

3	שפות תכנות	2	מכונת מונים (CM) ו-STs	1	הבונה החרוץ ואחרות
6	בעיות לא כריעות	5	משפט הרקורסיה	4	משפט רייס
9	סבוכיות קולמגורוב	8	הוכחת R/RE/Co-RE	7	מנייה רקורסיבית (RE)
12	שקילות כוח החישוב	11	מכונת RAM	10	מכונת טיורינג TM
15	אוטומטים ורגולריות	14	רדוקציות	13	סבוכיות זמן
18	הערות כלליות	17	דוגמאות	16	דקדוק חסר הקשר

7 מנייה רקורסיבית (RE)
 הטענות הבאות שקולות: (1) שפה ניתנת למנייה רקורסיבית. כלומר, קיים מונה רקורסיבי (על אבל לא בהכרח חח"ע). (2) כריעה למחצה. (3) קיימת תוכנית שמדפיסה את כל האיברים בשפה.
הערות: (א) לא כל קבוצה בת מניה ניתנת למספור רקורסיבי. יש קבוצות כמו קבוצת התוכניות שאינן עוצרות שהיא בת מנייה אך לא ניתנת לתכנות מניה רקורסיבי. **(ב)** אם L היא שפה הכריעה למחצה וגם המשלימה שלה כריעה למחצה אז L היא שפה כריעה. **(ג)** אינומרטור לשפה L היא פונקציה חשיבה ועל $f: N \rightarrow L$.
טענה: $L \in RE$ אינסופית אם יש L -אינומרטור מונוטוני. כלומר, אם $f(i) > f(j)$ בסדר לקסיקוגרפי.
טענה: לכל שפה אינסופית L ב-RE, קיימת תת שפה אינסופית L' כך ש-L' כריעה.

8 הוכחת R/RE/Co-RE
איך להראות כריעות? (1) אלגוריתם שפותר את הבעיה. (2) להראות שהבעיה היא "סופית".
 (3) רדוקציה אל בעיה הידועה ככריעה. (4) ע"י הוכחה שהיא והמשלימה שלה כריעות למחצה.
בעיות סופיות: (1) בעיות עם מספר סופי של שאליות הן תמיד כריעות. (2) בעיות שמספר השאליות בהם מוחזר כן/לא (מספיק אחד מהם) הוא סופי.
איך להראות אי-כריעות? (1) באמצעות שיטת האלכסון (כמו בבונה החרוץ/בעיית העצירה).
 (2) רדוקציה מבעיה הידועה כלא כריעה. (3) משפט רייס. (4) ע"י חיקוי הוכחת משפט רייס.
איך להראות כריעות למחצה? (1) אלגוריתם המכריע את הבעיה למחצה. (2) רדוקציה אל בעיה הידועה ככריעה למחצה. (3) ע"י הרצת שני מונים במקביל.
איך להראות Co-RE? (1) באמצעות שיטת האלכסון (כמו ב-SELF). (2) רדוקציות מיפוי מבעיה הידועה כ-Co-RE (כמו $halt^*$). (3) להראות שהמשלים שלה הוא RE ושזו אינה בעיה כריעה.

9 סבוכיות קולמגורוב
 הגדרה: $K(x)$ מחשבת את האורך המינמלי של תוכנית שמחזירה את x.
 מספר עובדות תחילה: ברור ש $|x| \leq K(x)$ כי אפשר לכתוב את המספר. בנוסף ברור שאין משמעות לשפה שבה אנו עובדים מכיון שבין כל שפה לשפה קיים מפרש ואורכו קבוע ונסמנו ב i אם כך לדוגמה $K_{Scheme}(x) \leq K_{C++}(x) + i$ ישנם מילים שלא ניתן לדחוס ועבורם $|x| > K(x)$.
 מספר דוגמאות: $c + K(x) \leq K(xx)$, $K(1^n) \leq \log(n) + c$, $K(2^n) \leq \log(n) + c$.
 אבל מחרוזות עם פתרון פשוט יחסית כזה הם נדירות. הפונקציה $K(\cdot)$ מוגדרת לכל מחרוזת ולא חסומה. אבל האם היא חשיבה? **טענה:** הפונקציה לא חשיבה. **הוכחה:** נניח בשלילה שהיא כן חשיבה. אם כך לכל $n \in N$ נגדיר את y_n להיות המחרוזת y המחושבת מבחינה לקסיקוגרפית שמקיימת $n < K(y)$. אם כך יש לנו את המצב הנוכחי z-י מכיל את $\{y_n\}_{n=1}^\infty$ מוגדרת היטב. כיון ש K חשיבה זה אומר שלכל n קיים קבוע c שעבורו $K(y_n) \leq \log(n) + c$. ואם כך קיבלנו סתירה לאופן שבה הוגדר y ומכאן הפונקציה לא חשיבה.

10 מכונת טיורינג TM
 $M = (Q, \Sigma, \Gamma, q_0, q_a, q_r, \delta)$ כאשר Q הוא מספר סופי של מצבים, Σ א"ב סופי שעל הסרט (ללא רווח). Γ א"ב סופי של המכונה (מכיל את הא"ב של הסרט ומכיל גם רווח). q_0, q_a, q_r – מצב התחלה, מקבל ודחיה. δ – פונקציות מעבר. **ייצוג מצב של TM:** מתבצע ע"י מחרוזת xyz. כאשר x מכיל את הקלט עד המיקום בו נמצא הראש, y מכיל את המצב הנוכחי z-י מכיל את סרט הקלט מהראש ועד הסוף. לדוגמה 1011q7011 אומר שאנו במצב q7, לפני הראש מופיע על הסרט 1011 ואחריו 011 (הראש נמצא על 0).
 שני סוגים של TM: **Acceptor** – מקבל קלט על הסרט (עד הרווח הראשון), מקבל את הקלט שכונס למצב קבלה, **ללא פלט**. **Computer** – קלט על הסרט (עד הרווח הראשון), מסיים ברגע שכונס למצב halt. פלט על הסרט (עד הרווח הראשון).
סימולציה: כל TM ניתן לסמלץ בסקים $B_m := \lambda z. tm([m], \epsilon, x, q_0)$. where [M] is the transition function of M as a list and strings are lists of symbols.
כמה אפשרויות קיימות למכונת טיורינג עם 3 סרטים (קלט – קריאה בלבד, טיטה – קריאה וכתובה ופלט לכתובה בלבד) $|Q| \leq |N| \cdot |S| \cdot |S|$ (משמאל לימין: כמות הקלטים האפשרית, מיקום הראש בקלט, מה שכתוב על סרט הטיטה, מיקום ראש הטיטה, המצב הפנימי).
 ב-QSAT כאשר אורך הסרט מוגבל, מספר הקונפיגורציות הוא $|Q| \cdot |\Gamma|^n$ (מצבים, תוכן הסרט, מיקום הראש) כאשר $n = |x|$ אורך הקלט.

11 מכונת RAM
תכונות RAM: זכרון פנימי- סדרה סופית ובלתי חסומה של אוגרים. זכרון חיצוני – רצף בלתי חסום של אוגרים. **תוכנית:** רצף סופי של הוראות ממוספרות. קלט/פלט – ברגיסטר הראשון. קיימת פקודת עצירה.

$x := y \text{ mod } 2$	$x := \lfloor y / 2 \rfloor$	$x := 2y$
t := y; x := 0	t := y; x := 0	u := y; x := 0
if t=0 then goto c else goto a; a: t--	a: t--	if u=0 then go to c else go to b
if t=0 then goto b else goto d	if t=0 then go to c else go to b	b: u--; x++; x++
d: t--	b: t--; x++; goto a	if u=0 then go to c else go to b
if t=0 then goto c else goto a	c: ...	c: ...
b: x ++; c: ...		

1 הבונה החרוץ ואחרות
(n) bb - פונקציה לא חשיבה המקבלת n אורך של תוכנית ומחזירה את המספר הגדול ביותר שניתן לחשב עם תוכנית באורך n לכל היותר. **הוכחת אי חשיבות:** נניח בשלילה שקיימת תוכנית B שמחשבת את bb, ונסמן $|B| = N$. נכתוב תוכנית חדשה: $c() := B(10^N) + 1$. נשים לב: מצד אחד, $c() := B(10^N) + 1 \leq 2N + \text{const} \leq 10N$. מצד שני, לפי הגדרת הבונה החרוץ $bb(10N) \geq c()$.
מכונת מונים (CM) ו-STs
מכונת מונים: פעולות קיימות במכונה: ++ (קידום ב-1), -- (חיסור ב-1), 0 נשאר 0, 0= (האם שווה ל-0). **תכונות:** זיכרון מספר סופי של מונים אינסופיים. **תוכנית:** מספר סופי של רצף הוראות. קלט/פלט: מתבצע ברגיסטר הראשון בלבד. שאר המונים מאופסים. פעולת עצירה = סוף תוכנית. שיטות לתרגום ברגיסטר: (א) ייצוג של כל הרגיסטרים כמספר יחיד באמצעות הראשונים $Z = 2^{m_1} 3^{m_2} \dots$. (ב) הפיכת כל 2 מספרים למספר 1 וכל 2 זוגות לזוג חדש וכך הלאה $pair(x,y) = 2^x(2y+1)$. כדי לקבל חזרה, בודקים כמה פעמים ניתן לחלק ב-2 זהו x. מהמספר הנותר מורידים 1 ומחלקים ב-2 ומתקבל y.
STS מכיל: Q – מצבים, $1 \leq Q$. קלט, $0 \leq Q$. פלט, $Q \rightarrow Q$. t – פונקציות מעברים.

2 מכונת מונים (CM) ו-STs
מכונת מונים: פעולות קיימות במכונה: ++ (קידום ב-1), -- (חיסור ב-1), 0 נשאר 0, 0= (האם שווה ל-0). **תכונות:** זיכרון מספר סופי של מונים אינסופיים. **תוכנית:** מספר סופי של רצף הוראות. קלט/פלט: מתבצע ברגיסטר הראשון בלבד. שאר המונים מאופסים. פעולת עצירה = סוף תוכנית. שיטות לתרגום ברגיסטר: (א) ייצוג של כל הרגיסטרים כמספר יחיד באמצעות הראשונים $Z = 2^{m_1} 3^{m_2} \dots$. (ב) הפיכת כל 2 מספרים למספר 1 וכל 2 זוגות לזוג חדש וכך הלאה $pair(x,y) = 2^x(2y+1)$. כדי לקבל חזרה, בודקים כמה פעמים ניתן לחלק ב-2 זהו x. מהמספר הנותר מורידים 1 ומחלקים ב-2 ומתקבל y.
STS מכיל: Q – מצבים, $1 \leq Q$. קלט, $0 \leq Q$. פלט, $Q \rightarrow Q$. t – פונקציות מעברים.

3 שפות תכנות
 ניתן לחלק שפות תכנות ל-2 חלקים: שפות מלאות עובורן בעיית העצירה לא כריעה אך ניתן לכתוב מפרש לשפה בשפה ושפות עוצרות עובורן בעיית העצירה קריאה אבל לא ניתן לכתוב מפרש לשפה בשפה עצמה.

4 משפט רייס
בעיה טריוויאלית – היא בעיה שהתשובה עליה היא תמיד 'כן' או תמיד 'לא'.
בעיה סמנטית – היא בעיה שבהניתן 2 תוכניות שקולות (מחזירות אותו ערך לכל קלט), מתקיים $P \neq g \Rightarrow P(f) = P(g)$. כלומר, מראהם דקדוקיה: $f \neq g \Rightarrow P(f) = P(g)$.

תנאי משפט רייס: (1) זו שפה של תוכניות (הקלט והפלט היא תוכנית). (2) הבעיה אינה טריוויאלית. (2) הבעיה סמנטית. (4) הבעיה אינה תלויה רק במספר הקלטים.
טענת המשפט: בהתקיים התנאים על בעיה P, אזי P אינה כריעה.
שלבי הוכחת המשפט: מחלקים את עולם הבעיות ל-2 טיפוסים a ו-b (בעיה היא מסוג a כאשר $P(\perp) = F$). P לא טריוויאלית לכן, קיימת תוכנית y כך ש- $P(y) = T$ (בבעיות מסוג b: $P(y) = F$). מראים דקדוקיה:

$$halt(f) := P \left(\lambda x. \begin{cases} f(x) & \text{if } x = \perp \\ f(y) & \text{if } x = y \\ f(\perp) & \text{if } x = w \end{cases} \right)$$
 עוצרת $f \Leftarrow g \Leftarrow y \Leftarrow T = P(y) = P(g) = halt(f)$.
 לא עוצרת $f \Leftarrow g \Leftarrow \perp = F = P(\perp) = P(g) = halt(f)$.

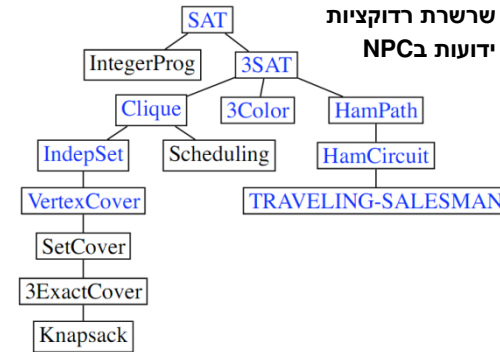
5 משפט הרקורסיה
 לכל תוכנית שמקבלת תוכנית ומחזירה תוכנית קיימת נקודת שבת. כלומר קיים $c \equiv f(c)$.
 הוכחה לתוכנית שרצות בסדר נורמלי:
 Let $c = k(k)$ where $k = \lambda w. f(w(w))$
 $c \equiv k(k) \equiv (\lambda w. f(w(w)))(k) \equiv f(k(k)) \equiv f(c)$
 שלבי הוכחה לתוכנית בסדר אפלקטיבי:
 Let $c := h(k)$ where $k(w) := f(h(w))$
 $h(y) := \lambda z. y(y)(z)$

6 בעיות לא כריעות
Pconst(p) - מוחזר T אם קיים n כך שלכל x, $p(x) = \perp$ או $p(x) = \perp$.
 $eqv(g,h) := \forall x. [g(x) = h(x)]$
same($\lambda c. check(f,c), \lambda c. T$) – האם 2 תוכניות שעוצרות, אי פעם מחזירות אותו ערך עבור אותו x.
Halt(f,x,n) = not($eqv(\lambda n. stop(f,x,n), \lambda c. F)$) – האם 2 תוכניות שעוצרות, תמיד מחזירות אותו ערך.

halt(f,x)	$f(x) \neq \perp$	$p(x) = \perp$
halt ₀ (f)	$f() \neq \perp$	$\forall x. [g(x) = h(x)]$
halt*(f)	$\forall x. [f(x) \neq \perp]$	$check(f,c), \lambda c. T$
loop(f)	$\exists x. [f(x) = \perp]$	מחזירות אותו ערך עבור אותו x.
empty(f)	$\forall x. [f(x) = \perp]$	$Halt(f,x,n) = not(eqv(\lambda n. stop(f,x,n), \lambda c. F))$

- $TM_2 \leq TM$ [1 tape; 2 channels]
 - $CM_2 \leq TM_2$ [111...1BBB...]
 - $CM_n \leq CM_2$ [2^3 5^k 7^l ...]
 - $RAM \leq CM_n$ [2^k(2y+1)]
 - $Scheme \leq RAM$ [Abelson & Sussman]
 - $TM \leq Scheme$ [Interpreter]
- סקים $TM \leq CM$ – כותבים משערך. $CM \leq TM$ – קידוד מונה כרצף של אחדות.
 $RAM \leq CM$ – קידוד RAM כרצף של מספרים. $RAM \leq CM$ – שימוש במחסנית כדי לדמות מיני סקים.

NP – הרמז צריך להיות בגודל פולינומיאלי והידיוי גם צריך להיות בזמן פולינומיאלי (כל זה ב-TM).
טענה: אם יש TM עם הרבה סרטים הפותרת בעיה ב- $t(n)$, אם נגביל את מספר הסרטים לסרט אחד, זמן הריצה הוא לכל היותר $t^2(n)$ (במכונה עם הסרט היחיד).
טענה: הפרש בין מודלים "סבירים" הוא רק פולינומיאלי.
NP-Hard – בעיה שמכל בעיה אחרת ב-NP ניתן להגיע אליה ע"י טרנספורמציה פולינומיאלי.
NP-Complete – בעיה שהיא NP-קשה וגם שייכת ל-NP.



טענה: $G=(V,E)$ לא מכוון. נאמר ש- $S \subseteq V$ הוא IS אם m V/S הוא VC.
חשוב: אם תוכנית רצה בזמן שאינו תלוי בקלט שלה, אז היא פולינומיאלי!

CNF – דרך לכתוב נוסחאות. יש קבוצות ביטויים שמחוברות בקשר וגם ובתוך כל קבוצת ביטויים בין הליטרלים יש קשר או, הליטרל עצמו יכול להיות גם תחת שלילה.
CSP – בעיית הספיקות של נוסחאות עם או,וגם, שלילה ושווה. בהינתן נוסחא האם קיימת לה הצבה שעבורה התוצאה היא אמת. הבעיה ב-NP.
SAT – בעיית הספיקות של נוסחאות עם או,וגם,ושלילה. בהינתן נוסחא האם קיימת לה הצבה שעבורה התוצאה היא אמת. הבעיה ב-NP.
3SAT – בעיית הספיקות של נוסחאות מהצורה של 3CNF מה שאומר שהם מהצורה של הרבה קבוצות ביטויים שמחוברות בקשר וגם. כאשר כל קבוצה היא 3 ליטרלים מחוברים בקשר או. הליטרל עצמו יכול להיות גם תחת שלילה. הבעיה ב-NP.
2SAT – בעיית הספיקות של נוסחאות מהצורה של 2CNF מה שאומר שהם מהצורה של הרבה קבוצות ביטויים שמחוברות בקשר וגם. כאשר כל קבוצה היא 2 ליטרלים מחוברים בקשר או. הליטרל עצמו יכול להיות גם תחת שלילה. הבעיה ב-P.
QBF – בעיית הספיקות של נוסחאות עם או,וגם,שלילה, קיים ולכל. בהינתן נוסחא האם קיימת לה הצבה שעבורה התוצאה היא אמת. הבעיה ב-PSpace.

- רדוקציה רגילה:** $A \leq B$. בהינתן B חשיבה ניתן לתכנת בעזרה פונקציה חשיבה A ל- B .
 (1) אם $A \in R$ אז $B \in R$ אם (2) $A \notin R$ אז $B \notin R$.
רדוקציות מיפוי: $A \leq_m B$. קיימת פונקציה חשיבה f כך ש $x \in A$ אם $x \in B$ ו- $f(x) \in B$.
 (1) אם $A \leq_m B$ אז $A \leq_m B$ אם (2) $A \in RE$ אז $B \in RE$ אם $A \notin RE$ אז $B \notin RE$.
 (3) אם $A \in CoRE$ אז $B \in CoRE$ אם $A \notin CoRE$ אז $B \notin CoRE$.
 * קיום רדוקציות מיפוי גורר קיום של רדוקציה רגילה. קיומה של רדוקציה רגילה לא גורר קיום רדוקציות מיפוי.
רדוקציה פולינומיאלית: $A \leq_p B$. קיימת פונקציה חשיבה פולינומיאלית f כך ש $x \in A$ אם $x \in B$ ו- $f(x) \in B$.
 (1) אם $A \in P$ אז $B \in P$ אם $A \notin P$ אז $B \notin P$ אם (2) $A \in NP$ אז $B \in NP$ אם $A \notin NP$ אז $B \notin NP$.
משפט: אם $A \in NP - Hard$ אז $A \leq_p B$ ו- $B \in NP - Hard$.
משפט: אם $A \in NPC$ ו- $B \in NP$ אז $A \leq_p B$ ו- $B \in NPC$.

- אוטומט סופי דטרמיניסטי (DFA)**
 Q – קב' סופית של מצבים
 Σ – א"ב קלט סופי
 δ – פונקציות המעברים ($Q \times \Sigma \rightarrow Q$)
 q0 – מצב התחלתי
- אוטומט סופי לא-דטרמיניסטי (NFA)**
 כמו אוטומט דטרמיניסטי עם השינויים הבאים:
 Σc – א"ב קלט סופי (כולל ε)
 δ – פונקציות המעברים ($Q \times \Sigma_c \rightarrow P(Q)$)
 בפונ' המעבר כל מעבר הוא לקבוצת מעברים.

שפה היא רגולרית אם יש DFA שמקבל אותה. **משפט:** שפה רגולרית אם יש ביטוי רגולרי המתאר אותה.

הערות: (1) שפה סופית היא שפה רגולרית. (2) **באוטומטים** השפה $L(A) \neq \emptyset$ היא בעיה כריעה, כי ניתן לבדוק אם יש מצב מקבל ולוודא שקיים מסלול ממצב התחלה למצב המקבל.

הפכת אוטומט אי דטרמיניסטי לדטרמיניסטי (עולה 2ⁿ – עלות מערכת):

שלב 1: נפטרים מכל מעברי אפסילון. נניח שאני במצב A ואפסילון מעביר אותי למצב B וממצב B מעביר אותי למצב C. אז באוטומט ביניים שלי יהיה לי מעבר ממצב A למצב C ע"י s. כך נטפל בכל מעברי האפסילון באוטומט ונקבל אוטומט ביניים, כעת נעבור לשלב הבא.

שלב 2: מעבר לאוטומט דטרמיניסטי. כעת כל מצב יהיה לנו קבוצה של מצבים שאליה יהיה ניתן להגיע.

נניח שבאוטומט הביניים שלנו יש שני מעברי s ממצב A, אחד למצב B ואחד למצב C. אז כעת יהיה לנו באוטומט הדטרמיניסטי שלנו מעבר מ A ל B, C ע"י s. מכל קבוצת מצבים שיש לנו נבדוק מה המעברים האפשריים באוטומט הביניים וככה נתקדם. נעבור על כל הריצות האפשריות במקביל וכך נבנה לנו אוטומט דטרמיניסטי.
 מצבים מקבלים: מצבים שבקבוצת מצבים שלהם מכילים מצב שבמקור הוא מקבל.

למת הניפוח לרגולריות: לכל שפה רגולרית קיים מספר n (מס' המצבים באוטומט) כך שלכל מלה w המקיימת $|w| \geq n$ יש פירוק $w=xyz$ המקיים $|xy| \leq n-1$ ו- $|y| \geq 1$ מתקיים לכל i : $w^i = xyz^i yz^i$ ששייך לשפה.

שילת למת הניפוח לשפות רגולריות: (1) נניח בשלילה ש L רגולרית ו P קבוע הניפוח שלה. (2) נמצא $w \in L$ ונראה שלכל חלוקה $w=xyz$ קיים i כך ש $XY^iZ \notin L$ בסתירה ללמת הניפוח.

סגירות רגולריות: איחוד – ניתן להוסיף מצב חדש עם מעברי אפסילון. **שרשר** – חיבור מצבי הסיום של הראשון עם מצב התחלה של השני במעבר אפסילון. **משלים** – באוטומט דטרמיניסטי הנפוץ את כל המצבים המקבלים ללא מקבלים ולהפך. **חיתוך** – ע"פ דה-מורג (בהשתמש באיחוד ומשלים). בניית חיתוך עולה 2^m . יש בניה זולה יותר (וישירה) בגודל $m!$. **קלין** **סטאר *** – מוסיפים ממצבי הסיום מעבר אפסילון למצב התחלה חדש העובר למצב התחלה הישן באמצעות אפסילון. מצב ההתחלה החדש הוא מצב מקבל. **הפרש** – שפות רגולריות סגורות על הפרש.

שקילות שפות רגולריות: A, B אוטומטים. הבעיה $L(A)=L(B)$ היא בעיה כריעה כי $L(A)=L(B) \Leftrightarrow$

$$L(A) \cap \overline{L(B)} = \emptyset \Leftrightarrow L(B) \subseteq L(A) \wedge L(A) \subseteq L(B)$$

אוטומט מינימאלי: בעיית מציאת אוטומט מינימאלי לשפה רגולרית היא כריעה (ניתן לעבור על כל האוטומטים לפי גודל ולבדוק שקילות עד שלכל היותר מגיעים לאוטומט שעבורו מחפשים מינימאלי).

(* ניתן למצוא גם אוטומט מינימאלי בצורה יעילה ע"י מחיקת מצבים לא נחוצים (כאלו שלא ניתן להגיע ממצב התחלה אליהם או מהם למצב סיום) ולאחר מכן, זריקת מסלולים כפולים.
 (*) 2 אוטומטים מינימאליים לשפה הם זהים עד כדי שמות מצבים.

בשפות רגולריות הבעיות הבאות כריעות:

- **Membership** ($x \in L(A)$)
- **Emptiness** ($L(A) = \emptyset$)
- **Fullness** ($L(A) = \Sigma^*$)
- **Eqv** ($L(A) = L(B)$)
- **Subset** ($L(A) \subseteq L(B)$)

דקדוק חסר הקשר הוא רביעייה: (V, Σ, R, S) . V – קבוצת המשתנים (אותיות גדולות). Σ – קבוצת הטרמינלים (א"ב של המחורזת הסופית). R – קבוצה סופית של חוקי גזירה. S – המשתנה (מצב) התחלתי.

שפה חסרת הקשר: $L(G)=\{w \mid S \Rightarrow w\}$ (ניתן לגזור מ-S את w).

הצורה הנורמלית של חומסקי: (1) רק S (התחלת) יכול לעבור ל-ε. (2) S לא יופיע בצד ימין של גזירה. (3) מעבר לטרמינלים יתבצע לבד (לא בצמוד למשתנה). (4) כל גזירה תבוצע ל-2 משתנים (בדיוק) או טרמינל בודד אחד.

אופן הביצוע: (1) הוספת מצב חדש S0 שמפנה ל-S המקורי. (2) ביטול כל חוקי A->ε (כל חוק כזה חוזרים אחורה למי שמפנה ל-A ומוסיפים גזירה ללא A. (3) ביטול כל גזירות יחידות A->B (מבטלים את ההפניה ל-B, ובמקומה רושמים B-A את כל האפשרויות של B). (4) ביטול גזירות של 3 ומעלה. כל גזירה A->BCD הופכים לכללים חדשים: A->BA1 ו- A->CD A1. כמו כן, אם יש A->aB, אזי הנפוץ ל-A1B כאשר a-A1. כך יש בדיוק חוקים כפולים (גזירה ל-2 משתנים).

דקדוק לינארי A->x, A->aB לינארי ימני A->a, A->AB. משפחת השפות הנוצרות ע"י דקדוקים לינאריים ימניים/שמאליים היא משפחת השפות הרגולריות. נשתמש בדקדוק לינארי כדי לייצג אוטומט כדקדוק ח"ה: (1) נהפוך למשתנה גזירה ראשוני R0. (2) $R_i \rightarrow aR_j \Leftrightarrow \delta(q_i, a) = q_j$. (3) לכל מצב מסוים qi נגדיר $R_i \rightarrow \epsilon$.

סגירות ח"ה: ח"ה סגורות לאיחוד $(S_g \rightarrow S_A / S_B)$, קלין סטאר $(S \rightarrow \epsilon | SS)$, שרשר $(S_g \rightarrow S_A S_B)$, חיתוך עם שפה רגולרית, היפוך. **אינן סגורות** תחת חיתוך ומשלים.

הערה: כיוון שאין סגירות תחת המשלים, הבעיה "האם קיימת מילה שאינה שייכת לשפה" אינה כריעה.

למת הניפוח לח"ה: תהי L שפה ח"ה אזי קיים קבוע $p > 0$ כך שלכל מילה $w \in L$, $|w| \geq p$ יש פירוק

$w = uvxyz$, כך ש: (1) $|v| \geq 1$, (2) $|vxy| \leq p$, (3) לכל $i \in N$ $uv^i xy^i z \in L$.

הנחת הבסיס איתה מוכיחים את הلمה: אם מספר המשתנים הוא d ובכל צעד אנו יכולים להתפצל ל- c אפשרויות לכל היותר, הרי המילה הארוכה ביותר ללא חזרות היא לכל היותר c^d .

שילת למת הניפוח לח"ה: נניח בשלילה ש L ח"ה" וק קבוע הניפוח שלה. נמצא מילה w כך ש $w \in L$, $|w| \geq p$ ונראה שלכל חלוקה $w = uvxyz$, כך ש- $|v| \geq 1$, $|vxy| \leq p$, קיים i כך ש $uv^i xy^i z \notin L$.

בשפות ח"ה הבעיות הבאות כריעות: **Membership** ($x \in L(G)$). **Emptiness** ($L(G) = \emptyset$).
האם המילה שייכת לשפה של דקדוק – הפוך לצורת חומסקי (ואז אין מחיקות) ובדוק את כל אפשרויות הגזירה עד האורך של המילה הספציפית $|x|$.

האם יש דקדוק שלא נותן כלום – נמחק משתנים מיותרים (לפי חומסקי) ואם קיבלנו שמשנתה ההתחלה מיותר אז השפה ריקה (אפשר גם למצוא חסם למילה הקצרה ביותר לפי למת הניפוח).

בשפות ח"ה הבעיה הבאה לא כריעה: **Fullness** ($L(G) = \Sigma^*$).

פישוט דקדוקים: (1) סילוק משתנים שלא ניתן להגיע מהם למילה טרמינלית. (2) סילוק סימנים שלא ניתן להגיע אליהם ממשנתה ההתחלה.

א17 דוגמאות - האם בעיות P- או B-NP

1. האם בגרף קיים גם **click** וגם **Independent Set (IS - קב' בת"ל)** בגודל k כל אחד? הבעיה ב-**NPC**. עושים רדוקציה מ-**click** אחר. מוסיפים לגרף k קודקודים שאינם קשורים לכלום, ובהם בהכרח ה-**IS**. בגרף החדש יש **click** אם ורק-אם היה במקור **click**.

2. **בהינתן גרף, מספר k ושני קודקודים s ו- t - האם יש $click$ בגודל k והאם יש IS בגודל k של שכני t .** הבעיה ב-**NPC**.
עושים רדוקציה מ-**click**. יוצרים צומת חדש s' ומחליפים בין s לבין s' (כל הקשתות s -מנותקים ומעבירים ל- s'). עכשיו מחברים את s לכל הצמתים האחרים בגרף. בנוסף, מחברים ל- t k צמתים חדשים שמחוברים בקשת רק ל- t . מאכן כי בהכרח יש **IS** בגודל k שכנים של t , והיה **click** כנדרש אם ורק-אם היה במקור **click** שהוא (כי כולם עכשיו שכנים של s , ובגודל הגרף המקורי לא שונה כלל).

3. **האם בגרף קיים $click$ וגם IS בגודל $n/2+1$ כ"א?** הבעיה ב-**P**. אם יש מספר זוגי של צמתים זה בלתי-אפשרי. אם יש מס' אי-זוגי זה אפשרי אבל אז זה בדיוק $n/2+1$ בכל קב'. נספור כמה קשתות יש לכל צומת ואת כל אלו עם יותר מ- $n/2+1$ נשים בקבוצה אחת ואת השאר בשניה. אם קיימים קליק ו-**IS** נדרש, אז זו החלוקה ביניהם, ובדיקה האם בגרף בעל d צמתים יש קליק בגודל d היא קלה ופולי כי צריך לבדוק שבדיוק כולם מחוברים לכולם. כל התהליך פולי ולכן הכל פולי. משהו כזה, צריך לשפצר מעט בהוכחה.

4. **האם בגרף יש צומת מדרגה $lg_2(n)$ לפחות או קליק בגודל m ?** הבעיה ב-**P**. לבדוק דרגת צומת זה פשוט. אם אין, אז הקליק המקסי' הוא פחות מ- $lg_2(n)$. לכן אם m גדול מזה – אין. אם m קטן מזה אז צריך לבדוק את כל האפשרויות. יש בסה"כ $lg_2(n)$ מעל m אפשרויות, וזה יוצר $O(n)$. כל מס' פולי של בדיקות וכל בדיקה היא בזמן פולי ולכן ב-**P**.
$$\binom{\log_2 n}{m} = O(2^{\log_2 n}) = O(n)$$

5. **האם בגרף יש שני קליקים זרים בגודל k .** בעיה ב-**NPC**. רדוקציה מקליק. לוקחים את הגרף המקורי ומשכפלים אותו (ללא קשר בין שני החלקים). אם במקור היה קליק אז עכשיו יש שניים ואחרת לא.

6. **האם בגרף יש קליק בגודל k או קב' בת"ל בגודל k ?** הבעיה ב-**NPC**.
רדוקציה מקליק. מוסיפים לגרף עוד $|V|$ צמתים ומקשרים אותם לכולם (כולל עצמם). עכשיו מחפשים בגרף החדש את הבעיה עבור $k+|V|$. בהכרח אין **IS** בגודל הזה כי קישורנו את הצמתים החדשים לכל הצמתים האחרים. לגבי קליק, אם היה במקור בגודל k אז עכשיו יהיה בגודל $k+|V|$ כי הוספנו קליק $|V|$ וחיברנו לכולם. זהו.

7. **האם בגרף דו"צ יש IS בגודל k - הבעיה ב- P .** לוקחים צד אחד, מוסיפים לכמות את כל מה שבצד שני ולא מחובר לא. אם לא טוב, עושים כ"ל רק עם הצד השני. אם גם עכשיו לא טוב אז אין, ואחרת ממצאן בדרך.

8. **האם לגרף יש תת-גרף מושרה דו"ל עם k קודקודים בכל צד.** בעיה ב-**NPC**. רדוקציה מבעיית **IS** לקב' בגודל k . מוסיפים קב' קודקודים B בגודל k ומחברים בקשתות לכל הקודקודים במקור. אם במקור היה קב' בת"ל A בגודל k אז עכשיו קב' דו"צ עם B (אין קשתות בין חברי A כי הוא **IS** ואין בין B -כי בבניה הגדרנו שאין). אם בגרף החדש יש גרף-מושרה דו"צ בגודל k אז לפחות צד אחד של הגרף הוא צמתים מהגרף המקורי, וזה אומר שהיה בו **IS** בגודל k .

9. **נתונה קב' S של אברים H -אוסף תתי-קבוצות לא ריקות. האם יש קב' T חלקית ל- S כל שכל תתי-קבוצה H -מכילה לפחות אבר אחד T -** הבעיה ב-**NPC**. רדוקציה מבעיית **Vertex Cover**. הרדוקציה לוקחת את הגרף ומגדירה את S להיות קבוצת הקודקודים וכל תתי-קבוצה להיות קשת (ככה שהקבוצות H -הן כל הקשתות האפשריות). אם בגרף המקורי יש **vertex-cover** בגודל k אז זה בדיוק קב' t בגודל k כך שלכל תתי-קב' H -יש אבר אחד ב- t . וצד שני – אם יש קב' t כזו, אז זה אומר שיש **vertex-cover**.

10. **כ"ל עבור קב' H בגודל $n-5$.** הבעיה ב-**P**. בגלל הגודל סך כל תתי-הקב' האפשריות הוא פולינומי (n מעל $n-5$ או מעל 5). לכן אפשר לעבור על כל הקבוצות, לספור מופעים של כל אבר, ובכל פעם לקחת את האבר המקסימלי

ולמחוק את הקבוצות שממנו הגיע. נפסיק כשיגמרו כל הקבוצות, וזה גודל הקב' t המינימלי. נשאר רק לבדוק האם k גדול-שווה מהמספר המינימלי הזה.

נוספים:

$Np = Co_np$ - אמ"מ קיימת בעיה ב-**NPC** שהיא גם **Co_np** -
 $halt \in NP_hard$ אבל לא ל-**NPC**.

בעיות חשובות

PATH – בהינתן גרף מכוון ושני קודקודים – האם יש מסלול בין שני הקודקודים (אפשר שלא חוזר באותו צומת פעמיים).
RELPRIME – שני מספרים יוגדרו **RELPRIME (relative primality)** אם המכנה המשותף המקסימלי שלהם הוא 1.
SUBSET-SUM – נתונה קבוצת מספרים ומספר יעד. האם יש שילוב של המספרים שחיבורם נותן את מספר היעד. הבעיה היא **NPC**.

הבעיות הבאות הן בעיות NP-Complete

HAMPATH = $\{G, s, t \mid G \text{ has Hamilton path from } s \text{ to } t\}$ האם בגרף יש מסלול בין s ל- t שגם עובר בכל הקודקודים וגם לא עובר באף קודקוד פעמיים.

HAMCIRCUIT = $\{G \mid G \text{ has Hamilton circuit}\}$ בהינתן גרף מכוון רוצים לדעת האם יש מעגל שעובר בכל קודקודי הגרף וגם לא עובר באף קודקוד פעמיים. הערת: **hamcircuit** מוגדר לגרף מכוון אבל הוכח בתרגול כי הוא נכון גם לגרפים לא מכוונים. בעקרון גם **hampath** לא מכוון הוא **NPC**, אבל זה לא הוכח בתרגול, ובעיקרון צריך להוכיח...
CLIQUE – האם בגרף יש תת-גרף קשיר לחלוטין (שבו כל הקודקודים מחוברים לכל שאר הקודקודים בתת-הגרף).

K-CLIQUE – לרוב זו תהיה השאלה, והכוונה להאם יש **clique** עם k קודקודים.
SAT – בהינתן ביטוי בוליאני, האם יש לו השמה מספקת.

3SAT – כ"ל, כאשר הביטוי הבוליאני הוא מהצורה הבאה: מורכב מביטויים שקשורים ביניהם בקשר "וגם" (\wedge), כאשר בכל ביטוי יש 3 ליטרלים הקשורים ביניהם בקשר "או" (\vee).

INDEPENDENT-SET – נתון גרף ושואלים האם יש תת-קבוצה של קודקודי הגרף, המורכבת מקודקודים שאין ביניהן קשתות המקשרות ביניהן באופן ישיר (כלומר בין כל 2 קודקודים בתת-הקבוצה) אין קשת בגרף המקורי.
Vertex-Cover – נתון גרף ומספר k ושאלת השאלה האם ניתן לכסות את הגרף ע"י k קודקודים. כיסוי הוא תת-קבוצה של הקודקודים כך שלכל קשת בגרף אחד הקודקודים (לפחות) של הקשת יהיה בתת-הקבוצה. כלומר:
 $\forall (a,b) \in E: a \in V' \text{ or } b \in V'$

K-coloring – האם ניתן לצבוע את קודקודי הגרף ב- k צבעים. צביעה חוקית היא כזו שלקודקוד יש צבע שונה מכל שכניו (הקודקודים הקשורים אליו בקשת) – במילים אחרות זה לכל קשת (u,v) אז u ו- v יש צבע שונה.
דוגמת חשובה: למקרה שרוצים רדוקציות מ-**3SAT** לבעיות אחרות שקשורות לנוסחאות **SAT** או **CNF** אז כדאי לבצע העברות כאלו: לוקחים נוסחה מהצורה $(y_1 \vee y_2 \vee y_3)$ ומעבירים למשהו מהצורה הבאה: $(y_1 \vee y_2 \vee z_1) \wedge (\bar{z}_1 \vee y_3 \vee b)$ (למשל רדוקציה מ-**3SAT** ל-**3SAT** שבה בכל ביטוי מספק יש לכל היותר שני ליטרלים מאותו ערך ושלישי ערך שונה).

א17 דוגמאות – בעיות הכרעה

קלט: תוכנית p . שאלה: האם קיים x עבורו p עוצרת? זוהי שפה הכריעה ולמחצה (**RE/R-** שייכת ל-**RE**).
הוכחת שייכות ל-RE: נראה אלגוריתם המשמש בשיטת השכלול באמצעות סטפ. יהי $s1...si$ קלטם אפשריים עבור p . נוזח על הקלטים באופן הבא: עבור כל $i \in \mathbb{N}$ נוזח על הקלטים $s1...si$ למשך i צעדים. במידה ועבור אחד הקלטים הסטפ עצר באופן טבעי, נחזיר T ונסיים את הבדיקה, אחרת נמשיך הלאה.

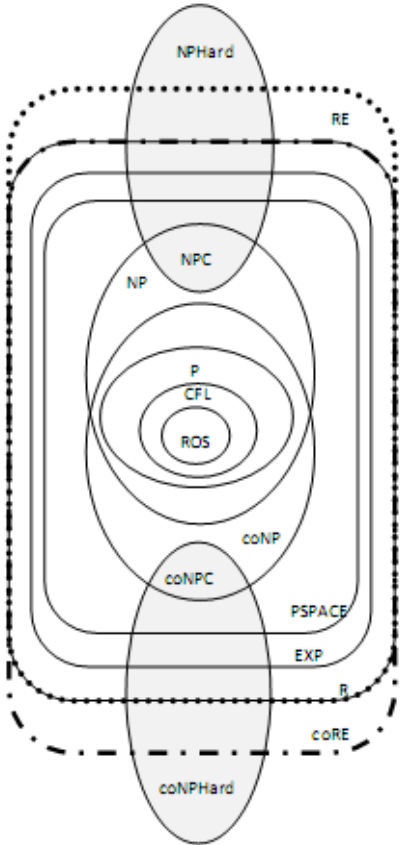
הוכחת אי כריעות: נניח בשלילה שהשפה P כריעה ונראה רדוקציה **half**. כלומר קיים פרויקט **stopsForSomeX** המכריע את השפה (מחזיר T אם בשפה קיים x שעבורה התוכנית עוצרת F -אחרת).
 $Halt(p,x) := stopsForSomeX(g) \text{ where } g := \lambda z. \text{if } z=x \text{ then return } p(x) \text{ else return } loopy()$

קלט: תוכנית p . שאלה: האם כל מספר זוגי הגדול מ-2, ניתן להציג כסכום של 2 מספרים ראשוניים? **פתרון: R**;
הנחת גולדבאך אומנם עדיין לא הוכחה או הפוכה אך באופן כללי לשאלה זו יש תשובה חיידית T או F . לכן שפה זו היא שפה טריוויאלית: אין תלות בקולט p ובהכרח ניתן להכריע אותה ע"י התוכנית $\lambda p, x. T$ או $\lambda p, x. F$. נשים לב, שאין צורך שנדע את התשובה לשפה טריוויאלית, כמו בדוגמה של שפת חברי הכנסת ה-18 שטרם הוכרעה רשמית (נכון לכתובת שורות אלו) כפי שהוצגה בכיתה.

קלט: תוכנית p ו-2 משתנים x ו- y . שאלה: האם p עוצרת בדיוק על 1 משני המשתנים? **פתרון:** אף קבוצה.
הוכחת אי שייכות ל-RE: באמצעות רדוקציית מיפוי. נניח בשלילה שקיים $haltsOnOne(p,x,y)$ ונראה רדוקציה.
 $Halt(p,x) := haltsOnOne(f(p,x)) \text{ where } f := \lambda p, x. \langle g, x, \epsilon \rangle \quad g := \lambda z. \text{if } (z=x) \text{ then return } p(x) \text{ else return } loopy()$

רדוקציה מ SAT3 ל Clique: בהינתן נוסחאת CNF3. ניצור גרף שבו כל מופע של ליטרל (משתנה או שלילתו) מייצג קודקוד. הקשתות יהיו בין כל קודקוד לכל הקודקודים שלא מייצגים את הליטרלים שבשלושה שלו ובנוסף הם לא השלילה שלו. לדוגמא $(x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee w)$ אז לקודקוד שמייצג את x יהיו קשתות לקודקודים של w ו של השלושה השנייה. הנוסחא ספיקה-היהיה לנו קליק בגודל של k כאשר k זה מספר השלושות בנוסחא. הנוסחא לא ספיקה-לא יהיה קליק כזה.

יחסי הכלה של הקבוצות השונות

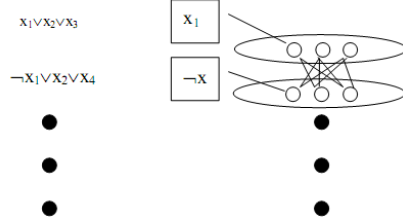


שאלה מ מבחן: נתבונן בשפה הבאה:
 $3\text{-SAT-C} = \{\varphi \mid \varphi \text{ is a 3-CNF formula and each variable appears exactly } C \text{ times in } \varphi \text{ and } \varphi \text{ is satisfied}\}$
 כלומר כל משתנה מופיע בדיוק C פעמים בנוסחא (הופעה של משתנה x היא x או שלילתו $\neg x$).
 ידוע כי 3-SAT-7 (כלומר נוסחא 3CNF כאשר כל משתנה מופיע בדיוק 7 פעמים) שלמה ב NP . הראו כי קיים קבוע $D > 0$ שעבורו הבעיה הבאה שלמה ב- NP :

$D\text{-IS} = \{(G, k) \mid D \leq k \leq \text{דרגת כל קודקוד ב } G \text{ היא } \geq D\}$

שנלמדה בכיתה. $Clique$ רמז: העזר ברדוקציה ל NP - קל לוודא שאכן דרגת כל קודקוד היא לכל היותר $D=8$ ובהינתן קבוצת קודקודים בגודל לפחות k ניתן לבדוק אם היא ב"ת. נראה קושי ל NP ע"י רדוקציה $3\text{-SAT-}7 \leq_p 8\text{-IS}$. למעשה נחלק את הרדוקציה לשני חלקים: $Clique(1) \leq_p 3\text{-SAT-}7 \leq_p (n-8)$ כאשר יש קליקה בגודל $k \leq 8$ ודרגת כל קודקוד ב G היא $< n-8$ | $Clique(2) = (G, k) \leq_p (n-8) - Clique$ כאשר n הוא מספר הקודקודים ב G .

$3\text{-SAT-}7 \leq_p (n-8) - Clique$: הרדוקציה זהה לזו שראינו בכיתה (1) $Clique(3)$. הרדוקציה הופכת כל הסגר לשלושה קודקודים ב"ת (אין קשת בין כל אחד מהקודקודים) וקיימת קשת בין שני קודקודים x, y שמקורם בהסגרים שונים אם $x \neq y$.



נשים לב שאם כל משתנה מופיע בדיוק 7 פעמים אזי לכל קודקוד יהיו יותר מ $n-8$ שכנים שכן 2 הקודקודים שהם מאותו הסגר הם ב"ת - למשל בדוגמא ל x_1 לא נחבר את הקודקוד x_2 ולא את x_3 . וכיוון שכל משתנה מופיע בדיוק 7 פעמים אזי יהיו לכל היותר 6 קודקודים סותרים ($y = \neg x$) שלא תהיה אליהם קשת. הנוכחות נובעת מהרדוקציה שהראנו בכיתה.

2. $(n-8) - Clique \leq_p 8\text{-IS}$. ניקח את הגרף המשלים (הוכח בכיתה). כיוון שדרגת כל קודקוד בגרף המקורי היא $< n-8$ אזי דרגתו ב גרף המשלים היא לכל היותר 8.

*הערה: קל לבדוק אם בהינתן נוסחא כל משתנה מופיע 7 פעמים ואם לא הרדוקציה תייצר גרף למשל משולש ותדרוש $k=4$. (לא הורדו נקודות למי שלא התייחס לנקודה זו).

הוכחת אי שייכות RE/R: נוכיח באמצעות רדוקציית מיפוי.

$$\overline{halt}(p, x) := haltsOnOne(f(p, x)) \text{ where } f := \lambda p, x. \langle g, x, \epsilon \rangle \quad g := \lambda z. \text{if } (z = x) \text{ then return } p(x) \text{ else return } T$$

קלט: תוכנית p . **שאלה:** $|L(p)| > 3$? **פתרון:** נראה שזוהי שפה הכריעה למחצה (שייכת ל-RE/R).

הוכחת שייכות RE/R: נראה אלגוריתם המשתמש בשיטת השבלול באמצעות סטפר. יהי s_1, s_2, \dots, s_i קלטים אפשריים עבור p . נרוץ על הקלטים באופן הבא: עבור כל $i \in \mathbb{N}$ נרוץ על הקלטים s_1, \dots, s_i למשך i צעדים. בכל פעם שעבור אחד הקלטים הסטפר עצר באופן טבעי לראשונה, נקדם מונה ב-1. אם המונה יגיע ל-4 נחזיר T נסיים את הבדיקה, אחרת נמשיך הלאה.

הוכחת אי שייכות R: נניח בשלילה שהשפה p כריעה ונראה רדוקציה $halts$. כלומר קיים פרידיקט $moreThan3$ המכריע את השפה (מחזיר T אם בשפה יש יותר מ-3 קלטים התוכנית p עוצרת, ו- F אחרת).

$$Halt(p, x) := moreThan3(g) \text{ where } g := \lambda z. \text{if } z = x \text{ then return } p(x) \text{ else if } z = next(x) \text{ then } T \text{ else if } z = next(next(x)) \text{ then } T \text{ else if } z = next(next(next(x))) \text{ then } T \text{ else return loopy()}$$

קלט: תוכנית p . **שאלה:** $|L(p)| \leq 3$? **פתרון:** זוהי שפה שהמשלים שלה היא שפה הכריעה למחצה (שייכת ל-Co-RE/R). **הוכחת שייכות Co-RE/R:** נשים לב כי המשלים הוא $|L(p)| > 3$. כפי שהוכחנו בסעיף הקודם, המשלים שייך ל-RE/R ואינו שייך ל-R. לכן, באופן מידי (ע"פ מה שהוכחנו בכיתה) נסיק כי שפה זו שייכת ל-Co-RE/R.

הוכחת אי שייכות R: נניח בשלילה שהשפה p כריעה. מכאן שגם המשלים של השפה שייך ל-R אך בסעיף הקודם הראנו כי הוא אינו שייך ל-R. קיבלנו סתירה ולכן השפה אינה שייכת ל-R.

קלט: פרידיקט P . **שאלה:** האם $L(P)$ היא כריעה? **פתרון:** שפה זו אינה שייכת לאף אחת מהקבוצות $R, RE/R, co-RE/R$. **הוכחת אי שייכות Co-RE/R:** נוכיח באמצעות רדוקציית מיפוי $halts$: נניח שהשפה p כריעה ומוכרעת ע"י הפרידיקט Q . נסמן את הפרידיקט של השפה $halts$ ב- P (שפת התוכניות שעוצרות עבור הערך אפסילון).

$$halt(p, x) := Q(g(p, x)) \text{ where } g := \lambda p, x. \lambda z. P \mathcal{E}(z) \vee f(z) \quad f := \lambda z. \begin{cases} T & p(x) = p(x) \\ loopy(x) & o.w. \end{cases}$$

הוכחת אי שייכות RE/R:

נסמן את השפה L_R . בכיתה הראנו והוכחנו כי קיימת רדוקציית מיפוי מ- $halts$ לשפה המשלימה L_R . כלומר מתקיים $halts \leq_m L_R$ ומכאן (ע"פ מה שהוכחנו בכיתה) גם מתקיים $halts \leq_m L_R \Rightarrow halt \leq_m L_R$. היות ו- $halts$ אינה כריעה ואינה שייכת ל-

RE/R נסיק כי גם L_R אינה שייכת כריעה ואינה שייכת ל- RE/R .

קלט: פרידיקט p . **שאלה:** האם $L(P)$ מוכרעת למחצה? **פתרון:** זוהי שפה כריעה (שייכת ל-R). **הוכחה:** ע"פ ההגדרה, כל שפה $L(p)$ מוכרעת למחצה ע"י הפרידיקט שלה p . כלומר, השפה $L(P)$ מוגדרת להיות כל הערכים להם P מחזיר אמת. לכן, השפה $L(p)$ מעצם ההגדרה מוכרעת למחצה ע"י P . מכאן ששאלה זו היא שאלה טריוויאלית ומוכרעת ע"י הפרידיקט $\lambda x. T$. הראנו כי קיים פרידיקט המכריע את השפה ולכן שפה זו שייכת ל-R.

קלט: תוכנית p ללא קלט, שמקיימת $|p| < bb(1000)$. **שאלה:** האם p עוצרת? **פתרון:** זוהי שפה כריעה (שייכת ל-R). **הוכחה:** מספר התוכניות ללא קלט שמקיימות את התנאי $|p| < bb(1000)$ הוא מספר סופי. לכן, קיימים מספר סופי של שאלות שניתן לבצע על הבעיה. ע"פ מה שנלמד בכיתה, כל בעיה עם מספר סופי של שאליות היא בעיה כריעה (בדומה לבעיית חברי הכנסת ה-18). לכן, זוהי שפה כריעה. **הערה:** נשים לב, שמשפט רייס אינו תופס במקרה זה כיוון שהבעיה אינה סמנטית. תהי תוכנית עוצרת באורך $bb(1000)$. התוכנית $g = \lambda x. f(x)$ שקולה לתוכנית f היות והערכים שהיא מחזירה זהים לערכים של f . עם זאת, ברור שהאורך של g גדול מ- $bb(1000)$. לכן, למרות שהתוכניות שקולות סמנטית, הבעיה תחזיר עבור Fg כי היא אינה שייכת לשפה.

18 הערות כלליות

* יהי E ביטוי רגולרי ו- G דקדוק ח"ה, הבעיה $L(G) \subseteq L(E)$ היא Ra והבעיה $L(E) \subseteq L(G)$ היא $co-RE$.
 * במכונת טיורינג כאשר מספר הצעדים הוא סופי, מספר הקלטים חסום עבור א"ב קבוע. $halt^*$ היא ב- $NP-Hard$.
 * רדוקציה ממבחן:

$$L = \{ \langle M \rangle \mid SAT \leq_p L(M) \}$$

$$\overline{Halt}(f, x) = P(g(f, x)); g(f, x) := \lambda y. \begin{cases} F & S(f, x, |y|) \text{ עצר טבעית} \\ T & O/W \end{cases}$$