, \begin{pmatrix} L \\ S \end{pmatrix}\right), \\ \delta\left(q_{\text{back}}, \begin{pmatrix} _ \\ _ \end{pmatrix}\right) &= \left(q_{\text{check}}, \begin{pmatrix} _ \\ _ \end{pmatrix}, \begin{pmatrix} R \\ S \end{pmatrix}\right), \\ \delta\left(q_{\text{check}}, \begin{pmatrix} \sigma \\ \sigma \end{pmatrix}\right) &= \left(q_{\text{check}}, \begin{pmatrix} \sigma \\ \sigma \end{pmatrix}, \begin{pmatrix} R \\ L \end{pmatrix}\right), \\ \delta\left(q_{\text{check}}, \begin{pmatrix} \sigma \\ \tau \end{pmatrix}\right) &= \left(q_{\text{rej}}, \begin{pmatrix} \sigma \\ \tau \end{pmatrix}, \begin{pmatrix} S \\ S \end{pmatrix}\right), \\ \delta\left(q_{\text{check}}, \begin{pmatrix} _ \\ _ \end{pmatrix}\right) &= \left(q_{\text{acc}}, \begin{pmatrix} _ \\ _ \end{pmatrix}, \begin{pmatrix} S \\ S \end{pmatrix}\right).\end{aligned}$$

נשים לב כי הסיבוכיות זמן של המכונה עם שני סרטים, M_2 היא $O(|w|)$, כאשר w האורך של המילה.

כעת נבנה מ"ט עם סרט יחיד שמכריעה את השפה L_{WR} .

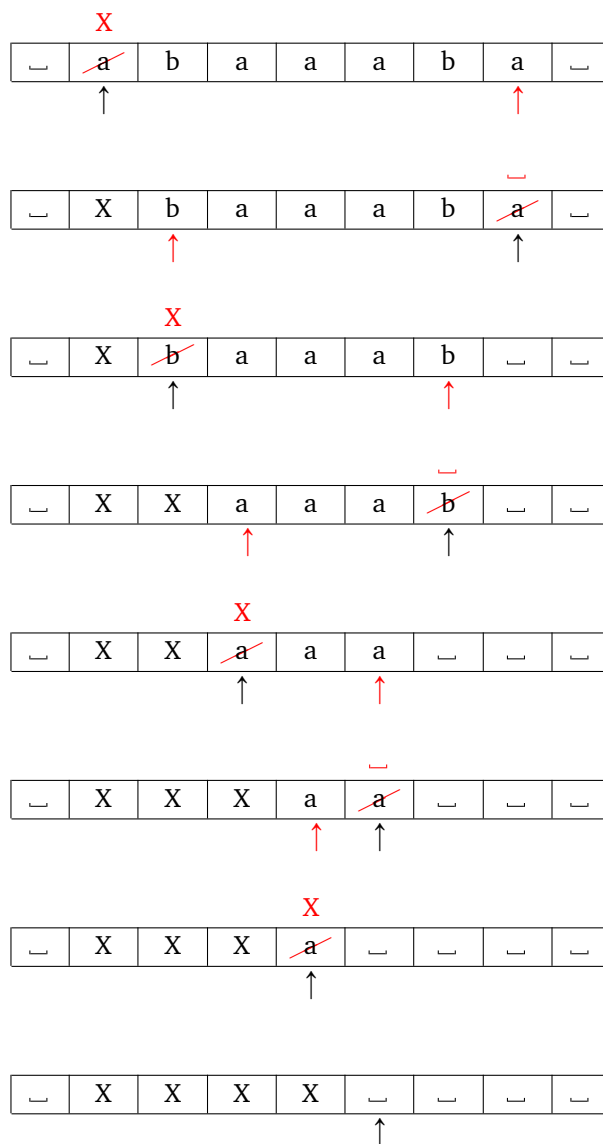
תאור המכונה:

נסמן M_1 המכונה עם סרט יחיד שמכריעה את השפה L_{wR} .

$M_1 =$ על הקלט w :

- (1) אם התו שמתחת לראש הוא $_$ אז $\text{acc} \leftarrow M_1$
 - (2) זוכרת את התו שמתחת לראש ומוחקת אותו ע"י X .
 - (3) מזיזה את הראש ימינה עד התו הראשון משמאל ל- $_$.
- אם התו שמתחת לראש הוא X אז $\text{acc} \leftarrow X$

- אם התו שונה מהתו שזכרנו $\Leftarrow \text{rej}$.
- מוחקת את התו שמתחת לראש ע"י $_$, מזיזה את הראש שמאולה עד התו הראשון מימין ל- X וחוזרת לשלב (1).



3.4 שקילות בין מטמ"ס למ"ט עם סרט יחיד

מ"ט עם סרט יחיד היא מקרה פרטי של מטמ"ס.

משפט 3.1 שקילות בין מטמ"ס למ"ט עם סרט יחיד

לכל מטמ"ס M קיימת מ"ט עם סרט יחיד M' השקולה ל- M .

כלומר, לכל קלט $w \in \Sigma^*$:

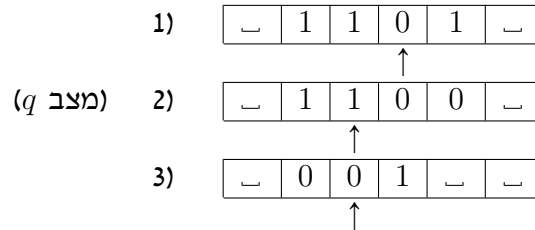
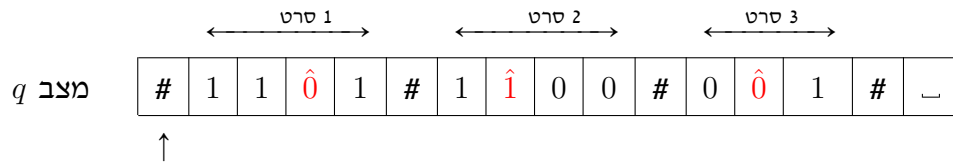
- אם M מקבלת את $w \Leftarrow M'$ מקבלת את w .
- אם M דוחה את $w \Leftarrow M'$ דוחה את w .
- אם M לא עוצרת על $w \Leftarrow M'$ לא עוצרת על w .

הוכחה:

בהינתן מטמ"ס $M = (Q, \Sigma, \Gamma, \delta_k, q_0, q_{acc}, q_{rej})$ עם k סרטים, נבנה מטמ"ס עם סרט יחיד $M' = (Q', \Sigma, \Gamma', \delta', q'_0, q'_{acc}, q'_{rej})$ השקולה ל- M באופן הבא:

רעיון הבנייה:

בהינתן קלט $w \in \Sigma^*$, M' תבצע "סימולציה" של ריצה M על w .

ב- M ב- M' 

- M' תשמור את התוכן של k הסרטים של M על הסרט, רק שהתוכן של סרט i יופיע בין $\#_i$ ל- $\#_{i+1}$.
- M' תשמור את המיקום של הראשים של M ע"י הכפלת הא"ב Γ .
- כלומר, לכל אות $\alpha \in \Gamma$, M' תשמור שתי אותיות α ו- $\hat{\alpha}$ ב- Γ' , כך ש- $\hat{\alpha}$ תסמן את התו שמתחת לראש בכל סרט.
- בכל צעד חישוב, M' סורקת את הסרט שלה משמאל לימין כדי ללמוד מהם התווים שמתחת לראשים (התווים שמסומנים ב- $\hat{\alpha}$).
- M' משתמשת בפונקצית המעברים δ_k של M כדי לחשב את המעבר הבא.
- M' סורקת את הסרט שלה משמאל לימין כדי לעדכן את הסרטים ואת המיקום הראשים בהם.

תאור הבנייה של M' :**(1) שלב האיתחול**

בהינתן קלט $w = \sigma_1 \sigma_2 \cdots \sigma_n$, M' מאתחלת את הקונפיגורציה ההתחלתית של M על הסרט שלה.

ב- M

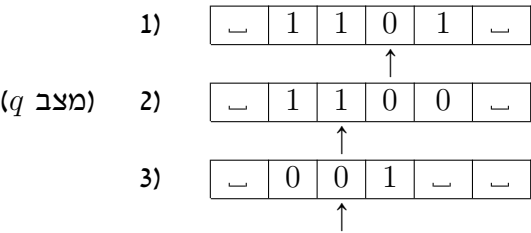


ב- M'

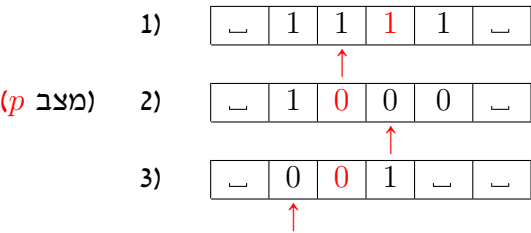


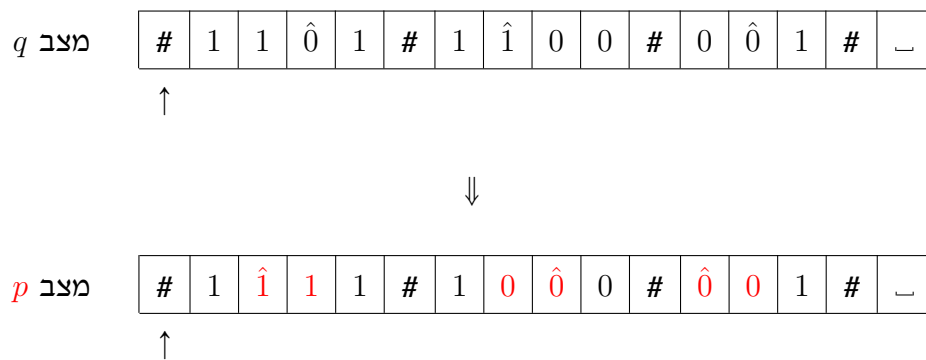
(2) תאור צעד חישוב של M

ב- M



$$\delta_k \left(q, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \right) = \left(p, \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} L \\ R \\ L \end{pmatrix} \right)$$



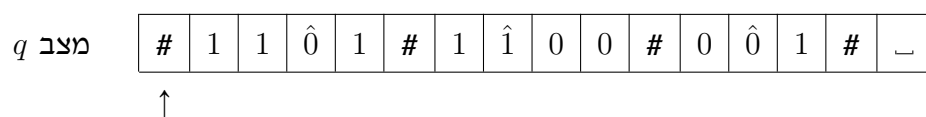
M' -ב

- איסוף מידע
- M' סורקת את הסרט שלה משמאל לימין ומזהה את התווים שמסומנים ב- $\hat{\alpha}$. מידע זה ניתן לשמור במצבים. לדוגמה:

$$q, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

זה אפשרי מכיוון שמספר המצבים הנדרש הוא סופי:

$$|Q| \times |\Gamma|^k \quad .$$





#	1	$\hat{1}$	1	1	#	1	0	$\hat{0}$	0	#	$\hat{0}$	0	1	#	␣
---	---	-----------	---	---	---	---	---	-----------	---	---	-----------	---	---	---	---

↑

M' סורקת את הסרט שלה פעם נוספת כדי לפעול על פי פונקצית המעברים, כלומר, לעדכן את התאים שמתחת לראשים ולעדכן את מיקום הראשים.

- אם בכל שלב M' מזיזה אחד מן הראשים הווירטואליים ימינה אל סימן ה- $\#$, פעולה זו מציינת שמכונת M הזיזה את הראש המתאים אל החלק הריק שטרם נקרא של הסרט. לכן M' כותבת תו $_$ על המשבצת הזו ומזיזה את כל התוכן של הסרט בין התא הזה לבין ה- $\#$ הימני ביותר בתא אחד ימינה. לאחר מכן היא ממשיכה את הסימולציה כרגיל.

שיעור 4

מכונת טיורינג אי-דטרמיניסטית

4.1 הגדרה של מכונת טיורינג אי-דטרמיניסטית

4.1 מכונת טיורינג אי-דטרמיניסטית

מכונת טיורינג אי-דטרמיניסטית (מ"ט א"ד) היא שביעייה

$$M = (Q, \Sigma, \Gamma, \Delta, q_0, q_{acc}, q_{rej})$$

כאשר $Q, \Sigma, \Gamma, q_0, q_{acc}, q_{rej}$ מוגדרים כמו במ"ט דטרמיניסטי (ראו הגדרה 1.2).

Δ היא פונקצית המעברים

$$\Delta : (Q \setminus \{q_{acc}, q_{rej}\}) \times \Gamma \rightarrow P(Q \times \Gamma \times \{L, R, S\}) .$$

$$\Delta(q, a) = \{(q_1, a, S) , (q_2, b, L) , \dots\} .$$

כלומר, לכל זוג $q \in Q, a \in \Gamma$ ייתכן מספר מעברים אפשריים, 0, 1 או יותר.

- קונפיגורציה של מ"ט א"ד זהה לקונפיגורציה של מ"ט דטרמיניסטית.

- לכל קונפיגורציה ייתכן מספר קונפיגורציות עוקבות.

- לכל מילה $w \in \Sigma^*$ ייתכן מספר ריצות שונות:

- * ריצות שמגיעות ל- q_{acc} .

- * ריצות שמגיעות ל- q_{rej} .

- * ריצות שלא עוצרות.

- * ריצות שנתקעות.

4.2 הגדרה

מילה $w \in \Sigma^*$ מתקבלת במ"ט א"ד M אם קיימת לפחות ריצה אחת שמגיעה ל- q_{acc} .

השפה של מ"ט א"ד M היא

$$L(M) = \{w \in \Sigma^* \mid \exists u, v \in \Gamma^* : q_0 w \vdash_* u q_{acc} v\}$$

כלומר,

$w \in L(M)$ אם קיימת ריצה אחת שבה M מקבלת את w .

$w \notin L(M)$ אם בכל ריצה של M על w , M דוחה או לא עוצרת, או נתקעת.

הגדרה 4.3 מ"ט א"ד המכריעה שפה L

תהי M מ"ט א"ד.
אומרים כי מ"ט א"ד M מכריעה שפה L אם לכל $w \in \Sigma^*$:

• אם $w \in L$ אז M מקבלת את w .

• אם $w \notin L$ אז M דוחה את w .

הגדרה 4.4 מ"ט א"ד המקבלת שפה L

תהי M מ"ט א"ד.
אומרים כי מ"ט א"ד M מקבלת שפה L אם לכל $w \in \Sigma^*$:

• אם $w \in L$ אז M מקבלת את w .

• אם $w \notin L$ אז M דוחה את w או M לא עוצרת על w .

דוגמה 4.1

נתונה השפה

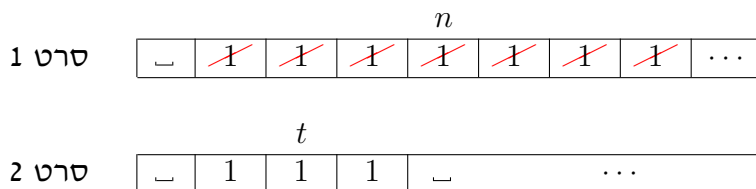
$$L = \{1^n \mid n \text{ אינו ראשוני}\}, \quad \Sigma = \{1\}.$$

בנו מ"ט המכריעה את השפה L .

פתרון:הרעיון

נבנה מ"ט א"ד N המכריעה את L .

N תבחר באופן א"ד מספר $1 < t < n$ ותבדוק האם t מחלק את n .

תאור הבניה

N = על קלט $w = 1^n$

שלב 1)

• N בוחרת באופן א"ד מספר $1 < t < n$.

• מעתיקה את w לסרט 2.

• עוברת על העותק משמאל לימין, ובכל תא מחליטה באופן א"ד האם להשאיר את ה-1 או למחוק אותו ע"י X (לדאוג שהמספר שנבחר הוא לא 1 ולא n).

- בסוף המעבר המספר t שנבחר הוא כמות ה-1 ים שלא נמחקו.

סרט 1

␣	1	1	1	1	...	1	1	␣	...
---	---	---	---	---	-----	---	---	---	-----

סרט 2

␣	1	1	1	1	...	1	1	␣	...
---	---	--------------	--------------	---	-----	--------------	---	---	-----

שלב 2 N בודקת האם t שנבחר מחלק את n .

- אם כן $N \Leftarrow$ מקבלת.

- אם לא $N \Leftarrow$ דוחה.

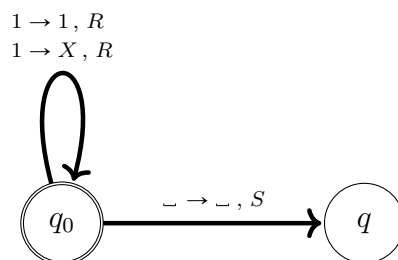
4.2 עץ החישוב של מ"ט א"ד

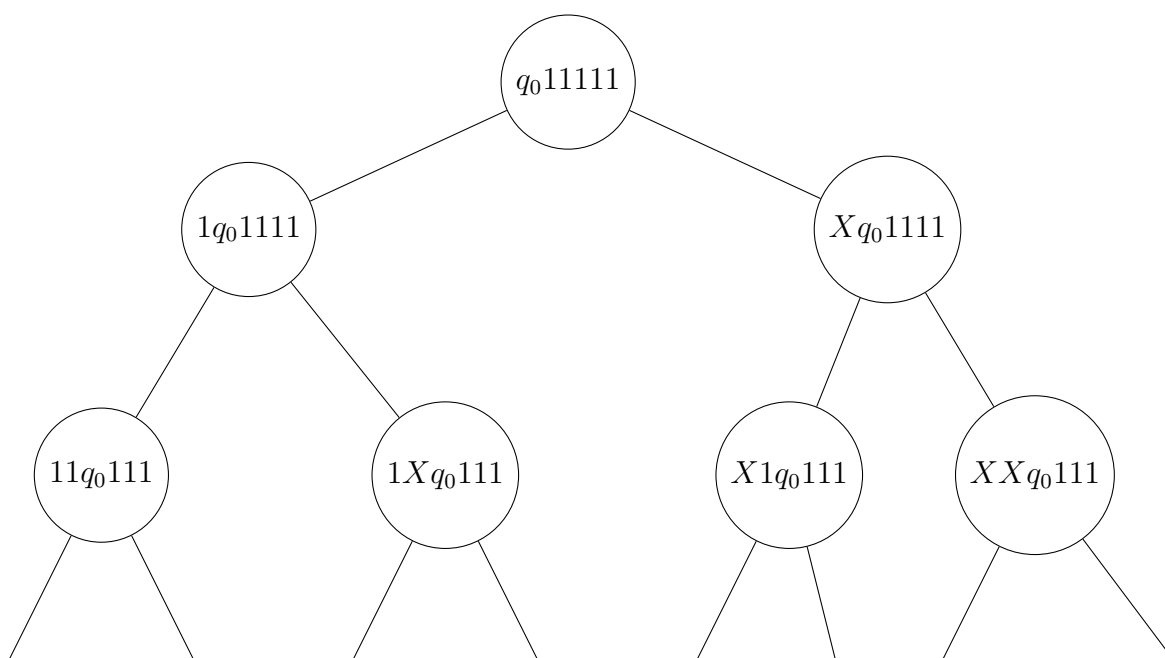
הגדרה 4.5 עץ החישוב של מ"ט א"ד

בהינתן מ"ט א"ד M ומילה $w \in \Sigma^*$, עץ החישוב של M ו- w הוא עץ מושרש שבו:

- (1) כל קדקוד בעץ מתאר קונפיגורציה בחישוב של M על w .
- (2) שורש העץ מתאר את הקונפיגורציה ההתחלתית $q_0 w$.
- (3) לכל קדקוד v בעץ הבנים של v הם כל הקדקודים הנובעים מהקונפיגורציה המתוארת ע"י v .

דוגמה 4.2





4.3 שקילות בין מכונת טיורינג אי-דטרמיניסטית למכונת טיורינג דטרמיניסטית

משפט 4.1 שקילות בין מ"ט אי-דטרמיניסטית למ"ט דטרמיניסטית ב- RE

לכל מכונת טיורינג אי-דטרמיניסטית N קיימת מכונת טיורינג דטרמיניסטית D כך ש-

$$L(N) = L(D) .$$

כלומר לכל $w \in \Sigma^*$:

- אם N מקבלת את w $D \Leftarrow w$ תקבל את w .
- אם N לא מקבלת את w $D \Leftarrow w$ לא תקבל את w .

הוכחה: בהינתן מכונת טיורינג אי-דטרמיניסטית N נבנה מכונת טיורינג דטרמיניסטית D ונוכיח כי

$$L(N) = L(D) .$$

רעיון ההוכחה

בהינתן קלט $w \in \Sigma^*$, D תבצע ריצה של כל החישובים האפשריים של N על w , ואם אחד החישובים מסתיים ב- q_{acc} אזי D תעצור ותקבל.

מכיוון שייכתנו חישובים אינסופיים, לא נוכל לסרוק את עץ החישוב לעומק. במקום זה נסרוק את העץ לרוחב. כלומר, נבדוק את כל החישובים באורך 1, ואחרי זה נבדוק את כל החישובים באורך 2, וכן הלאה. אם אחד החישובים הסתיים ב- q_{acc} , אזי D תעצור ותקבל.

תאור הבניה

מכיוון שלכל $q \in Q$ ולכל $\alpha \in \Gamma$:

$$\Delta(q, \alpha) \subseteq Q \times \Gamma \times \{L, R, S\} .$$

אזי

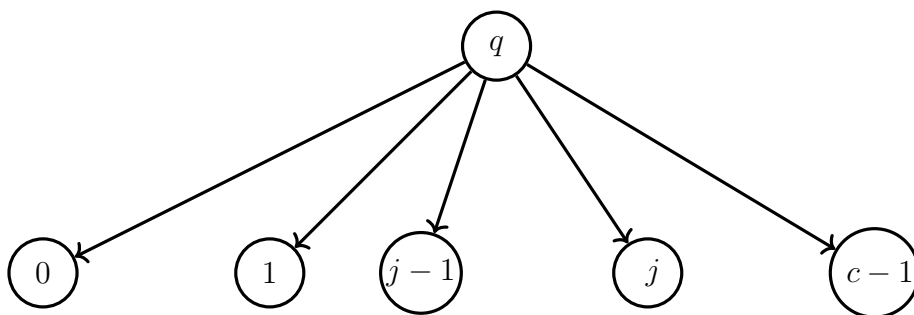
$$|\Delta(q, \alpha)| \leq |Q| \cdot |\Gamma| \cdot |\{L, R, S\}| = 3|Q| \cdot |\Gamma| .$$

נסמן:

$$C = 3|Q| \cdot |\Gamma| .$$

- לכל מצב $q \in Q$ ולכל אות $\alpha \in \Gamma$ נמספר את המעברים ב- $\Delta(q, \alpha)$ שרירותית

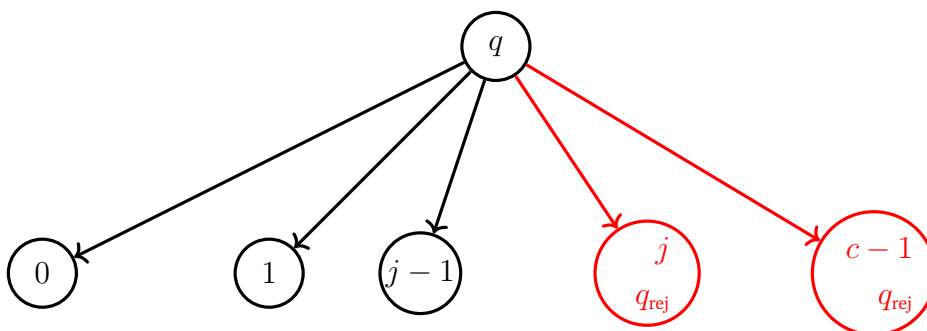
$$\{0, 1, 2, \dots, C-1\} .$$



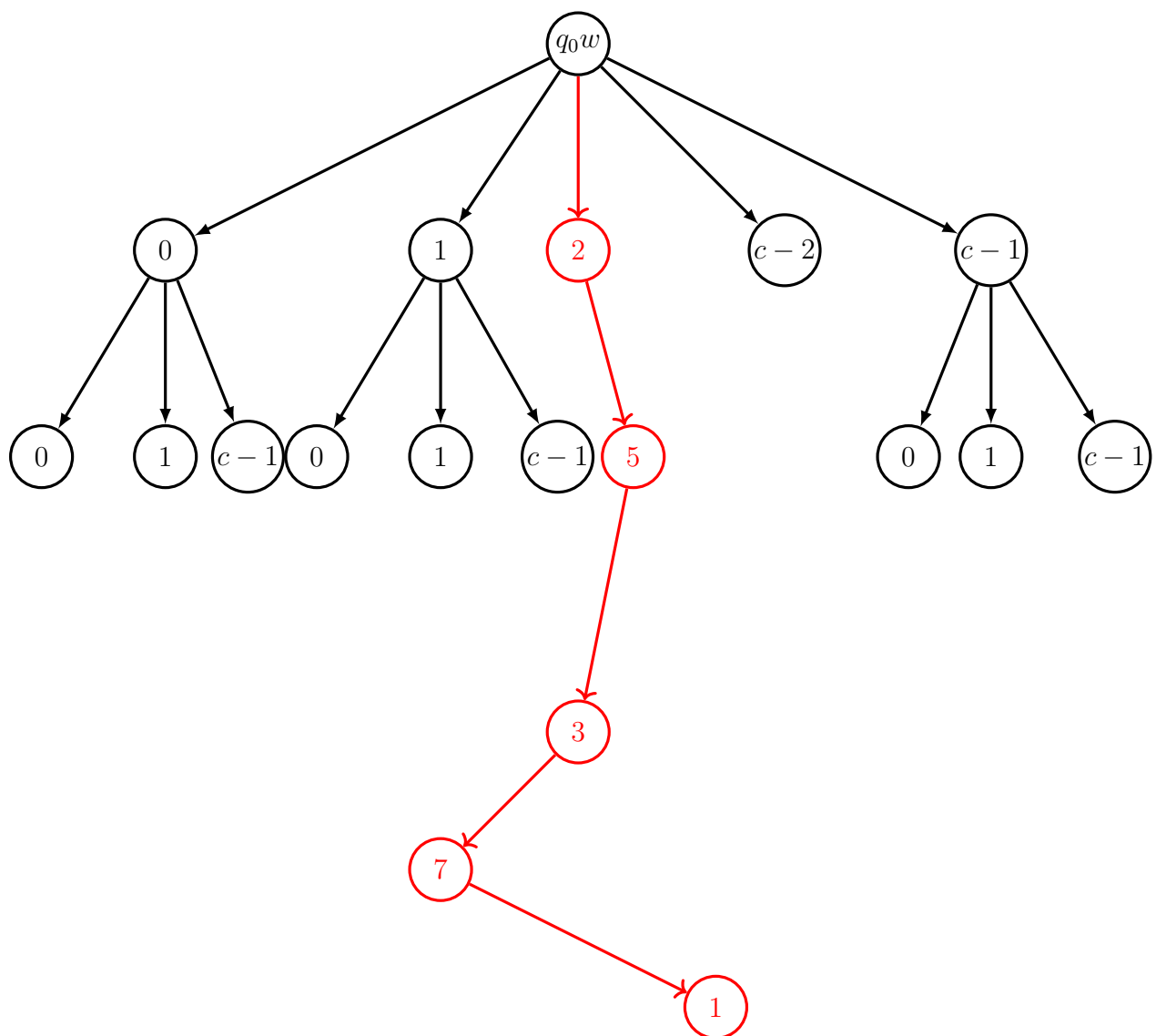
- אם $|\Delta(q, \alpha)| = j < C$,

אזי לכל $j \leq k \leq C-1$

נקבע $k = (q_{\text{rej}}, \alpha, S)$.



- נשים לב כי שינוי זה לא משנה את השפה של N .



קידום לקסיקוגרפי:

0	00	10	...	$(C-1)0$	000	
1	01	11	...	$(C-1)1$	001	
2	02	12	...	$(C-1)2$	002	...
\vdots	\vdots	\vdots	...	\vdots	\vdots	
$C-1$	$0(C-1)$	$1(C-1)$...	$(C-1)(C-1)$	$00(C-1)$	

הבנייה של D

D מכילה 3 סרטים:

סרט 1	n								
	␣	1	1	1	1	1	1	1	...
סרט 2	t								
	␣	1	1	1	␣				...
סרט 3	␣	1	1	1	␣				...

$D = \text{"על קלט } w\text{:}$

(1) מאתחלת את המחרוזת בסרט 3 ל-0.

(2) מעתיקה את w לסרט 2.

(3) מריצה את N על w לפי המחרוזת בסרט 3.

• אם N קיבלה את $w \Leftarrow D$ עוצרת ומקבלת.

• אחרת D מוחקת את סרט 2, מקדמת את המחרוזת בסרט 3 לקסיקוגרפית וחוזרת לשלב 2)



שיעור 5

התזה של צרף טיורינג ודקדוקים כלליים

5.1 היחס בין הכרעה וקבלה

משפט 5.1 כל שפה כריעה היא גם קבילה

כל שפה כריעה היא גם קבילה.

■ **הוכחה:** המכונה טיורינג שמכריעה את L גם מקבלת אותה. נשאל שאלה. האם כל שפה קבילה היא גם כריעה? זאת שאלה שכרגע אין לנו מספיק כלים לענות עליה. נחזור לשאלה הזו בפרק הבא. לבינתיים נוכיח טענה חלשה יותר.

משפט 5.2

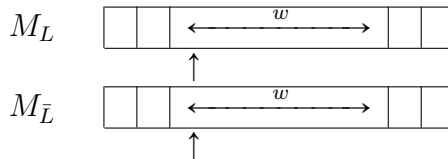
תהי L שפה.

אם גם L וגם \bar{L} קבילות אזי L כריעה.

הוכחה: תהי M_L מ"ט שמקבלת את L , ותהי $M_{\bar{L}}$ מ"ט שמקבלת את \bar{L} . נבנה מ"ט D_L שמכריעה את L .

כיצד תעבוד המ"ט D_L המכריעה?

- נריץ במקביל את M_L ואת $M_{\bar{L}}$.
- אם M_L מקבלת את המילה אז נעבור ל-acc.
- אם $M_{\bar{L}}$ מקבלת את המילה אז נעבור ל-rej.



- הסימולציה מתבצעת ע"י סימלוץ צעד צעד.

* צעד במכונה M_L .

* צעד במכונה $M_{\bar{L}}$.

- נמשיך בסימולציה המקבילית עד שאחת המכונות מגיעה למצב acc.

* אם M_L מקבלת $\leftarrow .acc$.

* אם $M_{\bar{L}}$ מקבלת $\leftarrow .rej$.

- לא יכול להיות מצב כי אף אחת מהמכונות לא מגיעה למצב acc כי כל מחרוזת $w \in L$ או $w \in \bar{L}$.

5.2 שקילות של מכונת טיורינג ותוכנית מחשב

- מכונת טיורינג היא מודל חישובי למחשב.

- מחשב = תוכנית מחשב.

- תוכנית מחשב כתובה בשפת תכנות, למשל

* ג'אווה

* פייתון

* C

* SIMPLE

- המרכיבים של שפת תכנות הם

* משתנים

* פעולות

* תנאים

* זרימה

נוכיח כי מכונת טיורינג ותוכנית מחשב שקולים חישובי.

5.3 SIMPLE

משתנים

- טבעיים:

¹ i, j, k, \dots

מקבלים כערך מספר טבעי.

מערכים

- המערכים אין-סופיים

¹ $A[]$,

² $B[]$,

³ $C[]$,

- בכל תא ערך מתוך א"ב Γ .

אתחול

- כל המשתנים מאותחלים ל-0.

- הקלט נמצא בתאים הראשונים של $A[]$.

פעולות

- השמה בקבוע:

```
1 i=3, B[i]="#"
```

- השמה בין משתנים:

```
1 i=k, A[k]=B[i]
```

- פעולות חשבון:

```
1 x = y + z , x = y - z , x = y.z
```

תנאים

- $B[i] == A[j]$ (מערכים).

- $x \geq y$ (משתנים טבעיים).

זרימה

- סדרה פקודות ממוספרות.

- goto : מותנה ולא מותנה.

- stop : עצירה עם ערך חזרה.

```
1 one = 1
2 zero = 0
3 B[zero] = "0"
4 i=0
5 j=i
6 if A[i] == B[zero] goto 9
7 i=j + one
8 goto 3
9 C[one] = A[j]
10 if C[one] == A[zero] goto 12
11 stop(0)
12 stop(1)
```

כעת נגדיר את מושגי הקבלה והדחייה של מילים בשפה SIMPLE, ונגדיר את מושגי הכרעה והקבלה של שפות בשפה SIMPLE.

הגדרה 5.1 קבלה ודחייה של מחרוזת בשפה SIMPLE

עבור קלט w ותוכנית P בשפת SIMPLE. אומרים כי

- P **מקבלת** את w אם הריצה של P על w עוצרת עם ערך חזרה 1.
- P **דוחה** את w אם הריצה של P על w עוצרת עם ערך חזרה 0.

הגדרה 5.2 הכרעה וקבלה של מחרוזת בשפה SIMPLE

עבור שפה L ותוכנית P בשפת SIMPLE. אומרים כי

- P **מכריעה** את L אם היא מקבלת את המילים שב- L ודוחה את אלה שלא ב- L .
- P **מקבלת** את L אם היא מקבלת את כל ורק המילים ב- L .

משפט 5.3

המודלים של מכונת טיורינג ותוכניות SIMPLE שקולים.

הוכחה:

כיוון ראשון:

נוכיח כי לכל מ"ט M קיימת תוכנית P שקולה.
נבצע סימולציה של מ"ט M במחשב P .

בלי להכנס לפרטים, די ברור שבשפה עילית, כגון ג'אווה, ניתן להגדיר מבני נתונים עבור כל מרכיבי מכונת טיורינג:

- הסרט.
- המצבים.
- מיקום הראש.
- טבלת המעברים.

וברור שניתן לבצע סימולציה של פעילות המכונה.
ואם ניתן לעשות זאת בשפה עילית, ניתן לעשות זאת גם בשפת SIMPLE.

כיוון שני:

נוכיח כי לכל תוכנית P בשפה SIMPLE קיימת מ"ט שקולה.

אנחנו צריכים להראות כיצד ניתן לממש את הרכיבים השונים של תוכניות SIMPLE במ"ט.

הרכיבים הם:

- משתנים.
- פעולות.
- תנאים.
- זרימה.

משתנים

לכל משתנה יהיה סרט משלו.
המספר שהמשתנה יחזיק ייוצג בבסיס אונרי.
בהתחלה הסרט יהיה רק עם רווחים, זה מייצג את המספר אפס בבסיס אונרי.

לכל מערך יהיה סרט משלו.
בכל תא בסרט המערך תהיה אות.
בהתחלה כל המערכים יהיו מאופסים למעט הסרט הראשון, שיחזיק את הקלט.

למשל ההשמה הבאה של משתנים בשפה SIMPLE:

¹ $A[1] = a, A[2] = b, A[3] = b, A[4] = a$
² $B[1] = b, B[2] = a$
³ $i = 3$
⁴ $j = 1$
⁵ $k = 2$

ניתן לממש במ"ט על ידי לכתוב על סרטים, שרט אחד לכל משתנה:



פעולות

כעת נניח שנשים

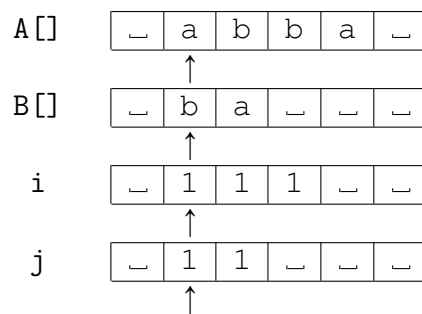
¹ $j = k$

אפשר לממש את ההשמה הזאת על ידי להעתיק את תוכן הסרט של המשתנה k לסרט של המשתנה j .

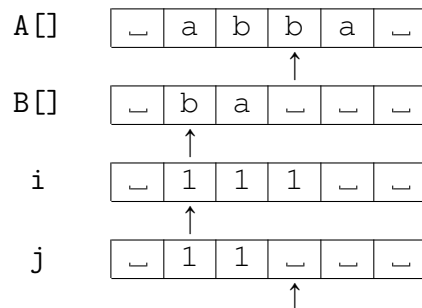


כעת נניח שנשים $B[i] = A[j]$, ז"א $B[3] = A[2]$.

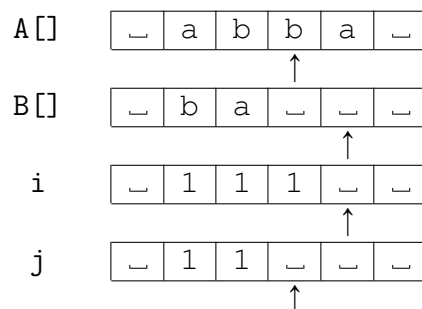
נממש זה במ"ט ע"י להעתיק את תוכן משבצת 2 בסרט של $A[]$ למשבצת 3 בסרט של $B[]$.



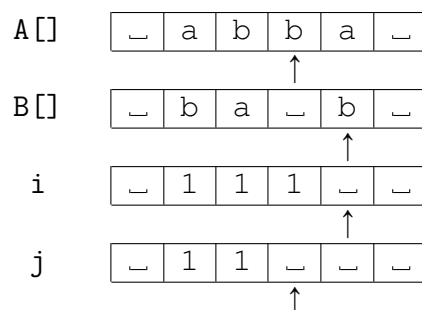
(שלב 2)



(שלב 3)



(שלב 4)



נניח עכשיו שאנחנו רוצים לשים

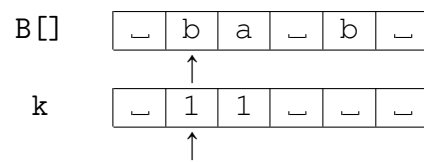
`1 B[k] <- "a"`

ז"א

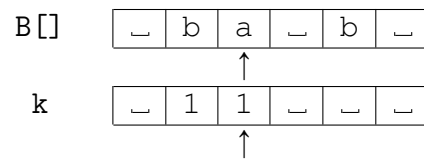
`1 B[3] <- "a"`

נממש זה במ"ט ע"י על ידי הפעולות הבאות עם הסרט של B[] והסרט של k.

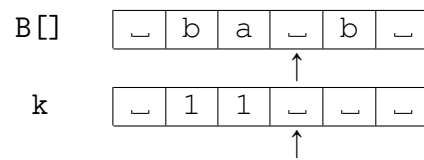
(שלב 1)



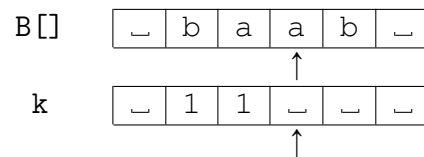
שלב (2)



שלב (3)



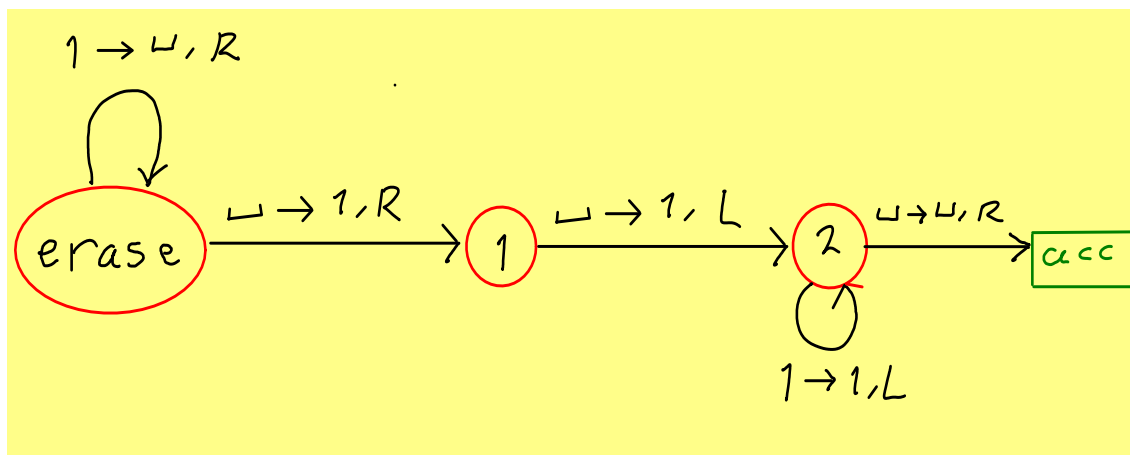
שלב (4)



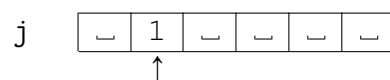
כעת נניח שאנחנו רוצים לשים

$j=2$

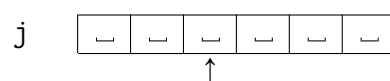
אז נממש זה במ"ט עם הפעולות הבאות:



שלב (1)



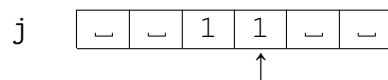
שלב (2)



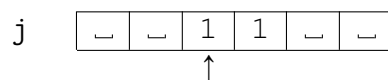
(שלב 3)



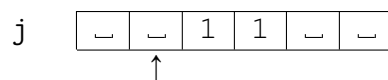
(שלב 4)



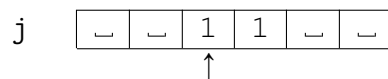
(שלב 5)



(שלב 6)



(שלב 7)

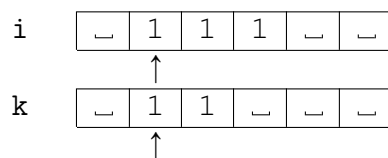
תנאים

נניח שאנחנו רוצים לממש את התנאי

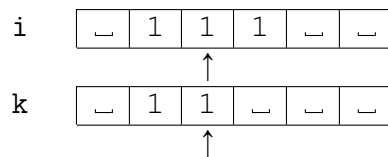
`i >= k`

ניתן לבדוק את התנאי במ"ט על ידי הפעולות הבאות:

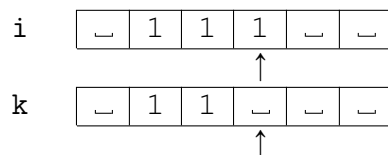
(שלב 1)



(שלב 2)



(שלב 3)



הגדרה 5.3 זקדוקים חסרי קשר

דקדוק חסר קשר הוא קבוצה

$$(V, \Sigma, R, S)$$

כאשר

- V קבוצה סופית של **משתנים** שמורכב מאותיות גדולות של אלפיבית.
- Σ קבוצה סופית של **טרמינלים** שמורכב מאותיות קטנות וסימנים של אלפיבית.
- R קבוצה של כללים. כל כלל הוא מצורה

$$\gamma \rightarrow u$$

כאשר $\gamma \in V$ משתנה בודד בצד שמאל ו- $u \in (V \cup \Sigma)^*$ מחרוזת של משתנים וטרמינלים בצד ימין

- $S \in V$ המשתנה ההתחלתי.

דוגמה 5.1

נתון הדקדוק חסר קשר:

$$G_1 = (\{A, B\}, \{0, 1, \#\}, R, A)$$

הקבוצת משתנים היא $V = \{A, B\}$, הקבוצת טרמינלים היא $V = \{0, 1, \#\}$, המשתנה ההתחלתי הוא $S = A$ והכללים של הדקדוק הם

$$R = \begin{cases} A \rightarrow 0A1 \\ A \rightarrow B \\ B \rightarrow \# \end{cases}$$

הגדרה 5.4 יצירה של מילה על ידי זקדוק חסר קשר(1) כתבו את המשתנה ההתחלתי S .

(2) מצאו משתנה וכלל אשר מתחיל אם משתנה זה, והחליפו אותו עם המחרוזת בצד ימין של הכלל.

(3) חזרו על שלבים 1 ו-2 עד שלא נשאר אף משתנים של V .**דוגמה 5.2**הדקדוק G_1 יוצר את המחרוזת $000\#111$:

$$A \xrightarrow{A \rightarrow 0A1} 0A1 \xrightarrow{A \rightarrow 0A1} 00A11 \xrightarrow{A \rightarrow 0A1} 000A111 \xrightarrow{A \rightarrow B} 00B11 \xrightarrow{B \rightarrow \#} 000\#111$$

דוגמה 5.3

נתון את הדקדוק

$$G_2 = (\{S, T, F\}, \{(\cdot), +, \times, a\}, R, S)$$

כאשר הכללים הם

$$R = \begin{cases} S \rightarrow S + T \\ S \rightarrow T \\ T \rightarrow T \times F \\ T \rightarrow F \\ F \rightarrow (S) \\ F \rightarrow a . \end{cases}$$

G_2 יוצר את המילה: $a + a$

$$S \xrightarrow{S \rightarrow S+T} S + T \xrightarrow{SA \rightarrow T} T + T \xrightarrow{T \rightarrow F} F + F \xrightarrow{F \rightarrow a} a + a$$

בדקדוק כללי, גם בצד ימין וגם בצד שמאל יכולה להופיע מחרוזת של משתנים וטרמינלים. פורמלי:

הגדרה 5.5 זקדוקים כלליים

דקדוק חסר קשר הוא קבוצה

$$(V, \Sigma, R, S)$$

כאשר

- V קבוצה סופית של **משתנים** שמורכב מאותיות גדולות של אלפיבית.
- Σ קבוצה סופית של **טרמינלים** שמורכב מאותיות קטנות וסימנים של אלפיבית.
- R קבוצה של כללים. כל כלל הוא מצורה

$$\gamma \rightarrow u$$

כאשר $\gamma \in (V \cup \Sigma)^+$, $u \in (V \cup \Sigma)^*$. מחרוזת של משתנים וטרמינלים בצד ימין

- $S \in V$ המשתנה ההתחלתי.

דוגמה 5.4

נתון את הדקדוק

$$G = (\{S, [,]\}, \{a\}, R, S)$$

שבו הקבוצת משתנים היא $V = \{S, [,]\}$, הקבוצת טרמינלים היא $\Sigma = \{a\}$ והכללים הם

$$R = \begin{cases} S \rightarrow [S] \\ S \rightarrow a \\ [a \rightarrow aa[\\ [] \rightarrow \varepsilon . \end{cases}$$

G יוצר את המילה: $aaaa$

$$\begin{array}{ccccccc} S & \xrightarrow{S \rightarrow [S]} & [S] & \xrightarrow{S \rightarrow [S]} & [[S]] & \xrightarrow{S \rightarrow a} & [[a]] \\ & \xrightarrow{[] \rightarrow \varepsilon} & [aa] & \xrightarrow{[a \rightarrow aa[} & aa[a] & \xrightarrow{[a \rightarrow aa[} & aa aa[] \\ & & & & & & \xrightarrow{[] \rightarrow \varepsilon} & aaaa \end{array}$$

דוגמה 5.5

נתון את הדקדוק

$$G = (\{S, [,]\}, \{a\}, R, S)$$

שבו הקבוצת משתנים היא $V = \{S, [,]\}$, הקבוצת טרמינלים היא $\Sigma = \{a\}$ והכללים הם

$$R = \begin{cases} S \rightarrow [S] \\ S \rightarrow a \\ [a \rightarrow aa[\\ [] \rightarrow \varepsilon. \end{cases}$$

מהן המילים שניתן לצור בעזרת הדקדוק הזה, או במילים אחרות: מהי השפה של הדקדוק?

פתרון:

תשובה:

$$L(G) = \{a^n \mid n = 2^k, k \geq 1\}.$$

הסבר:

$$\begin{array}{ccccccc} S & \xrightarrow{S \rightarrow [S]} & [S] & \xrightarrow{S \rightarrow [S]} & \dots & \xrightarrow{S \rightarrow [S]} & [^k S]^k \\ & \xrightarrow{S \rightarrow a} & [^k a]^k & & & & \\ & \xrightarrow{[a \rightarrow aa[} & [^{k-1} aa [^k] & \xrightarrow{[] \rightarrow \varepsilon} & [^{k-1} aa]^{k-1} & & \\ & \xrightarrow{[a \rightarrow aa[} & [^{k-2} aa [a]^{k-1} & \xrightarrow{[a \rightarrow aa[} & [^{k-2} aaaa [^k]^{k-1} & \xrightarrow{[] \rightarrow \varepsilon} & [^{k-2} aaaa]^{k-2} \\ & & & & \dots & \rightarrow & a^k \end{array}$$

5.6 דוגמה

בנו דקדוק כללי אשר יוצר את הפשה

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

פתרון:

$$G = (\{S\}, \{a, b\}, R, \{S\})$$

$$S \rightarrow abS, \quad (1)$$

$$ab \rightarrow ba, \quad (2)$$

$$ba \rightarrow ab, \quad (3)$$

$$S \rightarrow \varepsilon. \quad (4)$$

$$S \xrightarrow{1} abS \xrightarrow{1} ababS \xrightarrow{2} baabS \xrightarrow{4} baab$$

שימו לב: בדקדוק כללי אנו מאפשרים גם כלליצירה בהם בצד שמאל יש רק טרמינלים. לכן, יתכן גם שנמשיך ונפתח מחרוזתשכולה טרמינללים. למשל

$$S \xrightarrow{1} abS \xrightarrow{1} ababS \xrightarrow{4} abab \xrightarrow{2} baab$$

נשאל שאלה כללית:

- אלו שפות ניתן לצור בעזרת דקדוק כללי?
- האם יש שפות שלא ניתן לצור בעזרת דקדוק כללי?
- האם יש מודל חישובי שמקבל שפות שנוצרות ע"י דקדוקים כלליים?

דוגמה 5.7

בנו דקדוק כללי שיוצר את השפה

$$w = \{w \in \{a, b, c\}^* \mid w = a^n b^n c^n\}$$

פתרון:

נראה דקדוק כללי עבור שפה זו.

שפה זו אינה חסרת הקשר.

לכן, לא ניתן לבנות עבורה דקדוק חסר הקשר.

אנו נבנה לה דקדוק כלל.

נגזור את האותיות a, b, c יחד.

נעשה זאת בצורה כזו שכדי לסיים את תהליך הגזירה יש לסדר את האותיות בסדר הרצוי:

תחילה a ,

אחר כך b ,

ובסוף c .

$$S \rightarrow S'] \quad (1)$$

$$S' \rightarrow aS'bC \mid \varepsilon \quad (2)$$

$$Cb \rightarrow bC \quad (3)$$

$$C] \rightarrow]c \quad (4)$$

$$] \rightarrow \varepsilon \quad (5)$$

$$\begin{array}{llllll} S & \xrightarrow{1} & S'] & \xrightarrow{2} & aS'bC] & \xrightarrow{2} & aaS'bCbC] & \xrightarrow{2} & aaaS'bCbCbC] \\ & \xrightarrow{3} & aaaS'bbCCbC] & \xrightarrow{3} & aaaS'bbCbCC] & \xrightarrow{3} & aaaS'bbbCCC] & & \\ & \xrightarrow{4} & aaaS'bbbCC]c & \xrightarrow{4} & aaaS'bbbC]cc & \xrightarrow{4} & aaaS'bbb]ccc & & \\ & \xrightarrow{5} & aaaS'bbbccc & \xrightarrow{1} & aaabbbccc & & & & \end{array}$$

דוגמה 5.8

בנו דקדוק כללי אשר יוצר את שפת המילים

$$L = \{uu \mid u \in \{a, b\}^*\}$$

פתרון:

דוגמא זאת תמחיש ביצד דקדוק כללי יכול "לפעול בדומה" למכונת טיורינג.

בדקדוק נשתמש במשתנים וכללי גזירה שיאפשרו מעין תנועה על גבי המחרוזת הנגזרת, בדומה לתנועת הראש של מכונת טיורינג על גבי הסרט.

$S \rightarrow [H\{$	כלל גזירה יחיד מהמשתנה ההתחלתי. המשתנה H ידמה את הראש של המ"ט ש"יזוז" מצד לצד על המחרות הנגזרות. הסוגר המרובע $[$ מסמן את הקצה השמאלי של המילה השמאלית. הסוגר המסולסל $\{$ מסמן את הקצה השמאלי של המילה הימינית.	1
$[H \rightarrow [aH_a$	כלל זה מאפשר הוספת אות a לקצה השמאלי של המילה השמאלית. המשתנה H מוחלף במשתנה H_a כדי "לזכור" שיש עכשיו להוסיף גם במחרות הימינית. (בדומה לזיכרונות של מ"ט).	2
$H_a a \rightarrow aH_a$	כלל זה מאפשר לראש "לזוז" ימינה.	3
$H_a \{ \rightarrow H\{a$	כאשר המשתנה H_a "יגיע" לסוגר המסולסל, הוא יגזור אות a נוספת מימין לסוגר, שהוא הקצה השמאלי של המחרות הימינית. כך אפשרנו להוסיף שתי אותיות a : אחת מימין לסוגר $[$ ואחת תואם ימין לסוגר $\{$. כלומר אות a בקצה השמאלי של כל אחת המחרות.	4
$aH \rightarrow Ha$	כעת צריך לאפשר למשתנה H לחזור שמאלה על גבי האותיות שבין הסוגרים, עד הסוגר $[$.	5

ברגע "שהראש" H חזר לתחילת המחרות ועומד ליד הסוגר $[$ עוברים על השלבים 2-5 שוב. בסבב הבא נחק בחשבון גם יצירה של שתי אותיות b .

$[H \rightarrow [bH_b$	כלל זה מאפשר הוספת אות b לקצה השמאלי של המילה השמאלית. המשתנה H מוחלף במשתנה H_b כדי "לזכור" שיש עכשיו להוסיף גם במחרות הימינית.	'2
$H_a a \rightarrow aH_a$ $H_a b \rightarrow bH_a$ $H_b a \rightarrow aH_b$ $H_b b \rightarrow bH_b$	כללים האלה מאפשרים לראש "לזוז" ימינה.	'3
$H_b \{ \rightarrow H\{b$	כאשר המשתנה H_b "יגיע" לסוגר המסולסל, הוא יגזור אות b נוספת מימין לסוגר, שהוא הקצה השמאלי של המחרות הימינית. כך אפשרנו להוסיף שתי אותיות b : אחת מימין לסוגר $[$ ואחת תואם ימין לסוגר $\{$. כלומר אות b בקצה השמאלי של כל אחת המחרות.	'4
$bH \rightarrow Hb$	כעת צריך לאפשר למשתנה H לחזור שמאלה על גבי האותיות שבין הסוגרים, עד הסוגר $[$.	'5

בכדי לסיים את הגזירה יש להפטר ממשני העזר על ידי הכללים הבאים:

$H \rightarrow \varepsilon$ $[\rightarrow \varepsilon$ $\{ \rightarrow \varepsilon$	הכללים האלה מאפשרים להעלים את המשתנים $H, [, \{$	6
--	--	---

למשל:

$$\begin{array}{ccccccc}
 S & \xrightarrow{1} & [H\{ & \xrightarrow{2} & [aH_a\{ & \xrightarrow{4} & [aH\{a & \xrightarrow{5} & [Ha\{a \\
 & & & \xrightarrow{2} & [aH_a a\{a & \xrightarrow{3} & [aaH_a\{a & \xrightarrow{4} & [aaH\{aa & \xrightarrow{5} & [Haa\{aa \\
 & & & \xrightarrow{2} & [bH_b aa\{aa & \xrightarrow{3} & [baaH_b\{aa & \xrightarrow{4} & [baaH\{baa & \xrightarrow{5} & [Hbaa\{baa \\
 & & & & & & & & & & & \xrightarrow{6} & baabaa
 \end{array}$$

5.5 דקדוקים כלליים ומכונת טיורינג

משפט 5.4 קדוקים כלליים ומכונת טיורינג

תהי L שפה. L קבילה אם ורק אם קיים דקדוק כללי G כך ש- $L(G) = L$.

הוכחה: כיוון ראשון.

נוכיח שאם קיים דקדוק כללי G אז $L(G)$ קבילה.

נניח שקיים דקדוק כללי G . נוכיח כי $L(G)$ קבילה על ידי להוכיח שקיימת תוכנית מחשב P שמקבלת $L(G)$.

נתון דקדוק כללי G . נבנה תוכנית מחשב שמקבלת את $L(G)$.
יהי הקלט $w \in L(G)$, כלומר w מילה בשפה $L(G)$.

(1) $u = S$.

(2) repeat

- פצל באופן לא דטרמיניסטי את u ל- xyz .
- בחר באופן לא דטרמיניסטי גזירה $t \rightarrow v$ של G .
- אם $y \neq t$ דחה.
- $u = xvz$
- אם $w = u$ קבל.

כיוון שני.

נוכיח שאם $L(G)$ קבילה אז קיים דקדוק כללי G .

ז"א, נניח שקיימת מ"ט M שמקבלת את השפה L . נוכיח שקיים דקדוק כללי G כך ש- $L(G) = L(M)$, כלומר השפה המתקבלת על ידי M היא השפה של דקדוק כללי G .

נתונה מ"ט M בעלת הטבלת המעברים להלן. נבנה דקדוק כללי G שמממש אותם צעדים.

תזוזה	כתיבה	מצב חדש	סימן	מצב
R	a	q_0	a	q_0
R	a	q_1	b	q_0
L	$_$	acc	$_$	q_0
L	b	q_0	a	q_1
L	b	q_1	$b, _$	q_1

לפי הטבלת המעברים קיים הצעד

$$q \ q_0 \ b \ a \ b \vdash_M \ a a q_1 \ a b$$

נניח שבדקדוק כללי G קיים אותו הצעד

$$q \ q_0 \ b a b \xrightarrow{G} \ a a q_1 \ a b$$

ניתן לממש צעד זה על ידי הכלל

$$q_0 b \rightarrow a q_1$$

באופן כללי,

- עבור כל פונקצית המעברים של M שגוררת תזוזה ימינה מצורה

$$\delta(q, \sigma) = (p, \pi, R)$$

נממש מעבר זה על ידי כלל של הדקדוק G מצורה

$$q\sigma \rightarrow \pi p.$$

- עבור כל פונקצית המעברים של M שגוררת תזוזה שמאלה מצורה

$$\delta(q, \sigma) = (p, \pi, L)$$

אז לכל $\tau \in \Gamma$ ב- G נממש מעבר זה על ידי הכלל

$$\tau q\sigma \rightarrow p\tau\pi.$$



5.6 ההיררכיה של חומסקי

משפחת שפות	דקדוק	מודל חישובי
קבילות	כללי	מכונת טיורינג
חסרות הקשר	חסר הקשר	אוטומט מחסנית
רגולריות	רגולרי	אוטומט סופי

- היררכיה של חומסקי קושרת לנו בין משפחות של שפות דקדוקים ומודלים חישוביים.
- בתחתית ההיררכיה נמצאות השפות הרגולריות שנוצרות על ידי דקדוקים רגולריים ומתקבלות על ידי אוטומטים סופיים.
- מעליהן נמצאות השפות חסרות ההקשר שנוצרות על ידי דקדוקים חסרי הקשר ומתקבלות על ידי אוטומטי מחסנית.
- ומעליהן נמצאות השפות הקבילות שנוצרות על ידי דקדוקים כלליים ומתקבלות על ידי מכונות טיורינג.
- כל רמה בהיררכיה מכילה ממש את הרמה שמתחתיה.
- * כל שפה רגולרית היא גם חסרת הקשר, אבל יש שפות חסרות הקשר שאינן רגולריות.
- * כל שפה חסרת הקשר היא קבילה, אבל יש שפות קבילות שאינן חסרות הקשר.

5.7 כל שפה חסרת הקשר הינה כריעה

לפי ההיררכיה של חומסקי אנחנו יודעים לקבוע שכל שפה חסרת הקשר היא קבילה.

האם כל שפה חסרת הקשר הינה כריעה?

משפט 5.5

יהי $G = (V, \Sigma, S, R)$ דקדוק חסר הקשר ו- $w \in L(G)$. אזי קיים עץ גזירה של w שעומקו לכל היותר $(|V| + 1)(|w| + 1)$.

הוכחה: יהי T עץ הגזירה הקטן ביותר (מבחינת מספר קדקודים) של w .

בשלילה נניח שב- T יש מסלול מהשורש לעלה שמכיל לפחות $(|V| + 1)(|w| + 1)$ קודקודים פנימיים.

נסמן מסלול זה ב-

$$p = (u_1, u_2, \dots, u_m).$$

עבור קודקוד u_i במסלול נסמן ב- $s(u_i)$ את תת-המחרוזת של w שנוצרת מ- u_i .

מתקיים ש- $s(u_{i+1})$ היא תת-מחרוזת של $s(u_i)$. אומרים שקדקוד u_i הוא משמעותי אם $s(u_i)$ מכיל ממש את $s(u_{i+1})$.

כל קודקוד משמעותי מוסיף לפחות אות אחת ל- w .

לכן, ישנם לכל היותר $|w|$ קודקודים משמעותיים.

לכן, ברצף הקודקודים הפנימיים (u_1, u_2, \dots, u_m) שאורכן לפחות $(|V| + 1)(|w| + 1)$, בהכרח ישנו תת רצף $(u_i, u_{i+1}, \dots, u_{i+|V|})$ באורך $|V| + 1$, שבו כל הקודקודים לא משמעותיים.

ברצף זה בהכרח ישנם שני קודקודים, נאמר u_j, u_k , $j < k$ שמסומנים עם אותו משתנה.

לכן בעץ הגזירה, ניתן להחליף את הקודקוד u_j יחד עם כל תת העץ שמתחתיו- בקודקוד u_k , יחד עם כל תת העץ שמתחתיו.

כיוון שכל הקודקודים שבין u_j ל- u_k (כולל) הם לא משמעותיים, החלפה זו לא משנה את המחרוזת הנוצרת.

כלומר, העץ החדש גם הוא עץ הגזירה עבור w .

בסתירה להנחת המינימליות של העץ.

משפט 5.6

כל שפה חסרת הקשר היא כריעה.

הוכחה: בהינתן דקדוק חסר הקשר $G = (V, \Sigma, S, R)$, התוכנית הלא דטרמיניס הבאה מכריעה את $L(G)$.

קלט: מחרוזת w .

פלט: כן או לא.

(1) נחש עץ גזירה של הדקדוק G בעומק לכל היותר $(|V| + 1)(|w| + 1)$.

(2) בדוק האם העץ יוצר את המחרוזת w . אם כן, החזר "כן" איתר החזר "לא".

שני שלבי התוכנית בהכרח מסתיימים. לכן, זו תוכנית להכרעה. ישנו חישוב שמחזיר "כן" אם ורק אם $w \in L(G)$ לכן זו תוכנית שמכריעה את $L(G)$.

שיעור 6

תכונות סגירות של R ו- RE 6.1 הגדרה של השפות R ו- RE הגדרה 6.1 R

אוסף השפות הכריעות מסומן R ומוגדר

$$R = \{L \subseteq \Sigma^* : L \text{ מכריעה את } L\}.$$

הגדרה 6.2 RE

אוסף השפות הקבילות מסומן RE ומוגדר

$$RE = \{L \subseteq \Sigma^* : L \text{ מקבלת את } L\}.$$

6.2 היחס בין הכרעה וקבלה

למה 6.1 היחס בין הכרעה וקבלה

אם $L \in RE$ וגם $\bar{L} \in RE$ אזי $L \in R$.

הוכחה: תהי M מ"ט המקבלת את L ותהי \bar{M} מ"ט המקבלת את \bar{L} .

נבנה מ"ט D המכריעה את L .



$D = \text{על קלט } w$:

(1) D מעתיקה את w לסרט נוסף.

(2) מריצה במקביל את M על w ואת \bar{M} על העותק של w .

• אם M מקבלת $\Leftrightarrow D$ מקבלת.

• אם \bar{M} מקבלת $\Leftrightarrow D$ דוחה.

• אם M דוחה $\Leftrightarrow D$ דוחה.

• אם \bar{M} דוחה $\Leftrightarrow D$ מקבלת.

נוכיח כי D מכריעה את L .

אם $w \in L$

$w \in L(M) \Leftrightarrow$

$\Leftrightarrow (M \text{ מקבלת את } w) \text{ או } (\bar{M} \text{ דוחה את } w)$

$\Leftrightarrow D \text{ עוצרת ומקבלת את } w$.

אם $w \notin L$

$w \in \bar{L} \Leftrightarrow$

$\Leftrightarrow w \in L(\bar{M})$

$\Leftrightarrow (\bar{M} \text{ מקבלת את } w) \text{ או } (M \text{ דוחה את } w)$

$\Leftrightarrow D \text{ עוצרת ודוחה את } w$.

■

6.3 סגירות של שפות כריעות ושפות קבילות

משפט 6.1 סגירות של השפות הכריעות

R סגורה תחת:

(1) איחוד

(2) חיתוך

(3) משלים

(4) שרשור

(5) סגור קליין

משפט 6.2 סגירות של השפות הקבילות

RE סגורה תחת:

(1) איחוד

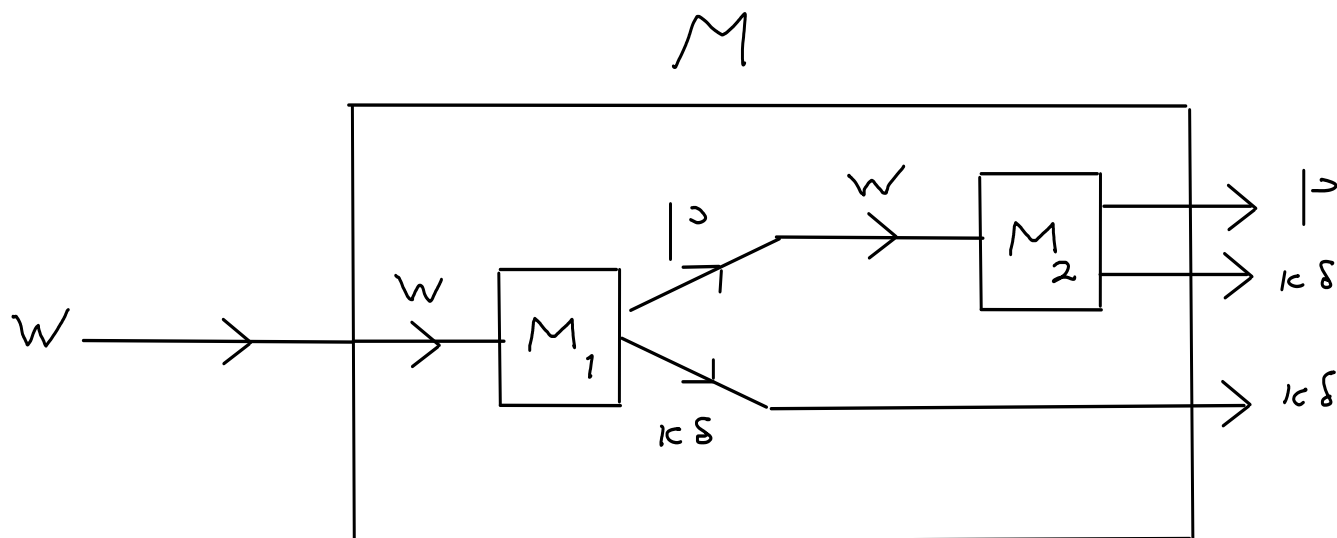
(2) חיתוך

(3) שרשור

(4) סגור קלין

הוכחה:

(1) חיתוך:

(א) סגור תחת חיתוךנוכיח כי לכל שתי שפות $L_1, L_2 \in R$ מתקיים $L_1 \cap L_2 \in R$.תהי M_1 ו- M_2 מ"ט המכריעות את L_1 ו- L_2 בהתאמה. נבנה מ"ט M המכריעה את $L_1 \cap L_2$.תאור הבנייה $M = \text{על קלט } w$:(1) M מעתיקה את w לסרט נוסף.(2) M מריצה את M_1 על w .• אם M_1 דוחה $M \Leftarrow$ דוחה.• אחרת M מריצה את M_2 על העותק של w ועונה כמוה.נכונות:נוכיח כי M מכריעה את $L_1 \cap L_2$.אם $w \in L_1 \cap L_2$ $w \in L_2$ וגם $w \in L_1 \Leftarrow$

M_1 מקבלת את w וגם M_2 מקבלת את $w \Leftarrow$
 M מקבלת את $w. \Leftarrow$
 אם $w \notin L_1 \cap L_2$
 $w \notin L_2$ או $w \notin L_1 \Leftarrow$
 M_1 דוחה את w או M_2 דוחה את $w \Leftarrow$
 M דוחה את $w. \Leftarrow$

(ב) RE סגורה תחת חיתוך

נוכיח כי לכל שתי שפות $L_1, L_2 \in RE$ מתקיים $L_1 \cap L_2 \in RE$.

תהיינה M_1 ו- M_2 שתי מכונות טיורינג המקבלות את L_1 ו- L_2 בהתאמה.
 נבנה מ"ט M המקבלת את $L_1 \cap L_2$ באותו אופן כמו (א).

(2) איחוד:**(א) R סגורה תחת איחוד**

נוכיח כי לדל שתי שפות $L_1, L_2 \in R$ מתקיים $L_1 \cup L_2 \in R$.

תהיינה M_1 מ"ט המכריעה את L_1 ו- M_2 מ"ט המכריעה את L_2 .
 נבנה מ"ט M המכריעה את $L_1 \cup L_2$.

תאור הבנייה

$M =$ על קלט w :

(1) M מעתיקה את w לסרט נוסף.

(2) M מריצה את M_1 על w .

• אם M_1 מקבלת $M \Leftarrow$ מקבלת.

• אחרת, M מריצה את M_2 על העותק של w ועונה כמוה.

(ב) RE סגורה תחת איחוד

נוכיח כי לכל שתי שפות $L_1, L_2 \in RE$ מתקיים $L_1 \cup L_2 \in RE$.

תהיינה M_1 מ"ט המקבלת את L_1 ו- M_2 מ"ט המקבלת את L_2 .
 נבנה מ"ט א"ד M המקבלת את $L_1 \cup L_2$.

תאור הבנייה

$M =$ על קלט w :

(1) M בוחרת באופן א"ד $i \in \{1, 2\}$

(2) M מריצה את M_i על w ועונה כמוה.

(3) שרשור:**(א) R סגורה תחת שרשור**

נוכיח כי לכל שתי שפות $L_1, L_2 \in R$ מתקיים $L_1 \cdot L_2 \in R$ כאשר

$$L_1 \cdot L_2 = \{w = w_1 w_2 \mid w_1 \in L_1 \wedge w_2 \in L_2\}.$$

תהייה M_1 מ"ט המכריעה את L_1 ו- M_2 מ"ט המכריעה את L_2 .
נבנה מ"ט א"ד M המכריעה את $L_1 \cdot L_2$.

תאור הבנייה

$$M = \text{על קלט } w:$$

(1) M בוחרת באופן א"ד חלוקה של w ל- $w_1 w_2$.

(2) M מריצה את M_1 על w_1 .

• אם M_1 דוחה $\Leftarrow M$ דוחה.

• אחרת, M מריצה את M_2 על w_2 ועונה כמוה.

(ב) RE סגורה תחת שרשור

RE סגורה תחת שרשור באותו אופן כמו ב- (א)

(4) * קליני

(א) R סגורה תחת * קליני

נוכיח כי לכל שפה L :

$$L \in R \Rightarrow L^* \in R$$

כאשר

$$L^* = \{w = w_1 w_2 \cdots w_k \mid \forall 1 \leq i \leq k, w_i \in L\}.$$

תהי M מ"ט המכריעה את L .

נבנה מ"ט M^* א"ד המכריעה את L^* .

תאור הבנייה

$$M^* = \text{על קלט } w:$$

(1) אם $w = \varepsilon$ אז M^* מקבלת.

(2) אחרת M^* בוחרת באופן א"ד חלוקה של w ל- $w_1 \cdots w_k$.

(3) לכל $1 \leq i \leq k$:

M^* מריצה את M על w_i .

• אם M דוחה את w_i אז M^* דוחה.

• אחרת חוזרים לשלב (3).

(4) אם M קיבלה את כל המחזוריות $\{w_i\}$ אזי M^* מקבלת.

(ב) RE סגורה תחת * קליני

(5) משלים

(א) R סגורה תחת המשלים

נוכיח כי

$$L \in R \Rightarrow \bar{L} \in R,$$

כאשר

$$\bar{L} = \{w \in \Sigma^* \mid w \notin L\}.$$

תהי M מ"ט המכריעה את L .נבנה מ"ט \bar{M} המכריעה את \bar{L} .

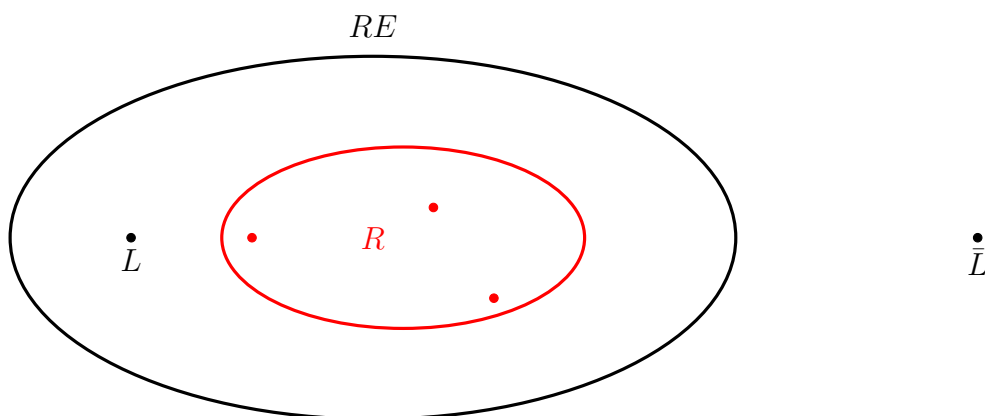
$$\bar{M} = \text{על קלט } w$$

(1) \bar{M} מריצה את M על w .• אם M מקבלת $\bar{M} \Leftarrow$ דוחה.• אם M דוחה $\bar{M} \Leftarrow$ מקבלת.(ב) RE אינה סגורה תחת המשלים

■

משפט 6.3 RE אינה סגורה תחת המשלים

$$L \in RE \setminus R \Rightarrow \bar{L} \notin RE.$$



הוכחה:

נניח כי $L \in RE \setminus R$ ונניח בשלילה כי $\bar{L} \in RE$.אזי לפי טענת עזר (למה 6.1), $L \in R$ וזו סתירה.

■

הגדרה 6.3 $CoRE$

$$CoRE = \{L \subseteq \Sigma^* \mid \bar{L} \in RE\}.$$



$$RE \cap Co RE = R.$$

6.4 סגירות של שפות כריעות תחת DROPOUT

משפט 6.4 סגירות של שפות כריעות תחת DROPOUT

תהי L שפי כריעה ותהי $DROPOUT(L)$ השפה הבאה:

$$DROPOUT(L) = \{uv \mid uv \in L, \sigma \in \Sigma^* : u\sigma v \in L\}$$

אזי השפה $DROPOUT(L)$ גם כריעה.

הוכחה:

השפה L כריעה אזי קיימת מכונת טיורינג M_L המכריעה אותה. נבנה מכונת טיורינג $M_{DROPOUT}$ המכריעה את $DROPOUT(L)$ באופן הבא:

בניית המכונת טיורינג

$$M_{DROPOUT} = \text{על כל קלט } w:$$

(1) בוחרת חילוק של המילה w לשרשור של שני מילים

באופן אי-דטרמיניסטי כך ש: $w = uv$.

(2) בוחרת אות $\sigma \in \Sigma^*$ באופן אי-דטרמיניסטי ובונה את המילה $w = u\sigma v$.

(3) מריצה M_L על המילה $w = u\sigma v$ ועונה כמורה.

הוכת הנכונות

נניח ש- $w \in DROPOUT(L)$.

\Leftarrow קיימים מילים $u, v \in L$ וקיים $\sigma \in \Sigma^*$ כך ש- $u\sigma v \in L$.

\Leftarrow קיימת ריצה של $M_{DROPOUT}$ עבורה $M_{DROPOUT}$ תבחר חילוק $w = uv$ ו- $M_{DROPOUT}$ תבחר $\sigma \in \Sigma^*$ כך ש- $u\sigma v \in L$.

\Leftarrow $M_{DROPOUT}$ תקבל את w .

נניח ש- $w \notin DROPOUT(L)$.

\Leftarrow לא קיימים מילים $u, v \in L$ ו- $\sigma \in \Sigma^*$ כך ש- $u\sigma v \in L$.

\Leftarrow כל ריצה של $M_{DROPOUT}$ עוברת ל- q_{rej} .

\Leftarrow $M_{DROPOUT}$ תדחה את w .

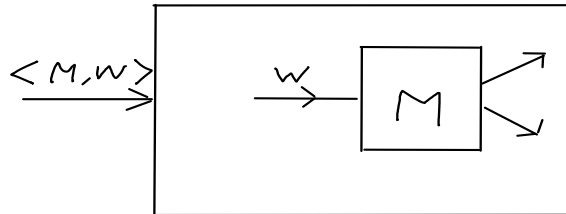
6.5 קידוד של מ"ט דטרמיניסטית

הגדרה 6.4 קידוד של מ"ט

בהינתן קבוצה O של עצמים מופשטים (למשל מכונת טיורינג, תוכנית מחשב, גרף). הקידוד של O , מסומן $\langle O \rangle$, הוא מיפוי של O אל מחרוזת מעל אלפבית סופי שיש בו לפחות שני סימנים.

במידה ויש רב עצמים O_1, \dots, O_k נסמן את הקידוד שלהם $\langle O_1, O_2, \dots, O_k \rangle$.

6.6 מ"ט אוניברסלית U



מ"ט אוניברסלית U מקבלת כקלט זוג, קידוד של מ"ט $\langle M \rangle$ וקידוד של מילה $\langle w \rangle$, ומבצעת סימוציה של ריצה של M על w ועונה בהתאם.

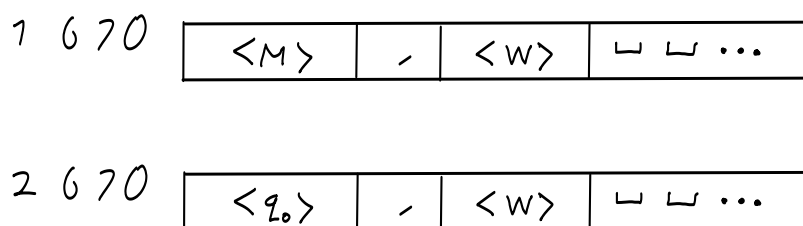
תאור הפעולה של U

$U = \text{על קלט } x$:

(1) בודקת האם x הוא קידוד של מ"ט $\langle M \rangle$ וקידוד של מילה $\langle w \rangle$.

• אם לא \Leftarrow דוחה.

(2) מבצעת סימוציה של M על w :



- רושמת את הקונפיגורציה ההתחלתית $q_0 w$ על סרט 2.
- מחשבת את הקונפיגורציה הבאה בעזרת טבלת המעברים.
- בסוף כל מעבר בין שתי קונפיגורציות, U בודקת האם המצב הנוכחי הוא q_{acc} .
- אם כן U עוצרת ומקבלת.

- * אחרת U בודקת האם המצב הוא q_{rej} .
- * אם כן U עוצרת ודוחה.
- * אחרת U ממשיכה לקונפיגורציה הבאה.

מהי השפה של U ?

לכל x :

(1) אם $U \Leftarrow x \neq \langle M, w \rangle$ דוחה את x .

(2) אם $x = \langle M, w \rangle$:

- אם M מקבלת $w \Leftarrow U$ מקבלת את x .
- אם M דוחה את $w \Leftarrow U$ דוחה את x .
- אם M לא עוצרת על $w \Leftarrow U$ לא עוצרת על x .

$$L(U) = \{ \langle M, w \rangle \mid w \in L(M) \} .$$

הגדרה 6.5 L_{acc}

$$L_{acc} = \{ \langle M, w \rangle \mid w \in L(M) \} \in RE \setminus R$$

הגדרה 6.6 L_{halt}

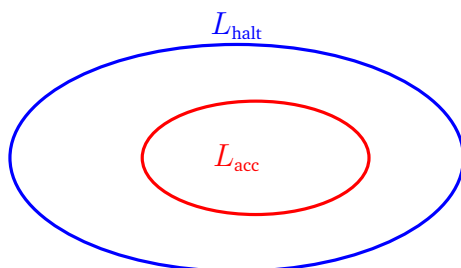
$$L_{halt} = \{ \langle M, w \rangle \mid M \text{ עוצרת על } w \} \in RE \setminus R$$

הגדרה 6.7 L_d

$$L_d = \{ \langle M \rangle \mid \langle M \rangle \notin L(M) \} \notin RE$$

אבחנה:

$$L_{acc} \subseteq L_{halt} .$$



משפט 6.5

$$L_{\text{acc}} \in RE.$$

הוכחה: מכיוון ש- $L(U) = L_{\text{acc}}$, U מקבלת את L_{acc} ולכן $L_{\text{acc}} \in RE$. ■

משפט 6.6

$$L_{\text{halt}} \in RE.$$

הוכחה: נבנה מ"ט U' שהיא למעשה U פרט למקום שבו U עצרה ודחתה, U' תעצור ותקבל.

נוכיח כי U' מקבלת את L_{halt} :

אם $x \in L_{\text{halt}}$

$x = \langle M, w \rangle \Leftarrow$ M עוצרת על w

U' עוצרת ומקבלת את x .

אם $x \notin L_{\text{halt}}$ שני מקרים:

• $x = \langle M, w \rangle \Leftarrow U'$ דוחה את x .

• $x = \langle M, w \rangle \Leftarrow$ M לא עוצרת על $w \Leftarrow U'$ לא עוצרת על x .

■

שיעור 7

אי-כריעות

7.1 השפות L_d, L_{halt}, L_{acc} לא כריעות

הגדרה 7.1 L_{acc}

$$L_{acc} = \{ \langle M, w \rangle \mid w \in L(M) \} \in RE \setminus R$$

הגדרה 7.2 L_{halt}

$$L_{halt} = \{ \langle M, w \rangle \mid M \text{ עוצרת על } w \} \in RE \setminus R$$

הגדרה 7.3 L_d

$$L_d = \{ \langle M \rangle \mid \langle M \rangle \notin L(M) \} \notin RE$$

משפט 7.1 $L_{acc} \in RE$

$$L_{acc} \in RE.$$

הוכחה: מכיוון ש- $L(U) = L_{acc}$, כאשר U המכונת טיורינג האוניברסלית אשר מקבלת את L_{acc} , לכן $L_{acc} \in RE$.

משפט 7.2 $L_{halt} \in RE$

$$L_{halt} \in RE.$$

הוכחה: נבנה מ"ט U' שהיא למעשה U פרט למקום שבו U עצרה ודחתה, U' תעצור ותקבל.

נוכיח כי U' מקבלת את L_{halt} :

אם $x \in L_{halt}$

$\Leftarrow x = \langle M, w \rangle$ ו- M עוצרת על w

$\Leftarrow U'$ עוצרת ומקבלת את x .

אם $x \notin L_{\text{halt}} \Leftarrow$ שני מקרים:

• $U' \Leftarrow x \neq \langle M, w \rangle$ דוחה את x .

• $x = \langle M, w \rangle$ ו- M לא עוצרת על $w \Leftarrow U'$ לא עוצרת על x .

■

משפט 7.3 $L_d \notin RE$

$$L_d \notin RE .$$

הוכחה:

נניח בשלילה כי $L_d \in RE$.

$\Leftarrow \exists$ מ"ט M_d המקבלת את L_d .

$\Leftarrow L(M_d) = L_d$.

נבדוק ריצה של M_d על $\langle M_d \rangle$:

• אם $\langle M_d \rangle \in L(M_d) \Leftarrow \langle M_d \rangle \notin L_d \Leftarrow L(M_d) \neq L_d$.

• אם $\langle M_d \rangle \notin L(M_d) \Leftarrow \langle M_d \rangle \in L_d \Leftarrow L(M_d) \neq L_d$.

■

בשני המקרים קיבלנו סתירה לכך ש- $L(M_d) = L_d$ ולכן $L_d \notin RE$.

משפט 7.4 L_{acc} לא כריעה

$$L_{\text{acc}} = \{ \langle M, w \rangle \mid w \in L(M) \} \notin R .$$

הוכחה:

נניח בשלילה כי $L_{\text{acc}} \in R$ ותהי M_{acc} המ"ט המכריעה את L_{acc} .

נשתמש ב- M_{acc} כדי לבנות מ"ט M_d המכריעה את L_d (בסתירה לכך ש- $L_d \notin RE$ כפי שהוכחנו במשפט 7.3).

$$L_d = \{ \langle M, w \rangle \mid \langle M \rangle \notin L(M) \} .$$



התאור של M_d

$M_d =$ על קלט x :

(1) בודקת האם $x = \langle M \rangle$. אם לא \Leftarrow דוחה.

(2) מחשבת את $\langle x \rangle = \langle \langle M \rangle \rangle$.

(3) מריצה את M_{acc} על הזוג $\langle M, \langle M \rangle \rangle$:

• אם M_{acc} מקבלת $\Leftarrow M_d$ דוחה.

• אם M_{acc} דוחה $\Leftarrow M_d$ מקבלת.

כעת נוכיח כי M_d מכריעה את L_d :

אם $x \in L_d$

$\Leftarrow x = \langle M \rangle$ ו- $\langle M \rangle \notin L(M)$

$\Leftarrow M_{acc}$ דוחה את הזוג $\langle M, \langle M \rangle \rangle$

$\Leftarrow M_d$ מקבלת את x .

אם $x \notin L_d$ שני מקרים:

מקרה (1): $\Leftarrow x \neq \langle M \rangle$ M_d דוחה את x .

מקרה (2): $\Leftarrow x = \langle M \rangle$ ו- $\langle M \rangle \in L(M)$

$\Leftarrow M_{acc}$ מקבלת את זוג $\langle M, \langle M \rangle \rangle$

$\Leftarrow M_d$ דוחה את x .

משפט 7.5 L_{halt} לא כריעה

$$L_{\text{halt}} = \{ \langle M, w \rangle \mid M \text{ עוצרת על } w \} \notin R.$$

הוכחה:

נניח בשלילה כי $L_{\text{halt}} \in R$ ותהי M_{halt} מ"ט המכריעה את L_{halt} .נשתמש ב- M_{halt} כדי לבנות מ"ט M_{acc} המכריעה את L_{acc} (בסתירה לכך ש- $L_{\text{acc}} \notin R$ כפי שהוכחנו במשפט 7.4).התאור של M_{acc} M_{acc} = על קלט x :(1) מריצה את M_{halt} על x .

- אם M_{halt} דוחה $\Leftarrow M_{\text{acc}}$ דוחה.
- אם M_{halt} מקבלת $\Leftarrow M_{\text{acc}}$ מריצה את U על x ועונה כמוה.

אבחנהנוכיח כי M_{acc} מכריעה את L_{acc} :אם $x \in L_{\text{acc}}$

$$\Leftarrow x = \langle M, w \rangle \text{ ו- } \langle w \rangle \in L(M)$$

$$\Leftarrow M_{\text{halt}} \text{ מקבלת את } x \text{ וגם } U \text{ מקבלת את } x$$

$$\Leftarrow M_{\text{acc}} \text{ מקבלת את } x.$$

אם $x \notin L_{acc} \Leftarrow$ שני מקרים:

מקרה (1): $x \neq \langle M, w \rangle$

M_{halt} דוחה את $x \Leftarrow$

M_{acc} דוחה את $x \Leftarrow$

מקרה (2): $x = \langle M, w \rangle$ ו- $\langle w \rangle \notin L(M) \Leftarrow$ שני מקרים:

מקרה (א): M לא עוצרת על $w \Leftarrow M_{halt}$ דוחה את $x \Leftarrow M_{acc}$ דוחה את x .

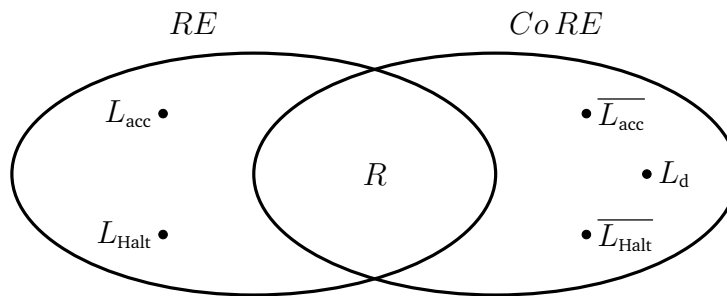
מקרה (ב): M דוחה את $w \Leftarrow M_{halt}$ מקבלת את x אבל U דוחה את $x \Leftarrow M_{acc}$ דוחה את x .

הראנו כי M_{acc} מכריעה את L_{acc} בסתירה לכך ש- $L_{acc} \notin R$.

לכן $L_{halt} \notin R$.

משפט 7.6

$$\begin{aligned} L_{acc} \in RE \setminus R &\Rightarrow \bar{L}_{acc} \notin RE, \\ L_{halt} \in RE \setminus R &\Rightarrow \bar{L}_{halt} \notin RE, \\ L_d \notin RE \setminus R. \end{aligned}$$



7.2 השפה L_E לא כריעה

הגדרה 7.4 השפה L_E

$$L_E = \{ \langle M \rangle \mid L(M) = \emptyset \}.$$

משפט 7.7 $L_E \notin R$

$$L_E \notin R.$$

כלומר L_E לא כריעה.

הוכחה:

נניח בשלילה כי L_E כריעה. אז נבנה מ"ט M_{acc} המכריעה את L_{acc} באופן הבא.

בנייה של M_w

ראשית נגדיר את המ"ט M_w :

$$M_w = \text{על כל קלט } x:$$

(1) אם $x \neq w \Leftarrow$ דוחה.

(2) אם $x = w$ אז מריצה M על w ועונה כמוה.

אבחנה

אם $x = w$ ו- M מקבלת את w אז $L(M_w) = \Sigma^*$.

אם $x \neq w$ או אם $x = w$ ו- M דוחה את w אז $L(M_w) = \emptyset$.

בנייה של M_{acc}

נניח כי קיימת מ"ט M_E המכריעה את L_E . אז נבנה מ"ט M_{acc} המכריעה את L_{acc} :

$$M_{\text{acc}} = \text{על כל קלט } x:$$

(1) אם $x \neq \langle M, w \rangle \Leftarrow$ דוחה.

(2) אם $x = \langle M, w \rangle$, בעזרת התאור $\langle M, w \rangle$, בונה מ"ט M_w .

(3) מריצה M_E על $\langle M_w \rangle$:

(4) • אם M_E מקבלת \Leftarrow דוחה.

• אם M_E דוחה \Leftarrow מקבלת.

נכונות

אם $x \in L_{\text{acc}} \Leftarrow x = \langle M, w \rangle \Leftarrow x = \langle M, w \rangle$ ו- $w \in L(M) \Leftarrow L(M_w) = \Sigma^* \neq \emptyset \Leftarrow M_E$ דוחה $\langle M_w \rangle$ מקבלת. M_{acc}

אם $x \notin L_{\text{acc}} \Leftarrow$ שני מקרים:

מקרה 1: $x \neq \langle M, w \rangle \Leftarrow L(M_w) = \emptyset \Leftarrow M_E$ מקבלת $\langle M_w \rangle \Leftarrow M_{\text{acc}}$ דוחה.

מקרה 2: $x = \langle M, w \rangle$ ו- $w \notin L(M) \Leftarrow L(M_w) = \emptyset \Leftarrow M_E$ מקבלת $\langle M_w \rangle \Leftarrow M_{\text{acc}}$ דוחה.

לסיכום:

אם L_E כריעה אז אפשר לבנות מ"ט M_{acc} המכריעה את L_{acc} בסתירה לכך ש- $L_{\text{acc}} \notin R$.
לכן $L_E \notin R$.

משפט 7.8 $L_E \notin RE$

$L_E \notin RE$

הוכחה:
הרעיון

נבנה מכונת טיורינג אי-דטרמיניסטית N המקבלת את

$$\bar{L}_E = \{ \langle M \rangle \mid L(M) \neq \emptyset \}.$$

N = על קלט x :

(1) אם $\langle M \rangle \neq x$ דוחה.

(2) אם $x = \langle M \rangle$ אז N בוחרת מילה $w \in \Sigma^*$ באופן אי-דטרמיניסטית.

(3) מריצה M על w .

• אם M מקבלת N מקבלת.

• אם M דוחה N דוחה.

הוכחת הנכונות

אם $x \in \bar{L}_E$

$$\Leftrightarrow x = \langle M \rangle \text{ ו- } L(M) \neq \emptyset$$

$$\Leftrightarrow \text{קיימת מילה } w \in \Sigma^* \text{ כך ש- } w \in L(M)$$

$$\Leftrightarrow \exists \text{ ניחוש } w \in \Sigma^* \text{ כך ש- } M \text{ מקבלת את } w$$

$$\Leftrightarrow \text{קיים חישוב של } N \text{ המקבל את } \langle M \rangle = x$$

$$\Leftrightarrow x \in L(N)$$

לכן קיימת מ"ט א"ד N המקבלת את השפה \bar{L}_E לכן $\bar{L}_E \in RE$.

כעת נוכיח כי $L_E \notin RE$.

נניח בשלילה כי $L_E \in RE$. הוכחנו למעלה ש- $\bar{L}_E \in RE$. לכן לפי משפט 6.1, $L_E \in R$.

זו בסתירה לכך ש- $L_E \notin R$.

לכן $L_E \notin RE$.

■

7.3 השפה L_{EQ} לא כריעה

הגדרה 7.5 L_{EQ}

$$L_{EQ} = \{ \langle M_1, M_2 \rangle \mid L(M_1) = L(M_2) \}$$

משפט 7.9 $L_{EQ} \notin R$

$$L_{EQ} \notin R$$

השפה L_{EQ} לא כריעה.

הוכחה:

נניח בשלילה כי L_{EQ} כריעה. תהי M_{EQ} מ"ט המכריעה את L_{EQ} . אז נבנה מ"ט M_E המכריעה את L_E באופן הבא.

בנייה של M_E

$M_E = \text{על כל קלט } x:$

(1) אם $x \neq \langle M \rangle \Leftarrow \text{דוחה.}$

(2) אם $x = \langle M \rangle$, מריצה M_{EQ} על $\langle M, M_\emptyset \rangle$ כאשר M_\emptyset המ"ט שדוחה כל קלט.

(3) • אם M_{EQ} מקבלת \Leftarrow מקבלת.

• אם M_{EQ} דוחה \Leftarrow דוחה.

נכונות

אם $x \in L_E$

$\Leftarrow x = \langle M \rangle$ ו- $L(M) = \emptyset$

$\Leftarrow L(M) = L(M_\emptyset)$

$\Leftarrow \langle M, M_\emptyset \rangle \in L_{EQ}$

$\Leftarrow M_{EQ}$ מקבלת $\langle M, M_\emptyset \rangle$

$\Leftarrow M_E$ מקבל.

אם $x \notin L_E$ שני מקרים:

מקרה 1: $x \neq \langle M \rangle \Leftarrow M_E$ דוחה.

מקרה 2: $x = \langle M \rangle$ ו- $L(M) \neq \emptyset$

$\Leftarrow L(M) \neq L(M_\emptyset)$

$\Leftarrow \langle M, M_\emptyset \rangle \notin L_{EQ}$

$\Leftarrow M_{EQ}$ דוחה $\langle M, M_\emptyset \rangle$

$\Leftarrow M_E$ דוחה.

לסיכום:

אם L_{EQ} כריעה אז אפשר לבנות מ"ט M_E המכריעה את L_E בסתירה למשפט 7.7 האומר ש- $L_E \notin R$.
לכן $L_{EQ} \notin R$.

משפט 7.10 $L_{EQ} \notin RE$

$L_{EQ} \notin RE$

L_{EQ} לא קבילה.

הוכחה:

נניח בשלילה כי L_{EQ} קבילה. תהי M_{EQ} מ"ט המקבלת את L_{EQ} . אז נבנה מ"ט M_E המקבלת את L_E באופן הבא.

בנייה של M_E

$$M_E = \text{על כל קלט } x:$$

(1) אם $\langle M \rangle \neq x \Leftarrow$ דוחה.

(2) אם $x = \langle M \rangle$, מריצה M_{EQ} על $\langle M, M_\emptyset \rangle$ כאשר M_\emptyset המ"ט שדוחה כל קלט.

(3) • אם M_{EQ} מקבלת \Leftarrow מקבלת.

נכונות

אם $x \in L_E$

$$\Leftarrow \langle M \rangle = x \text{ ו- } L(M) = \emptyset$$

$$\Leftarrow L(M) = L(M_\emptyset)$$

$$\Leftarrow \langle M, M_\emptyset \rangle \in L_{EQ}$$

$$\Leftarrow M_{EQ} \text{ מקבלת } \langle M, M_\emptyset \rangle$$

$$\Leftarrow M_E \text{ מקבל.}$$

לסיכום:

אם L_{EQ} קבילה אז אפשר לבנות מ"ט M_E המקבלת את L_E בסתירה למשפט 7.8 האומר ש- $L_E \notin RE$.
לכן $L_{EQ} \notin RE$.

משפט 7.11 $\bar{L}_{EQ} \notin RE$

$$\bar{L}_{EQ} \notin RE.$$

הוכחה:

נניח בשלילה כי \bar{L}_{EQ} קבילה. תהי M_{EQ} מ"ט המקבלת את \bar{L}_{EQ} . אז נבנה מ"ט M_{acc} המקבלת את \bar{L}_{acc} באופן הבא.

בנייה של M_1

ראשית נגדיר מ"ט M_1 באופן הבא:

$$M_1 = \text{על קלט } x:$$

(1) מריצה M על w ועונה כמוה.

בנייה של M_{acc}

$$M_{acc} = \text{על כל קלט } x:$$

(1) אם $\langle M, w \rangle \neq x \Leftarrow$ מקבלת.

(2) אם $x = \langle M, w \rangle$ אז בונה M_1 .

(3) מריצה $M_{\overline{EQ}}$ על $\langle M_1, M^* \rangle$ כאשר M^* המ"ט שמקבלת כל קלט.

(4) • אם $M_{\overline{EQ}}$ מקבלת \Leftarrow מקבלת.

נכונות

אם $x \in L_{\text{acc}}$

$\Leftarrow M$ לא מקבלת w

$\Leftarrow L(M_1) = \emptyset$

$\Leftarrow \langle M_1, M^* \rangle \in L_{\overline{EQ}}$

$\Leftarrow M_{\overline{EQ}}$ מקבלת $\langle M_1, M^* \rangle$

$\Leftarrow M_{\text{acc}}$ מקבל.

לסיכום:

אם $L_{\overline{EQ}}$ קבילה אז אפשר לבנות מ"ט M_{acc} המקבלת את L_{acc} בסתירה למשפט 7.6 האומר ש- $L_{\text{acc}} \notin RE$.
לכן $L_{\overline{EQ}} \notin RE$.



7.4 סיכום: כריעות וקבילות של שפות

קבילה	כריעה	
✓	×	L_{acc}
×	×	$\overline{L_{\text{acc}}}$
×	×	L_d
✓	×	L_{Halt}
×	×	$\overline{L_{\text{Halt}}}$
×	×	L_E
✓	×	$\overline{L_E}$
×	×	L_{EQ}
×	×	$\overline{L_{\text{EQ}}}$
×	×	L_{REG}
×	×	L_{NOTREG}

שיעור 8

רדוקציה

8.1 טבלה של רדוקציות

טבלה של רדוקציות

רדוקציה	עמוד
$L_{\text{HALT}} \leq L_{\text{acc}}$	דוגמה 8.6 עמוד 84
$\bar{L}_{\text{acc}} \leq L_{\text{NOTREG}}$	דוגמה 8.11 עמוד 88
$L_{\text{acc}} \leq L_{\text{NOTREG}}$	דוגמה 8.12 עמוד 89
$L_{\text{HALT}} \leq L_{\text{NOTREG}}$	דוגמה 8.13 עמוד 90
$L_{\text{acc}} \leq L_{\text{REG}}$	דוגמה 8.15 עמוד 92
$\bar{L}_{\text{acc}} \leq L_{\text{REG}}$	דוגמה 8.14 עמוד 91
$\bar{L}_{\text{acc}} \leq L_{M_1 \cap M_2}$ כאשר $L_{M_1 \cap M_2} = \{ \langle M_1, M_2, w \rangle \mid w \in L(M_1) \wedge w \notin L(M_2) \}$	דוגמה 8.16 עמוד 93
$\bar{L}_{\text{acc}} \leq L_{M_1 \subset M_2}$ כאשר $L_{M_1 \subset M_2} = \{ \langle M_1, M_2 \rangle \mid L(M_1) \subset L(M_2) \}$	דוגמה 8.17 עמוד 93

8.2 מכונת טיורינג המחשבת את פונקציה

הגדרה 8.1 מכונת טיורינג המחשבת פונקציה

בהינתן פונקציה $f : \Sigma^* \rightarrow \Sigma^*$ אומרים כי מ"ט M מחשבת את f אם לכל $x \in \Sigma^*$:

- M מגיעה ל- q_{acc} בסוף החישוב של $f(x)$ וגם
- על סרט הפלט של M רשום $f(x)$.

הערה 8.1

מכונת טיורינג שמחשבת פונקציה עוצרת תמיד.

הגדרה 8.2 פונקציה חשיבה

בהינתן פונקציה $f : \Sigma^* \rightarrow \Sigma^*$ אומרים כי f חשיבה אם קיימת מכונת טיורינג המחשבת את f .

8.1 דוגמה

$$f_1(x) = xx \quad (8.1)$$

$f_1(x)$ חשיבה.

8.2 דוגמה

$$f_2(x) = \begin{cases} x & |x| \text{ זוגי} \\ xx & |x| \text{ אי-זוגי} \end{cases} \quad (8.2)$$

$f_2(x)$ חשיבה.

8.3 דוגמה

$$f_3(x) = \begin{cases} \langle M' \rangle & x = \langle M \rangle \\ \langle M^* \rangle & x \neq \langle M \rangle \end{cases} \quad (8.3)$$

כאשר

• M^* מ"ט שמקבלת כל קלט.

• M' מ"ט המקבלת את השפה

$$L(M') = \{w \in \Sigma^* \mid ww \in L(M)\}$$



$f_3(x)$ חשיבה כי ניתן לבנות מ"ט שבודקת האם $x = \langle M \rangle$. אם לא, מחזירה קידוד קבוע $\langle M^* \rangle$. ואם כן, מחזירה קידוד $\langle M' \rangle$ ע"י הוספת מעברים המשכפלים את הקלט בתחילת הקידוד $\langle M \rangle$.

8.4 דוגמה

$$f_4(x) = \begin{cases} 1 & x = \langle M \rangle \wedge \langle M \rangle \in L(M) \\ 0 & \text{אחרת} \end{cases} \quad (8.4)$$

$f_4(x)$ לא חשיבה כי ייתכנו קלטים $x = \langle M \rangle$ ו- M לא עוצרת על $\langle M \rangle$.

8.3 רדוקציות

הגדרה 8.3 רדוקציות

בהינתן שתי שפות $L_1, L_2 \subseteq \Sigma^*$ אומרים כי L_1 ניתנת לרדוקציה ל- L_2 , ומסמנים

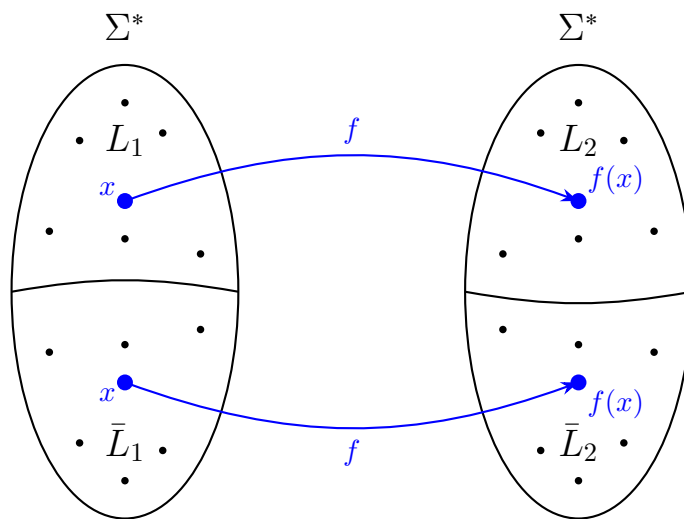
$$L_1 \leq L_2 ,$$

אם \exists פונקציה $f : \Sigma^* \rightarrow \Sigma^*$ המקיימת:

(1) f חשיבה

(2) לכל $x \in \Sigma^*$:

$$x \in L_1 \iff f(x) \in L_2 .$$



8.5 דוגמה

נתונות השפות

$$L_1 = \{x \in \{0, 1\}^* \mid |x| \text{ זוגי} \} ,$$

$$L_2 = \{x \in \{0, 1\}^* \mid |x| \text{ אי-זוגי} \} .$$

הוכיחו כי

$$L_1 \leq L_2 .$$

פתרון:

נגדיר את הפונקציה

$$f(x) = \begin{cases} 1 & |x| \text{ זוגי} \\ 10 & |x| \text{ אי-זוגי} \end{cases}$$

הוכחת הנכונות:

$$f(x) \in L_2 \Leftarrow |f(x)| \text{ אי-זוגי} \Leftarrow f(x) = 1 \Leftarrow |x| \text{ זוגי} \Leftarrow x \in L_1$$

$$f(x) \notin L_2 \Leftarrow |f(x)| \text{ זוגי} \Leftarrow f(x) = 10 \Leftarrow |x| \text{ אי-זוגי} \Leftarrow x \notin L_1$$

משפט 8.1 משפט הרדוקציהלכל שתי שפות $L_1, L_2 \subseteq \Sigma^*$, אם קיימת רדוקציה

$$L_1 \leq L_2$$

אזי התנאים הבאים מתקיימים:

$$L_1 \in R \Leftarrow L_2 \in R \quad (1)$$

$$L_1 \in RE \Leftarrow L_2 \in RE \quad (2)$$

$$L_1 \notin R \Rightarrow L_2 \notin R \quad (3)$$

$$L_1 \notin RE \Rightarrow L_2 \notin RE \quad (4)$$

הוכחה: מכיוון ש-

$$L_1 \leq L_2$$

קיימת פונקציה f חשיבה המקיימת:

$$x \in L_1 \Leftrightarrow f(x) \in L_2$$

לכל $x \in \Sigma^*$.תהי M_f מ"ט המחשבת את f .

$$(1) \quad \underline{L_1 \in R \Leftarrow L_2 \in R} \quad \text{נוכיח}$$

תהי M_2 מ"ט המכריעה את L_2 .נבנה מ"ט M_1 המכריעה את L_1 .התאור של M_1

$$M_1 = \text{על קלט } x:$$

1. מחשבת את $f(x)$ בעזרת M_f .2. מריצה את M_2 על $f(x)$ ועונה כמוה.נוכיח כי M_1 מכריעה את L_1 .

$$\bullet \quad \text{אם } x \in L_1 \Leftarrow f(x) \in L_2 \Leftarrow M_2 \text{ מקבלת את } f(x) \Leftarrow M_1 \text{ מקבלת את } x.$$

• אם $x \notin L_1 \Leftrightarrow f(x) \notin L_2 \Leftrightarrow M_2$ דוחה את $f(x) \Leftrightarrow M_1$ דוחה את x .

(2) נוכיח $L_1 \in RE \Leftrightarrow L_2 \in RE$

תהי M_2 מ"ט המקבלת את L_2 .

נבנה מ"ט M_1 המקבלת את L_1 .



התאור של M_1

$M_1 =$ על קלט x :

1. מחשבת את $f(x)$ בעזרת M_f .

2. מריצה את M_2 על $f(x)$ ועונה כמוה.

נוכיח כי M_1 מקבלת את L_1 :

• אם $x \in L_1 \Leftrightarrow f(x) \in L_2 \Leftrightarrow M_2$ מקבלת את $f(x) \Leftrightarrow M_1$ מקבלת את x .

• אם $x \notin L_1 \Leftrightarrow f(x) \notin L_2 \Leftrightarrow M_2$ לא מקבלת את $f(x) \Leftrightarrow M_1$ לא מקבלת את x .

(3)

(4)

כלל 8.1

• אם רוצים להוכיח כי שפה כלשהי $L \in RE$, בוחרים שפה אחרת $L' \in RE$ ומראים שקיימת רדוקציה

$$L \leq L'.$$

לדוגמה:

$$L \leq L_{acc}$$

(כנ"ל לגבי R)

- אם רוצים להוכיח כי שפה כלשהי $L \notin RE$ בוחרים שפה אחרת $L' \notin RE$ ומראים שקיימת רדוקציה

$$L' \leq L.$$

לדוגמה

$$L_d \leq L$$

(כנ"ל לגבי R).

8.6 דוגמה

נתונות השפות $L_{acc} = \{\langle M, w \rangle \mid w \in L(M)\}$ ו- $L_{halt} = \{\langle M, w \rangle \mid M \text{ עוצרת על } w\}$.
הוכיחו כי $L_{acc} \notin R$ ע"י רדוקציה $L_{acc} \leq L_{halt}$.

פתרון:

נבנה פונקציה f חשיבה ומקיימת

$$x \in L_{acc} \iff f(x) \in L_{halt}.$$

$$M \text{ מקבלת את } w \iff M' \text{ תעצור על } w'.$$

$$M \text{ דוחה את } w \iff M' \text{ לא תעצור על } w'.$$

$$M \text{ לא עוצרת את } w \iff M' \text{ לא תעצור על } w'.$$

$$f(x) = \begin{cases} \langle M', w \rangle & : x = \langle M, w \rangle \\ \langle M_{loop}, \varepsilon \rangle & : x \neq \langle M, w \rangle \end{cases}$$

כאשר

- M_{loop} מ"ט שלא עוצרת על אף קלט.

- M' מ"ט המתנהגת כמו M פרט למקומות בהם M עצרה ודחתה, M' תיכנס ללולאה אינסופית.

נכונות הרדוקציה

f חשיבה כי ניתן לבנות מ"ט שתבדוק האם $x = \langle M, w \rangle$.

אם לא, תחזיר קידוד קבוע $\langle M_{loop}, w \rangle$

ואם כן, תחזיר קידוד $\langle M', w \rangle$ ע"י ביצוע שינויים לוקלים בקידוד של M .

$$x \in L_{acc} \iff f(x) \in L_{halt} \text{ נוכיח כי}$$

אם $x \in L_{acc}$:

$$w \in L(M) \text{ ו-} x = \langle M, w \rangle \Leftarrow$$

$$w \text{ עוצרת ומקבלת את } M' \text{ ו-} f(x) = \langle M', w \rangle \Leftarrow$$

$$f(x) \in L_{\text{halt}} \Leftarrow$$

אם $x \notin L_{\text{acc}}$ אז שני מקרים:

מקרה 1:

$$f(x) \notin L_{\text{halt}} \Leftarrow f(x) = \langle M_{\text{loop}}, \varepsilon \rangle \Leftarrow x \neq \langle M, w \rangle$$

מקרה 2:

$$\text{שני מקרים:} \quad f(x) = \langle M', w \rangle \Leftarrow w \notin L(M) \text{ ו-} x = \langle M, w \rangle$$

$$\text{מקרה א:} \quad M \text{ לא עוצרת על } w \Leftarrow M' \text{ לא עוצרת על } w \Leftarrow f(x) \notin L_{\text{halt}}$$

$$\text{מקרה ב:} \quad M \text{ דוחה את } w \Leftarrow M' \text{ לא עוצרת על } w \Leftarrow f(x) \notin L_{\text{halt}}$$

לסיכום, הוכחנו רדוקציה $L_{\text{acc}} \leq L_{\text{halt}}$. ומכיוון ש- $L_{\text{acc}} \notin R$ (משפט 7.4) אז ממשט הרדוקציה 8.1, מתקיים $L_{\text{halt}} \notin R$.

8.7 דוגמה

נתונה השפה

$$L_{\Sigma^*} = \{ \langle M \rangle \mid L(M) = \Sigma^* \}$$

ונתונה השפה

$$\bar{L}_{\Sigma^*} = \{ \langle M \rangle \mid L(M) \neq \Sigma^* \} \cup \{ x \neq \langle M \rangle \}.$$

הוכיחו כי:
 $L_{\Sigma^*} \notin R$

פתרון:

נוכיח כי $L_{\Sigma^*} \notin R$ ע"י רדוקציה

$$L_{\text{acc}} \leq L_{\Sigma^*}.$$

נבנה פונקציה חשיבה f המקיימת

$$x \in L_{\text{acc}} \Leftrightarrow f(x) \in L_{\Sigma^*}.$$

$$L(M') = \Sigma^* \Leftarrow w \in L(M)$$

$$L(M') \neq \Sigma^* \Leftarrow w \notin L(M)$$

$$f(x) = \begin{cases} \langle M' \rangle & : x = \langle M, w \rangle \\ \langle M_{\emptyset} \rangle & : x \neq \langle M, w \rangle \end{cases}$$

כאשר

• M_{\emptyset} מ"ט שדוחה כל קלט.

• M' היא מ"ט שעל כל קלט x , מתעלמת מ- x ומריצה את M על w ועונה כמוה.

אבחנה:

$$L(M') = \begin{cases} \Sigma^* & : w \in L(M) \\ \emptyset & : w \notin L(M) \end{cases}$$

נכונות הרדוקציה:

f חשיבה כי ניתן לבנות מ"ט שתבדוק האם $x = \langle M, w \rangle$.

אם לא תחזיר קידוד קבוע $\langle M_\emptyset \rangle$.

אם כן, תחזיר קידוד $\langle M' \rangle$ ע"י הוספת קוג ל- M שמוחק את הקלט מהסרט וכותב w במקומו.

נוכיח כי

$$x \in L_{\text{acc}} \Leftrightarrow f(x) \in L_{\Sigma^*}$$

$$\Leftrightarrow L(M') = \Sigma^* \text{ ולפי האבחנה } f(x) = \langle M' \rangle \Leftrightarrow w \in L(M) \text{ ו- } x = \langle M, w \rangle \Leftrightarrow \begin{matrix} \text{אם } x \in L_{\text{acc}} \\ f(x) \in L_{\Sigma^*} \end{matrix}$$

$$\text{אם } x \in L_{\text{acc}} \Leftrightarrow \text{שני מקרים:}$$

$$\text{מקרה 1: } f(x) \notin L_{\Sigma^*} \Leftrightarrow L(M_\emptyset) = \emptyset \text{ ו- } f(x) = \langle M_\emptyset \rangle \Leftrightarrow x \neq \langle M, w \rangle$$

$$\text{מקרה 2: } f(x) \notin L_{\Sigma^*} \Leftrightarrow L(M') = \emptyset \text{ ולפי האבחנה } f(x) = \langle M' \rangle \Leftrightarrow w \notin L(M) \text{ ו- } x = \langle M, w \rangle$$

לסיכום, הוכחנו רדוקציה $L_{\text{acc}} \leq L_{\Sigma^*}$. ומכיוון ש- $L_{\text{acc}} \notin R$ (משפט 7.4) אז ממשט הרדוקציה 8.1, מתקיים $L_{\Sigma^*} \notin R$.

8.8 דוגמה

נתונה השפה

$$L_d = \{ \langle M \rangle \mid \langle M \rangle \notin L(M) \}$$

ונתונה השפה

$$\bar{L}_{\text{acc}} = \{ \langle M, w \rangle \mid w \notin L(M) \} \cup \{ x \neq \langle M, w \rangle \}.$$

הוכיחו כי

$$\bar{L}_{\text{acc}} \notin RE \text{ ע"י רדוקציה}$$

$$L_d \leq \bar{L}_{\text{acc}}.$$

פתרון:

נבנה פונקציה חשיבה f המקיימת

$$x \in L_d \Leftrightarrow f(x) \in L_{\text{acc}}.$$

$$w' \notin L(M') \Leftrightarrow \langle M \rangle \notin L(M)$$

$$w' \in L(M') \Leftrightarrow \langle M \rangle \in L(M)$$

$$f(x) = \begin{cases} \langle M, \langle M \rangle \rangle & : x = \langle M \rangle \\ \langle M^*, \varepsilon \rangle & : x \neq \langle M \rangle \end{cases}$$

כאשר M^* המ"ט שמקבלת כל קלט.

נכונות הרדוקציה:

f חשיבה כי ניתן לבנות מ"ט שתבדוק האם $x = \langle M, w \rangle$

אם לא תחזיר קידוד קבוע $\langle M^*, \varepsilon \rangle$.

אם כן, תחשב $\langle M, \langle M \rangle \rangle$.

נוכיח כי

$$\begin{aligned} x \in L_d &\Leftrightarrow f(x) \in \bar{L}_{acc} \\ \Leftrightarrow \langle M \rangle \notin L(M) \text{ ו- } f(x) = \langle M, \langle M \rangle \rangle &\Leftrightarrow \langle M \rangle \notin L(M) \text{ ו- } x = \langle M \rangle \Leftrightarrow \begin{aligned} &x \in L_d \text{ אם } \\ &f(x) \in \bar{L}_{acc} \end{aligned} \end{aligned}$$

אם $x \notin L_d$ שני מקרים:

$$\text{מקרה 1: } f(x) \notin \bar{L}_{acc} \Leftrightarrow \varepsilon \in L(M^*) \text{ ו- } f(x) = \langle M^*, \varepsilon \rangle \Leftrightarrow x \neq \langle M \rangle$$

$$\text{מקרה 2: } f(x) \notin \bar{L}_{acc} \Leftrightarrow \langle M \rangle \in L(M) \text{ ו- } f(x) = \langle M, \langle M \rangle \rangle \Leftrightarrow \langle M \rangle \in L(M) \text{ ו- } x = \langle M \rangle$$

לסיכום, הוכחנו רדוקציה $L_d \leq \bar{L}_{acc}$, ומכיוון ש- $L_d \notin RE$ (משפט 7.3) אז ממשט הרדוקציה 8.1, מתקיים $\bar{L}_{acc} \notin RE$.

משפט 8.2 משפט הרדוקציה בין שפות משלימות

אם קיימת רדוקציה $L_1 \leq L_2$, אזי קיימת רדוקציה $\bar{L}_1 \leq \bar{L}_2$.

הוכחה:

אם \exists רדוקציה

$$L_1 \leq L_2$$

אזי \exists פונקציה חשיבה f המקיימת

$$x \in L_1 \Leftrightarrow f(x) \in L_2$$

ולכן עבור אותה פונקציה f היא גם חשיבה וגם מקיימת

$$x \in \bar{L}_1 \Leftrightarrow f(x) \in \bar{L}_2$$

ולכן

$$\bar{L}_1 \leq \bar{L}_2 .$$

8.4 דוגמאות בשימוש של משפט הרדוקציה בין שפות משלימות (משפט 8.2)

8.9 דוגמה

הוכחנו בדוגמה 8.7 רדוקציה

$$L_{acc} \leq L_{\Sigma^*}.$$

לכן לפי משפט 8.2 קיימת רדוקציה

$$\bar{L}_{acc} \leq \bar{L}_{\Sigma^*}.$$

מכיוון ש- $\bar{L}_{acc} \notin RE$, אזי ממשפט הרדוקציה 8.1 מתקיים $\bar{L}_{\Sigma^*} \notin RE$.

8.10 דוגמה

הוכחנו בדוגמה 8.8 רדוקציה

$$L_d \leq \bar{L}_{acc}.$$

לכן לפי משפט 8.2 קיימת רדוקציה

$$\bar{L}_d \leq L_{acc}.$$

מכיוון ש- $L_{acc} \in RE$, אזי ממשפט הרדוקציה 8.1 מתקיים $\bar{L}_d \in RE$.

8.5 דוגמאות בשימוש של משפט הרדוקציה (משפט 8.1)

8.11 דוגמה $\bar{L}_{acc} \leq L_{NOTREG}$

תהי L_{NOTREG} השפה

$$L_{NOTREG} = \{ \langle M \rangle \mid L(M) \text{ לא רגולרית} \}.$$

הוכיחו כי השפה L_{NOTREG} לא כריעה על ידי רדוקציה מ- \bar{L}_{acc} .

פתרון:

השפה \bar{L}_{acc} מוגדרת

$$\bar{L}_{acc} = \{ \langle M, w \rangle \mid w \text{ לא מקבלת } M \} \cup \{ x \neq \langle M, w \rangle \}.$$

והשפה L_{NOTREG} מוגדרת

$$L_{NOTREG} = \{ \langle M \rangle \mid L(M) \text{ לא רגולרית} \}.$$

נגדיר הפונקציה הבאה:

$$f(x) = \begin{cases} \langle M' \rangle & x = \langle M, w \rangle \\ w' \in PAL & x \neq \langle M, w \rangle \end{cases}$$

כאשר M' מ"ט הבאה:

$M' =$ על כל קלט y :

(1) אם $y \in PAL \Leftarrow$ מקבלת.

(2) אחרת מריצה M על w ועונה כמוה.

הוכחת נכונות הרדוקציה

אם $x \in \bar{L}_{acc}$ \Leftarrow שני מקרים:

מקרה 1: $x = \langle M, w \rangle$

$\Leftarrow M$ לא מקבלת w

$\Leftarrow L(M') \in PAL$

$\Leftarrow \langle M' \rangle \in PAL$

$\Leftarrow f(x) \in PAL$

$\Leftarrow f(x) \in L_{NOTREG}$

מקרה 2: $x \neq \langle M, w \rangle$ $\Leftarrow f(x) \in PAL \Leftarrow f(x) \in L_{NOTREG}$

אם $x \notin \bar{L}_{acc}$ $\Leftarrow M$ מקבלת w $\Leftarrow L(M') = \Sigma^*$ $\Leftarrow f(x) \in \Sigma^*$ $\Leftarrow f(x) \in L_{NOTREG}$

לכן הוכחנו כי $x \in \bar{L}_{acc} \Leftrightarrow f(x) \in NOTERG$, ז"א $f(x)$ היא רדוקציה מ- L_{acc} ל- L_{NOTREG} .

השפה \bar{L}_{acc} לא כריעה. לפיכך, לפי משפט הרדוקציה גם L_{NOTREG} לא כריעה.

8.12 דוגמה $L_{acc} \leq L_{NOTREG}$

תהי L_{NOTREG} השפה

$$L_{NOTREG} = \{ \langle M \rangle \mid L(M) \text{ לא רגולרית} \}.$$

הוכיחו כי השפה L_{NOTREG} לא כריעה על ידי רדוקציה מ- L_{acc} .

פתרון:

השפה L_{acc} מוגדרת

$$L_{acc} = \{ \langle M, w \rangle \mid M \text{ מקבלת } w \}.$$

והשפה L_{NOTREG} מוגדרת

$$L_{NOTREG} = \{ \langle M \rangle \mid L(M) \text{ לא רגולרית} \}.$$

נגדיר הפונקציה הבאה:

$$f(x) = \begin{cases} \langle M' \rangle & x = \langle M, w \rangle \\ \langle M_{\emptyset} \rangle & x \neq \langle M, w \rangle \end{cases}$$

כאשר M' מ"ט הבאה:

$M' =$ על כל קלט y :

(1) M' מריצה M על w .

(2) • אם M דוחה \Leftarrow דוחה.

• אם M מקבלת $\Leftarrow M'$ בודקת אם y פלינדרום.

* אם כן \Leftarrow מקבלת.

* אם לא \Leftarrow דוחה.

הוכחת נכונות הרדוקציה

אם $x \in L_{acc}$ $\Leftarrow M$ מקבלת w $\Leftarrow L(M') = PAL$ $\Leftarrow f(x) \in PAL$ $\Leftarrow f(x) \in L_{NOTREG}$.

$x \notin L_{acc}$ \Leftarrow שני מקרים.

מקרה 1: $x \neq \langle M, w \rangle \Leftarrow f(x) = \langle M_\emptyset \rangle$ ו- $L(M_\emptyset) = \emptyset \Leftarrow \langle M_\emptyset \rangle \notin L_{NOTREG}$ $\Leftarrow f(x) \notin L_{NOTREG}$.

מקרה 2: $x = \langle M, w \rangle$ ו- M לא מקבלת w $\Leftarrow L(M') = \emptyset \Leftarrow \langle M' \rangle \notin L_{NOTREG}$ $\Leftarrow f(x) \notin L_{NOTREG}$.

8.13 דוגמה $L_{HALT} \leq L_{NOTREG}$

תהי L_{NOTREG} השפה

$$L_{NOTREG} = \{ \langle M \rangle \mid L(M) \text{ לא רגולרית} \}.$$

הוכיחו כי השפה L_{NOTREG} לא כריעה על ידי רדוקציה מ- L_{HALT} .

פתרון:

השפה L_{HALT} מוגדרת

$$L_{HALT} = \{ \langle M, w \rangle \mid M \text{ עוצרת על } w \}.$$

והשפה L_{NOTREG} מוגדרת

$$L_{NOTREG} = \{ \langle M \rangle \mid L(M) \text{ לא רגולרית} \}.$$

נגדיר הפונקציה הבאה:

$$f(x) = \begin{cases} \langle M' \rangle & x = \langle M, w \rangle \\ \langle M_\emptyset \rangle & x \neq \langle M, w \rangle \end{cases}.$$

כאשר M' מ"ט הבאה:

$M' =$ על כל קלט y :

(1) M' מריצה M על w .

(2) אם M דוחה \Leftarrow דוחה.

• אם M מקבלת \Leftarrow ממשיכה לשלב (3).

(3) M' בודקת אם $y \in PAL$.

• אם כן \Leftarrow מקבלת.

• אם לא \Leftarrow דוחה.

הוכחת הנכונות

$$L(M') \in L_{\text{NOTREG}} \iff L(M') \in PAL \iff x \in L_{\text{HALT}}$$

$$\text{שני מקרים:} \iff x \notin L_{\text{HALT}}$$

$$\langle M_\emptyset \rangle \notin L_{\text{NOTREG}} \iff L(M_\emptyset) = \emptyset \text{ ו- } f(x) = \langle M_\emptyset \rangle \iff x \neq \langle M, w \rangle \quad \text{מקרה 1:}$$

$$f(x) \notin L_{\text{NOTREG}} \iff$$

$$\langle M_\emptyset \rangle \notin L_{\text{NOTREG}} \iff L(M_\emptyset) = \emptyset \iff w \text{ לא עוצרת על } M \text{ ו- } x = \langle M, w \rangle \quad \text{מקרה 2:}$$

$$f(x) \notin L_{\text{NOTREG}} \iff$$

דוגמה 8.14 $\bar{L}_{\text{acc}} \leq L_{\text{REG}}$ תהי L_{REG} השפה

$$L_{\text{REG}} = \{ \langle M \rangle \mid L(M) \text{ רגולרית} \}.$$

הוכיחו כי השפה L_{REG} לא כריעה על ידי רדוקציה מ- \bar{L}_{acc} .**פתרון:**השפה \bar{L}_{acc} מוגדרת

$$\bar{L}_{\text{acc}} = \{ \langle M, w \rangle \mid w \text{ לא מקבלת } M \} \cup \{ x \mid x \neq \langle M, w \rangle \}.$$

והשפה L_{REG} מוגדרת

$$L_{\text{REG}} = \{ \langle M \rangle \mid L(M) \text{ רגולרית} \}.$$

נגדיר הפונקציה הבאה:

$$f(x) = \begin{cases} \langle M' \rangle & x = \langle M, w \rangle \\ \langle M_\emptyset \rangle & x \neq \langle M, w \rangle \end{cases}$$

כאשר M_\emptyset המ"ט שדוחה כל קלט ו- M' מ"ט הבאה:

$$M' = \text{על כל קלט } y:$$

(1) מריצה M על w .(2) אם M דוחה \Leftarrow דוחה.• אם M מקבלת \Leftarrow בודקת אם y פלינדרום:• אם כן \Leftarrow מקבלת.• אם לא \Leftarrow דוחה.אבחנה

$$L(M') = \begin{cases} PAL & w \in L(M) \\ \emptyset & w \notin L(M) \end{cases}$$

הוכחת נכונות הרדוקציה

אם $x \in \bar{L}_{acc}$ שני מקרים: \Leftarrow

מקרה 1: $x \neq \langle M, w \rangle \Leftarrow f(x) = \langle M_{\emptyset} \rangle \Leftarrow L(M_{\emptyset}) = \emptyset \Leftarrow \langle M_{\emptyset} \rangle \in L_{REG} \Leftarrow f(x) \in L_{REG}$

מקרה 2: $x = \langle M, w \rangle \Leftarrow x \notin L(M) \Leftarrow f(x) = \langle M' \rangle \Leftarrow L(M') = \emptyset$ ולפי האבחנה $\langle M_{\emptyset} \rangle \in L_{REG} \Leftarrow f(x) \in L_{REG}$

אם $x \notin \bar{L}_{acc} \Leftarrow w \in L(M) \Leftarrow f(x) = \langle M' \rangle \Leftarrow L(M') \in PAL \Leftarrow f(x) \in PAL$
 $f(x) \notin L_{REG} \Leftarrow$

8.15 דוגמה $L_{acc} \leq L_{REG}$

תהי L_{REG} השפה

$$L_{REG} = \{ \langle M \rangle \mid L(M) \text{ רגולרית} \}.$$

הוכיחו כי השפה L_{REG} לא כריעה על ידי רדוקציה מ- L_{acc} .

פתרון:

השפה L_{acc} מוגדרת

$$L_{acc} = \{ \langle M, w \rangle \mid M \text{ מקבלת } w \}.$$

והשפה L_{REG} מוגדרת

$$L_{REG} = \{ \langle M \rangle \mid L(M) \text{ רגולרית} \}.$$

נגדיר הפונקציה הבאה:

$$f(x) = \begin{cases} \langle M' \rangle & x = \langle M, w \rangle \\ \langle M_{PAL} \rangle & x \neq \langle M, w \rangle \end{cases}$$

כאשר M_{PAL} המ"ט שמכריעה את השפה של פלינדרומים, ו- M' מ"ט הבאה:

$$M' = \text{על כל קלט } y:$$

(1) M' בודקת אם y פלינדרום:

- אם כן \Leftarrow מקבלת.
- אם לא מריצה M על w ועונה כמזה.

הוכחת נכונות הרדוקציה

אם $x \in L_{acc} \Leftarrow M \text{ מקבלת } w \Leftarrow L(M') = \Sigma^* \Leftarrow f(x) \in REG$

$x \notin L_{acc} \Leftarrow$ שני מקרים:

מקרה 1: $x \neq \langle M, w \rangle \Leftarrow f(x) = \langle M_{PAL} \rangle \Leftarrow L(M_{PAL}) = PAL \Leftarrow \langle M_{PAL} \rangle \notin L_{REG}$
 $f(x) \notin L_{REG} \Leftarrow$

מקרה 2: $x = \langle M, w \rangle \Leftarrow M \text{ לא מקבלת } w \Leftarrow L(M') = PAL \Leftarrow \langle M' \rangle \notin L_{REG}$
 $f(x) \notin L_{REG} \Leftarrow$

דוגמה 8.16 $\bar{L}_{acc} \leq L_{M_1 \neg M_2}$

נתונה השפה הבאה:

$$L = \{ \langle M_1, M_2, w \rangle \mid w \in L(M_1) \wedge w \notin L(M_2) \} .$$

הוכיחו כי $L \notin RE$ ע"י רדוקציה מ- \bar{L}_{acc} .**פתרון:**פונקצית הרדוקציה:

$$f(x) = \begin{cases} \langle M^*, M, w \rangle & x = \langle M, w \rangle \\ \langle M^*, M_\emptyset, \varepsilon \rangle & x \neq \langle M, w \rangle \end{cases}$$

כאשר

• M^* היא מ"ט שמקבלת כל קלט• M_\emptyset היא מ"ט שדוחה כל קלט.נכונת הרדוקציה:

ראשית, f חשיבה כי ניתן לבנות מ"ט שתבדוק האם $x = \langle M, w \rangle$. אם לא, תחזיר קידוד קבוע $\langle M^*, M_\emptyset, \varepsilon \rangle$ ואם כן, תחזיר קידוד $\langle M^*, M, w \rangle$.

נוכיח כי

$$x \in \bar{L}_{acc} \iff f(x) \in L_{M_1 \neg M_2} .$$

אם $x \in \bar{L}_{acc}$ \Leftarrow שני מקרים:

$$f(x) \in L_{M_1 \neg M_2} \iff x \neq \langle M, w \rangle \iff f(x) = \langle M^*, M_\emptyset, \varepsilon \rangle \iff \varepsilon \notin L(M_\emptyset) \text{ ו- } \varepsilon \in L(M^*) \iff f(x) \in L_{M_1 \neg M_2} .$$

$$\text{מקרה 2: } x = \langle M, w \rangle \text{ ו- } w \notin L(M) \iff f(x) = \langle M^*, M, w \rangle \iff w \in L(M^*) \text{ ו- } w \notin L(M) \iff f(x) \in L_{M_1 \neg M_2} .$$

$$\text{אם } x \notin \bar{L}_{acc} \iff x = \langle M, w \rangle \text{ ו- } w \in L(M) \iff f(x) = \langle M^*, M, w \rangle \iff w \in L(M^*) \text{ ו- } w \in L(M) \iff f(x) \notin L_{M_1 \neg M_2} .$$

לסיכום, הוכחנו רדוקציה $\bar{L}_{acc} \leq L_{M_1 \neg M_2}$, ומכיוון ש- $\bar{L}_{acc} \notin RE$ ממשפט הרדוקציה מתקיים $L_{M_1 \neg M_2} \notin RE$.

דוגמה 8.17 $L_{acc} \leq L_{M_1 \subset M_2}$

נתונה השפה הבאה:

$$L = \{ \langle M_1, M_2, w \rangle \mid w \in L(M_1) \wedge w \notin L(M_2) \} .$$

הוכיחו כי $L \notin RE$ ע"י רדוקציה מ- L_{acc} .**פתרון:**פונקצית הרדוקציה:

$$f(x) = \begin{cases} \langle M_\emptyset, M' \rangle & x = \langle M, w \rangle \\ \langle M_\emptyset, M_\emptyset \rangle & x \neq \langle M, w \rangle \end{cases}$$

כאשר

- M_\emptyset היא מ"ט שדוחה כל קלט.
- M' היא מ"ט שעל קלט y מתעלמת מ- y ומריצה M על w ועונה כמוה.

אבחנה:

$$L(M') = \begin{cases} \Sigma^* & w \in L(M) \\ \emptyset & w \notin L(M) \end{cases}.$$

נכונת הרדוקציה:

ראשית, f חשיבה כי ניתן לבנות מ"ט שתבדוק האם $x = \langle M, w \rangle$. אם לא, תחזיר קידוד קבוע $\langle M_\emptyset, M_\emptyset \rangle$ ואם כן, תחזיר קידוד $\langle M^*, M' \rangle$, כאשר M_\emptyset היא קידוד קבוע ו- M' נוצר ע"י הוספת קוד ל- $\langle M \rangle$ המוחק את הקלט y ורושם w במקומו.

נוכיח כי

$$x \in L_{\text{acc}} \iff f(x) \in L_{M_1 \subset M_2}.$$

$$L(M') = \Sigma^* \text{ ולפי האבחנה } f(x) = \langle M_\emptyset, M' \rangle \iff x = \langle M, w \rangle \text{ ו- } w \in L(M) \iff x \in L_{\text{acc}} \\ f(x) \in L_{M_1 \subset M_2} \iff L(M_\emptyset) \subset L(M') \iff$$

$$x \notin L_{\text{acc}} \iff \text{שני מקרים:}$$

$$\text{מקרה 1: } x \neq \langle M, w \rangle \iff f(x) = \langle M_\emptyset, M_\emptyset \rangle \text{ ו- } L(M_\emptyset) = L(M_\emptyset) \iff f(x) \notin L_{M_1 \subset M_2}.$$

$$\text{מקרה 2: } x = \langle M, w \rangle \text{ ו- } w \notin L(M) \iff f(x) = \langle M_\emptyset, M' \rangle \text{ ולפי האבחנה } L(M') = \emptyset \\ f(x) \notin L_{M_1 \subset M_2} \iff L(M') = L(M_\emptyset) \iff$$

לסיכום, הוכחנו רדוקציה $L_{\text{acc}} \leq L_{M_1 \subset M_2}$, ומכיוון ש- $L_{\text{acc}} \notin R$ ממשפט הרדוקציה מתקיים $L_{M_1 \subset M_2} \notin R$.

שיעור 9

מבוא לסיבוכיות זמן

9.1 הגדרה של סיבוכיות זמן

עד כה כל הבעיות החישוביות שעסקנו בהן הניחו שהמשאבים שעומדים לרשות מכונת הטיורינג שפותרת אותן הם **בלתי מוגבלים**. כעת נעבור לעסוק בשאלה מה קורה כאשר אנחנו מגבילים חלק ממשאבים אלו. יש סוגים רבים של משאבים שניתן לעסוק בהם, אבל שני הנפוצים ביותר בתיאוריה של מדעי המחשב הם

- זמן החישוב,
- הזיכרון שנדרש לצורך החישוב.

אחת מהבעיות שבהן נתקלים: כשמעוניינים למדוד את צריכת המשאבים הללו של אלגוריתם מסויים היא שלא ברור כיצד למדוד אותם. האם זמן חישוב נמדד בשניות? אם כן, כיצד ניתן לחשב את זמן החישוב עבור אלגוריתם נתון? האם עלינו לקודד ולהריץ אותו על מחשב מסוים?

אבל במחשבים שונים האלגוריתם ירוץ זמנים שונים בשל

- יעילות המעבד,
- אופטימיזציות שמתבצעות ברמת המעבד,
- אופטימיזציות בזמן הקומפליצה,

וכיוצא בהן.

אפילו תנאים חיצוניים כמו החום בסביבת המעבד עשויים להשפיע על זמן הריצה. מכאן הרצון למצוא הגדרה **תיאורטית** של זמן ריצה, שאינה תלויה בחומרה זו או אחרת.

הערה 9.1

זמן הריצה של מ"ט M על קלט w , נמדד ביחס לגודל הקלט w , כלומר $f(|w|)$.

הגדרה 9.1 זמן הריצה של מכונת טיורינג

זמן הריצה של מ"ט M על קלט w היא פונקציה $f(|w|)$ השווה למספר הצעדים הנדרש בחישוב של M על w .

הגדרה 9.2 סיבוכיות זמן של בעיה/שפה

בהינתן קלט w באורך $n = |w|$. אומרים כי ניתן להכריעה שפה L בזמן $f(n)$ אם קיימת מ"ט M המכריעה את L כך שלכל $w \in \Sigma^*$, זמן הריצה של M על w חסום ע"י $f(|w|)$.

דוגמה 9.1 סיבוכיות זמן של השפה של מחרוזות האוניריות

נתבונן על השפה של מחרוזות האוניריות הבאה:

$$L = \{1^n \mid n \geq 0\}.$$

נבנה מכונת טיורינג הבאה שמכריעה אותה:

$$M = (Q, q_0, \Sigma, \Gamma, \delta, q_{acc}, q_{rej})$$

כאשר $\Sigma = \{0, 1\}$ ו- $\Gamma = \{0, 1, _ \}$. המצבים והמעברים מתוארים בהתרשים מצבים שלמטה:



בכדי לחשב את הסיבוכיות זמן של L נתאר את המכונה בפסאודוקוד.
 $M =$ על כל קלט w :

(1) • אם המילה היא מילת הריקה תקבל.

• אחרת ממשיכה לשלב 2.

(2) • אם התו הנקרא 0 תדחה.

• אחרת אם התו הנקרא הוא $_$ תקבל.

• אחרת חוזרת לשלב 2.

ככל שהקלט ארוך יותר, כך M תבצע צעדי חישוב רבים יותר.
 בפרט המספר הצעדים המקסימלי הוא במקרה שבקלט w יש רק תווי 1 ולכן המ"ט תבצעת $n = |w|$ צעדי חישוב. בכל מקרה אחר היא תבצעת פחות מ- n צעדים.

\Leftarrow חסם העליון של מספר צעדי חישוב של M על קלט w הוא n כאשר $n = |w|$.

$\Leftarrow M$ עוצרת בזמן $O(n)$

$\Leftarrow L$ כריעה בזמן $O(n)$.

$\Leftarrow L \in TIME(n)$

באופן כללי, אם מכונה על קלט w מבצעת פחות מ- $|w|$ צעדי חישוב, המשמעות היא המכונה כלל לא קראה את כל הקלט, וזה אינו מקרה נפוץ במיוחד (אם כי הוא בהחלט קיים). אם כן ברור שמדידת זמן הריצה היא תמיד ביחס לאורך הקלט.

דוגמה 9.2

בדוגמה הזו נבנה מכונת טיורינג M עם סרט יחיד שמכריעה את השפה $L = \{a^n b^n \mid n \geq 0\}$ ואז נחשב את הסיבוכיות זמן שלה.

בניית המכונת טיורינג

$M =$ "על קלט w :

(1) אם התו הנקרא הוא $_$ $M \Leftarrow$ תקבלת.

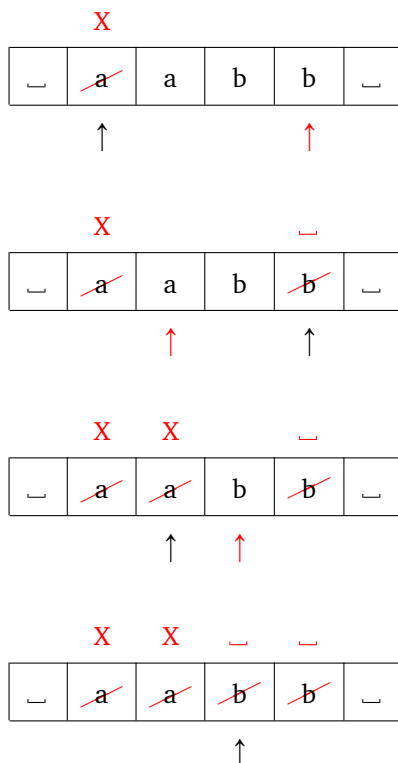
(2) אחרת אם התו הנקרא הוא $b \Leftarrow M$ דוחה.

(3) אחרת מוחקת את התו הנקרא ע"י X .

(4) מזיזה את הראש ימינה עד התו הראשון משמאל ל- $_$ בקצה הימין של הקלט.

• אם התו הוא a או $X \Leftarrow M$ דוחה.

• אחרת מוחקת את התו הנקרא ע"י $_$, מזיזה את הראש שמאלה עד התו הראשון מימין ל- X וחוזרת לשלב (1).



זמן הריצה

נסמן את אורך הקלט ב- $n = |w|$.

• M מבצעת $\left\lfloor \frac{n}{2} \right\rfloor$ איטרציות.

• בכל איטרציה M סורקת את הסרט פעמיים וזה עולה $O(n)$ צעדים.

• לכן סה"כ הזמן הריצה של M חסום ע"י

$$\frac{n}{2} \cdot O(n) = O(n^2).$$

9.3 דוגמה

כעת נבנה מכונת טיורינג מרובת סרטים M' שמכריעה את השפה $L = \{a^n b^n \mid n \geq 0\}$ ונראה שהסיבוכיות זמן שונה מזה של המכונת טיורינג עם סרט יחיד שראינו בדוגמה הקודמת.

בניית המכונה

$$M' = \text{"על קלט } w$$

(1) מעתיקה את ה- b -ים לסרט 2 (ותוך כדי בודקת האם w מהצורה a^*b^*). $O(n)$.

(2) מזיזה את הראשים לתחילת הסרטים. $O(n)$.

(3) אם שני הראשען מצביעים על $_$ \Leftarrow מקבלת.

(4) אם אחד הראשים מצביע על $_$ והשני לא \Leftarrow תדחה.

(5) מזיזה את שני הראשים ימינה וחוזרת לשלב (3). "שלבים (3-5): $O(n)$.

סרט (1)

$_$	a	a	b	b	$_$
------	---	---	---	---	------

סרט (2)

$_$	b	b			$_$
------	---	---	--	--	------

סיבוכיות זמן

נסמן את אורך הקלט ב- $n = |w|$.

הזמן הריצה של M' הוא $O(n)$.

הגדרה 9.3 $TIME(f(n))$

נגדיר הקבוצת השפות $TIME(f(n))$ כך שלכל שפה $L \in TIME(f(n))$ קיימת מכונת טירינג דטרמיניסטית שמכריעה את L בזמן לכל היותר $f(n)$, כאשר n הוא האורך של הקלט:

$$TIME(f(n)) = \{L \mid \text{קיימת מ"ט דטרמיניסטית המכריעה } L \text{ בזמן } O(f(n))\}.$$

דוגמה 9.4

עבור השפה בדוגמה 9.2:

$$L \in TIME(n^2).$$

דוגמה 9.5

עבור השפה בדוגמה 9.3:

$$L \in TIME(n).$$

דוגמה 9.6

הדוגמה הזו ראינו בדוגמה 1.5. נבנה מכונת טירינג M עם סרט יחיד שמכריעה את השפה $L = \{a^{2^n} \mid n \geq 0\}$ ואז נחשב את הסיבוכיות זמן שלה.

הרעיון

המכונה תסרוק את הסרט שלה משמאל לימין ובכל איטרציה תמחק חצי מה- a -ים שנשארו.

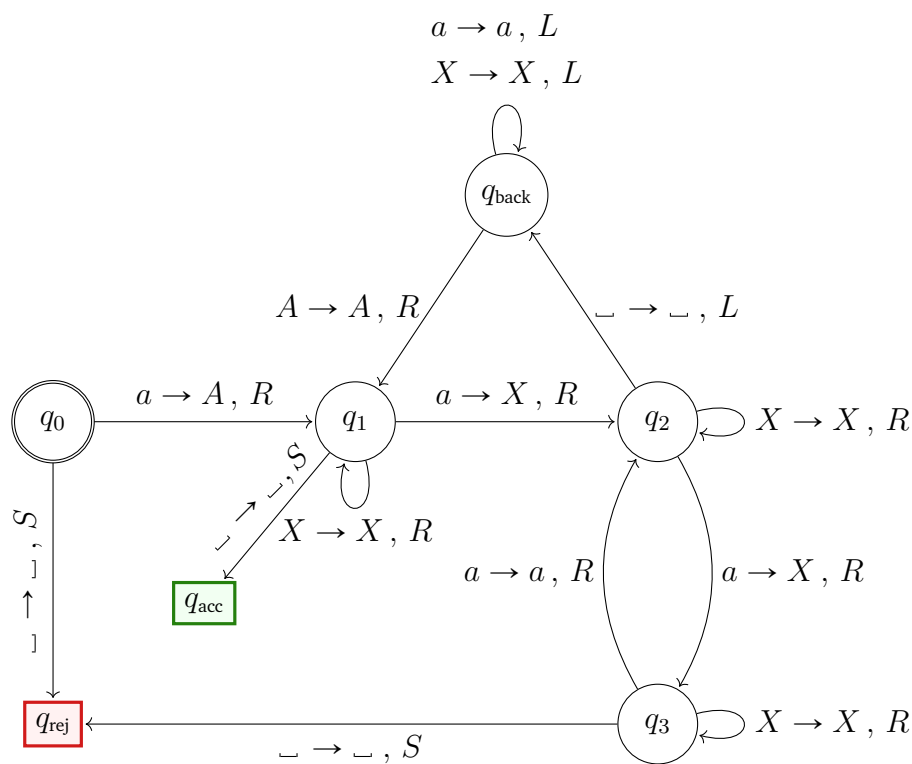
בניית המכונה

$M = "$ על קלט w :

- (1) אם התו הנקרא הוא $\sqcup \Leftarrow$ דוחה.
- (2) אחרת מחליפה את ה- a הראשון ב- A (כדי לסמן את תחילת הסרט).
- (3) סורקת את הסרט משמאל לימין ומוחקת כל a שני:

- אם הסרט מכיל a יחיד \Leftarrow מקבלת.
- אם הסרט מכיל מספר אי-זוגי של a -ים \Leftarrow דוחה.
- מזיזה את הראש שמאלה לתחילת הסרט וחוזרת לשלב (3).

התרשים מצבים של המכונה מתואר באיור למטה:



סיבוכיות זמן

- שלב (3) דורש n צעדים לכל היותר.
 - המכונה חוזרת על שלב (3) לכל היותר n פעמים.
 - לפיכך הסיבוכיות זמן של השפה L היא $O(n^2)$. כלומר:
- $$L \in TIME(n^2).$$

דוגמה 9.7 חיסור בינארי

בדוגמה זו נבנה מכונת טיורינג מ-3 סרטים שמקבלת כקלט שני מספרים שלמים x, y ($x > y$) בבסיס בינארי ומחשבת את החיסור $x - y$.

בניית המכונה

$SUBTRACT = "$ על קלט $\langle x, y \rangle$ כאשר x, y שלמים בבסיס בינארי ו- $x > y$:

(1) רושמת את x בסרט 1 ואת y בסרט 2 משמאל לימין כך שהתאים עם הקצוות הימניות של x ו- y מיושרים זה מתחת לזה. בתחילה סרט 3 ריק.

(2) מעמידה את ראשי סרטים 1, 2, 3 על התאים המיושרים של הקצוות הימניים של הקלטים. בפרט, ראשי סרטים 1 ו-2 נמצאים על הספרות הפחות משמעותיות של x ו- y , וראש סרט 3 נמצא בתא שמתחתם.

(3) שלב זה מבצע חיסור ללא חוב.

יהי $\sigma_1 \in \{0, 1, _ \}$ תו הנקרא בסרט 1 ויהי $\sigma_2 \in \{0, 1, _ \}$ תו הנקרא בסרט 2.

- אם $(\sigma_1, \sigma_2) = (_, _)$ אז M מקבלת.
- אם $(\sigma_1, \sigma_2) = (0, 0)$, כותבת $(0, 0, 0)$, מזיזה את הראשים (L, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (1, 0)$, כותבת $(1, 0, 1)$, מזיזה את הראשים (L, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (1, 1)$, כותבת $(1, 1, 0)$, מזיזה את הראשים (L, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (_, 0)$, כותבת $(_, 0, 0)$, מזיזה את הראשים (S, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (_, 1)$, כותבת $(_, 1, 1)$, מזיזה את הראשים (S, L, L) ועוברת לשלב הבא.
- אם $(\sigma_1, \sigma_2) = (0, _)$, כותבת $(0, _, 0)$, מזיזה את הראשים (L, S, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (1, _)$, כותבת $(1, _, 1)$, מזיזה את הראשים (L, S, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (0, 1)$, כותבת $(0, 1, 1)$, מזיזה את הראשים (L, L, L) ועוברת לשלב הבא.

(4) שלב זה מבצע חיסור כאשר קיים חוב.

- אם $(\sigma_1, \sigma_2) = (1, 0)$, כותבת $(1, 0, 0)$, מזיזה את הראשים (L, L, L) ועוברת לשלב הקודם.
- אם $(\sigma_1, \sigma_2) = (0, 1)$, כותבת $(0, 1, 1)$, מזיזה את הראשים (L, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (1, 1)$, כותבת $(1, 1, 1)$, מזיזה את הראשים (L, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (0, 0)$, כותבת $(0, 0, 1)$, מזיזה את הראשים (L, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (_, 0)$, כותבת $(_, 0, 1)$, מזיזה את הראשים (S, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (_, 1)$, כותבת $(_, 1, 0)$, מזיזה את הראשים (S, L, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (0, _)$, כותבת $(0, _, 1)$, מזיזה את הראשים (L, S, L) וחוזרת על שלב זה.
- אם $(\sigma_1, \sigma_2) = (1, _)$, כותבת $(1, _, 1)$, מזיזה את הראשים (L, S, L) ועוברת לשלב הקודם.

סיבוכיות זמן

יהי $n = \max(|x|, |y|)$. המכונה $SUBTRACT$ סורקת את השני סרטים שבהם כתובים המספרים x ו- y במקביל, ותוך כדי רושמת את הפלט על סרט 3. לכן $SUBTRACT$ מבצעת $O(n)$ צעדים.

לכן הסיבוכיות זמן של $SUBTRACT$ היא $O(n)$ (ליניארי).

דוגמה 9.8 האלגוריתם החילוק של אוקלידס

המשפט החילוק של אוקלידס אומר שבהינתן שני מספרים שלמים x, y , אזי קיימים שלמים q, r כך ש

$$x = qy + r, \quad 0 \leq r < |y|.$$

במקרה הפרטי ש- $0 < y < x$ אז:

$$q = \left\lfloor \frac{x}{y} \right\rfloor, \quad r = x \bmod y.$$

קיים אלגוריתם שמקבל כקלט x, y ונותן כפלט q ו- r . האלגוריתם עובד לכל שלמים x, y (בלי קשר לסימן שלהם). אנחנו נסתכל על האלגוריתם במקרה הפרטי ש- $0 < y < x$, כמתואר למטה.

בניית האלגוריתם

$DIVISION =$ "על קלט $\langle x, y \rangle$ כאשר x, y שלמים בביס בינארי ו- $0 < y < x$:

$$(1) \quad \left. \begin{array}{l} q \leftarrow 0 \\ r \leftarrow x \end{array} \right\} \text{מאתחל}$$

$$(2) \quad \text{כל עוד } r \geq y$$

$$(3) \quad r \leftarrow r - y$$

$$(4) \quad q \leftarrow q + 1$$

$$(5) \quad \text{פלט: } q, r$$

סיבוכיות זמן

נסמן $n = |\langle x, y \rangle|$ אורך הקלט.

- $DIVISION$ מבצעת r איטרציות לכל היותר.
- $r < y$ לכן y הוא חסם עליון של המספר האיטרציות המקסימלי של $DIVISION$.
- לכן $DIVISION$ מבצע $O(n)$ איטרציות לכל היותר.
- בכל איטרציה $DIVISION$ מבצע חיסור בינארי עם $SUBTRACT$ אשר (כפי שראינו בדוגמה 9.7) הוא $O(n)$.
- לכן הסיבוכיות זמן של $DIVISION$ היא

$$O(n^2).$$

9.2 יחס בין הסיבוכיות של מכונת טיורינג עם סרט יחיד ומכונת טיורינג מרובת סרטים

משפט 9.1

לכל מכונת טיורינג מרובת סרטים M הרצה בזמן $t(n)$ קיימת מכונת טיורינג סרט יחיד M' השקולה ל- M ורצה בזמן $O(t^2(n))$.

הוכחה:

תהי M מכונת טיורינג כלשהי עם k סרטים הרצה בזמן $O(t(n))$.

נבנה מכונת טיורינג S עם סרט אחד שרצה בזמן $O(t^2(n))$.

- ראשית נרשום את התוכן של ה- k סרטים של M על הסרט היחיד של S .
- התכנים של הסרטים מופרדים על ידי תו $\#$ על הסרט היחיד.
- המיקומים של הראשים של כל הסרטים של M מסומנים על הסרט היחיד על ידי חצים.

להלן יש דוגמה לכך של מכונת טיורינג עם 3 סרטים:

המכונת טיורינג M

0	1	0	1	0	␣	␣
---	---	---	---	---	---	---

↑

a	a	a	␣	␣	␣	␣
---	---	---	---	---	---	---

↑

a	b	b	␣	␣	␣	␣
---	---	---	---	---	---	---

↑

המכונת טיורינג S

#	0	1	0	1	0	#	a	a	a	#	a	b	b
				↑					↑			↑	

אפשר לסמלץ צעד חישוב אחד של M במכונת טיורינג S באופן הבא:

בניית המכונה S

(1) תחילה S מאתחלת את הסרט שלה בלכתוב את התכנים של ה- k סרטים על הסרט היחיד שלה, עם תו $\#$ להפריד בין סרטים שונים של M .

(2) בנוסף S רושמת תו $\#$ בקצה השמאלי כדי לסמן את התחילת הקלט ותו $\#$ בקצה הימין כדי לסמן את סוף הקלט.

(3) בכדי לסמלץ צעד חישוב אחד של M בהמכונה S , המכונה S סורקת את הסרט מ- $\#$ הראשון ל- $\#$ האחרון. בסריקה זו S זוכרת את המיקומים של ה- k ראשים על פי התאים שמסומנים עם חצים, באמצעות k תאי זכרון.

(4) אחר כך S מבצעת סריקה שנייה של הסרט. בסריקה זו, לפי הפונקציה המעברים של M , המכונה S מבצעת, לכל $1 \leq i \leq k$:

- כתיבה של הסימן החדש בסרט ה- i במיקום הנוכחי של הראש של סרט ה- i ,
- תזוזה של הראש של סרט ה- i .

(5) במצב שהראש של אחד הסרטים של M אז ימינה לתו רווח מצד ימין של סוף הקלט, אז S תוסיף תא אחד עם תו רווח מצד שמאל של ה- $\#$ המפריד בין סרטים, ואחר כך היא תזיז את כל התאים שבצד ימין של התא המוסף מקום אחד ימינה.

סיבוכיות זמן של המכונה S

יהי n האורך של התוכן הכי ארוך מתוך כל התכנים של ה- k סרטים של M .

- האורך של התוכן של S שווה לסכום הארכים של התכנים של ה- k סרטים של M .
- נתון שהזמן הריצה של M הוא $O(t(n))$.
- $\Leftarrow M$ עושה שימוש של $O(t(n))$ תאים.
- \Leftarrow אורך הקלט של S חסום מלמעלה ע"י $O(t(n))$.
- \Leftarrow סריקה שלמה של הקלט של S מתבצעת ב- $O(t(n))$ צעדים.
- \Leftarrow מכיוון שכל סריקה דורשת זמן $O(t(n))$ וכדי לדמות צעד אחד של M , המכונה S מבצעת שתי סריקות, אז S לוקחת זמן $O(t(n))$ לבצע צעד חישוב אחד של M .
- \Leftarrow הסיבוכיות זמן של S היא:

$$O(t(n)) \quad O(t(n)) = O(t^2(n))$$

■

9.3 יחס בין הסיבוכיות של מכונת טיורינג דטרמיניסטית ומכונת טיורינג אי דטרמיניסטית

משפט 9.2

לכל מכונת טיורינג אי-דטרמיניסטית N הרצה בזמן $f(n)$, קיימת מכונת טיורינג דטרמיניסטית D השקולה ל- N ורצה בזמן $2^{f(n)}$.

הוכחה:

בהינתן מכונת טיורינג אי-דטרמיניסטית N הרצה בזמן $f(n)$ מכונת טיורינג דטרמיניסטית D באותו אופן כמו בהוכחת השקילות במשפט 4.1.

כלומר, בהינתן קלט w , D תסרוק את עץ החישוב של N ו- w לרוחב ותקבל כל אחד החישובים של N המסתיים ב- q_{acc} .

בהינתן קלט w באורך n :

- כל מסלול בעץ החישוב של N על w חסום ע"י $f(n)$.
- מספר החישובים ש- D מבצעת חסום ע"י מספר הקודקודים בעץ החישוב של N ו- w .
- מכיוון שמספר הבנים של כל קודקוד בעץ החישוב חסום ע"י

$$C = 3|Q| \cdot |\Gamma|$$

מספר הקודקודים בעץ החישוב חסום ע"י

$$C^0 + C^2 + \dots C^{f(n)} \leq C^{f(n)+1} = C \cdot C^{f(n)}.$$

ולכן זמן הריצה של D חסום ע"י

$$f(n) \cdot C \cdot C^{f(n)} \leq C^{f(n)} \cdot C^{f(n)} = C^{2f(n)} = (C^2)^{f(n)} = 2^{C' \cdot f(n)} = 2^{O(f(n))}.$$

נתייחס כאן לשני החסמים הבאים:



- (1) חסם פולינומיאלי הוא חסם מהצורה n^c עבור $c > 0$ כלשהו.
- (2) חסם אקספוננציאלי הוא חסם מהצורה 2^{n^c} עבור $c > 0$ כלשהו.

9.4 המחלקה P

הגדרה 9.4 בעיית הכרעה

בעיית הכרעה מוגדרת באופן הבא:

"בהינתן קלט כלשהו, האם הקלט מקיים תנאי מסוים"

דוגמה 9.9

בהינתן מספר n , האם n ראשוני?

כל בעיית הכרעה ניתן לתאר כשפה שקולה:

$$L_{\text{prime}} = \{ \langle n \rangle \mid n \text{ ראשוני} \}.$$

משפט 9.3

. שפה \equiv בעיית הכרעה

הגדרה 9.5 אלגוריתם זמן פולינומיאלי

אומרים כי אלגוריתם A מכריעה בעייה בזמן פולינומיאלי אם קיים קבוע $c > 0$ כך שזמן הריצה של A על כל קלט w חסום ע"י $O(|w|^c)$.

משפט 9.4 התזה של צירץ' (Church Thesis)

אם קיים אלגוריתם המכריע בעייה בזמן פולינומיאלי, אז קיימת מכונת טיורינג דטרמיניסטית המכריעה את השפה השקולה לבעייה זו בזמן פולינומיאלי.

. מכונת טיורינג \equiv אלגוריתם מכריעה

הגדרה 9.6 המחלקה P

המחלקה P היא אוסף כל הבעיות (השפות) שקיים עבורן אלגוריתם (מכונת טיורינג דטרמיניסטית) המכריע אותן בזמן פולינומיאלי.

דוגמה 9.10

בדוגמה 9.2 הוכחנו כי קיימת מכונת טיורינג דטרמיניסטית שמכריעה את השפה

$$L = \{a^n b^n \mid n \geq 0\} \in P$$

בזמן $O(n^2)$. לכן $L \in P$.

דוגמה 9.11

תהי L_{prime} השפה הבאה:

$$L_{\text{prime}} = \{\langle n \rangle \mid n \text{ ראשוני}\}.$$

הוכיחו: $L_{\text{prime}} \in P$

פתרון:

כדי להוכיח כי $L_{\text{prime}} \in P$, נבנה אלגוריתם שמכריע את L_{prime} ואחר כך נראה כי האלגוריתם רץ בזמן פולינומיאלי.

האלגוריתם עצמו עושה שימוש של המשפט הבא:

משפט: למספר פריק יש מחלק הקטן מ- או שווה לשורשו

יהי n מספר שלם. אם n פריק אז קיים שלם $a \leq \sqrt{n}$ שמחלק את n .

הוכחה:

- נניח בשלילה כי n פריק ולא קיים מחלק d של n כך ש- $d \leq \sqrt{n}$.
- n פריק אז קיימים שלמים ab כך ש- $n = ab$, כאשר $1 < a, b < n$.
- נניח כי $1 < a \leq b < n$.
- על פי ההנחה שלנו, אם a מחלק n אז $a > \sqrt{n}$. בנוסף: $b \geq a > \sqrt{n}$.
- לכן

$$n = ab > (\sqrt{n})(\sqrt{n}) = n,$$

ז"א $n > n$ וזו סתירה!

על סמך המשפט הזה נבנה אלגוריתם $PRIME$ המכריע את L_{prime} בזמן פולינומיאלי. הרעיון של האלגוריתם הוא, בהינתן קלט n , לבדוק אם קיים מחלק של n אשר קטן מ- \sqrt{n} או שווה ל- \sqrt{n} . אם כן אז n פריק ואם לא אז n ראשוני.

בניית האלגוריתם

$PRIME = \text{על קלט } x$:

(1) בודק אם $x = \langle n \rangle$ ו- n מספר שלם חיובי.

• אם לא \Leftarrow דוחה.

(2) אם $n = 1 \Leftarrow$ דוחה.

(3) אם $n = 2 \Leftarrow$ מקבל.

(4) אחרת לכל $3 \leq d \leq \lfloor \sqrt{n} \rfloor$:

• מחשב $n \bmod d$.

• $n \bmod d = 0 \Leftarrow$ מקבל.

(5) מקבל.

סיבוכיות זמן

• $PRIME$ מבצע $O(n)$ איטרציות.

• בכל איטרציה $PRIME$ מחשב $n \bmod d$ באמצעות האלגוריתם $DIVISION$, אשר הוכחנו בדוגמה 9.8 שהוא רץ בזמן $O(n^2)$.

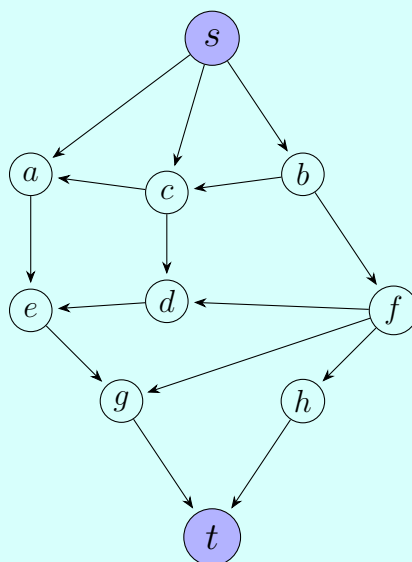
• לכן הסיבוכיות זמן של $PRIME$ היא $O(n^3)$.

לפיכך:

$$L_{\text{prime}} \in P.$$

9.5 בעיית PATH

הגדרה 9.7 בעיית המסלול בגרף מכוון



קלט: גרף מכוון $G = (V, E)$ ושני קודקודים $s, t \in V$.

פלט: האם קיים מסלול ב- G מ- s ל- t ?

$$PATH = \{ \langle G, s, t \rangle \mid \text{קיים מסלול ב- } G \text{ מ- } s \text{ ל- } t \}$$

משפט 9.5

$$PATH \in P.$$

הוכחה: נבנה אלגוריתם A שמכריע את $PATH$ בזמן פולינומיאלי.

בניית האלגוריתם

$A = \text{על קלט } \langle G, s, t \rangle$:

(1) צובע את s .

(2) מבצע $|V| - 1$ פעמים:

• לכל צלע $(u, v) \in E$:

* אם u צבוע \Leftarrow צבע את v .

(3) • אם t צבוע \Leftarrow מקבל.

• אחרת \Leftarrow דוחה.

הוכחת הנכונות

הוכחה של הכיוון \Leftarrow

אם $x \in PATH$:

$\Leftarrow x = \langle G, s, t \rangle$ ו- $G = (V, E)$ גרף מכוון וגם $s, t \in V$ כך שקיים מסלול מ- s ל- t .

\Leftarrow קיים קבוצת צלעות

$$C = \{u_1u_2, u_2u_3, \dots, u_{k-2}u_{k-1}, u_{k-1}u_k\} \subseteq E,$$

כאשר $u_1 = s, u_k = t$ ו- $k \leq |V|$.

\Leftarrow לכל $1 \leq i \leq k-1$, באיטרציה i הקודקוד u_{i+1} יצובע.

\Leftarrow מכיוון ש- $k \leq |V|$ אזי עד סוף הלולאה הקודקוד $u_k = t$ יצובע.

\Leftarrow מכיוון שקודקוד t צובע אז A מקבל.

הוכחה של הכיוון \Rightarrow

$x \notin PATH \Leftarrow 2$ מקרים:

מקרה 1 $x \neq \langle G, s, t \rangle$ ואז A דוחה.

מקרה 2 אחרת $x = \langle G, s, t \rangle$ כאשר $G = (V, E)$ גרף מכוון וגם $s, t \in V$ כך שלא קיים מסלול מ- s ל- t .

לאחר האיטרציה ה- i כל קודקוד שצובע הוא קודקוד שניתן להגיע אליו מ- s ע"י מסלול מכוון של אורך i לכל היותר.

לא ניתן להגיע לקודקוד t מקודקוד ע"י אף מסלול בגרף G .

\Leftarrow לא קיים סדרת קודקודים $s \rightarrow u_2 \rightarrow u_3 \rightarrow \dots \rightarrow t$ כך שבסוף הלולאה t יהיה צובע. $\Leftarrow A$ דוחה.

סיבוכיות זמן

- נסמן אורך הקלט $N = \langle G, s, t \rangle$.
- נסמן $n = |V|$ ו- $m = |E|$.
- A מבצע לולאה לכל $1 \leq i \leq |V| - 1$ שדורש $O(|V|)$ צעדים.
- בתוך הלולאה A מבצע לולאה מעל צלעות עבורן הנקודה ההתחלתית צובעת. \Leftarrow שלב זה דורש $O(|E|)$ צעדים.
- סבה"כ הסיבוכיות זמן של A היא $O(|E| \cdot |V|) = O(nm)$.
- בנוסף: $n \leq N$ וגם $m \leq N$ לכן A דורש N^2 צעדי חישוב לכל היותר.
- לכן A רץ בזמן $O(N^2)$.

■ מצאנו אלגוריתם שמכריע את $PATH$ בזמן $O(N^2) \Leftarrow PATH \in TIME(N^2) \Leftarrow PATH \in P$.

9.6 הבעיית $RELPRIME$

הגדרה 9.8 מספרים זרים (Relatively prime)

אומרים כי שני מספרים שלמים x, y הם זרים אם המחלק המשותף הגדול ביותר, מסומן $\gcd(x, y)$, שווה 1.

הגדרה 9.9 בעיית RELPRIME

קלט: שני מספרים x ו- y .

פלט: האם x ו- y זרים?

$$RELPRIME = \{ \langle x, y \rangle \mid \gcd(x, y) = 1 \} .$$

אנחנו נוכיח כי ניתן להכריע את $RELPRIME$ בזמן פולינומיאלי, כלומר נוכיח $RELPRIME \in P$ במשפט 9.8 למטה. לפניכן נסביר את האלגוריתם של אוקלידס למציאת ה- \gcd של שני שלמים, ומתוך זה נוכל לחשב את הסיבוכיות זמן של $RELPRIME$. ראשית נזכיר משפט שלמדנו בקורסים קודמים:

משפט 9.6 השמפט של האלגוריתם של אוקלידס

אם x, y שלמים אז

$$\gcd(x, y) = \begin{cases} x & y = 0 \\ \gcd(y, x \bmod y) & y \neq 0 \end{cases} .$$

■ **הוכחה**: ההוכחה היא לא חלק של הקורס ומופיע בסעיף האחרון "הוכחות של משפטים שימושיים" בדף 110.

האלגוריתם של אוקלידס הוא אלגוריתם, שמקבל כקלט שני מספרים x, y ופולט את $\gcd(x, y)$. הוא מתבוסס על המשפט 9.6. האלגוריתם עצמו הוא כדלקמן:

$EUCLID =$ על קלט x, y :

(1) כל עוד $y \neq 0$

(2) $x \leftarrow x \bmod y$

(3) $\text{swap}(x, y)$

(כלומר מחליפים בין x ו- y).

(4) מחזירים את x .

לדוגמה:

$$\gcd(18, 32) = \gcd(32, 18) = \gcd(18, 14) = \gcd(14, 4) = \gcd(4, 2) = \gcd(2, 0) = 2 .$$

כדי להוכיח כי $RELPRIME \in P$ נצטרך למשפט עזר הבא:

משפט 9.7 (משפט עזר)

אם $x > y$ אז $x \bmod y < \frac{x}{2}$

■ **הוכחה**: ההוכחה היא לא חלק של הקורס ומופיע בסעיף האחרון "הוכחות של משפטים שימושיים" בדף 111.

משפט 9.8

$$RELPRIME \in P.$$

הוכחה:

נבנה אלגוריתם A המכריע את $RELPRIME$ בזמן פולינומיאלי. $RELPRIME$ היא השפה של הבעיה, שמקבלת כקלט שני מספרים שלמים x, y ומחזירה תשובה לשאלה, האם x, y זרים. כלומר:

$$\langle x, y \rangle \in RELPRIME \iff \gcd(x, y) = 1.$$

לכן A משתמש בהאלגוריתם של אוקלידס $EUCLID(x, y)$ כדי לחשב $\gcd(x, y)$.

בניית האלגוריתם A המכריע $RELPRIME$:

$A =$ " על קלט $\langle x, y \rangle$: A מריץ את $EUCLID$ על x ו- y .

• אם $EUCLID(x, y)$ מחזיר $\gcd(x, y) = 1$ אז A מקבל.

• אחרת A דוחה.

הוכחת הנכונות

הנכונות של A מנובעת ישר מהנכונות של האלגוריתם האוקלידס, $EUCLID$.

סיבוכיות זמן

נראה כי A רץ בזמן פולינומיאלי בגודל הקלט $\langle x, y \rangle$. נסמן את אורך הקלט $n = |\langle x, y \rangle|$.

• לפי משפט 9.7: $x \bmod y < \frac{x}{2}$.

• בכל איטרציה, בשלב (2) המשתנה x מקבל את הערך החדש $x \leftarrow x \bmod y$.

ניתן לחשב את $x \bmod y$ בעזרת האלגוריתם החילוק של אוקלידס $DIVISION$ שרץ בזמן $O(n^2)$ (ראו דוגמה 9.8).

• לכן בכל איטרציה הערך החדש של x קטן ממש מחצי של הערך הקודם של x .

• לכן אחרי כל איטרציה, x קטן בלפחות חצי.

• בשלב (3), A מחליף בין x ו- y , אז אחרי כל 2 איטרציות, גם x קטן בלפחות חצי וגם y קטן בלפחות חצי.

• לכן המספר הפעמים המקסימלי שאפשר לבצע שלבים (2) ו-(3) היא $m \triangleq \min(2 \lfloor \log_2 x \rfloor, 2 \lfloor \log_2 y \rfloor)$.

• לכן m הוא חסם עליון של מספר האיטרציות ש $EUCLID$ מבצע.

• $m \leq n$ כאשר n הוא האורך של הקלט $\langle x, y \rangle$ לכן $EUCLID$ מבצע $O(n)$ איטרציות.

• כל איטרציה מבצעת $DIVISION$ כדי לחשב $x \bmod y$ אשר רץ בזמן $O(n^2)$.

$\Leftarrow A$ רץ בזמן $O(n^3)$. כלומר:

$$RELPRIME \in TIME(n^3).$$

לכן:

$$RELPRIME \in P.$$



9.7 *הוכחות של משפטים שימושיים

משפט 9.9 השמפט של האלגוריתם של אוקלידס

אם x, y שלמים אז

$$\gcd(x, y) = \begin{cases} x & y = 0 \\ \gcd(y, x \bmod y) & y \neq 0 \end{cases}.$$

הוכחה: (להעשרה בלבד)

המטרה של ההוכחה הזו היא רק להוסיף הבנה להוכחה של משפט 9.8 לסיבוכיות זמן של $RELPRIME$ למטה. היא לא הוכחה שאתם תיבחנו עליה ואפשר לדלג עליה.

נתחיל אם משפט החילוק של אוקלידס, שאומר שאם x, y שלמים אז קיימים שלמים q ו- $0 \leq r < y$ כך ש:

$$x = qy + r = \left\lfloor \frac{x}{y} \right\rfloor y + (x \bmod y). \quad (1*)$$

נגדיר $d \triangleq \gcd(x, y)$.

מכיוון ש- d הוא מחלק משותף של x ו- y אז $d \mid x$ וגם $d \mid y$. לכן בזכות משוואה (1*):

$$(d \mid x) \wedge (d \mid y) \xrightarrow{\text{משוואה (1*)}} d \mid (x \bmod y)$$

ז"א $d \mid y$ וגם $d \mid (x \bmod y)$ אז בהכרח:

$$d \mid \gcd(y, x \bmod y). \quad (2*)$$

כעת נגדיר $\bar{d} \triangleq \gcd(y, x \bmod y)$.

מכיוון ש- \bar{d} הוא מחלק משותף של y ו- $x \bmod y$ אז $\bar{d} \mid y$ וגם $\bar{d} \mid x \bmod y$. לכן בזכות משוואה (1*):

$$(\bar{d} \mid y) \wedge (\bar{d} \mid x \bmod y) \xrightarrow{\text{משוואה (1*)}} \bar{d} \mid x$$

ז"א $\bar{d} \mid y$ וגם $\bar{d} \mid x$ אז בהכרח:

$$\bar{d} \mid \gcd(x, y). \quad (3*)$$

לסיכום, לפי משוואות (2*) ו- (3*):

$$d \mid \bar{d} \quad \wedge \quad \bar{d} \mid d.$$

מכיוון ש- $d, \bar{d} > 0$ אז בהכרח $d = \bar{d}$, ז"א $\gcd(x, y) = \gcd(y, x \bmod y)$. ■

משפט 9.10 (משפט עזר)

אם $x > y$ אז $x \bmod y < \frac{x}{2}$.**הוכחה:** יש שני מקרים:

$$y \leq \frac{x}{2} \quad (1)$$

$$y > \frac{x}{2} \quad (2)$$

נוכיח את הטענה עבור שני המקרים.

מקרה 1: $y \leq \frac{x}{2}$.

לפי משפט החילוק של אוקלידס אם x, y שלמים עבורם $x > y$ אז קיימים $q = \left\lfloor \frac{x}{y} \right\rfloor$ ו- $r = x \bmod y$ כך ש $0 \leq r < y$ -

$$x = qy + r = \left\lfloor \frac{x}{y} \right\rfloor y + (x \bmod y) .$$

בפרט אם $r < y$ וגם $y \leq \frac{x}{2}$ לפיכך $x \bmod y < y \leq \frac{x}{2}$.

מקרה 2: $y > \frac{x}{2}$.

לפי משפט החילוק של אוקלידס אם x, y שלמים עבורם $x > y$ אז קיימים שלם $q = \left\lfloor \frac{x}{y} \right\rfloor$ ושלם $r = x \bmod y$ כך ש $0 \leq r < y$ -

$$x = qy + r = \left\lfloor \frac{x}{y} \right\rfloor y + (x \bmod y) .$$

בפרט אם $y > \frac{x}{2}$ אז $x < 2y$. אז בהכרח $q < 2$. מכיון ש- $x > y$ ו- $q = \left\lfloor \frac{x}{y} \right\rfloor$ אז הערך המינימלי של q הוא $q = 1$. לכן אם $q < 2$ בהכרח $q = 1$. לכן יש לנו

$$x = qy + r = (1)y + r + (x \bmod y) .$$

מכאן

$$x - y = x \bmod y .$$

כעת נציב את ההנחה ההתחלתית $y > \frac{x}{2} \Leftrightarrow x - y < \frac{x}{2}$ ונקבל

$$x \bmod y < \frac{x}{2} .$$



שיעור 10

המחלקה P והמחלקה NP

10.1 המחלקה P

הגדרה 10.1 בעיית הכרעה

בעיית הכרעה מוגדרת באופן הבא:

"בהינתן קלט כלשהו, האם הקלט מקיים תנאי מסויים ? "

דוגמה 10.1 דוגמה של בעיית הכרעה

לדוגמה, בהינתן מספר n , האם n ראשוני?

משפט 10.1 שקיות בין בעייה לשפה

כל בעייה הכרעה ניתן לתאר כשפה שקולה:

. בעיית הכרעה \equiv שפה

דוגמה 10.2

לדוגמה, הבעיית הכרעה הבאה:

"בהינתן מספר n , האם n ראשוני? "

ניתנת לרשום כשפה הבאה:

$$L_{\text{prime}} = \{ \langle n \rangle \mid n \text{ ראשוני} \}.$$

הגדרה 10.2 אלגוריתם זמן פולינומיאלי

אומרים כי אלגוריתם A מכריע בעייה בזמן פולינומיאלי אם קיים קבוע $c > 0$ כך שזמן הריצה של A על קלט w חסום ע"י $O(|w|^c)$.

התזה של צרף' טיורינג אומר שאם קיים אלגוריתם המכריע בעייה בזמן פולינומיאלי, אזי קיימת מכונת טיורינג דטרמיניסטית המכריעה את השפה השקולה לבעייה זו בזמן פולינומיאלי.

. אלגוריתם מכריע \equiv מכונת טיורינג דטרמיניסטית

הגדרה 10.3 המחלקה P

המחלקה P היא אוסף כל הבעיות (השפות) שקיים עבורן אלגוריתם (מכונת טיורינג דטרמיניסטית) המכריע אותן בזמן פולינומיאלי.

10.2 דוגמאות לבעיות ב- P

(1)

$$PATH = \{ \langle G, s, t \rangle \mid t \text{ -} s \text{ מסלול מ-} G \} \in P$$

(2)

$$RELPRIME = \{ \langle x, y \rangle \mid x \text{ ו-} y \text{ זרים} \} \in P$$

10.3 בעיית המסלול ההמילטוני $HAMPATH$

הגדרה 10.4 $HAMPATH$

בהינתן גרף מכוון $G = (V, E)$ ושני קודקודים $s, t \in V$. מסלול ההמילטוני מ- s ל- t ב- G הוא מסלול מ- s ל- t שעובר דרך כל קודקוד בגרף בדיוק פעם אחת.

לדוגמה:



הגדרה 10.5 בעיית $HAMPATH$

קלט: גרף מכוון $G = (V, E)$ ושני קודקודים $s, t \in V$.

פלט: האם G מכיל מסלול ההמילטוני מ- s ל- t ?

$$HAMPATH = \{ \langle G, s, t \rangle \mid t \text{ -} s \text{ מסלול ההמילטוני מ-} G \}$$

נשאל שאלה: האם $HAMPATH \in P$?

לא ידוע האם קיים אלגוריתם המכריע את $HAMPATH$ בזמן פולינומיאלי (שאלה פתוחה).

• בהינתן קלט $\langle G, s, t \rangle$, האם $\langle G, s, t \rangle \in HAMPATH$?

• נענה על שאלה אחרת:

בהינתן קלט $\langle G, s, t \rangle$, ומחרוזת y , האם $\langle G, s, t \rangle \in HAMPATH$?

• יתן לבדוק האם y היא מסלול ההמילטוני מ- s ל- t ב- G בזמן פולינומיאלי ולענות בהתאם.

• במקרה זה, אומרים כי $HAMPATH$ ניתנת לאימות בזמן פולינומיאלי.

10.4 אלגוריתם אימות

הגדרה 10.6 אלגוריתם אימות

אלגוריתם אימות עבור בעיית A הוא אלגוריתם V כך שלכל קלט $w \in \Sigma^*$ מתקיים:

אם $w \in A$ אז ורק אם קיימת מחרוזת (עדות) y באורך פולינומיאלי ב- $|w|$ כך ש- V מקבל את הזוג (w, y) כלומר:

• אם $w \in A \iff$ קיים y כך ש: $V(w, y) = T$.

• אם $w \notin A \iff$ לכל y מתקיים $V(w, y) = F$.

הערה 10.1

- זמן ריצה של אלגוריתם אימות נמדד ביחס לגודל הקלט $|w|$.
- אלגוריתם אימות פולינומיאלי אם הוא רץ בזמן פולינומיאלי.

10.5 המחלקה NP

הגדרה 10.7 המחלקה NP

המחלקה NP היא אוסף כל הבעיות שקיים עבורן אלגוריתם אימות פולינומיאלי.

משפט 10.2 $HAMPATH \in NP$

בעיית המסלול ההמילטוני $HAMPATH$:

קלט: גרף מכוון $G = (V, E)$ ושני קודקודים $s, t \in V$.

פלט: האם G מכיל מסלול המילטוני מ- s ל- t ?

$$HAMPATH = \{ \langle G, s, t \rangle \mid \text{גרף מכוון המכיל מסלול המילטוני מ- } s \text{ ל- } t \}$$

הוכיחו כי $HAMPATH \in NP$.

הוכחה: נבנה אלגוריתם אימות V עבור $HAMPATH$.

$V(\langle G, s, t \rangle, y) = T$ על קלט $(\langle G, s, t \rangle, y)$:

(1) בודק האם y היא סדרה של

$$u_1, u_2, \dots, u_n$$

השונים זה מזה.

• אם לא \Leftarrow דוחה.

(2) בודק האם $u_1 = s$ ו- $u_n = t$.

• אם לא \Leftarrow דוחה.

(3) בודק שכל הצלעות (u_i, u_{i+1}) (לכל $1 \leq i \leq n$) קיימות ב- G .

• אם כן \Leftarrow מקבל.

• אם לא \Leftarrow דוחה.

נכונות

• זמן הריצה של האלגוריתם הוא פולינומיאלי בגודל הקלט.

• אם $\langle G, s, t \rangle \in HAMPATH \Leftarrow G$ מכיל מסלול המילטוני מ- s ל- $t \Leftarrow$ עבור y שהוא קידוד של מסלול זה, V יקבל את הזוג $(\langle G, s, t \rangle, y)$.

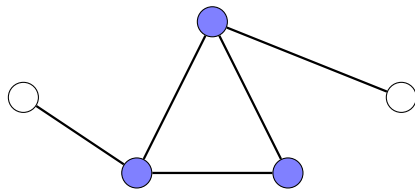
• אם $\langle G, s, t \rangle \notin HAMPATH \Leftarrow G$ לא מכיל מסלול המילטוני מ- s ל- $t \Leftarrow$ לכל y , האלגוריתם ידחה את הזוג $(\langle G, s, t \rangle, y)$.

■

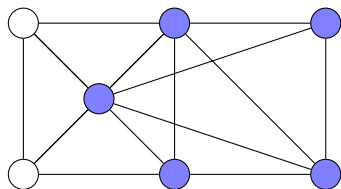
הגדרה 10.8 קליקה

בהינתן גרף לא מכוון $G = (V, E)$, קליקה ב- G היא תת-קבוצה של קודקודים $C \subseteq V$ כך שלכל שני קודקודים $u, v \in C$ מתקיים $(u, v) \in E$.

קליקה בגודל $k = 3$:



קליקה בגודל $k = 5$:





הגדרה 10.9 בעיית הקליקה

קלט: גרף לא מכוון $G = (V, E)$ ומספר k .

פלט: האם G קליקה בגודל k ?

$$CLIQUE = \{ \langle G, k \rangle \mid k \text{ גודל קליקה בגודל } k \}$$

משפט 10.3 $CLIQUE \in NP$

$$CLIQUE \in NP.$$

הוכחה: נבנה אלגוריתם אימות V עבור $CLIQUE$.

$$V = \text{על קלט } (\langle G, k \rangle, y)$$

(1) בודק האם y היא קבוצה של k קודקודים שונים מ- G .

• אם לא \Leftarrow דוחה.

(2) בודק האם כל שני קודקודים מ- y מחוברים בצלע ב- G .

• אם כן \Leftarrow מקבל.

• אם לא \Leftarrow דוחה.

הגדרה 10.10 בעיית $SubSetSum$

קלט: קבוצת מספרים $S = \{x_1, x_2, \dots, x_n\}$ ומספר t .

פלט: האם קיימת תת-קבוצה של S שסכום איבריה שווה t ?

$$SubSetSum = \left\{ \langle S, t \rangle \mid \sum_{x \in Y} x = t \text{ ש-} Y \subseteq S \text{ קיימת} \right\}$$

משפט 10.4 $SubSetSum \in NP$

$$SubSetSum \in NP.$$

הוכחה: נבנה אלגוריתם אימות V עבור $SubSetSum$.

$V = \text{על קלט } (\langle S, t \rangle, y)$:

(1) בודק האם y היא תת-קבוצה של S .

• אם לא \Leftarrow דוחה.

(2) בודק האם סכום המספרים ב- y שווה t .

• אם לא \Leftarrow דוחה.

• אחרת \Leftarrow מקבל.



10.6 הקשר בין NP למכונת טיורינג אי-דטרמיניסטית

NP=Non-deterministic polynomial-time.

משפט 10.5

לכל בעייה A :

$A \in NP$ אם ורק אם קיימת מכונת טיורינג אי-דטרמיניסטית המכריעה את A בזמן פולינומיאלי.

דוגמה 10.3

נבנה מכונת טיורינג אי-דטרמיניסטית M המכריעה את $CLIQUE$ בזמן פולינומיאלי.

$M = \text{על קלט } \langle G, k \rangle$:

• בוחרת באופן אי-דטרמיניסטי קבוצה y של k קודקודים מ- G .

• בודקת האם כל שני קודקודים מ- y מחוברים בצלע ב- G .

* אם כן \Leftarrow מקבלת.

* אחרת \Leftarrow דוחה.

אלגוריתם אימות \equiv מ"ט א"ד.

10.7 הקשר בין המחלקה P ו-NP

$P =$ כל הבעיות שניתן להכריע בזמן פולינומיאלי.

$NP =$ כל הבעיות שניתן לאמת בזמן פולינומיאלי.

משפט 10.6

$$P \subseteq NP.$$



שאלה פתוחה: האם $P = NP$?

משפט 10.7

P סגורה תחת משלים.

הוכחה: אם $A \in P$ אזי גם $\bar{A} \in P$.

הגדרה 10.11 $CoNP$

$$CoNP = \{A \mid \bar{A} \in NP\}.$$

לדוגמה:

$$\overline{HAMPATH} \in CoNP.$$

$$\overline{CLIQUE} \in CoNP.$$

שאלה פתוחה: האם $NP = CoNP$?

משפט 10.8

$$P \subseteq NP \cap CoNP.$$



שאלה פתוחה: האם $P = NP \cap CoNP$?

נדון בשאלה המרכזית: האם $P = NP$?

הגדרה 10.12 פונקציה פולינומיאלית

בהינתן פונקציה $f : \Sigma^* \rightarrow \Sigma^*$, אומרים כי f חשיבה בזמן פולינומיאלי אם קיים אלגוריתם (מ"ט דטרמיניסטי) המחשב את f בזמן פולינומיאלי.

הגדרה 10.13 רדוקציה פולינומיאלית

בהינתן שתי בעיות A ו- B . אומרים כי A ניתנת לרדוקציה פולינומיאלית ל- B , ומסמנים $A \leq_P B$, אם קיימת פונקציה $f : \Sigma^* \rightarrow \Sigma^*$ המקיימת:

(1) חשיבה בזמן פולינומיאלי

(2) לכל $w \in \Sigma^*$:

$$w \in A \iff f(w) \in B.$$

משפט 10.9 משפט הרדוקציה

לכל שתי בעיות A ו- B , אם $A \leq_P B$ אזי

(1) אם $B \in P$ אזי $A \in P$.

(2) אם $B \in NP$ אזי $A \in NP$.

מסקנה מ- (1) ו- (2):

(3) אם $A \notin P$ אזי $B \notin P$.

(4) אם $A \notin NP$ אזי $B \notin NP$.

הוכחה: מכיוון שקיימת רדוקציה $A \leq_P B$, קיימת פונקציה f חשיבה בזמן פולינומיאלי המקיימת, לכל $w \in \Sigma^*$,

$$w \in A \iff f(w) \in B.$$

יהי M_f האלגוריתם שמחשבת את f בזמן פולינומיאלי.

(1) נוכיח כי אם $B \in P$ אזי $A \in P$.

יהי M_B האלגוריתם שמכריע את B בזמן פולינומיאלי. נבנה אלגוריתם M_A המכריע את A בזמן פולינומיאלי.

התאור של M_A

$M_A =$ על כל קלט w :

1. מחשב את $f(w)$ ע"י M_f .

2. מריץ את M_B על $f(w)$ ועונה כמוה.

נוכיח כי M_A מכריע את A בזמן פולינומיאלי:

- אם $w \in A$ $\iff f(w) \in B \iff M_B$ מקבל את $f(w)$ $\iff M_A$ מקבל את w .
- אם $w \notin A$ $\iff f(w) \notin B \iff M_B$ דוחה את $f(w)$ $\iff M_A$ דוחה את w .

נוכיח כי זמן הריצה של M_A הוא פולינומיאלי בגודל הקלט $|w|$ בזמן פולינומיאלי:

- נסמן ב- P_f את הפולינום של M_f .
- נסמן ב- P_B את הפולינום של M_B .

זמן הריצה של M_A על קלט w שווה

$$P_f(|w|) + P_B(|f(w)|)$$

מכיוו ש- $|f(w)| \leq P_f(|w|)$, זמו הריצה של M_A על w חסום ע"י

$$P_f(|w|) + P_B(P_f(|w|)) = P_f(|w|) + (P_B \circ P_f)(|w|)$$

כאשר $P_B \circ P_f$ מסמן את ההרכבה של שני פולינומים. לכן M_A רץ בזמן פולינומיאלי בגודל $|w|$.



שיעור 11

NP שלמות

11.1 המחלקות NPC ו- NPH

הגדרה 11.1 NP-hard

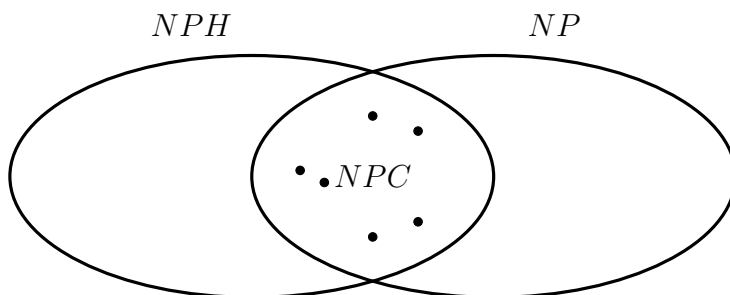
בעייה B נקראת NP קשה אם לכל בעייה $A \in NP$ קיימת רדוקציה $A \leq_p B$.

הגדרה 11.2 NP-complete

בעייה B נקראת NP שלמה אם

$$B \in NP \quad (1)$$

(2) לכל בעייה $A \in NP$ קיימת רדוקציה $A \leq_p B$.



משפט 11.1

אם $B \in NP$ שלמה וגם $B \in P$ אזי $P = NP$.

הוכחה:

• הוכחנו כבר ש- $P \subseteq NP$.

• נוכיח כי $NP \subseteq P$.

לכל בעייה $A \in NP$ קיימת רדוקציה $A \leq_p B$ ומכיון ש- $B \in P$, ממשפט הרדוקציה מתקיים $A \in P$.

■

מסקנה 11.1

אם $A \leq_p B$ אזי $\bar{A} \leq_p \bar{B}$.

משפט 11.2

אם $A \leq_p B$ וגם $B \leq_p C$ אזי $A \leq_p C$.

הוכחה:

משפט 11.3

תהי B בעייה $-NP$ שלמה. אזי לכל בעייה $C \in NP$, אם $B \leq_p C$ אזי גם C היא NP שלמה.

הוכחה: מכיוון ש- B היא $-NP$ שלמה, לכל בעייה $A \in NP$ קיימת רדוקציה $A \leq_p B$. מכיוון ש- $B \leq_p C$ מהטרנזיביות מתקיים $A \leq_p C$ לכל בעייה $A \in NP$. ולכן C היא $-NP$ שלמה.

11.2 בעיית הספיקות

הגדרה 11.3

נוסחת CNF ϕ היא נוסחה בוליאנית מעל n משתנים x_1, x_2, \dots, x_n המכילה m פסוקיות C_1, C_2, \dots, C_m , כאשר כל פסוקית מכילה אוסף של ליטרלים $(x_i \vee \bar{x}_i)$ המחוברים ע"י OR (\vee) בוליאני והפסוקיות מחוברים ע"י AND (\wedge) בוליאני.

לדוגמה

$$\phi = \left(x_1 \vee \bar{x}_2 \vee x_4 \vee \bar{x}_7 \right) \wedge \left(x_3 \vee x_5 \vee \bar{x}_8 \right) \wedge \dots$$

הגדרה 11.4 נוסחת CNF ספיקה

נוסחת CNF ϕ היא ספיקה אפ קימת השמה למשתנים x_1, x_2, \dots, x_n ע"י $T \setminus F$ כך ש- ϕ מקבלת ערך T , כלומר בכל פסוקית ישנו לפחות ליטרל אחד שקיבל ערך T .

11.3 בעיית SAT

הגדרה 11.5 בעיית SAT

קלט: נוסחת CNF ϕ .
פלט: האם ϕ ספיקה?

$$SAT = \{ \langle \phi \rangle \mid \text{נוסחת } CNF \text{ ספיקה} \}$$

משפט 11.4 $SAT \in NP$

$$SAT \in NP.$$

הוכחה: נבנה אלגוריתם אימות V עבור SAT .

$$V = \langle \langle \phi \rangle, y \rangle \text{ על קלט}$$

(1) בודק האם y היא השמה למשתנים x_1, x_2, \dots, x_n .

• אם לא $3CNF \Leftarrow$ דוחה.

(2) בודק האם השמה זו מספקת את ϕ .

• אם כן \Leftarrow מקבל.

• אם לא \Leftarrow דוחה.



11.4 משפט קוק לוין

משפט 11.5 (1973) משפט קוק לוין

הבעיית SAT היא NP - שלמה.

רעיון ההוכחה:

לכל $A \in NP$, $A \leq_p SAT$.
לכל $w \in \Sigma^*$:

$$w \in A \iff f(w) \in SAT,$$

כאן $f(w) = \langle \phi_w \rangle$.

11.2 מסקנה

$$P = NP \iff SAT \in P.$$

11.5 גרסאות של $kSAT$

ישנן לכל היותר k ליטרלים בכל פסוקית:

• $1SAT \in P$.

$$\phi = x_1 \wedge \bar{x}_2 \wedge x_3 \wedge \dots$$

• $2SAT \in P$.

$$\phi = (x_1 \vee \bar{x}_2) \wedge (x_2 \vee x_4) \wedge \dots$$

• $3SAT$ היא NP - שלמה.

11.6 בעיית $3SAT$

הגדרה 11.6 בעיית $3SAT$

קלט: נוסחת $3CNF$, ϕ .
פלט: האם ϕ ספיקה?

$$3SAT = \{ \langle \phi \rangle \mid \phi \text{ נוסחת } 3CNF \text{ ספיקה} \}$$

משפט 11.6 $3SAT$ היא NP שלמה. $3SAT$ היא NP שלמה.

הוכחה:

יש לקיים את השני תנאים הבאים:

$$3SAT \in NP \quad (1)$$

ניתן לבנות אלגוריתם אימות עבור $3SAT \in NP$ דומה לאלגוריתם האימות עבור SAT שבנינו בהוכחה של המשפט קוק-לויין 11.5 למעלה.

(2) $3SAT$ היא NP קשה ע"י רדוקציה

$$SAT \leq_p 3SAT.$$

ואז בגלל ש- SAT היא NP שלמה (לפי משפט קוק-לויין 11.5) ומכיוון ש- $3SAT \in NP$ אז לפי משפט האסימפטוטית 11.2 גם $3SAT$ היא NP -שלמה.

קיום פונקציה הרדוקציה $SAT \leq_p 3SAT$ כעת נבנה את פונקציה הרדוקציה מ- SAT ל- $3SAT$.ראשית נציין כי כל נוסחה בוליאנית ϕ ניתנת לרשום בצורה CNF בזמן פולינומיאלי.

בהינתן נוסחת CNF , ϕ (הקלט של SAT) נבנה בזמן פולינומיאלי נוסחת $3CNF$, ϕ' (הקלט של $3SAT$) ואז נוכיח שמתקיים

$$\langle \phi' \rangle \in 3SAT \iff \langle \phi \rangle \in SAT.$$

לכל פסוקית C ב- ϕ המכילה יותר מ-3 ליטרלים, ניצור אוסף C' ב- ϕ' של פסוקיות כך שכל פסוקית ב- C' תכיל 3 ליטרלים. למשל בהינתן הפסוקית C הבאה של ϕ :

$$C = x_1 \vee x_2 \vee x_3 \vee \bar{x}_4 \vee \bar{x}_5$$

ניצור את הפסוקית C' הבאה ב- ϕ' :

$$C' = (x_1 \vee x_2 \vee y_1) \wedge (\bar{y}_1 \vee x_3 \vee y_2) \wedge (\bar{y}_2 \vee \bar{x}_4 \vee \bar{x}_5).$$

באופן כללי, לכל פסוקית $C = a_1 \vee a_2 \vee \dots \vee a_k$ המכיל $k > 3$ ליטרלים, ניצור אוסף C' של פסוקיות שבו כל פסוקית מכילה 3 ליטרלים, ע"י הוספת $k-3$ משתנים y_1, y_2, \dots, y_{k-3} :

$$C' = (a_1 \vee a_2 \vee y_1) \wedge (\bar{y}_1 \vee a_3 \vee y_2) \wedge \dots \wedge (\bar{y}_{i-2} \vee a_i \vee y_{i-1}) \wedge \dots \wedge (\bar{y}_{k-3} \vee a_{k-1} \vee a_k).$$

בפרט, עבור כל פסוקית $C = (a_1 \vee a_2 \vee \dots \vee a_k)$ נניח $a_i = 1$ הוא הליטרל הראשון ששווה ל-1. אז

• נשים $y_j = 1$ לכל $1 \leq j \leq i-2$,• ונשים $y_j = 0$ לכל $i-1 \leq j \leq k-3$.

סיימנו להגדיר את הפונקציה הרדוקציה.

כעת נוכיח כי הפונקציה הזאת מקיימת את התנאי ההכרחי

$$\langle \phi' \rangle \in 3SAT \iff \langle \phi \rangle \in SAT.$$

כיוון \Leftarrow :נניח כי $\langle \phi \rangle \in SAT$ ותהי X השמה המספקת את ϕ .נוכיח שקיימת השמה X' מתאימה המספקת את ϕ' .

- בכל פסוקית C של ϕ , עבור הליטרלים a_1, a_2, \dots, a_k ניתן אותם ערכים כמו ב- X .
- מכיוון ש- X מספקת את ϕ , בכל פסוקית $C = (a_1 \vee a_2 \vee \dots \vee a_k)$ יש לפחות ליטרל אחד שקיבל ערך 1. נניח $a_i = 1$. אז על פי ההגדרה של פונקציה הרדוקציה:

$$\begin{aligned} & * \text{ נשים } y_j = 1 \text{ לכל } 1 \leq j \leq i - 2, \\ & * \text{ ונשים } y_j = 0 \text{ לכל } i - 1 \leq j \leq k - 3. \end{aligned}$$

באופן הזה אנחנו ניצור אוסף C' של פסוקיות עם המבנה הבא:

$$\begin{aligned} & \left(a_1 \vee a_2 \vee \overset{1}{y_1} \right) \wedge \left(\overset{0}{\bar{y}_1} \vee a_2 \vee \overset{1}{y_2} \right) \wedge \dots \wedge \left(\overset{0}{\bar{y}_{i-3}} \vee a_{i-1} \vee \overset{1}{y_{i-2}} \right) \wedge \left(\overset{0}{\bar{y}_{i-2}} \vee \overset{1}{a_i} \vee \overset{0}{y_{i-1}} \right) \wedge \left(\overset{1}{\bar{y}_{i-1}} \vee a_{i+1} \vee \overset{0}{y_i} \right) \\ & \wedge \dots \wedge \left(\overset{1}{\bar{y}_{k-3}} \vee a_{k-1} \vee a_k \right) \end{aligned}$$

ולכן השמה זו מספקת את C' ולכן $\langle \phi' \rangle \in 3SAT$.

כיוון \Rightarrow :

נניח כי $\langle \phi' \rangle \in 3SAT$ ותהי X' השמה המספקת את ϕ' .
נוכיח שקיימת השמה X המספקת את ϕ .

נסתכל על פסוקית $C = (a_1 \vee a_2 \vee \dots \vee a_k)$.
נניח בשלילה שלא קיימת השמה X המספקת את C . אז בהכרח

$$a_1 = a_2 = \dots = a_k = 0$$

לפי זה, באוסף פסוקיות C' שנקבל על פי ההגדרה של פונקציה הרדוקציה, לכל $1 \leq j \leq k - 3$.
כלומר מתקיים $y_1 = y_2 = \dots = y_{k-3} = 0$. לכן

$$C' = \left(\overset{0}{a_1} \vee \overset{0}{a_2} \vee \overset{1}{y_1} \right) \wedge \left(\overset{0}{\bar{y}_1} \vee \overset{0}{a_3} \vee \overset{1}{y_2} \right) \wedge \dots \wedge \left(\overset{0}{\bar{y}_{i-2}} \vee \overset{0}{a_i} \vee \overset{1}{y_{i-1}} \right) \wedge \dots \wedge \left(\overset{0}{\bar{y}_{k-3}} \vee \overset{0}{a_{k-1}} \vee \overset{0}{a_k} \right)$$

הפסוקית האחרונה $\left(\overset{0}{\bar{y}_{k-3}} \vee \overset{0}{a_{k-1}} \vee \overset{0}{a_k} \right)$ אינה מסופקת.
לכן C' אינה מסופקת, בסתירה לכך ש- X' מספקת את C' .

ולכן $\langle \phi \rangle \in SAT$.

הוכחנו שקיימת הרדוקציה $SAT \leq 3SAT$.

כעת נוכיח כי הרדוקציה הזו היא זמן פולינומיאלית.

סיבוכיות

החישוב של הפונקציה מתבצע בזמן פולינומיאלי. ספציפי, אם האורך של הנוסחה ϕ הוא $n = |\phi|$ אז הרדוקציה היא $O(n)$.



11.7 הוכחת משפט קוק לוין*

משפט 11.7 משפט קוק לוינ

הבעיית SAT היא NP - שלמה.

הוכחה:

חשיפה מלאה: ההוכחה הבאה מתבססת על ההוכחה שנתונה בהספר של Sipser.

על פי הגדרה 11.2 יש להוכיח ששני התנאים הבאים מתקיימים:

תנאי 1: $SAT \in NP$.תנאי 2: $A \leq_p SAT$ לכל $A \in NP$.ראשית נוכיח כי $SAT \in NP$:

כדי להוכיח כי SAT שייכת ל-NP, נוכיח כי אישור המורכב מהשמה מספקת עבור נוסחת קלט ϕ ניתן לאימות בזמן פולינומיאלי.

נניח כי $|\phi| = n$. כלומר ב- ϕ מופיעים n ליטרלים. ז"א השמה כלשהי דורשת n משתני בוליאניים לכל היותר.

- אלגוריתם האימות מחליף כל משתנה בנוסחה בערך המתאים לו על פי ההשמה. השלב הזה הוא $O(n)$.

- אחר כך האלגוריתם מחשב את ערכו של הביטוי:

* נניח כי הנוסחה ϕ מכילה k דורות של סוגריים בתוך סוגריים.

* החישוב מתחיל עם החישובים של הביטויים בתוך הסוגריים הכי בפנים.

* יש n סוגריים הכי-בפנים לכל היותר, וכל אחד של הסוגריים האלה מכיל n ליטרלים לכל היותר. לכן החישוב הזה הוא $O(n^2)$.

* יש k דורות של סוגריים לכן החישוב כולו הוא $O(kn^2)$

- בסה"כ הסיבוכיות זמן הריצה היא

$$O(n) + O(kn^2) = O(n^2)$$

לפיכך אישור של השמה כלשהי מתבצע בזמן פולינומיאלי.

- אם ערכו של הביטוי הוא 1 הנוסחה ספיקה.

הוכחנו כי $SAT \in NP$. עכשיו נוכיח כי $A \leq_p SAT$.

תהי N מ"ט אי-דטרמיניסטית זמן-פולינומיאלית שמכריעה שפה A כלשהי בזמן $O(n^k)$ עבור k טבעי. התרשים למטה מראה טבלה של קונפיגורציות של N . ברשימה הבאה רשומות ההגדרות של הטבלה:

- כל שורה מראה את תוכן הסרט בשלב מסוים של מסלול אחד של N .
- בשורה הראשונה יש את הקונפיגורציה ההתחלתית.
- אנחנו מניחים כי האורך של המילה, כלומר אורך הקלט הוא n .
- הסימנים w_1, \dots, w_n מסמנים את התווים של הקלט.

- בתא הראשון בכל שורה יש #, ואחר כך רשומה הקונפיגורציה של N .
בסוף הקונפיגורציה בכל שורה יש #.
- אחרי ה- # בקצה הימין של המילה, בכל תא יש תו רווח עד הסוף של השורה.
התווי רווח לפני המשבצת הראשונה של הקונפיגורציה לא מופיעים בטבלה.
- האורך של כל שורה הוא בדיוק n^k תאים.
- בטבלה יש בדיוק n^k שורות לסיבה הבאה:
 - המכונת טיורינג מבצעת n^k צעדים לכל היותר.
 - בכל צעד המ"ט עוברת לקונפיגורציה חדשה.
 - בכל שורה רשומה קונפיגורציה.
 - בסה"כ יש n^k שורות עבור ה- n^k קונפיגורציות שונות האפשריות.

#	q_0	w_1	w_2	\dots	w_n	\sqcup	\dots	\sqcup	#						
#	q_0								#						
#	q_0								#						
				<table><tr><td></td><td></td><td></td></tr><tr><td></td><td></td><td></td></tr></table>											
#									#						

אנחנו אומרים כי טבלה כלשהי היא **טבלה המקבלת** אם באחת השורות יש קונפיגורציה אשר N מקבלת אותה.

בעזרת הטבלה נתאר את הרדוקציה זמן-פולינומיאלית f משפה A כלשהי ל- SAT .

הפונקציה הרדוקציה f מקבלת קלט w ומחזירה נוסחה $\phi = f(w)$, אשר לפי ההגדרה של פונקצית הרדוקציה, עומדת בתנאי הבא:

$$w \in A \quad \Leftrightarrow \quad f(w) \in SAT \text{ .}$$

יהיו Q קבוצת המצבים ו- Γ האלפיבית של הסרט של N . נגדיר:

$$C = Q \cup \Gamma \cup \{\#\} \text{ .}$$

נסמן ב- s איבר כלשהו של C .

עבור כל תא (i, j) של הטבלת הקונפיגורציות נגדיר משתנה בוליאני $x_{i,j,s}$ לכל $1 \leq i, j \leq n^k$.
המשתנה $x_{i,j,s}$ מוגדר על פי התנאי

$$x_{ijs} = 1$$

אם בתא ה- ij של הטבלה יש $s \in C$. למשל, אם בתא ה- $(2, 5)$ של הטבלה מופיע התו a אז

$$x_{2,5,a} = 1$$

בעוד

$$x_{2,5,b} = 0 .$$

במובן הזה, התכנים של כל התאים של הטבלה מסומנים על ידי המשתנים של ϕ .

עכשיו נבנה נוסחה ϕ על סמך התנאי שהשמה מספקת של ϕ תהיה מתאימה לטבלה המקבלת של N .
נגדיר:

$$\phi = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{move}} \wedge \phi_{\text{acc}} . \quad (11.1)$$

נסביר את כל הנוסחאות ϕ_{cell} , ϕ_{start} , ϕ_{move} ו- ϕ_{acc} אחד אחד למטה.

• הנוסחה ϕ_{cell}

כפי שמצויין לעיל, אם המשתנה $x_{i,j}$ "דולק", כלומר אם $x_{i,j,s} = 1$, זאת אומרת שיש סימן s בתא ה- ij של הטבלה. אנחנו רוצים להבטיח שהשמה כלשהי בנוסחה אשר מתאימה לקונפיגורציה של הטבלה, מדליקה בדיוק משתנה אחד לכל תא של הטבלה. למטרה זו נגדיר ϕ_{cell} כך:

$$\phi_{\text{cell}} = \bigwedge_{1 \leq i, j \leq n^k} \left[\left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s, t \in C \\ s \neq t}} (\bar{x}_{i,j,s} \vee \bar{x}_{i,j,t}) \right) \right] \quad (11.2)$$

* האיבר הראשון בסוגריים מרובעים, $\bigvee_{s \in C} x_{i,j,s}$ מבטיח שלכל תא של הטבלה, לפחות משתנה אחד דולק.

* האיבר השני $\bigwedge_{\substack{s, t \in C \\ s \neq t}} (\bar{x}_{i,j,s} \vee \bar{x}_{i,j,t})$ מבטיח שעבור כל תא של הטבלה, משתנה אחד לכל היותר דולק.

לפיכך כל השמה מספקת עומדת בתנאי שיהיה בדיוק סימן אחד, s , בכל תא של הטבלה.

• הנוסחה ϕ_{start}

נוסחה ϕ_{start} מבטיחה ששורה הראשונה של הטבלה היא הקונפיגורציה ההתחלתית של N על הקלט w :

$$\begin{aligned} \phi_{\text{start}} = & x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \\ & \wedge \dots \wedge \\ & x_{1,n+2,w_n} \wedge x_{1,n+3,\sqcup} \wedge \dots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#} \end{aligned} \quad (11.3)$$

• הנוסחה ϕ_{acc}

הנוסחה ϕ_{acc} מבטיחה שקיימת טבלה קונפיגורציה אשר המ"ט N מקבלת אותה.

בפרט ϕ_{acc} מבטיחה שהסימן q_{acc} מופיע בתא אחד של הטבלה דרך התנאי שלפחות אחד המשתנים $x_{i,j,q_{\text{acc}}}$ דולק:

$$\phi_{\text{acc}} = \bigvee_{1 \leq i, j \leq n^k} x_{i,j,q_{\text{acc}}} \quad (11.4)$$

• הנוסחה ϕ_{move}

הנוסחה ϕ_{move} מבטיחה שכל שורה של הטבלה היא "שורה חוקית".

כלומר בכל שורה, הקונפיגורציה היא כך שאפשר להגיע אליה על ידי תזוזה חוקית של N מהקונפיגורציה הקודמת שמופיעה בשורה אחת למעלה.

תזוזה חוקית בין כל שתי קונפיגורציות נקבעת על ידי הפונקציה המעברים של המ"ט N .

בשפה פורמלית, אם c_i הקונפיגורציה של שורה i , ו- c_{i+1} הקונפיגורציה של השורה $i + 1$ אחת למטה, אז ϕ_{move} מבטיחה כי לכל $1 \leq i \leq n^k - 1$ מתקיים

$$c_i \vdash_N c_{i+1}.$$

במונחי הטבלה, אפשר להגדיר תזוזה חוקית בין כל שתי שורות על ידי תת-טבלה מסדר 2×3 שמכילה 3 תאים מתאימים של שתי שורות שכנות.

מכאן ואילך אנחנו נקרא לתת-טבלה כזאת "חלון".

למטה יש דוגמאות של חלונות חוקיים:

a	q ₁	b
q ₂	a	c

a	q ₁	b
a	a	q ₂

a	a	q ₁
a	a	b

#	b	a
#	b	a

a	b	a
a	b	q ₂

b	b	b
c	b	b

החלונות האלה למטה הם דוגמאות לחלונות לא חוקיים:

a	b	a
a	a	a

a	q ₁	b
q ₁	a	a

b	q ₁	b
q ₂	b	q ₂

הנוסחה ϕ_{move} קובעת שכל חלון של הטבלה חוקי. בפרט, כל חלון מכיל 6 תאים. לכן ϕ_{move} קובעת שהתכנים של ה-6 תאים של כל חלון מהווה חלון חוקי. ז"א

$$\phi_{\text{move}} = \bigwedge_{\substack{1 \leq i \leq n^k \\ 1 \leq j \leq n^k}} (\text{חלון ה- } i, j \text{ חוקי}) \quad (11.5)$$

אנחנו מציבים בטקסט "חלון ה- i, j חוקי" את הנוסחה הבאה, כאשר a_1, \dots, a_6 מסמנים את התכנים של ה-6 תאים של כל חלון:

$$\bigvee_{\substack{\{a_1, a_2, a_3, a_4, a_5, a_6\} \\ \text{חלון חוקי}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6}) \quad (11.6)$$

עד כה הוכחנו שקיים רדוקציה מכל שפה $A \in NP$ ל- SAT . כעת נוכיח כי הרדוקציה זו חישובית בזמן-פולינומיאלי.

הטבלה של N היא מסדר $n^k \times n^k$ ולכן היא מכילה n^{2k} תאים.

נחשב את הסיבוכיות של כל הנוסחאות $\phi_{\text{move}}, \phi_{\text{acc}}, \phi_{\text{start}}, \phi_{\text{cell}}$.

• הנוסחה ϕ_{cell}

הנוסחה (11.2) של ϕ_{cell} מכילה n^{2k} נוסחאות עם 3 ליטרלים. לכן

$$\phi_{\text{cell}} = O(n^{2k}) .$$

• הנוסחה ϕ_{start}

הנוסחה (11.3) של ϕ_{start} מכילה בדיוק n^k ליטרלים. לכן

$$\phi_{\text{start}} = O(n^k) .$$

• הנוסחה ϕ_{acc}

הנוסחה (11.4) של ϕ_{acc} מכילה בדיוק n^k ליטרלים. לכן

$$\phi_{\text{acc}} = O(n^k) .$$

• הנוסחה ϕ_{move}

הנוסחה (11.5,11.6) של ϕ_{move} מכילה n^{2k} נוסחאות עם 6 ליטרלים. לכן

$$\phi_{\text{move}} = O(n^{2k}) .$$

לכן בסה"כ

$$\phi = O(n^{2k}) + O(n^k) + O(n^k) + O(n^{2k}) = O(n^{2k}) .$$

לפיכך קיימת רדוקציה חישובית בזמן פולינומיאלי מכל שפה $A \in NP$ ל- SAT .



שיעור 12

רדוקציות פולינומיליות

12.1 $CLIQUE$ היא NP - שלמה

משפט 12.1 $CLIQUE \in NPC$

הבעיית $CLIQUE$ היא (ראו הגדרה 10.8):

$$CLIQUE = \{ \langle G, k \rangle \mid k \text{ מכיל קליקה בגודל } k \}.$$

$CLIQUE$ היא NP - שלמה

הוכחה:

(1) הוכחנו כי $CLIQUE \in NP$ במשפט 10.3.

(2) נוכיח כי $CLIQUE$ היא NP קשה ע"י רדוקציה $3SAT \leq_P CLIQUE$.

פונקציית הרדוקציה

בהינתן נוסחת $3CNF$ ϕ מעל n משתנים x_1, x_2, \dots, x_n המכיל m פסוקיות, ניצור זוג $\langle G, k \rangle$ ונוכיח כי

$$\langle \phi \rangle \in 3SAT \iff \langle G, k \rangle \in CLIQUE.$$

נבנה את הגרף G באופן הבא:

הקדקודים של G :

לכל פסוקית C_i ב- ϕ המכילה 3 ליטרלים ניצור שלשה t_i המכילה 3 קודקודים המתאימים לליטרלים של C_i :

$$x_1 \vee \bar{x}_3 \vee x_4 \longrightarrow \begin{array}{ccc} \textcircled{x_1} & \textcircled{\bar{x}_3} & \textcircled{x_4} \end{array}$$

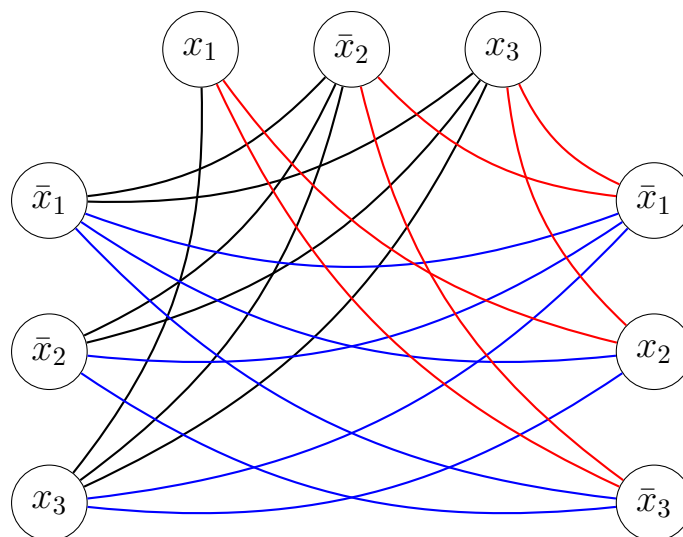
הצלעות של G :

נבחר בין כל שני קודקודים פרט לזוגות הבאים:

- זוג קודקודים המתאימים למשתנה ומשלמים שלו.
- זוג קודקודים שנמצאים באותה שלושה.

לדוגמה:

$$\phi = \underbrace{\left(\overset{T}{x_1} \vee \overset{T}{\bar{x}_2} \vee x_3 \right)}_{C_1} \wedge \underbrace{\left(\bar{x}_1 \vee \overset{T}{\bar{x}_2} \vee x_3 \right)}_{C_2} \wedge \underbrace{\left(\bar{x}_1 \vee x_2 \vee \overset{T}{\bar{x}_3} \right)}_{C_3}$$



נקבע $k = m$.

נכונות הרדוקציה

(1) ניתן לבנות את G בזמן פולינומיאלי בגודל ϕ .

(2) נוכיח כי

$$\langle \phi \rangle \in 3SAT \iff \langle G, k \rangle \in CLIQUE.$$

כיוון \Leftarrow

- נניח כי ϕ ספיקה ונסתכל על השמה המספקת את ϕ .
- בכל פסוקית C_i ב- ϕ יש לפחות ליטרל אחד שקיבל ערך T .
- נבחר מכל שלשה t_i בקודקוד אחד המתאים לליטרל שקיבל ערך T ב- C_i ונוסיף אותו לקליקה.
- מספר הקודקודים שבחרנו שווה k וכל שניים מהם מחוברים בצלע כי לא בחרנו שני קודקודים מאותה שלשה ולא בחרנו שני קודקודים המתאימים למשתנה ומשלים שלו.
- ולכן G מכיל קליקה בגודל k .

כיוון \Rightarrow

- נניח כי G מכיל קליקה בגודל k ונסתכל על קליקה כזו.
- לפי הבנייה הקליקה מכילה בדיוק קודקוד אחד מכל שלשה t_i . ניתן השמה למשתנים של ϕ כך שהליטרל שמתאים לקודקוד שנמצא בקליקה יקבל ערך T .
- השמה זו אפשרית מכיוון שהקליקה לא מכילה שני קודקודים המתאימים למשתנה ומשלים שלו.

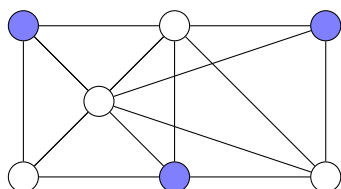
- בנוסף השם ϕ מספקת את ϕ מכיוון שהקליקה מכילה קודקוד מכל שלשה t_i ולכן הליטרל המתאים לקודקוד בשלשה t_i קיבל ערך T ולכן הוא מספק את הפסוקית C_i .
- לכן ϕ ספיקה.



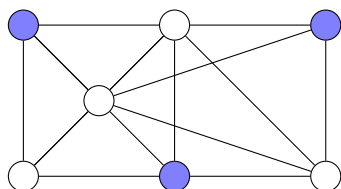
12.2 בעיית הקבוצה הבלתי תלויה

הגדרה 12.1 קבוצה בלתי תלויה

בהינתן גרף לא מכוון $G = (V, E)$, קבוצה בלתי תלויה ב- G היא תת-קבוצה של קודקודים $S \subseteq V$ כך שלכל שני קודקודים $u, v \in S$ מתקיים $(u, v) \notin E$.



קבוצה בלתי תלויה בגודל $k = 3$:



קבוצה בלתי תלויה בגודל $k = 3$:

הגדרה 12.2 בעיית IS

קלט: גרף לא מכוון $G = (V, E)$ ומספר k .
פלט: האם קיימת קבוצה בלתי תלויה ב- G בגודל k ?

$$IS = \{ \langle G, k \rangle \mid k \text{ קבוצה בלתי תלויה בגודל } k \}$$

משפט 12.2 $IS \in NPC$

הבעיה IS היא NP - שלמה.

הוכחה:

(1) נוכיח כי $IS \in NP$

נבנה אלגוריתם אימות V עבור IS .
 $V = \{ \langle G, k \rangle, y \}$ על קלט

- בודק האם y היא קבוצה של k קודקודים מ- G השונים זה מזה.

◦ אם לא \Rightarrow דוחה.

- בודק האם כל שני קודקודים מ- y לא מחוברים בצלע ב- G .

◦ אם כן \Rightarrow מקבל.

○ אם לא \Rightarrow דוחה.

(2) נוכיח כי $CLIQUE \leq_P IS$

פונקציית הרדוקציה:

בהינתן זוג $\langle G, k \rangle$ הקלט של $CLIQUE$, ניצור זוג $\langle G', k' \rangle$, הקלט של IS , ונוכיח כי:

$$\langle G, k \rangle \in CLIQUE \Leftrightarrow \langle G', k' \rangle \in IS.$$

הפונקציית הרדוקציה מוגדרת כך שהתנאים הבאים מתקיימים:

(1) נניח שהגרף הוא $G = (V, E)$.

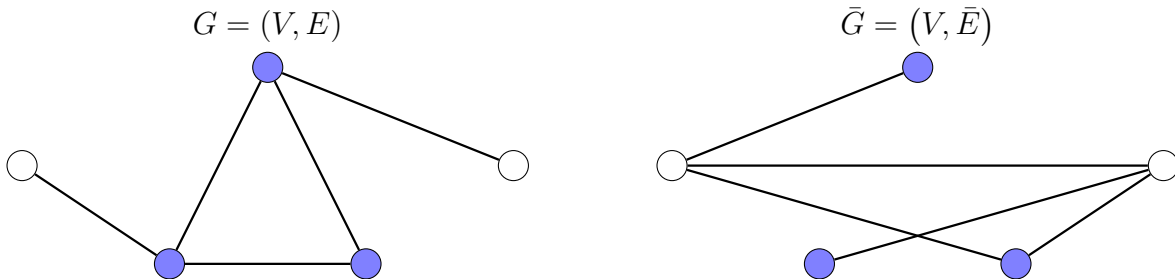
אז הגרף G' הוא הגרף המשלים של $G = (V, E)$.

ז"א $G' = \bar{G} = (V, \bar{E})$ כאשר

$$\bar{E} = \{(u_1, u_2) \mid (u_1, u_2) \notin E\}.$$

(2) $k' = k$.

לדוגמה, בהינתן הגרף $G = (V, E)$ שמכיל קליקה בגודל $k = 3$, הפונקציית הרדוקציה R מחזירה את הגרף $\bar{G} = (V, \bar{E})$ ואת המספר $k' = k = 3$, כמתואר בתרשים למטה:



נכונות הרדוקציה

(1) ניתן לבנות G' בזמן פולינומיאלי בגודל G .

(2) נוכיח כי $\langle G, k \rangle \in CLIQUE \Leftrightarrow \langle G', k' \rangle \in IS$.

כיוון \Leftarrow

בהינתן גרף $G = (V, E)$ ושלם k .

נניח כי $\langle G, k \rangle \in CLIQUE$.

\Leftarrow G מכיל קליקה S בגודל k .

\Leftarrow אם $u_1, u_2 \in S$ (אם u_1, u_2 שני קודקודים בקליקה S) אזי $(u_1, u_2) \in E$.

כלומר, כל שני קודקודים ב- S מחוברים בצלע של G .

\Leftarrow אם $u_1, u_2 \in S$ אזי $(u_1, u_2) \notin \bar{E}$.

כלומר, כל שני קודקודים ב- S לא מחוברים בצלע של המשלים של הגרף \bar{G} , דהיינו G' .

\Leftarrow אותה הקבוצה S היא קבוצה בלתי תלוייה ב- G' בגודל $k' = k$.

$\Leftarrow G' = \bar{G}(V, \bar{E})$ מכיל קבוצה בלתי תלוייה בגודל k' .

$\Leftarrow \langle G', k \rangle \in IS$

\Rightarrow כיוון

בהינתן גרף G' ושלים k' .

נניח כי $\langle G', k' \rangle \in IS$.

$\Leftarrow G'$ מכיל קבוצה בלתי תלוייה S בגודל k' .

\Leftarrow אם $u_1, u_2 \in S$ אזי $(u_1, u_2) \notin \bar{E}$.

כלומר, כל שני קדקודים ב- S לא מחוברים בצלע של G' .

\Leftarrow אם $u_1, u_2 \in S$ אזי $(u_1, u_2) \in E$.

כלומר, כל שני קדקודים ב- S מחוברים בצלע של $G(V, E)$.

\Leftarrow אותה הקבוצה S היא קליקה ב- G בגודל $k = k'$.

$\Leftarrow G$ מכיל קליקה בגודל k .

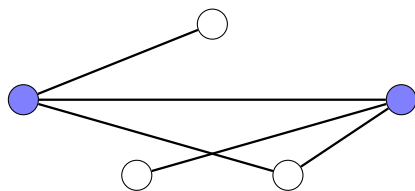
$\Leftarrow \langle G, k \rangle \in CLIQUE$

■

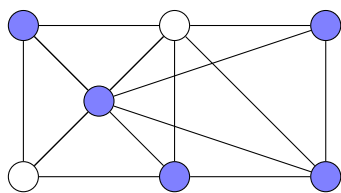
12.3 בעיית הכיסוי בקודקודים

הגדרה 12.3 כיסוי בקודקודים

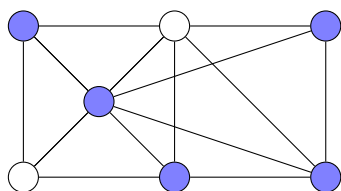
בהינתן גרף לא מכוון $G = (V, E)$, כיסוי בקודקודים ב- G הוא תת-קבוצה של קודקודים $C \subseteq V$ כך שלכל צלע $u, v \in S$ מתקיים $u \in C$ או $v \in C$.



כיסוי בקדקודים בגודל $k = 2$:



כיסוי בקדקודים בגודל $k = 5$:



כיסוי בקדקודים בגודל $k = 5$:

12.4 הבעיה VC

הגדרה 12.4 בעיית VC

קלט: גרף לא מכוון $G = (V, E)$ ומספר k .
פלט: האם קיים כיסוי בקודקודים ב- G בגודל k ?

$$VC = \{ \langle G, k \rangle \mid G \text{ גרף לא מכוון המכיל כיסוי בקודקודים בגודל } k \}$$

משפט 12.3 $VC \in NPC$

הבעיה VC היא NP - שלמה.

הוכחה:

נוכיח כי $VC \in NP$

נבנה אלגוריתם אימות V עבור VC .
 $V = \text{על קלט } \langle G, k \rangle, y$:

• בודק האם כל צלע ב- G מכילה לפחות קצה אחד ב- y .

◦ אם כן \Rightarrow מקבל.

◦ אם לא \Rightarrow דוחה.

נוכיח כי VC היא NP -קשה ע"י רדוקציה $IS \leq_P VC$

פונקציית הרדוקציה:

בהינתן זוג $\langle G, k \rangle$ הקלט של IS , ניצור זוג $\langle G', k' \rangle$ הקלט של VC ונוכיח כי

$$\langle G, k \rangle \in IS \iff \langle G', k' \rangle \in VC.$$

הפונקציית הרדוקציה מוגדרת כך שהתנאים הבאים מתקיימים:

(1) נניח שהגרף הוא $G = (V, E)$.

אז הגרף G' הוא אותו גרף $G = (V, E)$.

(2) $k' = |V| - k$.

נכונות הרדוקציה

(1) ניתן לבנות G' בזמן פולינומיאלי בגודל G .

(2) נוכיח כי $\langle G, k \rangle \in IS \iff \langle G', k' \rangle \in VC$.

כיוון \Leftarrow

בהינתן גרף $G = (V, E)$ ושלם k .

נניח כי $\langle G, k \rangle \in IS$

$\Leftarrow G$ מכיל קבוצה בלתי תלוייה S בגודל k .
 \Leftarrow אם $u_1 \in S$ וגם $u_2 \in S$ אז $(u_1, u_2) \notin E$.
 כלומר, כל שני קדקודים ב- S **לא מחוברים** בצלע ב- G .
 \Leftarrow השלילה הלוגית של הגרירה הזאת היא:
 אם $(u_1, u_2) \in E$ אז $u_1 \notin S$ או $u_2 \notin S$.
 \Leftarrow אם $(u_1, u_2) \in E$ אז $u_1 \in V \setminus S$ או $u_2 \in V \setminus S$.
 $\Leftarrow V \setminus S$ היא כיסוי קדקודים ב- G בגודל $k' = |V| - k$.
 \Leftarrow הגרף $G' = G$ מכיל כיסוי קדקודים בגודל k' .
 $\Leftarrow \langle G', k' \rangle \in VC$

\Rightarrow כיוון

בהינתן גרף G' ושלם k' .
 נניח כי $\langle G', k' \rangle \in VC$.
 $\Leftarrow G'$ מכיל כיסוי בקדקודים C בגודל k' .
 \Leftarrow אם $(u_1, u_2) \in E$ אז $u_1 \in C$ או $u_2 \in C$.
 \Leftarrow השלילה הלוגית של הגרירה הזאת היא:
 אם $u_1 \notin C$ וגם $u_2 \notin C$ אז $(u_1, u_2) \notin E$.
 \Leftarrow אם $u_1 \in V \setminus C$ וגם $u_2 \in V \setminus C$ אז $(u_1, u_2) \notin E$.
 \Leftarrow כל שני קדקודים ב- $V \setminus C$ לא מחוברים בצלע ב- G' .
 $\Leftarrow V \setminus C$ הוא קבוצת בלתי תלוייה ב- G' בגודל $k = |V| - k'$.
 \Leftarrow הגרף $G = G'$ מכיל קבוצה בלתי תלוייה בגודל k .



PARTITION 12.5

הגדרה 12.5 בעיית PARTITION

קלט: קבוצת מספרים שלמים $S = \{x_1, x_2, \dots, x_n\}$
 פלט: האם קיימת תת-קבוצה $Y \subseteq S$ כך ש- $\sum_{y \in Y} y = \sum_{y \in S \setminus Y} y$?

$$PARTITION = \left\{ S \mid \sum_{y \in Y} y = \sum_{y \in S \setminus Y} y \text{ ש- } Y \subseteq S \text{ קבוצת שלמים, וקיימת תת-קבוצה } Y \subseteq S \right\}$$

12.6 רדוקציות פולינומיאליות

משפט 12.4 רדוקציות פולינומאליות

$$\begin{aligned}
 SAT &\leq_P 3SAT \\
 3SAT &\leq_P CLIQUE \\
 CLIQUE &\leq_P IS \\
 IS &\leq_P VC \\
 SubSetSum &\leq_P PARTITION \\
 HAMPATH &\leq_P HAMCYCLE
 \end{aligned}$$

12.7 שפות NP שלמותמשפט 12.5 שפות NP - שלמות

SAT	-NP שלמה.	(משפט קוק לויין)
$3SAT$	-NP שלמה.	
$HAMPATH$	-NP שלמה.	
$CLIQUE$	-NP שלמה.	
IS	-NP שלמה.	
VC	-NP שלמה.	

שיעור 13

סיבוכיות מקום ושלמות ב PSPACE

13.1 הגדרה של סיבוכיות מקום

הגדרה 13.1 סיבוכיות מקום של מכונת טיורינג

הסיבוכיות מקום של מ"ט M על קלט w היא פונקציה $f(|w|)$ השווה למספר התאי סרט לכל היותר של המכונה M שבהם נעשה שימוש בחישוב של M על w .

הגדרה 13.2 המחלקה $SPACE(f(n))$

מחלקת $SPACE(f(n))$ היא אוסף כל השפות L עבורן קיימת מכונת טיורינג דטרמיניסטית M שמכריעה אותה כך ש:

על כל קלט w באורך $n = |w|$, המכונה M משתמשת לכל היותר $O(f(n))$ תאי סרט.

. $\{ \exists \text{ מ"ט } M \text{ שמכריעה } L \text{ ומשתמשת לכל היותר ב- } O(f(n)) \text{ תאי סרט.} \mid L \in SPACE(f(n)) \}$

דוגמה 13.1

נראה כי ניתן לפתור את הבעיה SAT ע"י אלגוריתם שהוא רץ במקום ליניארי. כלומר:

$$SAT \in SPACE(n).$$

תהי ϕ נוסחה בוליאנית כלשהי. נסמן $n = |\phi|$ ונסמן ב- m את מספר המשתנים ב- ϕ . נגדיר מכונה M שפועלת כך:

$$M = \text{על כל קלט } \langle \phi \rangle$$

(1) M רושמת את המחרוזת $\langle \phi \rangle$ על סרט הקלט.

(2) לכל השמה a_1, a_2, \dots, a_m (כאשר $a_i \in \{0, 1\}$ הוא הערך הנוכחי של x_i):

(א) M רושמת את מחרוזת של ההשמה $a_1 a_2 \dots a_m$ על סרט העבודה.

(ב) M מחשבת את הערך של ϕ עבור ההשמה הנוכחית a_1, \dots, a_m ע"י סריקה של הקלט $\langle \phi \rangle$ שרשום על סרט הקלט.

(ג) אם מתקבל $\phi(a_1, \dots, a_m) = 1$ אז M מקבלת.

(3) אם עבור כל ההשמות התקבל $\phi(a_1, \dots, a_m) = 0$ אז M דוחה.

מכאן אנחנו רואים כי המכונה M_1 רצה במקום ליניארי. בפרט:

• M שומרת על סרט העבודה את ההשמה $a_1 \dots a_m$ וזה נדרש $O(m)$ תאים.

• המספר המשתנים, m הוא n לכל היותר.

• לכן M רצה במקום $O(n)$.

לפיכך:

$$SAT \in SPACE(n).$$

הגדרה 13.3 המחלקה $NSPACE(f(n))$

מחלקת $NSPACE(f(n))$ היא אוסף כל השפות L עבורן קיימת מכונת טיורינג אי-דטרמיניסטית N שמכריעה אותה כך ש:
על כל קלט w באורך $n = |w|$ המכונה N משתמשת לכל היותר $O(f(n))$ תאי סרט מתוך כל המסלולי חישוב של N .

$$NSPACE(f(n)) = \{L \mid \exists \text{ מכריעה } N \text{ ומשתמשת לכל היותר ב- } O(f(n)) \text{ תאי סרט.}\}$$

דוגמה 13.2

תהי ALL_{NFA} השפה הבאה:

$$ALL_{NFA} = \{\langle A \rangle \mid L(A) = \Sigma^* \text{ עבור } NFA \text{ } A\}.$$

הוכיחו כי $ALL_{NFA} \in NSPACE(n)$.

פתרון:

הפתרון מתבוסס על זה שזה פשוט יותר לבנות אלגוריתם המכריע את השפה המשלימה:

$$\overline{ALL_{NFA}} = \{\langle A \rangle \mid w \in \Sigma^* \text{ עבור } A \text{ דוחה } w\}.$$

נשתמש במשפט מרכזי כדי לבנות אלגוריתם שמכריע את $\overline{ALL_{NFA}}$:

משפט 13.1

אם $M = (Q, \Sigma, \delta, q_0, F)$ הוא NFA וקיים מילה w שנדחה ע"י M אז האורך המילה $|w| \leq 2^q$ כאשר $q = |Q|$ הוא המספר המצבים של M , וקיימים אינסוף מלים שנדחות ע"י M .

לפני שנתאר את האלגוריתם עצמו, נגדיר סימון שנשתמש בו בבניית האלגוריתם. נניח ש- $M = (Q, \Sigma, \delta, q_0, F)$ היא מכונת NFA כלשהי. תהי $P(Q)$ הקבוצת החזקה של Q . עבור כל NFA הפונקציה המעברים היא מהצורה

$$\delta : Q \times \Sigma \rightarrow P(Q).$$

בהינתן מילה $w = a_1 a_2 \dots a_n$ כאשר $a_i \in \Sigma$ הוא התו ה- i של המילה, $1 \leq i \leq n$. נגדיר את הסדרה הבאה:

$$S_0, S_1, S_2, \dots, S_n$$

כאשר

$$S_0 \triangleq \{q_0\}, \quad S_{i+1} \triangleq \delta(S_i, a_i),$$

כאשר $S_i \in P(Q)$ לכל $0 \leq i \leq n$.

בניית האלגוריתם

נבנה אלגוריתם לא-דטרמיניסטי, N המכריע את $\overline{ALL_{NFA}}$ באופן הבא:

$N =$ "על כל קלט x :

(1) בודקת אם $x = \langle M \rangle$, כאשר $M = (Q, \Sigma, \delta, q_0, F)$ היא מכונת NFA .

• אם לא $N \Leftarrow$ תדחה.

(2) יהי $q = |Q|$ מספר המצבים של M . נגדיר $S_0 = \{q_0\}$.

(3) N מבצעת את הלולאה הבאה:

לכל $0 \leq i \leq 2^q - 1$

(א) בוחרת באופן אי-דטרמיניסטי תו קלט $a_i \in \Sigma$.

(ב) מחשבת

$$S_{i+1} = \delta(S_i, a_i).$$

(ג) אם $N \Leftarrow S_{i+1} \cap F \neq \emptyset$ תדחה.

בפועל N בודקת את התנאי הזה ע"י לסמן את כל המצבים שב- S_{i+1} . אם אחד מהמצבים המסומנים הוא מצב קבלה אז N תדחה.

(4) אם בסיום הלולאה לא היה מצב-קבלה באף אחת מן הקבוצות S_i אז N תקבל. "

אם $x \in \overline{ALL_{NFA}}$

$\Leftarrow \langle A \rangle = x$, כאשר A היא מכונת NFA . וקיימת מילה $w \in \Sigma^*$ כך ש- A תדחה.

\Leftarrow קיימת מילה w' באורך לכל היותר 2^q ש- A תדחה.

\Leftarrow קיימת ריצה של N שבה N בוחרת את התווים של w' בלולאה.

\Leftarrow במהלך הריצה של A על w' , אין מצב קבלה באף אחת מן הקבוצות S_i .

\Leftarrow N לא דחתה עד סוף הלולאה.

\Leftarrow בסופה N תקבל.

אם $x \notin \overline{ALL_{NFA}}$ אז שני מקרים:

(מקרה 1) $N \Leftarrow x \neq \langle A \rangle$ תדחה בשלב 1.

(מקרה 2) $x = \langle A \rangle$ ו- $L(A) = \Sigma^*$

\Leftarrow לכל מילה $w \in \Sigma^*$, קיים שלב שבו A נמצא במצב קבלה.

\Leftarrow בכל ריצה של N , קיימת איטרציה i עבורה $S_i \cap F \neq \emptyset$.

\Leftarrow באיטרציה זו N תדחה.

\Leftarrow בכל ריצה N תדחה.

\Leftarrow N דוחה את x .

סיבוכיות מקום

• נסמן ב- $n = |\langle M \rangle|$ את אורך הקלט, וב- $q = |Q|$ את מספר המצבים של ה- NFA .

• כל מצב וכל מעבר של M מופיעים בקידוד, מתקיים $q = O(n)$.

• במהלך כל ריצה, N שומרת רק את המידע הבא:

- * הקבוצה הנוכחית $S_i \subseteq Q$ של מצבים אפשריים.
לפועל N שומרת S_i בוקטור ביטים באורך q לכל היותר.
- * מונה של האיטרציות הלולאה עד 2^q , המאוחסן בייצוג בינארי ודורש $O(q)$ ביטים.
- * תו קלט אחד הנבחר באופן אי-דטרמיניסטי בכל איטרציה, ומשתני עזר לחישוב S_{i+1} , הדורשים מקום קבוע או לינארי ב- q .

לפיכך סיבוכיות המקום הכוללת של N היא

$$O(q) = O(n) .$$

לפיכך האלגוריתם N פועל במקום לינארי.

שימו לב: N לינארי במקום אף על פי שזמן הריצה שלו עלול להיות אקספוננציאלי.

13.2 משפט סביץ'

הגדרה 13.4 CANYIELD

בהינתן מכונת טיורנג אי-דטרמיניסטית N , מספר טבעי חיובי t , ושתי קונפיגורציות c_1, c_2 של N (ראו את ההגדרה של קונפיגורציה בהגדרה 1.3). האלגוריתם $CANYIELD$ הוא אלגוריתם דטרמיניסטי הבודק אם ניתן לעבור מ- c_1 ל- c_2 על ידי לכל היותר t צעדי חישוב של N . התאור פסאודוקוד של $CANYIELD$ הוא כדלקמן:

$CANYIELD = \text{על קלט } \langle N, c_1, c_2, t \rangle$:

(1) רושם את c_1, c_2 ו- t על מחסנית.

(2) בודק אם N היא מכונת טיורנג, c_1, c_2 קונפיגורציות ו- t מספר טבעי חיובי.

• אם לא אז הוא דוחה.

(3) אם $t = 1$:

• אם $c_1 = c_2$ אז הוא מקבל.

• אחרת אם $c_1 \vdash_N c_2$ (אם אפשר לעבור מ- c_1 ל- c_2 בצעד אחד [ראו הגדרה 1.4]) מקבל.

• אחרת הוא דוחה.

(4) אם $t > 1$, לכל קונפיגורציה c_k של הרצה של N על w אשר משתמשת במקום $f(n)$

(כאשר w היא המילה הנקראת של הקונפיגורציה c_k):

(5) מריץ $CANYIELD \left(N, c_1, c_k, \left\lfloor \frac{t}{2} \right\rfloor \right)$

כאשר $\left\lfloor \frac{t}{2} \right\rfloor$ הוא השלם הכי הקרוב ל- $\frac{t}{2}$ וקטן מ- או שווה ל- $\frac{t}{2}$.

(6) מריץ $CANYIELD \left(N, c_k, c_2, \left\lfloor \frac{t}{2} \right\rfloor \right)$

כאשר $\left\lceil \frac{t}{2} \right\rceil$ הוא השלם הכי הקרוב ל- $\frac{t}{2}$ וגדול מ- או שווה ל- $\frac{t}{2}$.

(7) אם שתי ההרצות בשלבי (4) ו- (5) הסתיימו בקבלה \Leftarrow מקבל.

(8) אחרת אם לא התקבלה תשובת קבלה \Leftarrow דוחה.

משפט 13.2 משפט סביץ'

לכל פונקציה $f: \mathbb{N} \rightarrow \mathbb{R}^+$, אם $f(n) \geq n$ אז

$$NSPACE(f(n)) \subseteq SPACE(f^2(n)).$$

הוכחה:

הריעון של ההוכחה:

תהי N מ"ט אי-דטרמיניסטית שמכריעה את השפה A במקום $O(f(n))$, כאשר n אורך הקלט w של N .
נבנה מכונת טיורינג דטרמיניסטית, M שמכריעה את A במקום $O(f^2(n))$.
כלומר, בהינתן $N \in NSPACE(f(n))$ המכריעה שפה A , נבנה $M \in SPACE(f^2(n))$ המכריעה A .
כלומר, אנחנו נראה שלכל $N \in NSPACE(f(n))$ קיימת $M \in SPACE(f^2(n))$.
באופן הזה אנחנו נוכיח כי

$$NSPACE(f(n)) \subseteq SPACE(f^2(n)).$$

בניית המכונה:

תהי N מכונת טיורינג אי-דטרמיניסטית שמכריעה השפה A .
תהי w מחרוזת שהיא הקלט של N .
בהינתן שתי קונפיגורציות c_1, c_2 של N ומספר טבעי חיובי t .

• אם ניתן לעבור מ- c_1 ל- c_2 בכלל היותר t צעדים $\Leftrightarrow CANYIELD(N, c_1, c_2, t)$ מקבל.

• אחרת $\Leftrightarrow CANYIELD(N, c_1, c_2, t)$ דוחה.

נגדיר מכונת טיורינג דטרמיניסטית M שמסמלצת את המכונה האי-דטרמיניסטית N באופן הבא.

ראשית נסמן ב- n את אורך הקלט w של N .

תהי c_0 הקונפיגורציה ההתחלתית.

נתקן את N כך שלאחר כל ריצה הראש חוזר לקצה השמאל של תוכן הסרט ו- N עוברת לקונפיגורציה c_{acc} .

נגדיר d כך ש- $2^{df(n)}$ הוא חסם עליון של מספר הקונפיגורציות שקיימות בריצות של N שדורשות $O(f(n))$ מקום.

המכונת טיורינג הדטרמיניסטית M תוגדר כך:

$$M = \text{"על קלט } w$$

(1) מריצה $CANYIELD$ על הקלט $\langle N, c_0, c_{acc}, 2^{df(n)} \rangle$ ועונה כמוהו.

הוכחת הנכונות:

נניח $w \in L(N)$ ו- $N \in NSPACE(f(n))$.

\Leftarrow לפי ההגדרה של d , ל- N יש $2^{df(n)}$ לכל היותר.

\Leftarrow קיים מסלול חישוב N על w מ- c_0 ל- c_{acc} .

\Leftarrow האלגוריתם $CANYIELD$ על הקלט $\langle N, c_0, c_{acc}, 2^{df(n)} \rangle$ יקבל.

M יקבל w . \Leftarrow

נניח $w \notin L(N)$ ו- $N \in NSPACE(f(n))$.

\Leftarrow לפי ההגדרה של d , ל- N יש $2^{df(n)}$ לכל היותר.

\Leftarrow לא קיים מסלול חישוב של N על w מ- c_0 ל- c_{acc} .

\Leftarrow האלגוריתם $CANYIELD$ על הקלט $\langle N, c_0, c_{acc}, 2^{df(n)} \rangle$ ידחה.

$\Leftarrow M$ ידחה w .

סיבוכיות מקום:

- כל פעם ש- $CANYIELD$ מפעיל את עצמו באופן רקורסיבי, הוא רושם את c_2, c_1 ו- t על מחסנית, כך שניתן יהיה לשחזר אותם לאחר הקריאה הרקורסיבית הבאה.
- בגלל ש- $N \in NSPACE(f(n))$ אזי הכתיבה של c_2, c_1 ו- t על המחסנית דורשת $O(f(n))$ מקום.
- בכל שלב של הרקורסיה, האלגוריתם $CANYIELD$ מחלק את t ב- 2.
- הערך ההתחלתי של t הוא $2^{df(n)}$ לכן העומק של הרקורסיה הוא $O(\log_2(2^{df(n)})) = O(f(n))$.
- לכן המכום הכולל ש- M דורש הוא $O(f^2(n))$.

לפיכך

$$M \in SPACE(f^2(n)) .$$

לסיכום: הוכחנו שבהינתן מכונת אי-דטרמיניסטית N כלשהי שמכריעה שפה A כלשהי עבודה

$$N \in NSPACE(f(n)) ,$$

קיימת מכונת טיורינג דטרמיניסטית M שמכריעה A במקום $O(f^2(n))$, כלומר:

$$M \in SPACE(f^2(n)) .$$

לפיכך:

$$NSPACE(f(n)) \subseteq SPACE(f^2(n)) .$$



13.3 המחלקה PSPACE

ההגדרה הבאה היא האנלוג של ההגדרה 10.2 של אלגוריתם זמן פולינומיאלי.

הגדרה 13.5 אלגוריתם מקום פולינומיאלי

אומרים כי אלגוריתם A מכריע בעייה במקום פולינומיאלי אם קיים קבוע $c > 0$ כך שהמקום הריצה של A על קלט w חסום ע"י $O(|w|^c)$.

התזה של צרף' טיורינג אומר שאם קיים אלגוריתם המכריע בעייה במקום פולינומיאלי, אזי קיימת מכונת טיורינג דטרמיניסטית המכריעה את השפה השקולה לבעייה זו במקום פולינומיאלי.

. אלגוריתם מכריע \equiv מכונת טיורינג דטרמיניסטית

הגדרה 13.6 המחלקה $PSPACE$

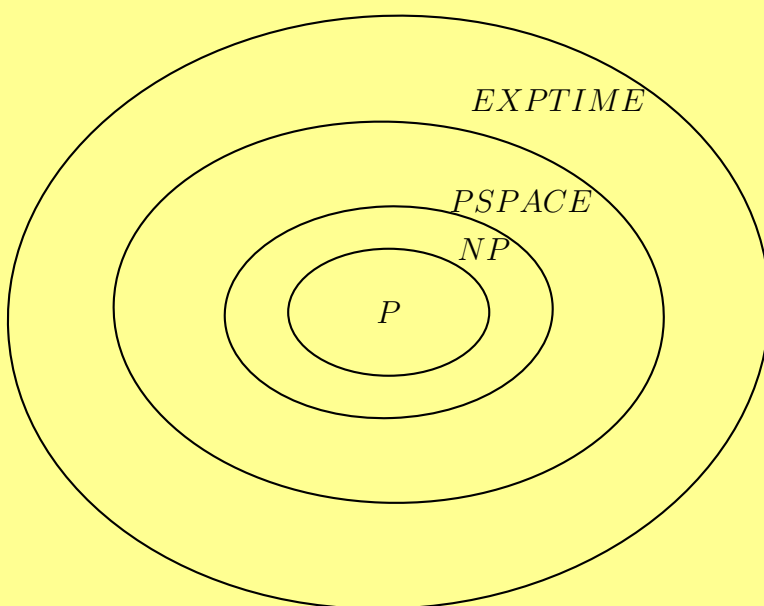
המחלקה $PSPACE$ היא אוסף כל הבעיות (השפות) שקיים עבורן אלגוריתם (מכונת טיורנג) דטרמיניסטי המכריע אותן במקום פולינומיאלי.

הגדרה 13.7 המחלקה $NPSPACE$

$NPSPACE$ היא אוסף כל הבעיות (השפות) שקיים עבורן אלגוריתם (מכונת טיורנג) אי-דטרמיניסטי המכריע אותן במקום פולינומיאלי.

משפט 13.3

$$P \subseteq NP \subseteq PSPACE \subseteq EXPTIME .$$

**13.4 שלמות ב- $PSPACE$** **הגדרה 13.8 $PSPACE$ קשה**

בעייה B נקראת $PSPACE$ קשה אם לכל בעייה $A \in PSPACE$ קיימת רדוקציה זמן פולינומיאלית מ- A ל- B . כלומר: $A \leq_p B$.

הגדרה 13.9 $PSPACE$ שלמות

בעייה B נקראת $PSPACE$ שלמה אם השני התנאים הבאים מתקיימים:

$$B \in PSPACE \quad (1)$$

(2) לכל בעייה $A \in PSPACE$ קיימת רדוקציה זמן פולינומיאלית מ- A ל- B . כלומר: $A \leq_p B$.

13.5 נוסחאות בוליאניות עם כמתים

בפרקים 11 ו-12 הגדרנו נוסחה בוליאנית כביטוי מתמטי שבנוי מהמרכיבים הבאים:

- משתנים בולינאיים, שמקבלים את הערכים 0 ו-1 (לעתים מסומנים F ו- T).
- אופרטורים בולינאיים עיקריים

וגם	\wedge
או	\vee
לא	\neg

כעת נכליל את ההגדרה הזו לסוג היותר מורחב של נוסחה בולינאית: נוסחה בולינאית עם כמתים.

הגדרה 13.10 נוסחת בוליאנית עם כמתים - QBF

בנוסחת בוליאנית עם כמתים מופיעה אחת מהשני כמתים העיקריים:

\forall	"לכל" (נקרא גם "כמת כולל")
\exists	"קיים" (נקרא גם "כמת ישי")

דוגמה 13.3 דוגמאות של נוסחאות בוליאניות עם כמתים

בדוגמאות הבאות x, y הם משתנים בולינאיים. כלומר $x, y \in \{0, 1\}$.

(1)

$$\phi = \forall x \exists y [(x \vee y) \wedge (\bar{x} \vee \bar{y})]$$

בדוגמה זו $\phi = 1$.

(2)

$$\phi = \forall x [x \vee \bar{x}] \rightarrow \phi = 1$$

(3)

$$\phi = \exists x (x \wedge \bar{x}) \rightarrow \phi = 0$$

הגדרה 13.11 $TQBF$

$\langle \phi \rangle$ בשפה $TQBF$ אם ϕ נוסחת בוליאנית עם כמתים והנוסחה מעורכת לאמת.

$$TQBF = \{ \langle \phi \rangle \mid \phi \text{ נוסחת בוליאנית עם כמתים ו-} \phi = 1 \}$$

הערה 13.1

בניגוד ל- SAT עבורה השאלה היא האם קיימת הצבת אמת, ב- $TQBF$ לכל נוסחה יש ערך אמת או שקר יחיד, מכיוון שאין משתנים חופשיים.

משפט 13.4

$$SAT \subseteq TQBF.$$

הוכחה: תרגיל בית.

משפט 13.5

השפה $TQBF$ היא NP שלמה.

הוכחה: נראה כי

$$TQBF \in PSPACE \quad (1)$$

$$(2) \text{ לכל } A \in PSPACE \text{ מקתיים } A \leq_P TQBF.$$

נבנה אלגוריתם רקורסיבי $A \in PSPACE$ שמכריע את $TQBF$ באופן הבא.

בניית האלגוריתם

$A = \text{"על קלט } \langle \psi \rangle \text{ כאשר}$

$$\psi = Q_1 x_1 Q_2 x_2 \dots Q_n x_n \phi(x_1, \dots, x_n),$$

כאשר לכל $1 \leq i \leq n$, Q_i הוא כמת \forall או \exists , x_i משתנה בוליאני ו- $\phi(x_1, \dots, x_n)$ נוסחה בוליאנית בלי כמתים:

(1) מקרה הבסיס:

אם $n = 0$ אז $\phi(x_1, \dots, x_n)$ הוא קבוע ומעריך אותה.

(2) מקרה הרקורסיבי:

• אם $Q_1 = \exists$:

• מריץ $A(\psi(x_1 = 0))$

• מריץ $A(\psi(x_1 = 1))$

• אם לפחות אחד מהם התקבל אז מקבל.

• אם $Q_1 = \forall$:

• מריץ $A(\psi(x_1 = 0))$

• מריץ $A(\psi(x_1 = 1))$

• אם שניהם התקבלו אז מקבל.

הוכחת הנכונות

ניתן להוכיח הנכונות של A ע"י אינדוקציה.

שלב הבסיס: $n = 0$.

אם $\psi = 1 \Leftarrow$ בשלב (1) מחשב את $A \Leftarrow \phi(x_1, \dots, x_n) = 1$ מקבל.

אם $\psi = 0 \Leftarrow$ בשלב (1) מחשב את $A \Leftarrow \phi(x_1, \dots, x_n) = 0$ דוחה.

שלב המעבר: $n = k$.

נניח ש- A מכריע נוסחה כלשהי עם כמתים $\psi = Q_1 x_1 Q_2 x_2 \dots Q_k x_k \phi(x_1, \dots, x_n)$ במקרה $n = k$. זוהי ההנחת האינדוקציה שלנו. נוכיח כי A גם מכריע נוסחה ψ כלשהי במקרה ש- $n = k + 1$ באופן הבא:

אם $\psi(x_1, x_2, \dots, x_k, x_{k+1}) = 1$ שני מקרים:

מקרה 1: $Q_1 = \exists$

$\psi(x_1 = 0) = 1$ או $\psi(x_1 = 1) = 1$ ושתייהן QBF בעלי k משתנים וכמתים.
 $A(\psi(x_1 = 0))$ או $A(\psi(x_1 = 1))$ התקבל או $A(\psi(x_1 = 1))$ התקבל.
 A מקבל $\psi(x_1, \dots, x_{k+1})$.

מקרה 2: $Q_1 = \forall$

$\psi(x_1 = 0) = 1$ וגם $\psi(x_1 = 1) = 1$ ושתייהן QBF בעלי k משתנים וכמתים.
 $A(\psi(x_1 = 0))$ וגם $A(\psi(x_1 = 1))$ התקבל וגם $A(\psi(x_1 = 1))$ התקבל.
 A מקבל $\psi(x_1, \dots, x_{k+1})$.

אם $\psi(x_1, x_2, \dots, x_k, x_{k+1}) = 0$ שני מקרים:

מקרה 1: $Q_1 = \exists$

$\psi(x_1 = 0) = 0$ וגם $\psi(x_1 = 1) = 0$ ושתייהן QBF בעלי k משתנים וכמתים.
 $A(\psi(x_1 = 0))$ דוחה וגם $A(\psi(x_1 = 1))$ דוחה.
 A דוחה $\psi(x_1, \dots, x_{k+1})$.

מקרה 2: $Q_1 = \forall$

$\psi(x_1 = 0) = 0$ או $\psi(x_1 = 1) = 0$ ושתייהן QBF בעלי k משתנים וכמתים.
 $A(\psi(x_1 = 0))$ דוחה או $A(\psi(x_1 = 1))$ דוחה.
 A דוחה $\psi(x_1, \dots, x_{k+1})$.

סיבוכיות מקום של האלגוריתם

כדי לחשב את הסיבוכיות מקום, נסמן השימוש מקום ב- $S(n, m)$ כאשר n המספר המשתנים של ψ ו- $m = |\psi|$ הוא האורך של ψ . אז יש לנו את היחס רקורסיבי הבא:

$$S(0, m) = O(m), \quad S(n, m) = S(n-1, m) + O(m).$$

מכאן

$$S(n, m) = O(nm).$$

הוכחה כי השפה $TQBF$ היא $PSPACE$ קשה: בניית הרדוקציה $L \leq_P TQBF$.

נראה כי לכל שפה $L \in PSPACE$ מתקיים

$$L \leq_P TQBF.$$

תהי L שפה השייכת ל- $PSPACE$ ותהי M מכונת טיורינג המכריעה את L במקום פולינומיאלי $S(n)$. נוכיח שקיימת פונקציה f כך ש:

$$x \in L \iff f(x) \in TQBF.$$

ז"א $f(x) = \psi$ כאשר ψ היא נוסחה בוליאנית עם כמתים כך ש:

$$\begin{aligned} x \in L &\Rightarrow \psi = 1, \\ x \notin L &\Rightarrow \psi = 0. \end{aligned}$$

כדי להגדיר את הפונקציה הרדוקציה, קודם נגדיר את הנוסחה הבוליאנית הבאה. יהיו c, c' שתי קונפיגורציות של המכונה M ויהי t מספר שלם חיובי.

$$\phi(c, c', t) \triangleq \begin{cases} 1 & : \text{אפשר לעבור מ-} c \text{ ל-} c' \text{ עם } t \text{ צעדים לכל היותר} \\ 0 & : \text{אחרת} \end{cases}.$$

הצורה המפורשת של הנוסחה $\phi(c, c', t)$ עצמה בנויה רקורסיבי באופן הבא.

המקרה $t > 1$

לכל $t > 1$ ולכל קונפיגורציות c, c' :

$$\phi(c, c', t) = \exists w \forall x, y \quad (x = c \wedge y = w) \vee (x = w \wedge y = c') \Rightarrow \phi\left(x, y, \left\lfloor \frac{t}{2} \right\rfloor\right).$$

כאן w מסמן קונפיגורציה ביניים בין הקונפיגורציה c לקונפיגורציה c' . היחס הרקורסיבי הזה של $\phi(c, c', t)$ אומר ש- M יכול לעבור מ- c ל- c' ב- t צעדים לכל היותר אם קיימת קונפיגורציה ביניים w כך ש- M יכול לעבור מ- c ל- w בכל היותר $\lfloor \frac{t}{2} \rfloor$ צעדים, ואחר כך M יכול לעבור מ- w ל- c' בכל היותר $\lfloor \frac{t}{2} \rfloor$ צעדים. הנוסחאות $\phi(c, w, \lfloor \frac{t}{2} \rfloor)$ ו- $\phi(w, c', \lfloor \frac{t}{2} \rfloor)$ הן נבנות בשלב הבא של הרקורסיה באותו אופן. התליך הזה ממשיך עד שנגיע לנוסחה $\phi(x, y, t)$ שבה $t = 0$ או $t = 1$.

המקרה $t = 1$

$$\phi(c, c', t = 1) = \begin{cases} 1 & : \text{אם } M \text{ יכול לעבור מ-} c \text{ ל-} c' \text{ בצעד חישוב בודד} \\ 0 & : \text{אחרת} \end{cases}$$

הצורה המפורשת של $\phi(c, c', t = 1)$ היא כדלקמן. תהיינה c, c' שתי קונפיגורציות כלשהן. נגדיר טבלת הקונפיגורציות:

#	c_1	c_2	...	c_{N-1}	c_N	#
#	c'_1	c'_2	...	c'_{N-1}	c'_N	#

נניח כי $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$ ותהי $C = Q \cup \Gamma \cup \{\#\}$. עבור כל תא (i, j) של הטבלת הקונפיגורציות נגדיר משתנה בוליאני $x_{i,j,s}$ לכל $1 \leq i \leq 2$ ולכל $1 \leq j \leq N$ כך ש:

$$x_{ijs} = \begin{cases} 1 & : \text{הסימן } s \in C \text{ כתוב בתא } ij \\ 0 & : \text{הסימן } s \in C \text{ לא כתוב בתא } ij \end{cases}$$

למשל אם בתא $(2, 5)$ כתוה a אז $x_{2,5,a} = 1$ בעוד $x_{2,5,b} = 0$. הנוסחה $\phi(c, c', t = 1)$ מוגדרת:

$$\phi(c, c', t = 1) = \phi_{\text{cell}} \wedge \phi_c \wedge \phi_{\text{move}} \wedge \phi_{c'}.$$

• $\phi_{\text{cell}} = 1$ אם ורק אם יש רק סימן אחד בדיוק שכתוב בכל תא ו-0 אחרת. בפרט:

$$\phi_{\text{cell}} = \bigwedge_{\substack{1 \leq j \leq 2 \\ 1 \leq j \leq n^k}} \left[\left(\bigvee_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s, t \in C \\ s \neq t}} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right] \quad (13.1)$$

מבטיח שלכל תא לפחות משתנה אחד דולק.

$$\bigwedge_{\substack{s,t \in C \\ s \neq t}} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \quad \text{מבטיח שלכל תא משתנה אחד לכל היותר דולק.}$$

לפיכך ϕ_{cell} מסופקת אם ורק אם יש בדיוק סימן אחד כתוב בכל תא. בניגוד זה אם שני משתנים של אותו תא דולקים בגלל שהסימן s וגם הסימן t כתובים בו זמנית בתא ה- ij אז $x_{i,j,s} = 1$ וגם $x_{i,j,t} = 1$ ואז ϕ_{cell} תהיה שווה ל-0.

• הנוסחה ϕ_c (קובעת כי המשתנים הספציפים של הקונפיגורציות c (c') הם דולקים:

$$\begin{aligned} \phi_c &= x_{1,0,\#} \wedge x_{1,1,c_1} \wedge x_{1,2,c_2} \wedge \dots \wedge x_{1,N-1,c_{N-1}} \wedge x_{1,N,c_N} \wedge x_{1,N+1,\#} , \\ \phi_{c'} &= x_{2,0,\#} \wedge x_{2,1,c'_1} \wedge x_{2,2,c'_2} \wedge \dots \wedge x_{2,N-1,c'_{N-1}} \wedge x_{2,N,c'_N} \wedge x_{2,N+1,\#} . \end{aligned} \quad (13.2)$$

• $\phi_{\text{move}} = 1$ אם אפשר להגיע מ- c ל- c' על ידי תזוזה חוקית אחת של M ו-0 אחרת. תזוזה חוקית בין כל שתי קונפיגורציות מוגדרת על פי סהפונקציה המעברים של M . במונחי הטבלה, אפשר להגדיר תזוזה חוקית בין השתי שורות על ידי תת-טבלה מסדר 2×3 שמכילה 3 עמודות. נקרא תת-טבלה כזו "חלון". אלה הם דוגמאות של חלונות חוקיים:

a	q_1	b
q_2	a	c

a	q_1	b
a	a	q_2

a	a	q_1
a	a	b

#	b	a
#	b	a

a	b	a
a	b	q_2

b	b	b
c	b	b

בעוד אלה הם דוגמאות של חלונות לא חוקיים:

a	b	a
a	a	a

a	q_1	b
q_1	a	a

b	q_1	b
q_2	b	q_2

$\phi_{\text{move}} = 1$ אם כל חלון של הטבלה חוקי ו-0 אם לא. בפרט, כל חלון מכיל 6 תאים. החלון המכיל את ה-3 עמודות i , $i+1$ ו- $i+2$ נקרא החלון- i . לכן:

$$\begin{aligned} \phi_{\text{move}} &= \bigwedge_{1 \leq i \leq N-2} (\text{חלון-} i \text{ חוקי}) \\ &= \bigwedge_{1 \leq i \leq N-2} \left(x_{1,i,c_i} \wedge x_{1,i+1,c_{i+1}} \wedge x_{1,i+2,c_{i+2}} \wedge x_{2,i,c'_i} \wedge x_{2,i+1,c'_{i+1}} \wedge x_{2,i+2,c'_{i+2}} \right) \end{aligned} \quad (13.3)$$

המקרה $t = 0$

אם $t = 0$ אז

$$\phi(c, c', t = 0) = \begin{cases} 1 & : c = c' \\ 0 & : c \neq c' \end{cases} .$$

כעת נחזור להגדרה עצמה של הרדוקציה פולינומיאלית. תהי L שפה כריעה ע"י מכונת טיורינג M במקום $O(n^k)$, אז קיימת רדוקציה $\psi = f(x)$ כך ש:

$$\psi = \phi(c_{\text{start}}, c_{\text{acc}}, h)$$

כאשר $h = 2^{df(n)}$ ו- $d > 0$ מספר ממשי חיובי הנבחר כך של- M יש לכל היותר $2^{df(n)}$ קונפיגורציות בהינתן קלט של אורך n , ו- $f(n) = n^k$.

הוכחת הנכונותאם $x \in L$ M תקבל $x \Leftarrow$ M יכולה לעבור מ- c_{start} ל- c_{acc} במספר צעדים פחות מ- h או שווה ל- h $\psi = 1 \Leftarrow$ $\psi \in TQBF \Leftarrow$ אם $x \notin L$ M תדחה $x \Leftarrow$ לא קיימים צעדים של M מ- c_{start} ל- c_{acc} . $\psi = 0 \Leftarrow$ $\psi \notin TQBF \Leftarrow$ סיבוכיות זמן של הרדוקציה

בפונקצית הרדוקציה מחשבת $f(x) = \psi = \phi(c_{\text{start}}, c_{\text{acc}}, h)$ כאשר $h = 2^{df(n)}$ ו- $f(n) = n^k$, באופן רקורסיבי. לרקורסיה יש מספר שלבים השווה ל:

$$\log_2(2^{df(n)}) = df(n) = dn^k.$$

- הנוסחה (13.1) של ϕ_{cell} מכילה n^{2k} נוסחאות עם 3 ליטרלים. לכן הסיבוכיות זמן של ϕ_{cell} היא:

$$O(n^{2k}).$$

- הנוסחאות ϕ_c ו- $\phi_{c'}$ במשוואה (13.2) מכילות בדיוק n^k ליטרלים. לכן הסיבוכיות זמן של ϕ_c ושל $\phi_{c'}$ הן:

$$O(n^k).$$

- הנוסחה (13.3) של ϕ_{move} מכילה n^{2k} נוסחאות עם 6 ליטרלים. לכן הסיבוכיות זמן של ϕ_{move} היא

$$O(n^{2k}).$$

לפיכך בסה"כ הרדוקציה ניתנת לחישוב בזמן $O(n^{2k})$ ולכן $f \in \text{TIME}(n^{2k})$ ולכן הרדוקציה היא רדוקציה פולינומיאלית.

13.6 המחלקה L**13.7 המחלקה NL****13.8 שלמות ב- NL****13.9 שיוויון NL ו- coNL**