חישוביות:

מ"ט מורכבת כך: $M = (Q, \Sigma, \Gamma, q_0, b, \delta, F)$

סוגי מכונות:

1. **המכונה המחשב פונקציה :** מקבלת את הקלוט x כרגיל וכשעוצרת, היא מוציאה פלט – הפלט הוא כל מה שכתוב בסרט הזיכרון מתא 1 ועד התא שבו נמצא הראש (לא כולל).
   - פונקציה מלאה - אם המכונה עוצרת (ואז מחזירה פלט) לכל קלט אז הפונקציה שהיא מחשבת נקראת פונקציה מלאה. (מוגדרת לכל קלט).
   - פונקציה הניתנת לחישוב – כל פונקציה שניתן לבנות לה מ"ט שתחשב אותה נקראת פונקציה ניתנת לחישוב.
2. **מכונה המכריעה שפה :** מקבל או לא מקבלת (כמו אוטומטים)
   יש שני מצבי עצירה שונים: q_{accept}, q_{reject}

אם המכונה לא עוצרת, אנחנו מחשיבים זאת (מתמטית) כאי קבלה אך לא ניתן לדעת את תשובת המכונה.

מושגים:

- **שפה של מכונה :** $L(M)$ – קבוצת כל המילים ב Σ^* שהמכונה מקבלת אותן (אם מכניסים מילה מהקבוצה כקלט למכונה אז היא עוצרת ומקבלת אותה).

- **קונפיגורציה:** תיאור של מצב נוכחי של המכונה באמצע עבודתה (או בהתחלה).
הקונפ' מורכבת מהמצב (בQ) שבו המכונה עומדת כרגע, מה כתוב בכל סרט הזיכרון, איפה עומד הראש קורא כותב.

- **מכונה מכריעה של שפה L:** מכונה שמקבלת כל מילה ששייכת לL ודוחה כל מילה שלא שייכת לL **ותמיד עוצרת.**

- **מכונה מזהה של שפה L:** מכונה שמקבלת כל מילה ששייכת לL אבל עבור מילים שלא שייכות לL היא **יכולה לדחות והיא יכולה לא לעצור.**

- **שפה כריעה**: שפה שקיימת עבורה מ"ט מכריעה (שתמיד עוצרת).

- **שפה מזוהה טיורינג:** שפה שקיימת לה מ"ט מזהה.

הערה: אם שפה היא כריעה אז היא גם מזהה. (כי מותר למכונה מזהה גם לעצור תמיד ולדחות במקרה והקלט לא בשפה).

שקילות מודלים:

ניתן להגדיר מספר מודלים של מ"ט שונות המתנהגות אחרת.

נאמר ש2 מודלים של מ"ט שקולים אם:

- כל פונקציה שהמודל הראשון יכול לחשב גם המודל השני יכול לחשב. ולהפך.
- כל שפה שהמודל הראשון יכול להכריע, גם המודל השני יכול להכריע ולהפך.

כדי להוכיח שקילות: צריך להוכיח שני כיוונים, להראות שאם קיים מודל 1 עבור פונקציה\שפה אז איך ניתן לבנות מי זה את מודל 2 לאותה הפונקציה\שפה.

כדי להוכיח אי שקילות : מספיק להראות דוגמה לשפה אחת\פונקציה שמודל אחד יכול לחשב והשני לא.

כלל: אם ניתן לראות את כל המילה של הקלט בבת אחת אז ניתן לקבל כל דבר.

הערה: אם כל הראשים כותבים למקום אחד, נחליט שהתו ילך לפי הראש הראשון ביותר שעומד מעל אותו תא (מבחינת מספר הראש).

מחלקות חישוביות:

- **המחלקה R (השפות כריעות)** - קבוצת כל השפות **שיש** להן מכונה שמכריעה אותן.
  - מ"ט מכריעה – **תמיד עוצרת** ומחזירה תשובה של "כן" על מילה בשפה ותשובה של "לא" על מילה שלא בשפה.
- **המחלקה RE (שפות מזוהות טיורינג)** – קבוצת כל השפות שיש להן מכונה שמזהה אותן.
  - מ"ט מזהה – מכונה שאם נותנים לה מילה בשפה היא תמיד עוצרת ועונה "כן" ואם נותנים לה מילה שלא בשפה אז היא יכולה לעצור ולומר "לא" ויכולה לא לעצור בכלל.
- **המחלקה $coRE$** – קבוצת כל השפות שהמשלימה שלהן ב RE.
  - אלו שפות שיש להן מכונה שאם נותנים לה מילה שלא בשפה אז היא תמיד עוצרת ועונה "לא". ואם נותנים לה מילה שהיא בשפה אז היא לעצור ולומר "כן" ויכולה לא לעצור.

תכונות:

1. $R = RE \cap coRE$ – כלומר אם הוכחנו ששפה היא בRE וגם ב$CORE$ אז היא בR.
2. $R \subseteq RE$ וגם $R \subseteq coRE$.
3. יש שפות שהן לא בRE ולא ב$coRE$.

הוכחה:

כמות השפות השונות בעולם היא כל תתי הקבוצות של Σ^* . כלומר: $|P(\Sigma^*)| = \aleph$.

בעצם יש לה סדר ולכן היא לא בת מנייה.

כמות מכונות הטיורינג השונות שיש בעולם (כמות התוכניות השונות ב$JAVA$ או בC שאפשר לייצר) היא : בגודל $\aleph_0$. (כי מדובר בכמות תווים סופים המרכיבה את קוד התוכנית).

מסקנה: יש יותר שפות ממכונות ולכן בהכרח יש שפה שאין לה מכונה שמקבלת אותה.

תכונות סגור: (סגירות של שפות בתוך המחלקות)

- **במחלקה R:**

  פעולות אונאריות: אם $L \in R$ אז:
  1. $\bar{L} \in R$
  2. $L^R \in R$
  3. $L^* \in R$

  פעולות בינאריות: אם $L_1, L_2 \in R$ אז:
  1. $L_1 \cup L_2 \in R$
  2. $L_1 \cap L_2 \in R$
  3. $L_1 \cdot L_2 \in R$
  4. $L_1 \backslash L_2 \in R$

- **במחלקה RE:**

  פעולות אונאריות: אם $L \in RE$ אז :
  1. אם סגירות בהכרח למשלים.
  2. $L^R \in RE$
  3. $L^* \in R$

  פעולות בינאריות: אם $L_1, L_2 \in RE$ אז:
  1. $L_1 \cup L_2 \in RE$
  2. $L_1 \cap L_2 \in RE$
  3. $L_1 \cdot L_2 \in RE$
  4. $L_1 \backslash L_2$ לא סגור.

- במחלקה $coRE$ הוא כמו RE.

הערה: בהכלה שום דבר לא סגור.

קידודים:

- לכל מכונת טיורינג יש קידוד (ואף אינסוף קידודים שיכולים לתאר את אותה מכונה).
- כל הקידודים הלא תקינים מתארים את אותה המכונה.

הרצה מבוקרת:

- **סוג 1:** להריץ את המכונה רק K צעדים ואז לעצור אותה (היא יכולה בזמן הזה לעצור ולקבל או לעצור ולדחות או לא לעצור).
- **סוג 2:** הרצה על מספר קלטים במקביל: מסדרים את כל המילים לפי סידור מילוני (לקסיקוגרפי: $\ldots ,\varepsilon, 0,1,00,01,10,11$) כאשר באינטרציה ה$i$ נרוץ על כל אחת מi המילים הראשונות.
- **סד"פ :**
- עבור t מ 1 עד אינסוף:
  - הרץ את M על ___ למשך t צעדים.

איך לדעת ולקבל אינטואיציה האם שפה היא ב$R, RE, coRE$?

1. **אם נותנים לנו מילה שהיא בשפה -** האם נצליח להוכיח לעצמינו שהיא בשפה? תוך כמות סופית של צעדים. (לא נצטרך להריץ מכונה הרצה מלאה כי היא עלולה לא לעצור ולא נצטרך לבדוק קבוצה אינסופית של אלמנטים).
אם הצלחנו להוכיח אז השפה ב RE אחרת היא לא בRE.

2. **אם הפעם המילה לא בשפה -** האם כעת נצליח להוכיח לעצמנו שהיא לא בשפה תוך כמות צעדים סופית.
אם הצלחנו להוכיח, השפה ב$coRE$ אחרת היא לא ב$coRE$.

3. **אם הוכחנו את שניהם -** אם בסוף הגעתם למצב שהשפה גם בRE וגם ב$coRE$ – אז היא בR.

שפות שהן ב$RE\backslash R$:

1. $A_{TM} = \boldsymbol{L_U} = \{< M, w > | M \ is \ truring \ machin \ and \ M \ accepts \ w\}$
2. $HALT_M = \boldsymbol{HP} = \{\ldots . | \ldots stops \ on \ w\}$
3. $\boldsymbol{L_D} = \{< M > | M \ is \ TM \ and \ M \ accepts < M >\}$
4. $\boldsymbol{L_{\neq\phi}} = \{< M > | L(M) \neq \phi\}$ קבוצת המכונות שמקבלות לפחות מילה אחת.

שפות שהן ב$coRE\backslash R$:

1. ההופכיות של השפות בRE.
2. $E_{TM} = L_\Phi = \{< M >: L(M) = \Phi\}$

שפות שהן לא בRE או ב$coRE$:

1. ניתן להשתמש בשפה הידוע: $L_\infty = \{< M >: |L(M)| = \infty\}$
2. $ALL_{TM} = L_{\Sigma^*} = \{< M >: L(M) = \Sigma^*\}$
3. $EQ_{TM} = L_{EQ} = \{< M_1, M_2 >: L(M_1) = L(M_2)\}$

הערה: ניתן לעשות 2 רדוקציות אחת לכל כיוון כדי להראות שייכות לכל RE ול$coRE$ ואז מכאן נראה שזה לא כאן ולא כאן ולכן זה בחוץ.

שיטת הוכחה ששפה כן שייכת למחלקה מסוימת:

1. **מראים מכונת טיורינג עבור השפה:**
   א. אם זה בR – מראים מכונה שתמיד עוצרת.
   ב. אם זה בRE –
      i. מראים מכונת טיורינג שתמיד עוצרת עם הקלט בשפה ואם הוא לא בשפה אז לא חייבת לעצור.
      ii. שימוש במ"ט א"ד.
   ג. אם זה ב$coRE$ – מראים מכונה עבור השפה המשלימה. (שהיא בRE).
2. **צריך להוכיח נכונות** (אם מילה בשפה למה היא מתקבלת ואם מילה לא בשפה למה היא נדחתה).
3. **באמצעות רדוקציה:** אפשר במקרים מסוימים להשתמש ברדוקציה דו כיוונית אם נתון לנו ששפה מסוימת שייכת למחלקה כלשהי.
   א. מראים רדוקציה מהשפה הידועה לשפה השנייה ולפי משפט הרדוקציות מראים אי שייכות למחלקות שהשפה הידועה לא שייכת אליהם.
   ב. מראים רדוקציה הפוכה (לאחר שאנחנו יודעים כבר שהשפה ה'לא ידועה' לא שייכת אליהן) ובכך מראים שהשפה הידועה גם לא שייכת אל המחלקות האלה.
   ג. מסקנה השפה הלא ידועה שייכת לאותה מחלקה של השפה הידועה.

איך מוכיחים ששפה לא שייכת ל$R, RE, coRE$?

1. **שיטת הלכסון (לא מומלץ):**
   א. מניחים בשלילה שהשפה כן נמצאת באותה מחלקה.
   ב. מהשייכות נובע שיש מכונה M_L עבור השפה.
   ג. מגדירים מכונה שמשתמשת בM_L ומבלבלת לה את התשובה. (כוונה מתנהגת הפוך מהשפה L אם M_L ענתה "כן" ומתנהגת כמו השפה L אם M_L ענתה לא.)
2. **רדוקציות:**

הערה: תמיד מה שרוצים להוכיח שהוא לא שייך למשהו יהיה בצד ימין

$L_1 \leq L_2$ (L_1 ניתנת לרדוקציה לL_2) אם קיימת פונקציה: $f: \Sigma^* \to \Sigma^*$ מלאה (מוגדרת על כל קלט) וניתנת לחישוב (יש מכונה שפולטת בדיוק את התוצאה של הפונקציה) כך שמתקיים:
$x \in L_1$ אם ורק אם $f(x) \in L_2$.

שימוש ברדוקציה להוכחה אי שייכות:
רעיון המרכזי שנשתמש בו במשפט הרדוקציה: אם השפה לא שייכת ל... אז כל שכן השפה הקשה יותר לא שייכת ל....

- **שפה לא בR:** צריך לקחת שפה שכבר ידוע שהיא לא בR ולעשות רדוקציה ממנה לשפה שעליה רוצים להוכיח. $(HP\ or\ L_D\ or\ L_u \leq L')$
- **שפה לא בRE:** ניקח את המשלימות של השפות הידועות ב$RE \backslash R$.

שימוש ברדוקציה להוכחת שייכות :

- הוכחת שייכות

תכונות של רדוקציה:

א. רפלקסיביות: $L \leq L$. הפונקציה תהיה פונקציית הזהות $f(x) = x$.
ב. טרנזיטיביות: $L_1 \leq L_2\ and\ L_2 \leq L_3 so\ L_1 \leq L_3$.
ג. הרדוקציה ממש לא חייבת להיות חח"ע ועל. היא יכולה אפילו למפות את כל הקלטים לאותו פלט.
ד. אם $L_1 \leq L_2$ אז $\bar{L_1} \leq \bar{L_2}$. (אפילו עם אותה פונקציה).
ה. **משפט הרדוקציה:** אם $L_1 \leq L_2$
   a. אם $L_1 \notin R\ so\ L_2 \notin R$ (או : אם $L_1 \in R\ so\ L_2 \in R$)
   b. אם $L_1 \notin RE\ so\ L_2 \notin RE$ (או : אם $L_1 \in RE\ so\ L_2 \in RE$)
   c. אם $L_1 \notin coRE\ so\ L_2 \notin coRE$ (או : אם $L_1 \in coRE\ so\ L_2 \in coRE$)
ו. לכל שפה $L \in R$ ולכל שפה $L' \neq \Phi, \Sigma^*$ מתקיים $L \leq L'$.

מה צריך להוכיח בכל רדוקציה?

א. **מלאה** - הפונק' עובדת על כל קלט.
ב. **ניתנת לחישוב** – ניתן לכתוב תוכנית מחשב\מכונה שתחשב את הפונק'.
ג. **תקפות** – אם הקלט בשפה אז התוצאה בשפה השנייה ואם הוא לא בשפה התוצאה לא בשפה השנייה.

4. **משפט רייס: (דרך להוכיח ששפה לא בR אבל בRE)**
כל תכונה $S \subseteq RE$ לא טריוויאלית של שפות מקיימת ש$L_s = \{< M >: L(M) \in S\} \notin R$.
במילים פשוטות: אם נתונה שפה המקיימת את התנאים הבאים:
   א. הקלט הוא רק קידוד מכונה אחת.
   ב. התנאי של השפה מדבר על שפות המכונה $L(M)$.
   ג. התנאי הוא לא טריוויאלי (טריוויאלי -תנאי שכל השפות בעולם מקיימות או שלא מקיימות).

אז: השפה לא בR.

הערה: אם S טריוויאלית אז $L_s \in R$.

שיטת עבודה :

- בודקים שמקבלים רק מכונה אחת.
- בודקים שהתנאי הוא על שפת מכונה.
  - אם התנאי על המכונה צ"ל שהתנאי הוא על המכונה ולא על השפה עצמה ! :
    1. בונים 2 מכונות שהשפות שלהן זהות.
    2. מראים שע"י שינוי בהתנהגות המכונה (קבלה\דחייה של מילה) התכונה תתקיים באחת ולא באחרת. (למרות שהשפות שלהן זהות).
- מגדירים את התכונה - כמו השפה רק שבמקום $< M >$ כותבים $L' \in RE$ ובמקום כל מקום שיש $L(M)$ כותבים פשוט L'.
- מראים שהשפה היא סמנטית:
  - מגדירים 2 מכונות שהשפות שלהן זהות.
  - מראים שאם השפה של מכונה 1 לא שייכת ל S אז איך זה משפיע על השפה של המכונה השניה(כי השפות שוות) ואיך גם היא לא שייכת ל S.
  - מראים אותו דבר על שייכות.
- מוכיחים ש S לא טריוויאלית:
  - $S \neq \Phi$ – דוגמה לשפה הRE שמקיימת את התנאי.
  - $S \neq RE$ – דוגמה לשפה בRE שלא מקיימת את התנאי.
- מסקנה: השפה המקורית לא בR.

<u>משפט רייס המורחב:</u>

- אם בנוסף מתקיים : $\Phi \in S$ אז: $L_s = L \notin RE$.
- אם בנוסף מתקיים : $\Phi \notin S$ אז: $L_s = L \notin coRE$.

<u>הוכחת משפט רייס:</u>

נתונה תכונה S לא טריוויאלית של שפות בRE. צ"ל: $L_s = \{< M >: L(M) \in S\}$ לא בR.
נחלק למקרים:

<u>מקרה 1: $\Phi \notin S$.</u>

מכיוון שS לא טריוויאלית אז קיימת שפה אחרת $L \in S$.

מכיוון שכל השפות בS הן RE אז ניתן להגיד $L \in RE$. לכן קיימת לה מכונה מזהה: M_L.

הוכחה: נראה רדוקציה - $L_u \leq L_s$. על ידי $f(< M, x >) =< M' >$ כאשר:

M' על קלט w:

- הרץ את M על x.
- אם M קיבלה את x הרץ את M_L על w וכנה כמוה.
- אחרת – דחה.

תקפות:

אם M מקבלת את x אז: M' תמיד תגיע לשורה שניה ותקבל רק מה שM_L מקבלת ולכן:

$L(M') = L(M_L) = L \in S$ ומכאן: $L(M') \in S$ ולכן: $< M >\in L_s$.

אם M לא מקבלת את x, אז אם M לא עוצרת על x אז M' תקוע בשורה הראשונה ולכן לא עוצרת על אף w ולכן : $L(M') = \Phi \notin S$ ולכן $< M' >\notin L_s$.

אם M עוצרת ודוחה את x אז M' תגיע תמיד לשורה שלישית ולכן תדחה הכל שוב: $L(M') = \Phi \notin S$ ולכן $< M' >\notin L_s$.

<u>מקרה 2: $\Phi \in S$.</u>

מכיוון שS לא טריוויאלית אז קיימת שפה אחרת $L \notin S$.

מכיוון שכל השפות שמדברים עליהן RE אז ניתן להגיד $L \in RE$. לכן קיימת לה מכונה מזהה: M_L.

הוכחה: נראה רדוקציה - $\overline{L_u} \leq L_s$. על ידי $f(< M, x >) =< M' >$ כאשר:

M' על קלט w:

- הרץ את M על x.

- אם M קיבלה את x הרץ את M_L על w וכנה כמוה.
- אחרת – דחה.

תקפות:

אם M מקבלת את x אז: M' תמיד תגיע לשורה שניה ותקבל רק מה שM_L מקבלת ולכן:

$L \notin S = L(M_L) = L(M')$ ומכאן: $L(M') \notin S$ ולכן: $<M> \notin L_s$.

אם M לא מקבלת את x, אז אם M לא עוצרת על x אז M' תקוע בשורה הראשונה ולכן לא עוצרת על אף w ולכן : $L(M') = \Phi \in S$ ולכן $<M'> \in L_s$.

אם M עוצרת ודוחה את x אז M' תגיע תמיד לשורה שלישית ולכן תדחה הכל שוב: $L(M') = \Phi \in S$ ולכן $<M'> \in L_s$.

בעיית הריצוף:

טענה: $\boldsymbol{Tiel \notin R}$- בעצם מראים כי אם קלט לבעיה גורם לעצירה זה לא ב$Tiel$ וקלט כן בשפה אז הריצוף הוא אינסופי.

הערה: $\boldsymbol{Tiel \in coRE}$ – קלט תקין המכונה לא עוצרת קלט, קלט לא תקין המכונה תעצור (ז"א הריצוף).

סיבוכיות

דגשים:

- כל השפות שנדבר עליהן נמצאות בR.
- סיבוכיות נמדדת בהתאם לגודל הקלט.
- סיבוכיות יעילה = פולינומיאלי . לא יעילה = אקספוננציאלי.
- חשוב לדעת ש: $\binom{n}{k} = \binom{n}{n-k} = O(n^k)$ לכל k קבוע.
- מ"ט מבצעת אלגו' לאט יותר ממחשב כי אין גישה לכל תא בסרט הזיכרון ב$O(1)$ אלה צריך לעבור על כל הסרט, למרות זאת מה שאפשר לעשות במחשב בזמן פולי' אפשר גם במ"ט.

הערה: יש הבדל בין מ"ט דטר' לבין מכונה אי-דטר'. צריך לשים לב לזה.

מ"ט אי דטרמיניסטית

- מ"ט שיש לה 2 פונקציות מעבר ובכל שלב היא "בוחרת\מנחשת" את הצעד הבא מבין 2 האופציות.
- לכן לכל קלט יש למכונה מספר מסלולי חישוב (יכול להיות גם ∞ מסלולי חישוב).
- אנו אומרים שהמכונה מקבלת מילה **אם יש אפילו מסלול אחד** מקבל, לכן ניתן להניח שהמכונה תמיד מנחשת נכון.
- המכונה יעילה - להחליף חיפוש בניחוש. לדוגמה: ניתן להחליף הרצה מבוקרת בניחוש אחד.
- מכונת טיורינג א"ד מכונה **לא עוצרת** אם אין מסלול מקבל ויש מסלול אחד לפחות שהוא אינסופי.
- אם יש מסלול מקבל אפילו אחד, המכונה תמיד עוצרת.
- המכונה נחשבת דוחה רק אם כל המסלולים סופיים ודוחים.
- מסלול אינסופי יכול להיווצר מהרצה של מכונה שעלולה לא לעצור או מניחוש איבר בתוך בטווח ערכים אינסופי.

מחלקות סיבוכיות

1. P – כל השפות שיש להן מ"ט **דטרמיניסטית** מכריעה שעובדת בסיבוכיות זמן פולינומית.
2. NP – כל השפות שיש להן מ"ט **אי דטרמיניסטית** מכריעה שעובדת בסיבוכיות זמן פולינומית.
3. $coNP$ – כל השפות שהמשלימה שלהן נמצאת ב NP.

קשר בין המחלקות:

- $P \subseteq NP$.
- $P \stackrel{?}{=} NP$ בעיה פתוחה.

- $P \subseteq coNP$.
- $coNP \cap NP$? בעיה פתוחה.

רדוקציה פולינומית:

רעיון המרכזי שנשתמש בו במשפט הרדוקציה: אם השפה לא שייכת ל... אז כל שכן השפה הקשה יותר לא שייכת ל....

הסימון: $L_1 \leq_p L_2$ (L_1 ניתנת לרדוקציה לL_2) אם קיימת פונקציה: $f: \Sigma^* \rightarrow \Sigma^*$ מלאה (מוגדרת על כל קלט) וניתנת לחישוב בזמן פולינומי (יש מכונה דטרמיניסטית שפולטת בדיוק את התוצאה של הפונקציה לאחר זמן פולינומי) כך שמתקיים:

$x \in L_1$ אם ורק אם $f(x) \in L_2$.

תכונות של רדוקציות פולינומיות:

- $L \leq_p L$
- אם $L_1 \leq_p L_2$ וגם $L_2 \leq_p L_3$ אז $L_1 \leq_p L_3$
- אם $L_1 \leq_p L_2$ אז $\overline{L_1} \leq_p \overline{L_2}$
- משפט הרדוקציה: אם $L_1 \leq_p L_2$ (פחות שימושי)
  - אם $L_1 \notin P$ אז גם $L_2 \notin P$.
- לכל שפה $L \in P$ ולכל שפה $L' \neq \Phi, \Sigma^*$ מתקיים $L \leq_p L'$.

שלמות שפות

שפה נקראת שלמה במחלקה שלה אם היא שייכת למחלקה וגם כל שפה אחרת באותה המחלקה ניתנת לרדוקציה אליה.

איך מוכחים שפה L היא שלמה?

1. מראים שהשפה שייכת לC. $L \in C$ (C מחלקה כלשהי).
2. אם יודעים כבר על שפה L' שהיא שלמה בC אז מראים : $L' \leq L$.

הערה: כשבודקים האם שפה היא NP - אם נצליח להוכיח רק את (2) אז נוכל להגיד כי $L \in NPH$

- **במלקה R:** כל השפות פרט לΦ, Σ^* הן שלמות (ברדוקציה של חישוביות)
- **במחלקה RE:** כל השפות שהן ב$RE \backslash R$ הן שלמות. (רדוקציה של חישוביות)
- **במחלקה $coRE$:** כל השפות שהן ב$coRE \backslash R$ הן שלמות. (רדוקציה של חישוביות)
- **במחלקה $RE \cup coRE$:** אין שפה שלמה.
- **במחלקה P:** כל השפות פרט לΦ, Σ^* הן שלמות (ברדוקציה פולינומית)
- **במחלקה NP:** נראה שפות שלמות אך לא ידוע האם כל השפות בNP (פרט לΦ, Σ^*) הן שלמות או רק חלק מהן וגם לא ידוע שפה שהיא בNP והיא לא NP שלמה.

שפות מוכרות שהן *NPC*

גרפים:

1. $CLIQUE = \{< G, k >: G\ contains\ clique\ of\ size\ k\}$
2. $IS = \{< G, k >: G\ contains\ independet\ sets\ of\ size\ k\}$ – כלומר יש בגרף קבוצה של k קודקודים שאף אחד מהם לא מחובר לאף אחד באופן ישיר.
3. $VC = \{< G, k >: G\ has\ vertex\ cover\ of\ size\ k\}$ – כלומר יש בגרף קבוצה של k קודקודים שמכסה את כל צלעות הגרף. = כל צלע נוגעת בלפחות אחד מקודקודי הקבוצה שנלקחה.
4. $k - Col = \{< G >: G\ is\ k - colorable\}$ – עבור $k > 2$. כל הגרפים שהם k צביעים.
5. $HC = \{< G >: G\ \text{undirected graph}\ with\ hmilton\ cycle\}$ – בגרף הלא מכוון יש מעגל המילטוני = עובר בכל הקודקודים בגרף בדיוק פעם אחת ופרט לקודקוד הראשון אליו חוזר בסוף המעגל.
6. $HL = \{< G >: G\ \text{undirected graph}\ with\ hamilton\ path\}$ – יש בגרף הלא מכוון מסלול המילטוני = מסלול שעובר **בכל** הקודקודים בגרף בדיוק פעם אחת.
7. $DHC = \{< G >: G\ \text{directed graph}\ with\ hmilton\ cycle\}$ – בגרף המכוון יש מעגל המילטוני = עובר בכל הקודקודים בגרף בדיוק פעם אחת ופרט לקודקוד הראשון אליו חוזר בסוף המעגל.

8. $DHL = \{< G >: G \text{ directed graph } with\ hamilton\ path\}$ - יש בגרף מכוון מסלול המילטוני = מסלול שעובר **בכל** הקודקודים בגרף בדיוק פעם אחת.

נוסחאות CNF:

9. $SAT = \{< \varphi >: \varphi \text{ is a CNF formula and } \varphi \text{has satisfiying assigment}\}$
10. $3SAT = \{< \varphi >: \varphi \text{ is a 3CNF formula and } \varphi \text{has satisfiying assigment}\}$ – כמו SAT פרט לכך שכל פסוקית בנויה מבדיוק 3 משתנים שונים (או שלילתם).

קבוצות:

11. $SS = \{< a_1, \dots a_n, s >: There\ is\ S \subseteq \{a_1 \dots a_n\} s.t\ \Sigma_{x \in S} x = s\}$
12. $SC = \{< A_1, \dots, A_t, n, k >$
$: There\ are\ k\ sets\ from\ A_{1 \dots t}\ s.t\ the\ union\ of\ all\ k\ sets\ is\ \{1,2 \dots n\}\}$

הערה: יש לנו שרשרת רדוקציות בין השפות הללו:

$$SAT \le_p 3SAT \le_p VC \le_p DHC \le_p HC \le_p HL \le_p DHL$$

שפות שהן *HARD NP*

- PLANARCIRCUITSAT: Given a boolean circuit that can be embedded in the plane so that no two wires cross, is there an input that makes the circuit output TRUE? This problem can be proved NP-hard by reduction from the general circuit satisfiability problem, by replacing each crossing with a small series of gates.
- NOTALLEQUAL3SAT: Given a 3CNF formula, is there an assignment of values to the variables so that every clause contains at least one TRUE literal *and* at least one FALSE literal? This problem can be proved NP-hard by reduction from the usual 3SAT.
- PLANAR3SAT: Given a 3CNF boolean formula, consider a bipartite graph whose vertices are the clauses and variables, where an edge indicates that a variable (or its negation) appears in a clause. If this graph is planar, the 3CNF formula is also called planar. The PLANAR3SAT problem asks, given a planar 3CNF formula, whether it has a satisfying assignment. This problem can be proved NP-hard by reduction from PLANARCIRCUITSAT.[7]

בשאלות שבהן צריך להוכיח האם שפה היא ב*P* או ב*NPC*:

תחילה נבדוק האם אנו מצליחים לבנות אלגוריתם (מ"ט דט') שמוכיח את *P*.

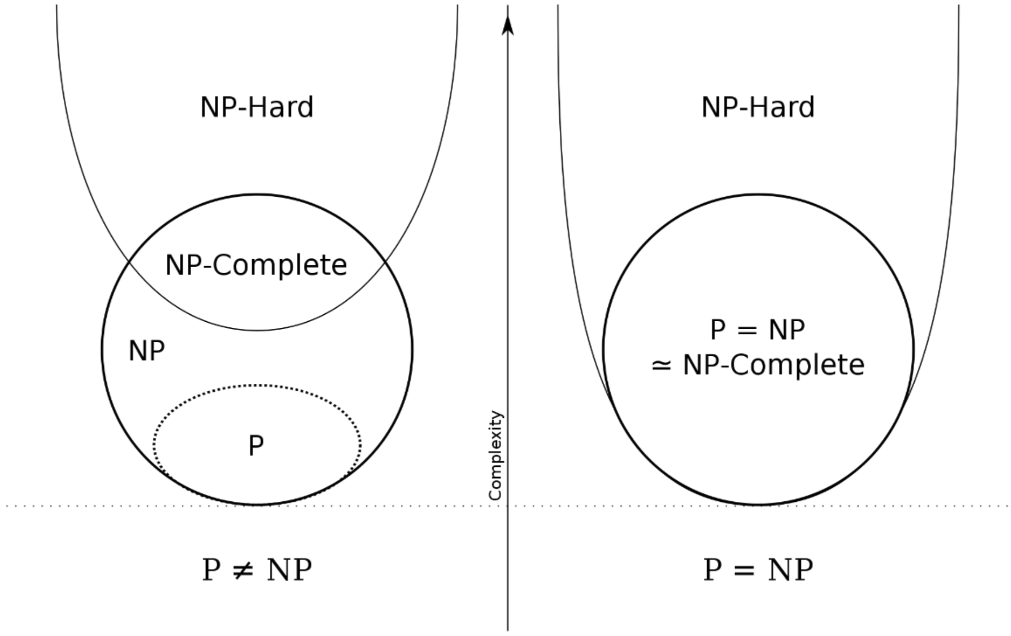
ייתכן שלכאורה קשה למצוא אלגוריתם אבל יש משהו בתנאים שהופך את הבעיה לטריוויאלית יחסית.

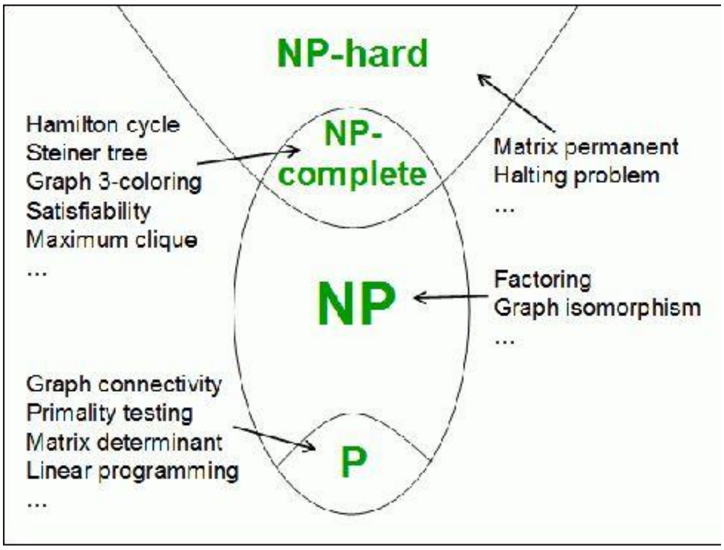
או שאם יש הגבלה של מספר קבוע אז לפעמים חיפוש שלם לא יהיה אקספוננציאלי.

אם לא הצלחנו את כל מה שנאמר לעיל אז מנסים למצוא אלגוריתם עם מ"ט אי דטר'.

(ניחושים + וידוא שהניחוש יצא נכון) וככה מוכיחים שזה ב*NP*.

לאחר מכן בוחרים את אחת מהשפות השלמות שהצגנו (הכי דומה לזו שבשאלה) ועושים רדוקציה פולינומית ממנה לשפה שבשאלה וזה כדי להוכיח שלמות.


NP-Hard
NP-Complete
NP
P
P ≠ NP
Complexity
NP-Hard
P = NP
≃ NP-Complete
P = NP



Here are some of my favorite **NP-Hard problems:**

SAT

Does a given boolean forumla, in CNF, have a satisfying assignment?

3-SAT

Does a given boolean forumla, in CNF with exactly three literals per clause, have a satisfying assignment?

Min Vertex Cover

In a given undirected graph, what is the (size of the) smallest subset of the vertices covering all of the edges?

Max Independent Set

In a given undirected graph, what is the (size of the) larges subset of the vertices having no edges in common?

Max Clique

What is the (size of the) largest complete subgraph of a given undirected graph?

Min Set Cover

Given a set SS and a collection of subsets of SS, what is smallest set of these subsets whose union is SS?

Min Hitting Set

Given a set SS and a collection of subsets of SS, what is smallest subset of SS containing at least one element from every subset?

Hamilton Path

Does a given graph have a Hamilton Path?

Hamilton Cycle

Does a given graph have a Hamilton Cycle?

Traveling Salesperson

What is the minimum cost Hamilton Cycle in a weighted, complete, graph?

Longest Path

What is the longest path between two given nodes in a weighted, undirected, graph?

Subset Sum

Does a given set of positive integers have a subset with sum kk?

Partition

Can a given set of positive integers be partitioned into two subsets each with the same sum?

3-Partition

Can a given set of 3n3n positive integers be partitioned into nn 3-element subsets each with the same sum?

Minesweeper

In a given Minesweeper configuration, is it safe to click on a particular square?

Sodoku

Does a given Sodoku puzzle have a solution?

סדפ לשפות ב NP :

אם נרצה להוכיח ששפה שייכת ל NP אז ניתן לעשות את זה ב 2 דרכים:

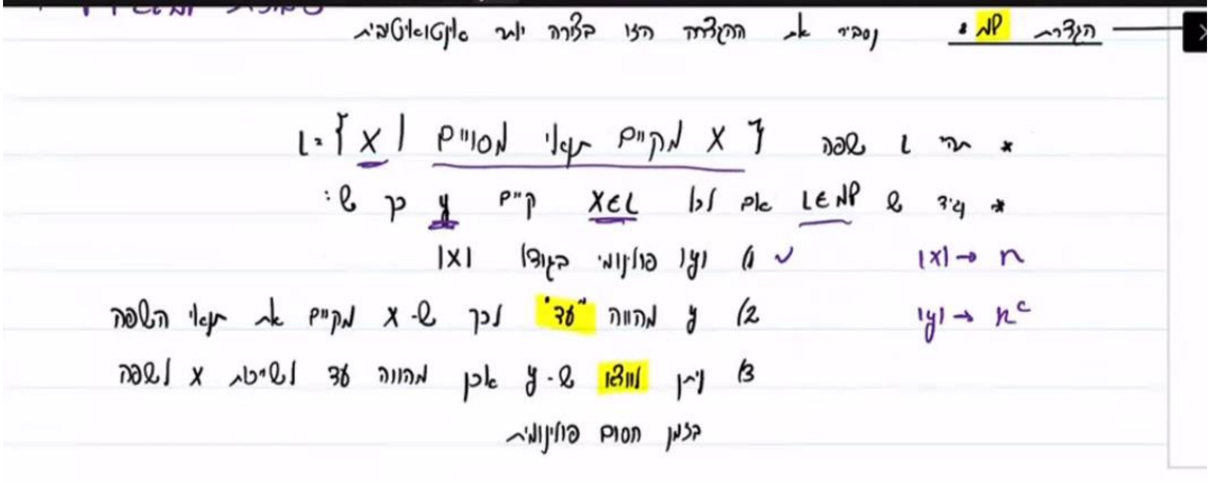
1. בניית מכונה אי דטרמיניסטית הפותרת את הבעיה בזמן ריצה פולינומיאלי (כמו שגיל הראה לנו) סד"פ:

N על <kelet>

א. נחש (X – משהו)
ב.
ג.
ד. (כל הבאים אח"כ זה תוודא שמה שניחשנו (ההנחה שניחש נכון) ניתן לבדיקה בזמן פולינום)
...

2. מגדירים שעבור בעיה מסוימת עם קלט- X נוכל להביא מאמת\מוודא Y שהוא בגודל פולינומי לקלט X (לדוגמא:
הבעיה: השפה מקבלת גרפים בהם קיים מעגל המילטוני.
הקלט: X גרף
המוודא\ מאמת: Y הוא קבוצת הקודקודים שמהווים מעגל המילטוני בקלט, גודל קבוצת הקודקודים פולינומי ביחס לגודל הקלט).

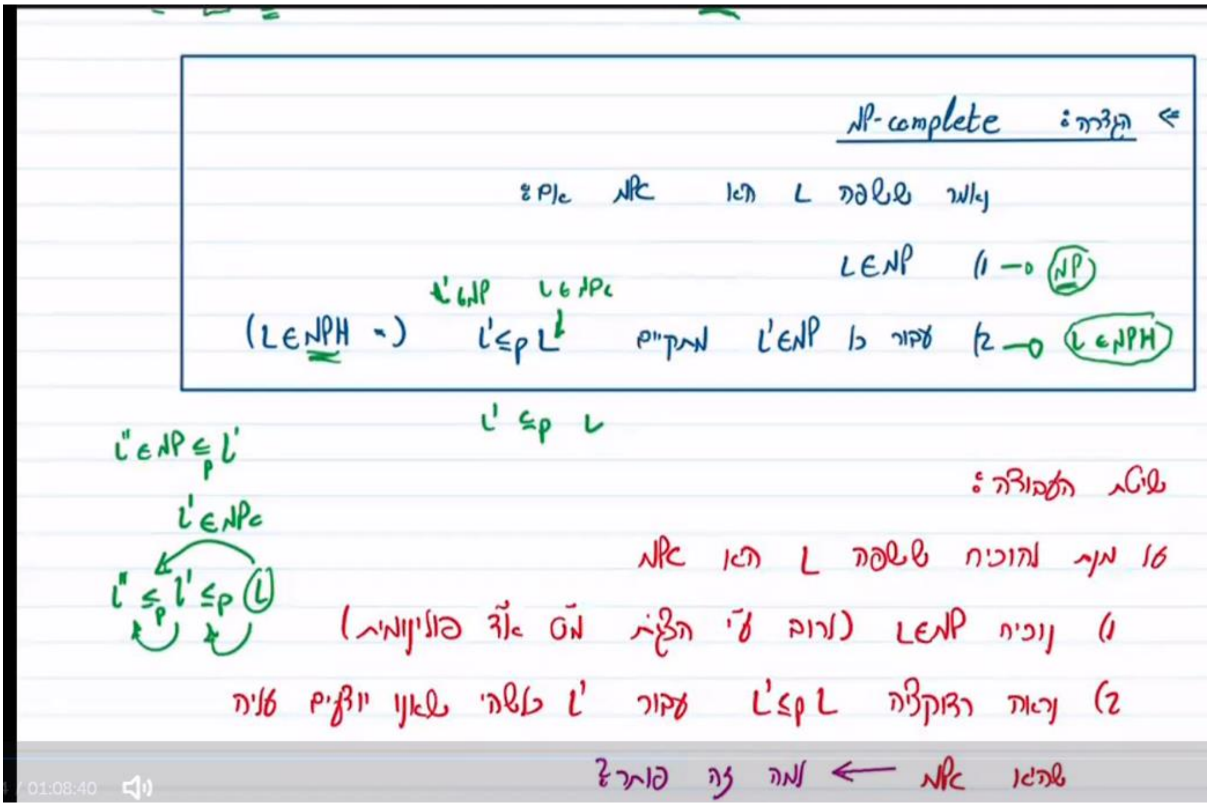
מראים בעצם שאם אני יכול לבנות מ"ט שמוודאת\מאמתת שאני שבהינתן Y אני יכול לבדוק בזמן פולינומי שהוא באמת קיים ב X אז הבעיה היא NP. (בניית אימות ומאמת)



NP COMPLETE

אם L היא NPC אז תן לי כל שפה בעולם 'L שהיא NP אז בהכרח ניתן לעשות לה רדוקציה ל L.

שיטת עבודה להוכיח ששפה היא NPC:



NPH – שפה נחשבת ל NPH אם כל שפה ב NP ישנה רדוקציה לשפה הנל. לא באמת נבדוק מול כל השפות ב NP פשוט נראה שפה שהיא NPC שיש לה רדוקציה לשפה שאנחנו רוצים להראות שהיא NPH

אלגוריתמי קירוב:

האם ייתכן שקיימת מכונה כזאתי?
בהינתן בעיה מסוימת האם קיימת פונקציית קירוב שלה או אלגוריתם קירוב עד רמה מסוימת.

לרוב התשובה תהיה לא.

כדי להוכיח שלא קיימת לרוב נוכיח בשלילה שכן קיימת ולהראות שמגיעים לאיזשהי סטייה או סתירה לעומת מה שאנחנו יודעים שהוא נכון.

לרוב השאלות בנושא יהיו הוכח\הפרך או הראו שזה בעיה פתוחה.

אם שפה גם NPC וגם שייכת ל P אז נגיע למסקנה ש P=NP וזה לא יתכן כי זו בעיה פתוחה.

<u>המשימה</u>: לבנות אלגוריתם המשתמש בקירוב ומכריע באמצעותו את השאלה.

נניח בשלילה שקיים G בהינתן גרף מכוון אני יוכל לדעת בקירוב את מספר המסלולים ההמילטונים אני רוצה להוכיח שאם אני יודע בקירוב שב +- 3 אני יוכל לדעת בוודאות שהוא קיים ל DHP .

שפות ממבחנים:

חישוביות:

$$L \in coRE \backslash R \ , \ L_{not-regular} = \{< M >| L(M) is\ not\ regular$$

????