Neta Roth: https://github.com/neta-r
Almog Jakov: https://github.com/AlmogJakov

חישוביות – דף נוסחאות

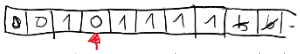
הגדרה פורמלית של מ"ט:

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

Q - קבוצת מצבים סופית לא ריקה.
Σ - א"ב קלט.
Γ – א"ב סרט (כולל את Σ ואת ƀ).
q_0- מצב התחלתי.
δ – פוני המעברים - מקבלת מצב ותו שאותו הראש קורא - עוברת למצב חדש, כותבת תו אחר במקום וזזה עם הראש ימינה או שמאלה.
q_{acc}- מצב מקבל, q_{rej}- מצב דוחה.
אם המכונה לא מגיעה למצבי עצירה זה היא לא עוצרת.

קונפיגורציה: תיאור מצב נוכחי של המכונה: (u,q,v) כאשר:
u- המילה הכתובה על הסרט עד מיקום הראש הקורא כותב
q- מצב נוכחי
v- המשך הסרט (מהמיקום של הראש)
לדוג':

$001q_801111$



* אם מגבילים את המכונה לא לעבור תא כלשהו אז ניתן לזהות מתי יש לולאה אינסופית (בעיית BHP).
אפשר להגדיר טיפה אחרת כמו שהגדירו בבעיית BHP.

מודלים שקולים:
1. מ"ט מרובה סרטים
2. מ"ט אי-דטרמיניסטית (לא שקולה בסיבוכיות!)
3. מ"ט עם מצב $stay$

מכונות שמחשבות פונקציות:
הפלט שלה הוא מה שכתוב על הסרט משמאל לראש קורא כותב (לא כולל התו שהוא רואה). הפלט מחושב כשהפונקציה עוצרת במצב עצירה.
*אם המכונה לא עוצרת על x אז הפלט לא מוגדר.
*פונקציה שלא מוגדרת על כל הקלטים נקראת פונקציה **לא מלאה**.
*פונקציה שמחזירה תוצאה לכל $x\in\Sigma^*$ נקראת פונקציה **מלאה**.

R – מחלקת השפות הכריעות:
שפה L כריעה היא שפה שיש מ"ט שמקבלת כל $x\in L$ ודוחה כל $x\notin L$. (בפרט עוצרת על כל קלט).
שפות מכריעות:
- כל השפות הסופיות.
- כל השפות הרגולריות וחסרות ההקשר.
- כל שפה שניתן לבנות לה תוכנית מחשב (פונקציה בוליאנית) שתמיד עוצרת.

סגירות: איחוד, חיתוך, הפרש, משלים, שרשור, חזקה, היפוך, איטרציה. אין סגירות להכלה.

RE – מחלקת השפות המקבלות:
שפה L כריעה היא שפה שיש מ"ט שמקבלת כל $x\in L$ ולא מקבלת כל $x\notin L$. (לא בהכרח עוצרת על קלט לא בשפה).
שפות מקבלות:
- כל מה שבR.
- שפות נוספות שקל לוודא שקלט אכן נמצא בשפה.

סגירות: איחוד, חיתוך, שרשור, איטרציה, היפוך, חזקה. אין סגירות להכלה, אין סגירות למשלים ואין סגירות להפרש!

:coRE
הגדרה פורמלית: $coRE=\{L|\bar{L}\in RE\}$.
הגדרה שקולה: היא קבוצת השפות שיש עבורן מכונה M כך ש:
-אם המכונה עוצרת, היא עונה נכון
-המכונה חייבת לעצור על כל קלט שלא בשפה.
לא בהכרח מתקיים $L(M)=L$.

תכונות של המחלקות:
- $R = RE\cap coRE$
- אם $L\in RE$ אז $\bar{L}\in coRE$
- $R\subseteq coRE, R\subseteq R$

שפות ללא מ"ט:
יש א שפות שונות בעולם (גודל קבוצת החזקה של Σ^*)
יש $\aleph_0$ מ"ט שונות (כגודל Σ^*)
יש יותר שפות ממ"ט-ים לכן יש שפות שלא ניתן להתאים להן מ"ט.
$|R|=|RE|=|coRE|=\aleph_0$
לכן רוב השפות הן לא ב$R,RE,coRE$.

הוכחת נכונות של מכונה
$x\in L \Rightarrow\cdots\Rightarrow x\in L(M)$
$x\notin L\Rightarrow\cdots\Rightarrow x\notin L(M)$

שפות מרכזיות
$L_u=\{<M,x>|M \text{ accepts } x\}\in RE\backslash R$
$HP=\{<M,x>|M \text{ halts on } x\}\in RE\backslash R$
$L_D=\{<M>|M \text{ accepts} <M>\}\in R\backslash RE$
$\overline{HP},\overline{L_u},\overline{L_D}\in coRE\backslash R$
$L_\infty=\{<M>||L(M)|=\infty\}\in\overline{RE\cup coRE}$
$L_{\Sigma^*}=\{<M>|L(M)=\Sigma^*\}\in\overline{RE\cup coRE}$
$L_{eq}=\{<M_1,M_2>|L(M_1)=L(M_2)\}\in\overline{RE\cup coRE}$

אינטואיציה מתי שפה שייכת ל$R,RE,coRE$:
דברים שפוסלים שייכות לRE:
- סימלוץ מכונה שלא ידוע אם תעצור
- מעבר על אינסוף מילים
באופן כללי: אם ניתן לוודא שקלט בשפה - RE,
* אם ניתן לוודא שקלט לא בשפה - $coRE$,
* אם ניתן לוודא את שניהם - R.
* תנאי טריוויאלי (תמיד מתקיים או תמיד לא מתקיים) - R.
* שפה סופית - R.

סימלוץ מכונה:
כל מ"ט יכולה להיות מקודדת כמחרוזת, נסמן - $<M>$.
מ"ט אוניברסלית היא מ"ט שמקבלת כקלט קידוד של מכונה אחרת ומסמלצת את ריצתה.

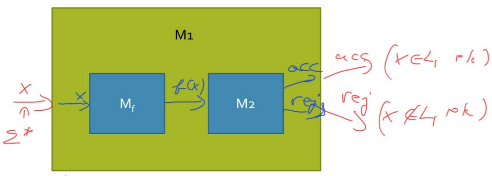
הרצה מבוקרת:
לכל $i\in\mathbb{N}^+$:
1. נריץ את M על x במשך i צעדים
2. .....

הרצה מבוקרת חזקה:
כשרוצים להריץ כל פעם מספר סופי של מילים מספר סופי של צעדים:
לכל $i\in\mathbb{N}^+$:
לכל $0<j\le i$:
הרץ את M על w_j במשך i צעדים

רדוקציות:
נאמר ש $L_1\le L_2$ (L_1 ניתנת לרדוקציה לL_2) אמ"ם:
קיימת $f:\Sigma^*\to\Sigma^*$ שמקבלת קלט לL_1 ומחזירה קלט לL_2
כך ש f מלאה (מוגדרת לכל קלט)
ניתנת לחישוב (יש M_f – מ"ט שמחשבת את הפונקציה הזו)
ותקפה כלומר מתקיים $\forall x\in\Sigma^*: x\in L_1\Leftrightarrow f(x)\in L_2$.

משפט הרדוקציה:
$L_1\in R \Leftarrow L_2\in R$, ובאותו אופן: $L_2\notin R\Leftarrow L_1\notin R$
$L_1\in RE \Leftarrow L_2\in RE$, ובאותו אופן: $L_2\notin RE\Leftarrow L_1\notin RE$
$L_1\in coRE \Leftarrow L_2\in coRE$, ו: $L_2\notin coRE\Leftarrow L_1\notin coRE$

הוכחה 1: נניח $L_1\le L_2$ ו$L_2\in R$ אז קיימת מ"ט M_f שמחשבת את פונקציית הרדוקציה, וקיימת מ"ט M_2 שמכריעה את L_2.
נתאר את M_1 מכונה מכריעה לL_1:
M_1 על קלט x:
1. מחשבת את $f(x)$ ע"י סימלוץ M_f על הקלט x. (שלב זה יסתיים כי M_f מלאה וניתנת לחישוב).
2. מריצה את M_2 על $f(x)$ ועונה כמוה.



תכונות של רדוקציות:
1. $L\le L$
2. אם $L_1\le L_2$ וגם $L_2\le L_3$ אז $L_1\le L_3$
3. אם $L_1\le L_2$ אז $\overline{L_1}\le\overline{L_2}$ (אותו פונקציה אפילו)
4. אם $L_1\le L_2$ **לא בהכרח** $L_2\le L_1$
5. $\forall L\in R$: $L\le L'$ כאשר L' היא סתם שפה שאינה טריוויאלית.
(תכונה 5 מתקיימת גם עבור $L\in P$)

מבנה הוכחת תקפות:
$x\in L_1\Rightarrow f(x)\in L_2$, $x\notin L_1\Rightarrow f(x)\notin L_2$

דוגמאות לרדוקציות:
1. רדוקציית "הכל או כלום": לרוב בשביל אי שייכות לR או אי שייכות ל$coRE$
לדוגמה השפה:
$L=\{<M>: |L(M)|>3\}$
נרצה להראות $L\notin R$, נרצה רדוקציה $HP\le L$ ע"י:
$f(<M,x>)=<M'>$
אינטואיציה: אם M עוצרת על x אז הגודל של M' גדול מ3.
אם M לא עוצרת על x אז הגודל של M' קטן שווה 3.
M' על קלט w:
1. הרץ את M על x // אם M לא עוצרת לא מקבלים כלום
2. קבל // מקבלים הכל
הרדוקציה מלאה וניתנת לחישוב - כתיבת קידוד של מכונה חוקית שמריצה מכונה על מילה ומקבלת.
הרדוקציה תקפה –
$<M,x>\in HP\Rightarrow M \text{ halts on } x\Rightarrow M' \text{ accepts all}$
$\Rightarrow L(M')=\Sigma^*\Rightarrow|L(M')|>3\Rightarrow M'\in L$

$<M,x>\notin HP\Rightarrow M \text{ doesn't halt on } x$
$\Rightarrow M' \text{ stuck in a loop for each input}$
$\Rightarrow L(M')=\Phi\Rightarrow|L(M')|=0\Rightarrow M'\notin L$

הערה: הרדוקציה לא מוכיחה כי $L\in RE$! צריך לעשות מ"ט מקבל

2. רדוקציה חשובה נוספת:
נרצה להראות $L\notin R$, נרצה רדוקציה $L_D\le HP$ ע"י:
$f(<M>)=<M',<M>>$

M' על קלט y:
1. הרץ את M על y
2. אם M קיבלה – קבל, אחרת- כנס ללולאה אינסופית

$<M>\in L_D\Rightarrow M \text{ accepts} <M>$
$\Rightarrow M' \text{ accepts} <M> \text{ and } \textbf{stoped} \text{ on it}$
$\Rightarrow<M',<M>\gg\in HP$

$<M>\notin l_d\Rightarrow M \text{ reject or loop on} <M>$
$\Rightarrow M' \text{ stuck in a loop on} <M> \text{ and}$
$\textbf{didn't stop} \text{ on it}\Rightarrow<M',<M>\gg\notin HP$

3. רדוקציה חשובה נוספת: לרוב בשביל אי שייכות לRE
נרצה להראות $L\notin R$, נרצה רדוקציה $\overline{HP}\le L_\infty$ ע"י:
$f(<M,x>)=<M'_x>$

M'_x על קלט y:
1. הרץ את M על x במשך $|y|$ צעדים
2. אם M לא עצרה בשלב 1- קבל
3. אחרת (M עצרה)- דחה

חשיבות: תמיד עוצרת!
משתמשים כשרוצים שאי-עצירה תגרור קבלת אינסוף מילים

$<M,x>\in\overline{HP}\Rightarrow M \text{ doesn't halt on } x$
$\Rightarrow \text{for each } x\ M \text{ won't stop on step } 1$
$\Rightarrow M'_x \text{ accepts all } x\in\Sigma^*\Rightarrow L(M'_x)=\Sigma^*$
$\Rightarrow|L(M'_x)|=\infty\Rightarrow<M'_x>\in L_\infty$

$<M,x>\notin\overline{HP}\Rightarrow M \text{ halts on } x$
$\Rightarrow\exists i\in\mathbb{N}: M \text{ halts on } x \text{ after } i \text{ steps}$
$\Rightarrow\forall y:|y|<i\ M'_x \text{ accepts } y,$
$\forall y:|y|\ge i\ M'_x \text{ reject } y$
$\Rightarrow L(M'_x)=\{y\in\Sigma^*||y|<i\}$
$\Rightarrow|L((M'_x))|=finite\Rightarrow<M'_x>\notin L_\infty$

4. שכלול של רדוקציה 3 כאשר רוצים שלא תמיד תעצור: לרוב בשביל אי שייכות לRE
לדוגמה השפה:
$L=\{<M<|M \text{ stops for every input } x\in\Sigma^*\}$
נרצה להראות $L\notin RE$, נרצה רדוקציה $\overline{HP}\le L$ ע"י:
$f(<M,x>)=<M'_x>$

M'_x על קלט y:
1. הרץ את M על x במשך $|y|$ צעדים
2. אם M לא עצרה בשלב 1- קבל
3. אחרת (M עצרה)- כנס ללולאה אינסופית

$<M,x>\in\overline{HP}\Rightarrow M \text{ doesn't halt on } x$
$\Rightarrow \text{for each } x\ M \text{ won't stop on step } 1$
$\Rightarrow M'_x \text{ accepts all } x\in\Sigma^*$
$\Rightarrow M'_x \text{ halts for each } x\in\Sigma^*\Rightarrow<M'_x>\in L_\infty$

$<M,x>\notin\overline{HP}\Rightarrow M \text{ halts on } x$
$\Rightarrow\exists i\in\mathbb{N}: M \text{ halts on } x \text{ after } i \text{ steps}$
$\Rightarrow\forall y:|y|<i\ M'_x \text{ accepts } y,$
$\forall y:|y|\ge i\ M'_x \text{ loop on } y$
$\Rightarrow M'_x \text{ doesn't halt on each input}$
$\Rightarrow<M'_x>\notin L_\infty$

5. רדוקציה חשובה נוספת: לרוב בשביל אי שייכות ל$RE,coRE$
נרצה להראות $L_{eq}\in\overline{RE\cup coRE}$, נרצה רדוקציה $L_{\Sigma^*}\le L_{eq}$ ע"י:
$f(<M>)=<M,M_{stam}>$

M_{stam}- תמיד מקבלת

$<M,x>\in L_{\Sigma^*}\Rightarrow L(M)=\Sigma^*, \text{and } L(M_{stam})=\Sigma^*$
$\Rightarrow L(M)=L(M_{stam})\Rightarrow<M,M_{stam}>\in L_{eq}$

$<M,x>\notin L_{\Sigma^*}\Rightarrow L(M)\ne\Sigma^*, \text{and } L(M_{stam})=\Sigma^*$
$\Rightarrow L(M)\ne L(M_{stam})\Rightarrow<M,M_{stam}>\notin L_{eq}$

כדי להראות:
אי שייכות לRE אפשר להראות: $HP\le L$
אי שייכות ל$coRE$ אפשר להראות: $\overline{HP}\le L$
אי שייכות ל$RE,coRE$ אפשר להראות:
$L_\infty,L_{\Sigma^*},L_{eq}\le L$

Neta Roth: https://github.com/neta-r
Almog Jakov: https://github.com/AlmogJakov

חישוביות – דף נוסחאות

שפות שלמות בRE:

שפה L היא שלמה ב-RE אם"ם היא מקיימת:
$L \in RE$
$\forall(L \in RE): L' \leq L$
דוגמה לשפה ב$RE - complete$: היא השפה L_u
ידוע כי $L_u \in RE$, נראה **רדוקציה כללית** לשפה $L' \in RE$:
תהי שפה $L' \in RE$, אז קיימת לה M' מקבלת, פונקציית הרדוקציה: $f(x) = < M', x >$.

$$x \in L' \Leftrightarrow M' \; accepts \; x \Leftrightarrow < M', x > \in L_u \Leftrightarrow f(x) \in L_u$$

* <u>לא יתכן ש$L \in R \cap RE - complete$:</u>
נניח בשלילה כי $L \in R \cap RE - complete$
לכן לכל $L' \in RE$ מתקיים $L' \leq L$, לכן אם $L' \notin R$, לפי משפט הרדוקציה $L \notin R$.

* <u>$RE - copmlete \neq RE \backslash R$</u>

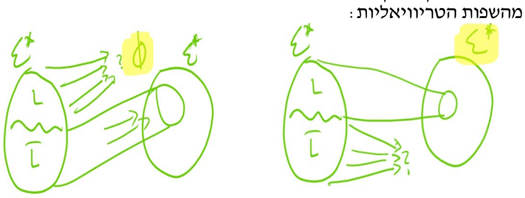
* <u>אם $L' \in RE$, $L \in RE - complete$ ומתקיים $L \leq L'$ אז $L' \in RE - complete$:</u>
נתון $L \leq L'$ ומכיוון ש$L' \in RE - complete$ מתקיים:
$\forall L * \in RE: L * \leq L'$
מטרנזיטיביות $\leq$ מתקיים: $\forall L * \in RE: L^* \leq l$.

הערה: יש רדוקציה מ$L_\infty \leq HP$ אבל זה לא אומר שL_∞ היא RE-שלמה כי היא בכלל לא בRE.

<u>שאלת מבחן בנושא:</u>
הוכח או הפרך: קיימת שפה במחלקה $(RE \cup coRE) - comp$:
הפרכה: נניח כי $L \in (RE \cup coRE) - complete$:
אם $L \in RE$: מכך ש$\overline{HP} \leq L$ (כי יש רדוקציה מכל שפה במחלקה ו$\overline{HP} \in$ $(RE \cup coRE)$ נקבל $\overline{HP} \in RE$ (משפט הרדוקציה בצורתו החיובית)
אם $L \in coRE$: מכך ש$HP \leq L$ נקבל $HP \in RE$.
סתירה בכל מקרה.

שפות שלמות בR:

$R - complete = R\{\phi, \Sigma^*\}$:
נשים לב שאין רדוקציה משפה כלשהי L שאינה טריוויאלית אל אחת מהשפות הטריוויאליות:



אם נתונות $L_1 \in R, L_2 \in R\{\Phi, \Sigma^*\}$ מתקיים $L_1 \leq L_2$:
M_f על קלט w:
1. בחר $x \in L_2, y \notin L_2$ (קיים כי L_2 לא טריוויאלית)
2. מסמלצת את ריצתה של M_1 (המכונה של L_1) אם M_1 קיבלה את w החזר את x אחרת החזר את y.

M_f תמיד מחזירה תשובה כי M_1 מכונה מכריעה.
$$w \in L_1 \Rightarrow M_1 \; accept \; w \Rightarrow M_f \; output \; x \Rightarrow x \in L_2$$
$$w \notin L_1 \Rightarrow M_1 \; reject \; w \Rightarrow M_f \; output \; y \Rightarrow y \notin L_2$$

משפט רייס וחבריו:

סימון- ϕ- קבוצה ריקה של שפות, φ- שפה ריקה
תכונה לא טריוויאלית בRE- קבוצת תכונות S כך ש$S \subseteq RE$, ומתקיים $\phi \neq S \neq RE$.

אבחנה: עבור תכונה לא טריוויאלית S קיימת לפחות שפה אחת בS ולפחות שפה אחת ב$RE \backslash S$.

השפה L_s של תכונה S מוגדרת כך: $L_s = \{< M > | L(M) \in S\}$

<u>משפט רייס:</u> לכל תכונה לא טריוויאלית $\phi \neq S \subsetneq RE$ מתקיים $L_s \notin R$
<u>משפט רייס המורחב:</u>
1. אם $\varphi \notin S$ מתקיים $L_s \notin coRE$,
2. אם $\varphi \in S$ מתקיים $L_s \notin RE$.

הוכחת מקרה 1:
נניח $\varphi \notin S$ ותהי $L_1 \in S$ ויהי M_{L_1} מ"ט מקבל לL_1 נוכיח $HP \leq L_s$
פונקציית הרדוקציה: $f(< M, x >) = < M_x >$
M_x על קלט $y \in \Sigma^*$:
1. מריצה את M על x
2. מריצה את M_{L_1} על y ועונה כמוה.

$$< M, x > \in HP \Rightarrow M \; halts \; on \; x \Rightarrow M_x \; gets \; to \; step \; 2$$
$$\Rightarrow L(M_x) = L_1 \Rightarrow < M_x > \in L_S$$
$$< M, x > \notin HP \Rightarrow M \; won't \; halts \; for \; all \; x$$
$$\Rightarrow L(M_x) = \varphi \Rightarrow < M_x > \notin L_S$$

הוכחת מקרה 2:
הרדוקציה כאן מ$\overline{HP} \leq L_s$, אותו דבר בדיוק רק בגלל ש$\varphi \in L_s$ הניתוח משתנה
$$< M, x > \in \overline{HP} \Rightarrow M \; won't \; halts \; every \; x$$
$$\Rightarrow L(M_x) = \varphi \Rightarrow < M_x > \in L_S$$
$$< M, x > \notin \overline{HP} \Rightarrow M \; halts \; for \; on \; x$$
$$\Rightarrow L(M_x) = L_2 \Rightarrow < M_x > \in L_S$$

תכונות טריוויאליות: <u>$S = RE$</u>: אז אפשר לקבל כל קידוד של מכונת טיורינג. מכיוון שהנחנו כי כל מחרוזת מתארת מכונת טיורינג כלשהי, נקבל כי השפה L_s מכילה את כל המחרוזות כלומר $L_s = \Sigma^*$ ואכן $\Sigma^* \in R$.

<u>$S = \phi$</u>: אז אפשר לדחות כל קידוד של מכונת טיורינג. כלומר, השפה L_S לא מכילה מילים כלל, כלומר $L_s = \varphi$, ואכן $\varphi \in R$.

מותר להשתמש ב$rice$ רק בתכונות של השפה ולא בתכונה של המכונה: דוגמאות:
1. $L = \{< M > | M \; halts \; on \; every \; input\}$ -אסור
2. $L = \{< M > || L(M)| \leq 3\}$ – מותר
3. $L = \{< M > || L(M)| \subseteq \Sigma^*\}$ -אסור תכונה טריוויאלית (שייך לR)
4. $L = \{< M > | M \; accept \; \varepsilon\}$ – מותר (זה שייך לRE)
5. $L = \{< M > | \varepsilon \in L(M)\}$ -מותר (זה שייך לRE)
6. $L = \{< M > | L(M) \in coRE\}$ -מותר
7. $L = \{< M > | L(M) \notin R\}$ -מותר
8. (אין צורך כי היא סופות ובהכרח שייכת אל R)
$L_7 = \{< M > || L(M)| \; is \; odd \; and \; | < M > | \leq 1000\}$

מתי לא להשתמש: תנאי השפה מתייחס להתנהגות של המכונה כגון: עצירה, זמן חישוב, מספר מצבים, תאי זיכרון....

דוגמה לשימוש במשפט רייס:

$$L1 = \{< M >: M \; accepts \; only \; even \; length \; inputs\}$$
התכונה:
$S1 = \{L \in RE: L \; contains \; only \; even \; length \; words\}$.
התכונה לא טרייויאלית: דוגמא לשפה שמקיימת את התכונה: $\{00\}$,
דוגמה לשפה שלא מקיימת את התכונה: $\{0\}$
לכן לפי משפט רייס, $L \notin R$
כאן ניתן להשתמש בהרחבה: כי φ מקיימת את התכונה (יש בה רק מילים באורך זוגי) ולכן: $L \notin RE$

בעיית BHP:

$$L_c = \{< M, x > | M \; halts \; on \; x \; within \; x \in \mathbb{N} \; steps\}$$
$$BHP = \{< M, x > | M \; halts \; on \; x, and \; M \; doesn't \; enter \; the \; (|x|^2 + 1)'th \; cell \; of \; it'stape\}$$
שפות אלו בR.

<u>$L_c \in R$:</u>
U על קלט $< M, x >$:
1. הרץ את M על x למשך c צעדים
2. אם M קיבלה קבל – אחרת דחה.

<u>$BHP \in R$:</u>
נחסום את מס' הקונפיגורציות האפשריות:
$c = \alpha, q, i$
α- תוכן הסרט, $\#(\alpha) \leq |\Gamma|^{|x|^2}$
q- מצב נוכחי, $\#(q) \leq |Q|$
i- מיקום הראש, $\#(i) \leq |x|^2$ (רק תאים מותרים)

חסם על מספר קונפיגוריות:
$$T = |\Gamma|^{|x|^2} \cdot |Q| \cdot |x|^2$$

נרצה לזהות לולאה אינסופית בין התאים המותרים ונעשה זאת באמצעות זיהוי חזרה על קונפיגוריצה (אפשרי כי זו מכונה דטרמיניסטית !)
U על קלט $< M, x >$:
1. הרץ את M על x במשך $T + 1$ צעדים:
2. בכל שלב: אם M נכנסה לתא ה$|x|^2 + 1$ עצור,
אם M עצרה- קבל
3. אם הסתיימו $T + 1$ צעדים ללא עצירה -דחה

נשים לב שU תמיד עוצרת (כי היא רצה גג $T + 1$ צעדים).

הוכחת נכונות:
אם $< M, x > \in BHP$ אז M עוצרת על x בלי לעבור את התא ה$|x|^2 +$ 1 . זה יקרה תוך גג $T + 1$ צעדים, לכן U תעצור ותקבל ולכן $< M, x > \in L(U)$.

אם $< M, x > \notin BHP$ אז או שM נכנסה לתא ה$|x| + 1$ ואז היא תזהה זאת ותדחה, או שהיא נכנסת ללולאה אינסופית בין התאים המותרים ולאחר $T + 1$ צעדים הקלט ידחה כי לא זוהתה עצירה.
בכל מקרה $< M, x > \notin L(U)$.

שיקולי ספירה - כמה שפות יש וכמה מ"ט יש

יש א שפות (ובפרט יש $א_0$ רדוקציה כי כל רדוקציה ניתנת לחישוב במ"ט). יש א שפות שונות כי $|P(\Sigma^*)| =$ א .
העובדה הזו יכולה לעזור בהוכחות כלליות.

סיבוכיות מ"ט לא דטרמיניסטית:

בכל צעד חישוב יש כמה אופציות לכן δ מוגדרת כך:
$$\delta: Q' \times \Gamma \rightarrow P(Q \times \Gamma \times \{L, R\})$$
מילה מתקבלת במ"ט א"ד אם קיים מסלול חישוב מקבל עבורה.
מילה לא מתקבלת במ"ט א"ד אם לא קיים מסלול חישוב מקבל.

<u>מ"ט א"ד מכריעה:</u> לכל $x \in L$ קיים מסלול חישוב מקבל, לכל קלט כל מסלולי החישוב סופיים (אין לולאות).

<u>שקילות המודל למודל הרגיל:</u>
כיוון 1: כמובן שכל מ"ט דטרמיניסטית כי גם מ"ט א"ד.
כיוון 2: סימלוץ המכונה באמצעות BFS (ולא DFS אחרת עלולים להתקע בלולואה) ואם מצאנו מסלול מקבל נקבל.

המודלים לא שקולים בסיבוכיות!!!!!!!!!!!!!!!!!!!
אם נהפוך את המצבים לא נקבל את המכונה לשפה המשלימה !

הבדל סיבוכיות בין מודלים שונים

זמן הסימולציה של מכונה מרובת סרטים על-ידי מכונה חד-סרטית הוא ריבועי היותר. לכן מותר לעבור חופשי בין המודלים.

מ"ט פולינומיות:

מ"ט תקרא פולינומית אם קיים $p(n) = n^c$, $c \in \mathbb{R}$, כך ש$\forall x \in \Sigma^*$ המכונה M עוצרת על x בתוך לכל היותר $p(|x|)$ צעדים!

דגש חשוב:

סיבוכיות זמן ריצה מודדים ביחס לגודל הקלט!!!
כלומר אם הקלט בגודל $N = 2^n$ ומבצעים N פעולות הסיבוכיות תהיה $O(N)$.
לדוגמא: אם הקלט הוא n - מספר המיוצג בבינארי.
ואנו רוצים לבדוק האם הוא ראשוני ולכן רצים על כל המספרים מ2 עד $n - 1$ ובודקים האם הוא מתחלק בהם. אז הסיבוכיות היא אקספוננציאלית.
גודל הקלט: $N = \log_2 n$ ביטים.
זמן ריצה: $O(n) = O(2^N)$ אקספוננציאלי !

תזכורת הגדרת מ"ט המכריעה שפה:

נזכיר כי מ"ט M מכריעה שפה L אם מתקיים:
1. M מקבלת את השפה
2. לכל $w \in \Sigma^*$, כל חישוב של M על w מסתיים תוך מס' סופי של צעדים (במכונה דט' יש חישוב אחד)

המחלקה P:

$$NTIME(t(n)) = \{L \,|\, L \; is \; a \; language, \; decided \; by \; an \; O(t(n)) - time \; DTM\}$$
$$P = \bigcup_{k \geq 0} DTIME(n^k)$$
קבוצת כל השפות שיש להן מ"ט דטרמיניסטית המכריעה אותן בזמן פולינומי.

סגירות: איחוד, חיתוך, משלים, שרשור, איטרציה, הפרש, היפוך, חזקה

המחלקה NP:

$$NTIME(t(n)) = \{L \,|\, L \; is \; a \; language, \; decided \; by \; an \; O(t(n)) - time \; NTM\}$$
$$NP = \bigcup_{k \geq 0} NTIME(n^k)$$
קבוצת כל השפות שיש להן מ"ט אי דטרמיניסטית המכריעה אותן בזמן פולינומי.

<u>הגדרה שקולה: $L \in NP$ אם קיים</u> יחס דו מקומי R_L המכיל זוגות (x, y) כך ש:
1. $x \in L$
2. y הינו עד עבור x. (y מהווה תעודת שייכות של x לשפה).
R_L מקיים:
1. R_L חסום פולינומית- לכל $(x, y) \in R_L$ קיים פולינום $p(\cdot)$ כך שמתקיים $|y| \leq p(|x|)$.
2. R_L ניתן לזיהוי פולינומי דטר'. כלומר קיימת מכונת טיורינג דטרמיניסטית פולינומית V המוודאת (=מכריעה) את היחס.
3. השפה L מקיימת $L = \{x \in \Sigma^* | \exists y \; s.t. (x, y) \in R_L\}$
כלומר לכל מילה בשפה קיים עד פולינומי, ולכל מילה שאיננה בשפה לא קיים עד פולינומי .
$$x \in L \Rightarrow \exists p(\cdot), \exists y \in \Sigma^{p(|x|)}: (x, y) \in R_L$$
$$x \notin L \Rightarrow \forall p(\cdot), \exists y \in \Sigma^{p(|x|)}: (x, y) \notin R_L$$

סגירות: איחוד, חיתוך, שרשור, איטרציה, היפוך, חזקה, לא ידוע אם סגורה למשלים והפרש !

המחלקה $coNP$:

$$coNP = \{L | \overline{L} \in NP\}$$
קבוצת כל השפות שהמשלימה שלהן בNP
סגירות: איחוד, חיתוך, שרשור, איטרציה, היפוך, חזקה, לא ידוע אם סגורה למשלים והפרש !

תכונות של המחלקות:

$P \subseteq NP$, $\quad P, NP \subseteq R$
$P \subseteq coNP$

תכונות לא ידועות של המחלקות:

$NP =^? P \qquad P =^? NP \cap coNP$
$NP =^? coNP \qquad P =^? coNP$

Neta Roth: https://github.com/neta-r
Almog Jakov: https://github.com/AlmogJakov

חישוביות – דף נוסחאות

הוכחת שייכות ל-NP ב2 דרכים:

<u>בניית מ"ט א"ד פולינומית:</u>

לדוגמה: נוכיח כי $IS \in NP$:

N על קלט $< G,k >$:

1. נחש תת קבוצה S של קודקודי G. (בדוק שגודל הקבוצה הינן k)
2. לכל זוג $v_1, v_2 \in S$ בדוק ש$\{v_1, v_2\} \notin E(G)$ – אם גילנו שצלע כלשהי קיימת דחה
3. קבל

<u>נכונות</u>: אם $< G,k > \in IS$ אזי קיימת תת קבוצה $S \subseteq V$ בגודל k שהיא בלתי תלויה. אזי, קיים מסלול חישוב שבו המכונה N מנחשת בדיוק את הקבוצה S. במסלול זה, הבדיקה של N את הקבוצה S יוצאת תקינה, ולכן N מקבלת. כלומר, קיים מסלול מקבל, ולכן $< G,k > \in L(N)$.

אם $< G,k > \notin IS$ אזי לכל תת קבוצה $S \subseteq V$ בגודל k מתקיים שהיא תלויה. אזי, לכל ניחוש של הקבוצה S המכונה N תזהה שקיימת צלע ב"תוך" S ותדחה. כלומר כל המסלולים בעץ החישוב דוחים, ולכן $< G,k > \notin L(N)$.

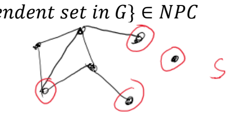
<u>סיבוכיות</u>: גודל הקלט הוא $O(n^2)$ כאשר n הוא מס' הקודקודים בגרף.
בשלב 1 - $O(k)$, בשלב 2- $O\left(\binom{k}{2}\right) = O(k^2)$, סה"כ: $O(k^2)$.

<u>הצגת עד פולינומי:</u>

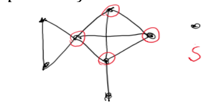
עד עבור IS יהיה קבוצה S בלתי תלויה בגודל k.
היחס R_{IS} מכיל זוגות מהצורה $(< G,k >, < s >)$ גודלו- k פולינומי בגודל הקלט.
אם $< G,k > \in IS$ אז קיימת תת קבוצה $S \subseteq V$ בגודל k שהיא בלתי תלויה. אז נבחר אותה להיות העד.
אם $< G,k > \notin IS$ אז לכל תת קבוצה $S \subseteq V$ בגודל k מתקיים שהיא תלויה. לכן לכל תת קבוצה שנבחר להיות העד לא יתקבל זוג אשר שייך לR_{IS}.

דוגמאות לשפות ושיוכן:

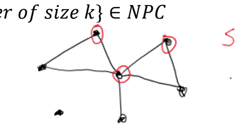
$IS = \{< G,k > | there\ is\ a\ subset\ of\ size\ k\ that\ is\ independent\ set\ in\ G\} \in NPC$



$CLIQUE = \{< G,k > | there\ is\ a\ subset\ of\ size\ k\ that\ is\ clique\ in\ G\} \in NPC$

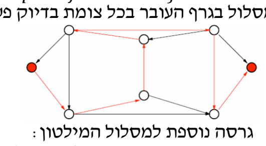


$VC = \{< G,k > | G\ is\ a\ graph\ that\ contain\ a\ vertex\ cover\ of\ size\ k\} \in NPC$



$HAMPATH = \{< G,s,t > | G\ is\ a\ un/directed\ graph\ with\ a\ Hamiltonian\ path\ from\ s\ to\ t\} \in NPC$

בעברית: מסלול בגרף העובר בכל צומת בדיוק פעם אחת



גרסה נוספת למסלול המילטון:

$HAMPATH = \{< G > | G\ is\ a\ un/directed\ graph\ with\ a\ Hamiltonian\ path\} \in NPC$

$HAMCYCLE = \{< G > | G\ is\ a\ un/directed\ graph\ with\ a\ Hamiltonian\ cycle\} \in NPC$

בעברית: כמו מסלול המילטון אבל חוזר לקודקוד הראשוני
לעומת זאת, מעגל אויילר (עובר בכל הצלעות פעם אחת) הוא בP

$3COL = \{< G > | G\ is\ graph\ and\ g\ is\ 3 - colorable\}$
$COMPOSITES = \{< n > | n\ is\ composite\} \in P$
$PRIMES = \{< n > | n\ is\ prime\} \in P$

$SUBSET-SUM = \{< S,t > | S = \{x_1, .., x_n\}, \exists\{y_1, \ldots, y_k\} \subseteq \{x_1, \ldots, x_n\}, \sum y_i = t\} \in NPC$

$SAT = \{< \varphi >: \varphi\ is\ CNF\ formula\ and\ \varphi\ has\ satisfiable\ assignment\} \in NPC$

$3SAT = \{< \varphi >: \varphi\ is\ 3CNF\ formula\ and\ \varphi\ has\ satisfiable\ assignment\} \in NPC$

$2SAT = \{< \varphi >: \varphi\ is\ 2CNF\ formula\ and\ \varphi\ has\ satisfiable\ assignment\} \in P$

$DNF-SAT = \{< \varphi >: \varphi\ is\ DNF\ formula\ and\ \varphi\ has\ satisfiable\ assignment\} \in P$

רדוקציה פולינומית:

בדיוק כמו רדוקציה רגילה רק שזמן הריצה שלה פולינומי- מסומנת $\leq_p$.
מקיימת את אותן תכונות כמו רדוקציה רגילה (כולל משפט הרדוקציה רק עם NP, P).

דוגמאות לרדוקציות פולינומיות:

$clique \leq_p IS$ ע"י הפונקציה: $f(< G,k >) = < \bar{G},k >$
תקפות הרדוקציה נובעת מהלמה הבאה: למה: עבור גרף $G = (V,E)$, קבוצת קודקודים S היא קליקה בG אס"ם S היא קבוצה בת"ל ב$\bar{G}$.

$IS \leq VC$ ע"י הפונקציה $f(< G,k >) = < G,n-k >$
תקפות הרדוקציה נובעת מהלמה הבאה: למה: עבור גרף $G = (V,E)$, קבוצת קודקודים S היא בת"ל בG אס"ם $V \backslash S$ היא כיסוי קודקודים בG.

המחלקה NPC:

שפה L תקרא NPC אם:
1. $L \in NP$
2. $\forall L' \in NP : l' \leq_P L$

<u>למה זה טוב?</u> בעזרת המחלקה נוכל להכריע בקלות יותר אם $NP = P$', איך? נניח כי ישנה שפה ב$P \cap NPC$ אז $P = NP$:
נניח $NPC \cap P \neq \phi$ לכן קיימת $L \in NPC \cap P$,
ידוע לנו גם ש$L \in P$
מכיוון ש$L \in NPC$ אז על פי הגדרת NPC: $\forall L' \in, L \in NP$ $NP: L' \leq_p L$
כלומר כל שפה בNP (L') ניתנת לרדוקציה לשפה בP (L) לכן לפי משפט הרדוקציה כל השפות בNP הן בP אז $NP = P$.

משפט קוק לוין
SAT היא שפה NP-שלמה

בהינתן שהשפה SAT היא NP שלמה, ניתן להוכיח בעזרתה שהרבה שפות אחרות הן גם NP-שלמות.

הראנו בתרגול שרשרת רדוקציה כשלמדנו את משפט קוק לוין:

$$SAT \leq_p 3SAT \leq_p VC \leq_p IS \leq_p Clique$$

בעיות חיפוש מול בעיות הכרעה:

בעיות הכרעה: מהסגנון "האם קיים בלה בלה בלה" לדוג': האם קיים מסלול המילטון בגרף.
בעיות חיפוש: מהסגנון "החזר את האובייקט מבעית החיפוש" לדוג': החזר את מסלול ההמילטון בגרף במידה וקיים.
<u>משפט</u>: אם $P = NP$ אזי קיים אלגוריתם דטרמיניסטי פולינומי לבעית החיפוש כלשהי.

נוכיח על $3SAT$:

נניח ש$P = NP$ אזי קיים אלגוריתם דטרמיניסטי פולינומי A המכריע את $3SAT$. נתאר אלגוריתם דטרמיניסטי (על קלט φ) המחזיר השמה מספקת לפסוק NPC נתון, או מחזיר את המחרוזת "לא קיימת השמה מספקת." כך:

1. סמלץ את A על φ. אם $A(\varphi) = 0$ החזר "לא קיימת השמה מספקת".
2. אתחלה השפה π בε.
3. כל עוד $n > 0$:
   - הצב ערך '0' במשתנה בעל האינדקס הנמוך ביותר של φ- יתקבל הפסוק φ_0.
   - סמלץ את ריצת A על φ_0 אם $A(\varphi_0) = 1$, השם למשתנה הנ"ל 0. אחרת ($A(\varphi_0) = 0$) השם למשתנה הנ"ל 1.
   - $n = n - 1$
4. החזר את π.

<u>נכונות</u>: אם $\varphi \notin CNF-SAT$ אז בשלב 1 יוחזר "לא קיימת השמה".
אם $\varphi \in CNF-SAT$ בכל שלב תינתן השמה למשתנה בודד ובסוף תהיה השמה מספקת.

<u>סיבוכיות</u>: נסמן ב$t(\cdot)$ את זמן הריצה של A (לפי ההנחה פולינומי).
האלגוריתם לעיל קורא לA n פעמים לכן זמן הריצה $O(n \cdot t(\cdot))$ וזה פולינומי.

נוכיח על $clique$:

נניח ש$P = NP$ אזי קיים אלגוריתם דטרמיניסטי פולינומי A המכריע את $clique$. נתאר אלגוריתם דטרמיניסטי (על קלט $< G,k >$) המחזיר קליקה בגודל k בגרף G, או מחזיר את המחרוזת "לא קיימת קליקה בגודל k." כך:

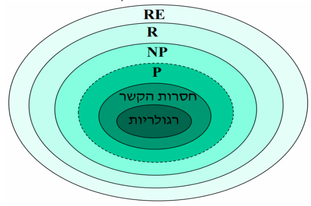
1. סמלץ את A על $< G,k >$ אם $A(< G,k >) = 0$ החזר "לא קיימת קליקה בגודל k".
2. $i = 1$
2. כל עוד $|V(G)| \neq k$:
   - הורד את הקודקוד הi ואת הצלעות החלות בו וקרא לגרף החדש שנוצר G'.
3. סמלץ את ריצת A על $< G',k >$ אם $A(< G',k >) = 1$, $G = G'$ אחרת ($A(< G',k >) = 0$) G נשאר ללא שינוי.
   - $i = i + 1$
4. החזר את G.

<u>נכונות</u>: אם $< G,k > \notin clique$ אז בשלב 1 יוחזר "לא קיימת קליקה בגודל k". אם $< G,k > \in clique$ בכל פעם שנחזר 0 אחרי הרצת האלגוריתם על G' נשאר פחות צלע, בסוף נקבל גרף בעל k קודקודים שהוא קליקה.

<u>סיבוכיות</u>: נסמן ב$t(\cdot)$ את זמן הריצה של A (לפי ההנחה פולינומי).
האלגוריתם לעיל קורא לA $n-k$ פעמים מקסימום לכן זמן הריצה $O((n-k) \cdot t(\cdot))$ וזה פולינומי.

תמונה כוללת

(לא ידוע אם כל ההכלות הן הכלות ממש!!!)



תרחישים אפשריים ליחס הקבוצות $P, NP, coNP$:

(לא ידוע איזה יחס הוא הנכון)

דוגמה לשאלת צביעה (שאלה 4 ממטלה 3):

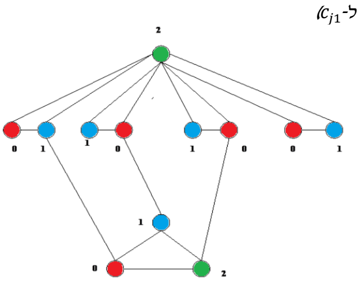
$$L = \{< \phi > | \phi\ is\ a\ 3CNF - Boolean\ formula\ that\ has\ an\ \neq -assignment\}$$

בפשטות, נזכיר כי זוהי שפת הפסוקים מצורת $3CNF$ כך שישנה השמה מספקת עבורה בכל פסוקית יש לפחות משתנה אחד $False$ ואחד $True$.

<u>נוכיח כי קיימת רדוקציה מהצורה הבאה:</u>

$$L \leq_P 3COLOR$$

עבור כל משתנה x_i נגדיר שני קודקודים. קודקוד c_{i1} עבור x_i וקודקוד c_{i0} עבור $\bar{x_i}$ וביניהם נעביר צלע.
נוסיף עוד קודקוד a אשר כל ה-c-ים שהגדרנו יתחברו אליו בצלע.
עבור כל פסוקית נגדיר שלושה משתנים k_{m1}, k_{m2}, k_{m3}, אחד עבור כל ליטרל, אשר יתחברו אחד לשני ע"י צלע וכל אחד מהם יתחבר ל-c של השלילה שהוא מוגדר בפסוק. (לדוגמה ליטרל $\bar{x_j}$ יתחבר ל-c_{j1})



הוכחת נכונות:

<u>כיוון אחד:</u>

אם $< \phi > \in L$ אזי קיים בכל פסוקית משתנה שערכו 1 ומשתנה שערכו 0 לכן נקבל צביעה חוקית ע"י כך שנצבע את a בצבע 2 ואת כל ה-c-ים שקיבלו 0 בצבע 0 ואת כל ה-c-ים שקיבלו 1 בצבע 1. עבור k שערכו 1 (אשר שולח צלע לצבע 1) נצבע בצבע 0 ואת הk שערכו 0 נצבע בצבע 1 (קיימים 2 כאלה לפי נכונות השפה. ואת הk הנותר נצבע בצבע 2. לכל קודקוד בעל צבע מסוים אין שכן עם אותו צבע ולכן מתקיימת צביעה חוקית אזי: $f(< \phi >) \in 3COLOR$

<u>כיוון שני:</u>

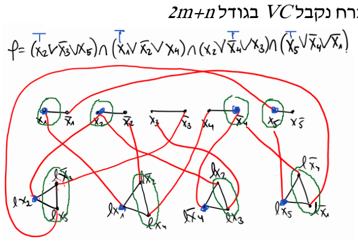
אם $f(< \phi >) \in 3COLOR$ אזי הגרף ניתן לצביעה חוקית, כלומר לקודקוד a יש צבע אחד נקרא לו 2 ולכל משתנה ושלילתו יש צבע 0 או 1 (כל אחד צבע שונה בגלל שהם יוצרים משולש). גם הליטרלים k_{m1}, k_{m2}, k_{m3} יוצרים משולש ולכן כל אחד מהם חייב להיות בצבע שונה (0, 1 או 2), שני הקודקודים המייצגים ליטרלים אשר קיבלו 0 או 1 יתחברו בצלע לקודקוד אשר מייצג משתנה והוא חייב לקבל 1 או 0 בהתאמה, כי הוא מחובר לa ולקודקוד הליטרל. כל קודקוד מקודקודי המשתנים אשר חייב לקבל את הערך 0 או 1 יקבל בפסוק גם 0(false) או 1(true) בהתאמה ועבור הליטרלים המייצגים קודקודים בצבע 2 שלאחר הבאת הערכים המשתנים שלהם לא קיבלו ערך יקבלו ערך רנדומלי (0 או 1) וכך עבור כל פסוקית נקבל שקיים ליטרל אחד לפחות עם ערך 0 וליטרל אחד לפחות עם ערך 1.
ולכן מתקיים $< \phi > \in L$

<u>סיבוכיות:</u>

עבור פסוק עם m פסוקיות ו-n ליטרלים נקבל בניה בגודל של $O(2n + 3m + 1)$ ולכן זו היא רדוקציה פולינומיאלית.

באופן יחסית דומה: הרדוקציה שעשינו עבור $3SAT \leq_P VC$

נניח שישנם n משתנים וm פסוקיות
עבור כל משתנה יש צלע מחברת לקודקודים המייצגים את המשתנה ושלילתו. ועבור כל פסוקית ישנו משולש של קודקודים המייצגים את המשתנים באותה פסוקית.
רק אם ישנה השמה מספקת אז בהכרח בכל פסוקית יש לפחות משתנה אחד שערכו $True$ ואז נוכל לקחת את המשתנה המייצג שלו [למעלה] + את שתי המשתנים האחרים במשולש המתאים ואז בהכרח נקבל VC בגודל $2m+n$



Neta Roth: https://github.com/neta-r
Almog Jakov: https://github.com/AlmogJakov

חישוביות – דף נוסחאות

תרגיל ממבחן 2020 מועד ב:

א. $L_1\{< M_1 > |\exists\ infinitely\ many < M_2 >' s\ s.t. L(M_1) = L(M_2)\}$

<u>פתרון:</u> השפה בR כי התנאי הוא טריוויאלי.
לכל שפה ב RE יש אינסוף מכונות שונות שמקבלות אותה.
ולכן התנאי מתקיים תמיד ואין צורך לבדוק אותו ומכאן השפה L_1 היא $\Sigma^* \in R$.

ב. $L_2 = \{< M_1 >, < M_2 > |There\ exsits\ at\ least\ 2\ words\ x_1 and\ x_2\ s.t. each\ x_i\ is\ in\ L(M_1) but\ not\ in\ the\ lenguague\ of\ the\ other\ M_j\ (j - 3 - i)\}$

קיימות 2 מילים כך שM_1 מקבלת אחת וM_2 לא, וM_2 מקבלת את השניה וM_1 לא.
<u>פתרון:</u> אינטואיציה: אפילו אם נניח ש 2 המילים הללו ידועות לנו, עדיין אם לדוגמא M_1 מקבלת את x_1ולא עוצרת על x_2 לא נוכל לדעת שהיא אכן לא עוצרת עליה ולכן זה לא ב RE.
השפה $L_2 \notin RE$
הוכחה: נראה רדוקציה: $\overline{HP} \leq L_2$ ע"י:

$$f(< M_1, x >) = < M_1, M_2 >$$

M_1 על קלט w:
1. אם $w = 0$ קבל
2. אם $w = 1$ הרץ את M על x ואם עצרה- קבל
3. אחרת- דחה

M_2 על קלט w:
1. אם $w = 0$ דחה
2. אם $w = 1$ קבל
3. אחרת- דחה

הרדוקציה מלאה וניתנת לחישוב - ...

תקפות: אם $< M.x >\in \overline{HP}$ אז M לא עוצרת על x. מכאן עבור $w = 0$ נקבל שM_1 מקבלת וM_2 דוחה ועבור $w = 1$ נקבל שM_1 לא עוצרת (לא מקבלת) וM_2 מקבלת לכן התנאי של L_2 מתקיים אז $< M_1, M_2 >\in L_2$.
אם $< M.x >\notin \overline{HP}$ אז M עוצרת על x אז לא קימת מילה שM_1 לא מקבלת ואילו M_2כן כי את 1,0 M_1 מקבלת ואת שאר המילים שתיהן דוחות. ולכן: $< M_1, M_2 >\notin L_2$.

לפי משפט הרדוקציה: $\overline{HP} \notin RE$ לכן $L_2 \notin RE$.

ב. $L_3 = \{< M_1 >, < M_2 > |There\ exsits\ at\ least\ 2\ words\ x_1 and\ x_2\ s.t. each\ x_i\ is\ in\ L(M_1)\ and\ is\ rejected\ by\ the\ other\ M_j\ (j - 3 - i)\}$

קיימות 2 מילים כך שM_1 מקבלת אחת וM_2 דוחה, וM_2 מקבלת את השניה וM_1 דוחה.
<u>פתרון:</u> מתקיים $L_3 \in RE\backslash R$
הוכחת שייכות ל RE: נראה מ"ט א"ד המקבלת את L_3
M_3 על קלט $< M_1, M_2 >$:
1. נחש את x_1, x_2
2. הרץ את M_1 על x_1. אם דחתה- דחה.
3. הרץ את M_1 על x_2. אם קיבלה- דחה.
4. הרץ את M_2 על x_1. אם קיבלה- דחה
5. הרץ את M_2 על x_2. אם דחתה- דחה.
6. קבל

<u>נכונות:</u> אם $< M_1, M_2 >\in L_3$ אז קיימות 2 מילים כך שאחת מתקבלת במכונה הראשונה ונדחית בשנייה
והמילה השנייה מתקבלת במכונה השנייה ונדחית בראשונה ולכן קיים מסלול חישוב בו ננחש בדיוק את 2
המילים הנ"ל בסדר הנ"ל ולכן M_1 תקבל את x_1 ותדחה את x_2 ואילו M_2 תדחה את x_1 ותקבל את x_2 ולכן נעבור את כל הבידקות ונקבל.
אם M_3 קיבלה אז קיים ניחוש של 2 מילים x_1, x_2 כך ש: M_1 קיבלה את x_1 ודחתה את x_2 וM_2 דחתה את x_1 וקיבלה את x_2. ולכן בפרט תנאי השפה L_3 מתקיים עבור 2 מילים אלו ולכן $< M_1, M_2 >\in L_3$

<u>הוכחת אי שייכות ל R:</u>
נראה רדוקציה: $HP \leq L_3$ ע"י:

$$f(< M_1, x >) = < M_1, M_2 >$$

M_1 על קלט w:
1. אם $w = 0$ הרץ את M על x ואם עצרה- קבל
2. אם $w = 1$ דחה
3. אחרת- דחה

M_2 על קלט w:
1. אם $w = 0$ דחה
2. אם $w = 1$ קבל
3. אחרת- דחה

אם $< M.x >\in HP$ אז M עוצרת על x. לכן M_1 תקבל את 0 ותדחה את 1 וM_2 תדחה את 0 ותקבל את 1 ולכן $< M_1, M_2 >\in L_3$
אם $< M.x >\notin HP$ אז M לא עוצרת על x ולכן M_1 לא תקבל אף מילה ומכאן בפרט: $< M_1, M_2 >\notin L_3$
לפי משפט הרדוקציה: $HP \notin R$ לכן $L_3 \notin R$.

תרגיל ממבחן 2021 מועד א:

א. הוכח או הפרך: לכל $L \in NP$ קיימים אינסוף יחסים $R \subseteq \Sigma^* \times \Sigma^*$ חסומים פולינומית, וניתנים לזיהוי פולינומי כך ש:

$$L = \{x \in \Sigma^* | \exists y, (x, y) \in R\}$$

<u>הוכחה:</u> אם $L \in NP$ אזי קיים יחס NP עבור השפה: R_L כלומר: $L = \{x \in \Sigma^* | \exists y, (x, y) \in R_L\}$, נגדיר את אוסף היחסים הבאים: $R_i = \{(x, 1^i y) | (x, y) \in R_L\}$ ברור ש $L = \{x | \exists y', (x, y') \in R_i\}$ לכל i כי נבחר $y' = 1^i y$. היחסים ניתנים לזיהוי פולינומי וחסומים פולינומית כיוון שR_L כזה.

ב. נניח $P \neq NP$ הוכח או הפרך: לכל זוג שפות $L_1, L_2 (NP \cup coNP)\backslash P$ מתקיים $L_1 \cup L_2 \notin P$
<u>דוגמא נגדית:</u> $L_1 = 3SAT \in NP\backslash P, L_2 = \overline{3SAT} \in coNP\backslash P$ (השפות לא ב P כי הן שלמות ומכיוון ש$P \neq NP$ אז כל שפה NPC או $coNPC$ לא שייכת לP). אבל $L_1 \cup L_2 = \Sigma^* \in P$.

ג. תנו דוגמה לשפות $L_1, L_1', L_2, L_2' \in NP$ כך ש$L_1 \subseteq L_1'$ וגם $L_2 \subseteq L_2'$ ודפונקציוץ $f \in POLY$ כך שf מהווה רדוקציה פולינומית מL_1 לL_2 וגם מL_1' לL_2' .
<u>דוגמה:</u> נבחר: $L_1 = L_2 = \{0\}, L_1' = L_2' = \{0,1\}$ ונבחר $f(x) = x$ מכאן היא פונקצית הרדוקציה $L_2 \leq_p L_1$ וגם $L_1' \leq L_2'$. השפות שייכות לNP כי הן סופיות, מתקיים $L_1 \subseteq L_1'$ וגם $L_2 \subseteq L_2'$ והפונקציה היא חישובית פולינומית (מחזיר את הקלט).

ד. נניח ש$P \neq NP$. הוכח או הפרך: השפה הבאה היא בP.

$$\{(G(V, E), k), 0^{2^{|V|}}) | (G, k) \in VC\}$$

השפה שייכת ל P. נראה מ"ט דטרמיניסטית עבור השפה:
M על קלט $< G, k, 0^{2^{|V|}} >$:
- עבור על הקלט ומנה את רצף האפסים בסוף ובדוק שהוא אכן שווה ל 2 בחזקת מספר קודקודי הגרף. אם לא - דחה.
- עבור על כל האפשרויות לבחירת k קודקודים מתוך $|V|$ ועבור כל אפשרות כזו:
  - $A = \Phi$
  - עבור כל קודקוד v נבחר , הכנס לA את כל הצלעות המחוברות אליו.
  - אם בסוף קיבלנו ש $A = E$ קבל
- דחה.

סיבוכיות: מניית האפסים: $O(2^{|V|}) = O(N)$ – לינארי בגודל הקלט.
מעבר על כל תתי הקבוצות של קודקודים חסום ב$O(2^{|V|}) = O(N)$ ועבור כל תת קבוצה יש לבדוק האם כל צלע נגעת בקודקוד כלשהו מתת הקבוצה: $O(|E| \cdot |V|)$ - חסום בגודל הקלט בריבוע.
סה"כ: פולינומי בגודל הקלט.

דוגמה לשאלה עם בניית פסוקיות (שאלה 3 ממטלה 3):

$$L = \{< \phi > |\phi\ is\ a\ 3CNF - Boolean\ formula\ that\ has\ an\ \neq -assignment\}$$

בפשטות, נזכיר כי זוהי שפת הפסוקים מצורת $3CNF$ כך שישנה השמה מספקת עבורה בכל פסוקית יש לפחות משתנה אחד *False* ואחד *True* .

<u>נוכיח כי $L \in NPC$:</u>
כלומר, צ"ל כי $L \in NP$ וגם $L \in NPH$.

<u>*שלב א: הוכחת $L \in NP$:*</u>
תחילה נראה כי קיימת מכונה NP שמכריעה את השפה:
1. ננחש עבור כל משתנה אם הוא true (1) או false (0).
2. אם עבור כל פסוקית יש ליטרל 0 וליטרל 1 – קבל.
3. אחרת – דחה.

נכונות:
אם $< \phi >\in L$ אזי קיים מצב בו נציב בכל משתנה 0 או 1 ונקבל שעבור כל פסוקית יש ליטרל 0 וליטרל 1, אזי ננחש את המצב הזה ואז המכונה תקבל.
אם $< \phi >\notin L$ אזי לא קיים מצב בו נציב בכל משתנה 0 או 1 ונקבל שעבור כל פסוקית יש ליטרל 0 וליטרל 1, אזי לא נוכל לנחש את המצב הזה ואז המכונה תדחה.

<u>*שלב ב: הוכחת $L \in NPH$:*</u>
כעת נבנה רדוקציה $3SAT \leq_p L$ בצורה הבאה:
עבור כל פסוקית $(x_1 \vee x_2 \vee x_3)$ נגדיר 2 פסוקיות חדשות: $(x_1 \vee x_2 \vee y_1) \wedge (\overline{y_1} \vee x_3 \vee z)$.
כלומר עבור פסוק עם m פסוקיות נוסיף עוד $y_1, y_2, \dots, y_m, z$ משתנים, כלומר m+1 משתנים.
הוכחת נכונות:
<u>כיוון אחד:</u>
אם $< \phi >\in 3SAT$ אזי עבור הפסוקיות החדשות

$$(x_i \vee x_{i+1} \vee y_k) \wedge (\overline{y_k} \vee x_{i+2} \vee z)$$

אם x_i או x_{i+1} שווים ל-1 אזי $y_k = 0$ ואז הפסוקית הראשונה מתקבלת וגם השנייה, נגדיר z להיות 0 ו$\overline{y_k} = 1$,
ואם x_{i+2} שווה ל-1 וכל השאר ל-0 נציב $y_k = 1$ וכך הפסוקית הראשונה תתקבל כי $y_k = 1$ ו$x_i = 0$ והפסוקית שניה תתקבל כי $x_{i+2} = 1$ וגם z שווה 0
ולכן $f(< \phi >) \in L$
<u>כיוון שני:</u>
אם $f(< \phi >) \in L$ אזי קיימת השמה שעבור כל פסוקית נקבל ערך 1 וערך 0, במידה ו-z הוא 1 נהפוך את כל ההשמות וכך z יהיה בעל ערך 0 (ניתן להפוך את כל ההשמות מכיוון שעבור כל פסוקית שקיבלה 0 ו1 זה יהפוך ל1 ו0 שזהו פסוק תקין).
במידה וערך x_{i+2} הוא 1 נקבל שערך הפסוקית $(x_i \vee x_{i+1} \vee x_{i+2})$ הוא 1.
במידה וערך x_{i+2} הוא 0 אזי $\overline{y_k} = 1 \rightarrow y_k = 0$
ולכן $x_i = 1$ או $x_{i+1} = 1$ והערך של הפסוקית תהייה 1.
מכיוון שכל הפסוקיות מקבלות את הערך 1 אזי מתקיים $< \phi >\in 3SAT$.
<u>סיבוכיות:</u>
נראה כי סיבוכיות הבניה היא פולינומיאלית: הוספנו m+1 משתנים פלוס N משתנים הקודמים כלומר:
$O(N + m + 1)$ שזו היא סיבוכיות פולינומיאלית.

תרגיל ממבחן 2021 מועד א:

א. $L_1 = \left\{(n, k, S_1, S_2, \dots, S_t) \middle| \begin{array}{l} (n, k, S_1, S_2, \dots, S_t) \in SC \\ and\ each\ S_i\ f\ even\ size \end{array}\right\}$

$SC = \{(n, k, S_1, S_2, \dots, S_t) | n, k \in \mathbb{N}, \forall i \in [t], S_i \subseteq [n], \exists I \subseteq [t]\ s.t. |I| = k\ and \cup_{i \in I} S_i = [n]\}$

השפה בNPC,
<u>שייכות לNP</u> נראה מ"ט א"ד:
N_1 על קלט $< n, k, S_1, .. S_t >$:
1. עבור i מ1 עד n:
- אם $|S_i| = 1 (mod\ 2)$ דחה
- נחש קבוצה I של k מספרים מתוך: $\{1, \dots, t\}$
- הגדר $A = \Phi$
- עבור $i \in I$:
  - $A = A \cup S_i$
- אם $A = \{1, \dots, n\}$ קבל
- אחרת דחה

<u>שייכות לNPH:</u> נראה $L_1 \leq_p SC$ ע"י:

$$f(< n, k, S_1, S_2, \dots, S_t >=< n', k', S_1', S_2', \dots, S_t' >$$
$$n' = 2n$$
$$k' = k$$
$$S_i' = S_i \cup \{n + a: a \in S_i\}$$

הרדוקציה מלאה וניתנת לחישוב.
סיבוכיות: הכפלת n ב 2 - פולינומי בגודל הקלט.
העתקת k.
הכפלת מספר האיברים ב S_i - פולינומי בגודל הקלט.

תקפות: אם $< n, k, S_1, S_2, \dots, S_t >\in SC$ אז קיימת קבוצת אינדקסים $|I| = k$ כך ש$\cup_{i \in I} S_i = [n]$.
לפי בניית S_i' נקבל שלכל $a \leq n$ מתקיים שאם $a \in S_i$ אז גם $a + n \in S_i'$ ולכן הכיסוי של המספרים 1 עד n מכסה גם את המספרים $n + 1$ עד $2n$ עם אותן קבוצות ולכן: $\cup_{i \in I} S'_i = [2n]$ ולכן זהו כיסוי חוקי. וכל הקבוצות בגודל זוגי)כי כמות האיברים בכל קבוצה היא כפולה ממה שהיה.
אם $< n', k', S_1', S_2', \dots, S_t' >\in L_1$ אזי קיים כיסוי $\cup_{i \in I} S'_i = [2n]$ ולפי הבנייה $\cup_{i \in I} S_i = [n]$ ולכן זהו כיסוי רגיל עבור $< n, k, S_1, S_2, \dots, S_t >$.

ב. $L_2 = \{(n, k, S_1, S_2, \dots, S_t) | \exists\ an\ even\ k \in \mathbb{N}\ s.t. (n, k, S_1, S_2, \dots, S_t) \in SC\}$

השפה שייכת ל P.
כי בהינן הקלט: $< n, k, S_1, S_2, \dots, S_t >$ אם נבחר t זוגי. נבחר $k = t$ ואז אם האיחוד של כולן מכסה את $\{1, \dots, n\}$ אז מצאנו את ה k המבוקש ולכן נקבל. ואחרת נדחה כי אם האיחוד של כולן לא כיסה את הכל אז גם אם נאחד חלק מהקבוצות , לא נצליח לכסות את הכל. אם נבחר t אי-זוגי אז נבחר: $k = t - 1$ ואז יש לעבור על $\binom{t}{t-1} = \binom{t}{1} = O(t)$ אפשריות לבחור את $t - 1$ הקבוצות מתוך הt, לאחד אותן ולבדוק האם האיחוד מכסה את $\{1, \dots, n\}$ {אם מצאנו - נקבל. ואם לא מצאנו בכלל - נדחה כי לא ייתכן שכמות קטנה יותר תכסה הכל.
סיבוכיות: פולינומית בגודל הקלט.

ג. $L_3 = \{(G, k) | There\ exist\ at\ least\ 2^{k/2}\ vertex\ cover\ of\ G\ of\ size\ k\}$

התנאי אינו טריוויאלי כיוון שיש גרפים)לדוגמא ללא צלעות(שבהם יש$2^{k/2}$ כיסויי צמתים. ומכיוון שכדי לוודא שאכו יש כאלה צריך לעבור על כל $2^{k/2}$ האפשרויות)אקספוננציאלי(אז השפה לא שייכת לNP ולכן היא שייכת לNPH.

הוכחת שייכות לNPH:
נראה רדוקציה $VC \leq_p L_3$ ע"י: $f(< G, k >) =< G', k' >$
G' יהיה גרף עם עוד $2k$ צמתים המחולקים לזוגות זוגות המחוברים בצלע אחת ביניהם.
$k' = 2k$.
סיבוכיות: העתקת הגרף עם תוספת $O(n)$ צלעות וקודקודים לכל היותר. העתקת k וכפולה שלו ב 2 ולכן סה"כ פולינומי בגודל הקלט.
תקפות: אם יש בG כיסוי בגודל k אזי אותו כיסוי עם כיסוי k הצלעות הנוספות מהווה כיסוי בגודל $2k$ בG'. בנוסף, ניתן לצרף לכיסוי של G כל קומבינציית כיסוי מהקודקודים הנוספים (עבור כל צלע חדשה יש 2אפשרויות לבחור את הקודקוד המכסה) ולכן סה"כ יש לפחות $2^k = 2^{k'/2}$ כיסויים שונים בגרף בגודל $k' = 2k$
אם יש ב G' לפחות 2^k כיסויים בגודל $k' = 2k$ אזי מכיוון שלכיסוי הצלעות החדשות מספיק k קודקודים בדיוק נובע שבתוך G יש כיסוי של כל הצלעות באמצעותk קודקודים ולכן יש ב G כיסוי בגודלk .