תורת הסיבוכיות

סיכום ההרצאות והתרגולים בקורס "תורת הסיבוכיות" בטכניון

סיכם: שיר בן ישראל

<u>הקדמה</u>

על ציר הזמן:

עד 1960 עסקו בנושא של החלק הראשון של תורת החישוביות - אילו חישובים בכלל ניתן לבצע ואילו לא ניתן.

החל מ 1960 התחילו לעסוק בחלק השני של תורת החישוביות - סיבוכיות הזמן של החישובים, חישוב יעיל, $P=?N$ (תחילת שנות ה-70), רדוקציות, NP-שלמות וכו'.

- עובדות בסיסיות על סיבוכיות (זמן) וזיכרון.
- מה זה זיכרון יעיל? $PSPACE$ - כל השפות שאותן ניתן לזהות באמצעות מכונה שיש לה זיכרון פולינומי. DL - זיכרון לוגריתמי.
- קשרים בסיסיים: $DL \subseteq P \subseteq NP \subseteq PSPACE$. לא ידוע האם ההכלות הללו הן אמיתיות או שוויון. כלומר, לא ידוע האם $DL = P$ והאם $P = NP$ והאם $NP = PSPACE$. נגדיר רדוקציות ומושגים של שלמות כדי לטפל בבעיה הזאת.
- הכוח של אי דטרמיניזם: $PSPACE = NPSPACE$. (מקרה פרטי של משפט $SAVITCH$).
- משפטי היררכיה:
  - $DSPACE\left(n^2\right) \subset DSPACE\left(n^3\right)$ - כלומר יש בעיות שניתן לפתור בזיכרון $O\left(n^3\right)$ אבל לא בזיכרון $O\left(n^2\right)$.
  - $DL \subset PSPACE$ (שוב, הכלה ממש).

תורת הסיבוכיות מתפתחת במקביל לתורת האלגוריתמים.

- אלגוריתמים הסתברותיים:
  - הרחבה של מושג החישוב היעיל.
  - BPP - מחלקת השפות הניתנות לחישוב הסתברותי יעיל.
  - האם P = BPP?

- סיבוכיות של הוכחות:
  - NP: SAT: אפשר לחשוב על הבעיה כעל מערכת הוכחה. לדוגמה, אם רוצים לבדוק האם $\varphi \in SAT$, אפשר להסתכל על זה כעל משחק בין שני משתתפים - המוכיח P והמוודא V.
    המוכיח P לא מוגבל חישובית.
    המוודא V משתמש במכונה יעילה.
    מתקיימות שלמות ונאותות:
    - שלמות - המוכיח מסוגל להוכיח עבור כל פסוק ספיק שהוא אכן ספיק.
    - נאותות - המוכיח לא מסוגל להוכיח עבור פסוק לא ספיק, שהוא כן ספיק. המוודא לעולם לא ישתכנע.
  - ניתן להראות שכל בעיה שיש לה מערכת הוכחה כזאת, היא ב NP ולכל בעיה ב NP יש מערכת הוכחה כזאת.
  - $\overline{SAT}$ - אין דרך לשכנע בקלות שפסוק הוא לא ספיק.
  - הוכחה אינטראקטיבית IP: מערכת הוכחה כמו קודם רק שהפעם V יכול להיות הסתברותי ושני הצדדים יכולים לשלוח הודעות הלוך ושוב.
  - $\overline{SAT} \in IP$.
  - $IP = PSPACE$.
  - PCP - מערכות הוכחה "מתקדמות".
  - אפליקציה: קושי של קירובים.

- שונות:
  - בעיות ספירה. דוגמה: $(\#SAT)$ - בעיית הספירה של SAT.
    נתון φ ורוצים לדעת כמה השמות מספקות יש ל φ. הסיבוכיות של חישוב הפונקציה היא בין NP לבין PSPACE.
  - סיבוכיות של מעגלים: מעגלים בוליאניים עם שערי $\wedge,\vee$.
    מוטיבציה:
    - חומרה. רוצים לדעת כמה קטן יכול להיות המעגל שמממש פונקציה כלשהי.
    - חישוב מקבלי:
      - עומק החישוב - מהו המרחק הכי גדול משער קלט אל הפלט.
      - גודל החישוב - מספר השערים המקסימאלי ברמה מסוימת (כלומר ה-"רוחב" של המעגל).
    - קשר לחישוב ע"י מכונת טיורינג.

<u>תורת החישוביות:</u>

- מודל: מכונת טיורינג (מ"ט) - יכולה להיות חד-סרטית או רב-סרטית.
  יכולה להיות דטרמיניסטית (דטר') או אי-דטרמיניסטית (א"ד)
- שפה של מכונה:
  - עבור מ"ט דטר' {החישוב של M על x מסתיים ב q_A} $L(M)=\{x \mid q_A$.
  - עבור מ"ט א"ד: {קיים חישוב של M על x המסתיים ב q_A} $L(M)=\{x \mid q_A$.
- סיבוכיות זמן: $t:\mathbb{N}\to\mathbb{N}$. אומרים שלמ"ט M יש סיבוכיות זמן $t(n)$ אם לכל קלט x, M עוצרת תוך $t(|x|)$ צעדים. [אם M א"ד - מתקיים בכל מסלולי החישוב].
- $L\in DTIME(t(n))$ אם קיימת מ"ט **דטר'** M כך ש $L=L(M)$, וסיבוכיות הזמן של M היא $O(t(n))$.
- $L\in NTIME(t(n))$ אם קיימת מ"ט **א"ד** M כך ש $L=L(M)$, וסיבוכיות הזמן של M היא $O(t(n))$.
- $P=\bigcup_{c\geq 1} DTIME(n^c)$
- $NP=\bigcup_{c\geq 1} NTIME(n^c)$
- רדוקציות פולינומיות: כלי להשוות קושי של בעיות. $L_1\leq_p L_2$ אם הקושי של L_1 קטן או שווה לקושי של L_2.
  פורמאלית: קיימת פונקציה f ניתן לחישוב פולינומי המקיימת $x\in L_1 \Leftrightarrow f(x)\in L_2$.
- $L_1\in P \Leftarrow \begin{cases} L_2\in P \\ L_1\leq L_2 \end{cases}$ (משפט הרדוקציה).
- שפה L היא NP-שלמה ($L\in NPC$) אם $L\in NP$ וגם לכל $L'\in NP$ מתקיים $L'\leq_p L$.
- $P=NP \Leftarrow \begin{cases} L\in NPC \\ L\in P \end{cases}$
  - מספיק להראות שפה NP-שלמה הנמצאת ב P בשביל להראות ש NP=P.

<u>סיבוכיות זיכרון:</u>
ההגדרה שהכרנו בקורס חישוביות לסיבוכיות הזיכרון של מכונה, היא התא המקסימאלי שהמכונה מגיע אליו. הבעיה בהגדרה הזאת היא שאי אפשר כך להגדיר מכונה הדורשת זיכרון הקטן מזיכרון ליניארי.

לכן נשתמש בהגדרה החדשה:
נשנה את המודל כך שהקלט ישב במקום אחד וזיכרון העבודה ישב במקום אחר.

מודל מ"ט "משופר":
המכונה היא בעלת סרט אחד לקלט, ועוד מספר סרטים עבור סרטי "עבודה" (זה לא משנה אם יש סרט עבודה אחד או יותר).

בסרט הראשון נמצא $\$x_1x_2x_3...x_n$ כאשר $x_1x_2x_3...x_n$ הוא הקלט ו $ הוא סימן מיוחד. סרט זה הוא **לקריאה בלבד**.

הזיכרון שהמכונה משתמשת בו הוא סכום התאים המקסימאליים שהמכונה מגיעה אליהם על סרטי העבודה.

סיבוכיות זיכרון: תהי $s:\mathbb{N}\to\mathbb{N}$.
אומרים שלמ"ט M (במודל המשופר) יש סיבוכיות זיכרון $s(n)$ אם לכל קלט x, סה"כ התאים בהם מבקרים בסרטי העבודה $\geq s(|x|)$.
[אם M א"ד - כנ"ל, בכל מסלולי החישוב].

<u>מחלקות הזיכרון:</u>

- $L\in DSPACE(s(n))$ אם קיימת מ"ט דטר' כך ש $L=L(M)$ ו M היא בעלת סיבוכיות זיכרון $O(s(n))$.
- $L\in NSPACE(s(n))$ אם קיימת מ"ט א"ד כך ש $L=L(M)$ ו M היא בעלת סיבוכיות זיכרון $O(s(n))$.
- $PSPACE=\bigcup_{c\geq 1}DSPACE(n^c)$

<u>קשרים בסיסיים:</u>

- $DTIME(t(n))\subseteq DSPACE(t(n))$ אם המכונה מסתפקת ב $t(n)$ זמן אז מספיק לה גם $t(n)$ זיכרון.
- באופן דומה עבור $NTIME(t(n))\subseteq NSPACE(t(n))$.
- $DSPACE(s(n))\subseteq DTIME\left(2^{O(s(n))}\right)$ - אם מספיק זיכרון $s(n)$ אז מספר הקונפיגורציות שהיא יכולה להיות בהן הוא בסדר גודל של $2^{O(s(n))}$. לכן מספיק להריץ את המכונה שרצה במקום $s(n)$ למשך $2^{O(s(n))}$ צעדים ואם היא עדין לא עצרה אז היא נכנסה ללולאה אין-סופית ולכן אפשר לעצור ולדחות. אם היא עצרה לפני זה, אז בוודאי שהזמן הדרוש הוא פחות מ $2^{O(s(n))}$.
- מסקנה: $P\subseteq PSPACE\subseteq EXPTIME$, $NP\subseteq PSPACE$.

דוגמה 1:

$$L=\left\{x\#y \mid x,y\in\{0,1\}^{*}, x=y\right\}$$

פתרון טריוויאלי (אבל לא הכי יעיל במקום) - העתק את x לסרט העבודה, ואז השווה בין x לבין y.
סיבוכיות הזמן והמקום היא ליניארית.
פתרון אלטרנטיבי - עבור כל $1\le i\le|x|$ השווה את x_i עם y_i ואם כולם שווים - קבל. אחרת - דחה.
סיבוכיות הזמן - $O\left(n^2\right)$.

סיבוכיות מקום - יש לזכור את i, ולכן צריך $\log_2\left(|x|\right)$ ביטים. לכן הזיכרון הוא $O\left(\log n\right)$.
נראה שיש $TRADE-OFF$ בין זיכרון לבין זמן. (עבור הבעיה הספציפית הזאת, אפשר להוכיח זאת).

דוגמה 2:

$CON=\{G,s,t \mid$ t אל s מ מסלול מכוון יש בגרף, s,t הם צמתים, n צמתים עם G} הוא גרף מכוון
$CON\in DSPACE\left(n\log n\right)$ - עוברים על כל $n!$ הפרמוטציות של הצמתים, ובודקים האם אחת מהן מהווה מסלול מתאים. יש צורך ב $\log\left(n!\right)=O\left(n\log n\right)$ ביטים בשביל לשמור את מספר הפרמוטציה.
אפשרות אחרת היא לבצע DFS מצומת s עד שנמצא את t. הזיכרון: $O\left(n\log n\right)$ - שמירה של מסלול בגרף.

$CON\in NSPACE\left(\log n\right)$:
האלגוריתם האי-דטרמיניסטי:
1. $v=s$
2. נחש צומת u. אם $(v,u)\notin E$ דחה.
3. אחרת אם $u=t$ קבל, אחרת $v=u$ וחזור ל 2.

נכונות - אם יש מסלול, אז באחד מהניחושים המכונה תעצור ותקבל.
יש לזכור בכל שלב רק את v ולכן יש צורך ב $O\left(\log n\right)$ מקום.

ממשפט SAVITCH: $CON\in DSPACE\left(\log^2 n\right)$.

<u>תזכורת:</u>

$DSPACE\left(s(n)\right)$ - אוסף השפות שניתן לזהות במ"ט **דטר'** בסיבוכיות זיכרון $O\left(s(n)\right)$.

$NSPACE\left(s(n)\right)$ - אוסף השפות שניתן לזהות במ"ט **א"ד** בסיבוכיות זיכרון $O\left(s(n)\right)$.

בסיבוכיות זיכרון מדובר על הזיכרון הדרוש לסרטי העבודה ולא לסרט הקלט.

$$PSPACE=\bigcup_{c\geq 1}DSPACE\left(n^c\right) \quad , NPSPACE=\bigcup_{c\geq 1}NSPACE\left(n^c\right)$$

<u>הגדרה:</u>

פונקציה $s(n)$ נקראת **פונקצית זיכרון** אם קיימת מ"ט M בעלת זיכרון $O\left(s(n)\right)$ שעל קלט 1^n (כלומר מחרוזת של n 1-ים), פולטת את $s(n)$.

פונקציה $s(n)$ נקראת **פונקצית שעון** אם קיימת מ"ט M בעלת זמן $O\left(s(n)\right)$ שעל קלט 1^n (כלומר מחרוזת של n 1-ים), פולטת את $s(n)$.

דוגמאות לפונקציות זיכרון: $\log^c n$, n^c , 2^n.

פונקציות שלא ניתנות לחישוב אינן פונקציות שעון ואינן פונקציות זיכרון.

משפט *SAVITCH*:

המשפט עוסק בכוח של אי-דטרמיניסטים בהקשר של זיכרון. בהקשר של זמן, אין לנו מושג קלוש לגבי הכוח של אי-דטרמיניסטים.

לכל פונקצית זיכרון $s(n)$ המקיימת $s(n)\geq \log n$:

$NSPACE\left(s(n)\right)\subseteq DSPACE\left(s^2(n)\right)$ - תמורת מחיר ריבועי בסיבוכיות, אפשר להיפטר מאי הדטרמיניזם.

<u>מסקנה 1:</u> $con\in DSPACE\left(\log^2 n\right)$.

תזכורת: *con* - $\{G,s,t\}$ אוסף של שלשות המכילים גרף G מכוון ושני צמתים t,s, כאשר יש מסלול מכוון מ s אל t.

<u>הוכחת מסקנה 1:</u>

ראינו ש $con\in NSPACE\left(\log n\right)$, ע"פ משפט *SAVITCH* מתקיים:

$NSPACE\left(\log n\right)\subseteq DSPACE\left(\log^2 n\right)$. לכן $con\in DSPACE\left(\log^2 n\right)$.

<u>מסקנה 2:</u> $PSPACE=NPSPACE$.

<u>הוכחה:</u>

כיוון ראשון: $PSPACE\subseteq NPSPACE$. טריוויאלי - אם אפשר לבצע את החישוב במכונה דטר' אז בוודאי שאפשר לבצע אותו במכונה א"ד בעלת אותה דרישת זיכרון.

כיוון שני: $NPSPACE\subseteq PSPACE$

$$PSPACE\underset{*}{=}\bigcup_{c\geq 1}DSPACE\left(n^c\right)\subseteq\bigcup_{c\geq 1}DSPACE\left(n^{2c}\right)\underset{**}{\subseteq}\bigcup_{c\geq 1}NSPACE\left(n^c\right)\underset{*}{=}NPSPACE$$

* - ע"פ הגדרה.

** - ע"פ משפט *SAVITCH*.

הוכחת משפט $SAVITCH$:

דוגמה עבור הוכחת $con \in DPSACE\left(\log^2 n\right)$:

נתאר פרוצדורה: $Reach(u,v,l)$ שמחזירה $True$ אם ניתן להגיע בגרף G מ u ל v ע"י מסלול שאורכו לכל היותר l.

באמצעותה נחשב את $Reach(s,t,n)$, כאשר n הוא מספר הצמתים בגרף.

$Reach(u,v,1) = True$ - אם $u = v$ או ש $(u,v) \in E$. ניתן לבדיקה בזיכרון לוגריתמי.

למה: להגיע מ u ל v תוך l צעדים, אפשרי אם ורק אם קיים צומת w כך שאפשר להגיע מ u אל w תוך $\frac{l}{2}$ צעדים וכן אפשר להגיע מ w אל v תוך $\frac{l}{2}$ צעדים.

מסקנה: על מנת לממש את $Reach(u,v,l)$ נוכל לבדוק באופן רקורסיבי לכל $w \in V$ האם

$$reach\left(u,w,\left\lceil \frac{l}{2} \right\rceil\right) \wedge reach\left(w,v,\left\lfloor \frac{l}{2} \right\rfloor\right) = True.$$

הנכונות מידית מהלמה.

זיכרון: מספר הרמות של הרקורסיה הוא $O(\log n)$.

בכל רמה של הרקורסיה יש לשמור לכל היותר את u,v,w, (אפשר לשמור אפילו פחות) הדורשים כל אחד $O(\log n)$ זיכרון.

לכן סה"כ יש צורך בשימוש ב $O\left(\log^2 n\right)$ זיכרון למימוש $Reach(u,v,l)$.

כמה זמן לוקח לאלגוריתם לעבוד? $time(l) = n \cdot time\left(\frac{l}{2}\right)$ - בכל שלב עוברים על כל הצמתים ועבור כל אחד מהם קוראים לפונקציה באופן רקורסיבי. $time(l) = n \cdot time\left(\frac{l}{2}\right) = n^{\log n}$.

חסכנו בזיכרון ונענשנו בסיבוכיות זמן גרועה.

הוכחת משפט $SAVITCH$, **המקרה הכללי**:

נתונה $L \in NSPACE\left(s(n)\right)$.

לכן קיימת מ"ט M א"ד כך ש $L(M) = L$ המשתמשת בזיכרון $O\left(s(n)\right)$.

כלומר לכל מילה w, $w \in L \Leftrightarrow$ קיים מסלול מקבל של M על w.

גרף קונפיגורציות של M על w:

צמתי הגרף הם קונפיגורציות - כולל את המצב של M, את מיקום הראשים ואת תוכן זיכרון העבודה. אין צורך לשמור את הקלט.

מספר הצמתים: לכל היותר כמספר הקונפיגורציות: $B = 2^{O(s(n))}$.

(השתמשנו בהנחה $s(n) \geq \log n$, כי שמירת מיקום הראשים דורשת $O(\log n)$).

קשתות: (c_1, c_2) היא קשת מכוונת בגרף אם בחישוב של M על w, ניתן לעבור בצעד אחד מקונפיגורציה c_1 אל קונפיגורציה c_2. (מכיוון ש M א"ד, יכול להיות שמהצומת- c_1 ניתן לעבור לכמה קונפיגורציות אחרות ולא רק לאחת יחידה).

על השאלה האם $(c_1, c_2) \in E$ ניתן לענות בזיכרון $O\left(s(n)\right)$:

נניח ש c_1 נמצא בסרט אחד ו c_2 נמצא בסרט שני: בודקים שהתוכן של הסרטים זהה פרט למקום אחד, הנמצא ליד הראש, ושהתנועה אחת מתאימה לאופן העבודה של M, ע"פ סרט הקלט.

קיימת קונפיגורציה קבועה (לא תלויה ב w) - c_{Start} שבה תמיד מתחיל החישוב.

ניתן להניח שקיימת קונפיגורציה מקבלת אחת - בהינתן M, ניתן להמיר אותה במ"ט שכשאר היא רואה שהיא אמורה להגיע ל q_{Accept}, היא מרוקנת את תוכן הסרטים, מזיזה את הראשים לקצה השמאלי, ורק אז נכנסת ל q_{Accept}.
לכן נניח בה"כ שקיימת קונפיגורציה מקבלת יחידה c_{Accept}.

מ"ט דטר' M: בהינתן קלט w תבדוק ביחס לגרף הקונפיגורציות של M על w (אך מבלי לבנות את הגרף בשום שלב), האם $Reach\left(c_{Start}, C_{Accept}, B\right)$ מתקיים.
אופן הפעולה של $Reach$ דומה ל $Reach$ שהוצגה בפתרון בעיית con.
בפרט: $Reach\left(c_1, c_2, 1\right)$ - ניתן למימוש ב $O\left(s\left(n\right)\right)$ זיכרון.

$Reach\left(c_1, c_2, l>1\right) = True$ אמ"מ קיים c_3 $Reach\left(c_1, c_3, \frac{l}{2}\right)$ וגם $Reach\left(c_3, c_2, \frac{l}{2}\right)$.

המעבר על כל קונפיגורציות הביניים האפשריות דורש חישוב של $S\left(n\right)$. העובדה שהיא פונקצית זיכרון, מבטיחה שזה יהיה אפשרי.

סיבוכיות הזיכרון: עומק הרקורסיה הוא $O\left(\log B\right) = O\left(s\left(n\right)\right)$ זיכרון ובכל שלב ברקורסיה מחזיקים $O\left(s\left(n\right)\right)$ זיכרון, ולכן בסה"כ צריך $O\left(s^2\left(n\right)\right)$ זיכרון.

<u>השפה TQBF:</u>

הגדרות:

נוסחת QBF היא נוסחה מהצורה $Q_1x_1Q_2x_2...Q_nx_n\varphi(x_1,x_2,...,x_n)$ כאשר $Q_i \in \{\exists,\forall\}$ ו φ הוא פסוק CNF התלוי במשתנים $x_1,x_2,...,x_n$ ובקבועים $0,1$.

$\{\psi$ היא נוסחת CNF שערך האמת שלה הוא T$\}=TQBF$.

$\{\psi$ היא נוסחה (לא בהכרח CNF) שערך האמת שלה הוא $T\}=TQBF'$

טענה: $TQBF \in PSPACE$, $TQBF' \in PSPACE$.

האלגוריתם: (רקורסיבי).

הרקורסיה היא על m, מספר הכמתים.

אם $m=0$ אז אין אף כמת. לכן לכל הקלטים של ψ כבר הושמו ערכים - 0 או 1. לכן קל לחשב את ערך האמת בזמן פולינומי (ואפילו ליניארי), ולכן גם במקום פולינומי.

אם $m \geq 1$ אז $\psi = Qx\psi'$. נחשב באופן רקורסיבי את $\psi'|_{x=0}$ ואת $\psi'|_{x=1}$ וע"פ הכמת אז:

אם $Q=\forall$ אז נחזיר T אם ורק אם שני הערכים הם T.

אם $Q=\exists$ אז נחזיר T אם ורק אם לפחות אחד מהערכים הוא T.

נכונות: נובעת באופן מידי מהגדרת ערך האמת של נוסחה עם כמתים.

זיכרון: עומק הרקורסיה הוא m ובכל צעד יש צורך היותר במספר תאים כאורך הנוסחה- n.

לכן: $space(m)=O(m\cdot n)$.

זמן: אקספוננציאלי.

$PSPACE$-שלמות:

ידוע ש $P \subseteq NP \subseteq PSPACE$ אולם לא ידוע האם $P=NP$ וכן האם $NP=PSPACE$.

שפה L היא $PSPACE$-שלמה אם מתקיימים התנאים הבאים:

1. $L \in PSPACE$.
2. לכל $L' \in PSPACE$ קיימת רדוקציה זמן פולינומית $L' \leq_P L$.

אם $L_1 \leq_P L_2$ וגם $L_2 \in P$ אז $L_1 \in P$.

אם $L_1 \leq_P L_2$ וגם $L_2 \in NP$ אז $L_1 \in NP$.

<u>מסקנה:</u>

אם L היא $PSPACE$-שלמה וגם $L \in P$ אז $P=PSPACE$.

אם L היא $PSPACE$-שלמה וגם $L \in NP$ אז $NP=PSPACE$.

<u>משפט:</u> $TQBF$ ו $TQBF'$ הן $PSPACE$-שלמות.

<u>הוכחה:</u>

תזכורת (הרצאה 12 בחישוביות): משפט $cook$: SAT (אוסף פסוקי ה CNF הספיקים) היא שפה NP-שלמה.

תהי $L \in NP$. צ"ל: $L \leq_P SAT$

תהי M מ"ט א"ד פולינומית מתאימה ל L.

ההוכחה מראה: בהינתן קלט w איך לבנות פסוק CNF, φ_w שמקיים $w \in L(M) \Leftrightarrow \varphi_w \in SAT$.

מסתכלים על החישוב בתור טבלה, כאשר כל שורה היא קונפיגורציה בחישוב.

מספר השורות הוא זמן החישוב - כמה קונפיגורציות דרושות בשביל להגיע מהקונפיגורציה התחילית אל הקונפיגורציה הסופית.

רוחב כל שורה הוא הזיכרון הדרוש למכונה.

$$\varphi_w : \exists c_0 c_1 ... c_t \left(c_0 = c_{start}, c_t = c_{Accept}, \forall c_i \mid -c_{i+1} \right)$$

נסתכל כעת על ההבדלים בין משפט *cook* למשפט שאנו מוכיחים.
נתונה $L \in PSPACE$ וצ"ל: $L \leq_P TQBF$'.
$L \in PSPACE$ לכן קיימת מ"ט דטר' M בעלת זיכרון פולינומי מתאימה. (הזמן אקספוננציאלי).
לכן לא ניתן לכתוב טבלת חישוב כנ"ל.
רוצים לחשב $Reach\left(c_{Start}, C_{Accept}, B = 2^{O(n^c)} \right)$?

<u>ניסיון 1 (כושל):</u> רוצים לבנות נוסחת $TQBF$' שמחזירה T אמ"מ מתקיים $Reach(c_1, c_2, l)$.
מקרה הבסיס $l = 1$ מתואר כבר במשפט *cook*: $c_i \mid -^1 c_{i+1}$.
$Reach(c_1, c_2, l)$: $\exists c_3 \left(Reach\left(c_1, c_3, \frac{l}{2} \right) \wedge Reach\left(c_3, c_2, \frac{l}{2} \right) \right)$.

<u>גודל הפלט:</u> $size(l) \geq 2 \cdot size\left(\frac{l}{2} \right) + O\left(n^c \right) \approx O\left(l \cdot n^c \right)$. זוהי סיבוכיות זמן גדולה מדי.

<u>ניסיון שני:</u>

$$Reach(c_1, c_2, l) \equiv \exists c_3 \forall c_4 \forall c_5 \left(\begin{pmatrix} (c_4 = c_1) \wedge (c_5 = c_3) \\ \vee (c_4 = c_3) \wedge (c_5 = c_2) \end{pmatrix} \rightarrow Reach\left(c_4, c_5, \frac{l}{2} \right) \right).$$

הנוסחה הנ"ל שקולה לנוסחה הקודמת (ההוכחה בקורס תורת השפות הפורמליות, הרצאה 11).

<u>נכונות:</u> נובעת מכך ששתי הנוסחאות שקולות.
<u>סיבוכיות:</u> $size\left(\frac{l}{2} \right) \leq size\left(\frac{l}{2} \right) + O\left(n^c \right) = O\left(n^c \cdot \log l \right)$.
$size(B) = O\left(n^{2c} \right)$.

<u>סיבוכיות זיכרון - תזכורת:</u>
דיברנו על $PSPACE$, על שלמות ב $PSPACE$.
ראינו את משפט $SAVITCH$: $NSPACE\left(s\left(n\right)\right)\subseteq\left(s^2\left(n\right)\right)$
אשר מראה בפרט ש $PSAPCE=NPSPACE$.

<u>משפטי היררכיה עבור זיכרון:</u>
תהינה f,g פונקציות, כאשר $f\left(n\right)\geq\log n$, $g\left(n\right)$ היא פונקצית זיכרון, $f\left(n\right)=o\left(g\left(n\right)\right)$,
כלומר $\lim_{n\to\infty}\frac{f\left(n\right)}{g\left(n\right)}=0$ (f יותר קטנה מ g).
אם כל התנאים הללו מתקיימים, אז $DSPACE\left(f\left(n\right)\right)\subset DSPACE\left(g\left(n\right)\right)$ - כלומר באמצעות $g\left(n\right)$ זיכרון, ניתן לבצע יותר מאשר ניתן לבצע באמצעות $f\left(n\right)$ זיכרון.

הוכחה: ההכלה טריוויאלית.
נשאר להוכיח שקיימות שפות ב $DSPACE\left(g\left(n\right)\right)$ אשר לא נמצאות ב $DSPACE\left(f\left(n\right)\right)$.
נוכיח זאת באמצעות ליכסון.
נבנה מ"ט U כך ש $L=L\left(U\right)$, שתהיה שונה מכל שפה ב $DSPACE\left(f\left(n\right)\right)$.

רוצים לעבור על כל המכונות M שיש להן זיכרון $f\left(n\right)$ ולוודא ש U "לא מסכימה" עם M (כלומר, מקבלת קלט ש M דוחה או להפך) על קלט אחד לפחות.

<u>בעיות:</u> (באדום, רעיון הפתרון)
- איך יודעים מיהן המכונות בעלות זיכרון $f\left(n\right)$? לא נדע - פשוט נעצור אם המכונה גולשת מהזיכרון.
- טכני: $f\left(n\right)<g\left(n\right)$ רק עבור n מספיק גדול (עבור n קטן, יכול להיות ש $f\left(n\right)>g\left(n\right)$).
נדאג שההבדל בין M ל U יהיה באינסוף קלטים ולכן ביניהם יש קלטים מספיק ארוכים.
- M היא בעלת זיכרון $c\cdot f\left(n\right)$ והקבוע c לא ידוע לנו.
- U עובדת בזיכרון $g\left(n\right)$.

U על קלט x באורך n:
1. מפרשים את x בצורה הבאה: $x=1^k0\left\langle M\right\rangle$, כלומר, רצף של 1-ים, אח"כ 0, ואח"כ קידוד של מכונה M כלשהי. אם לא ניתן לפרש את x באופן הנ"ל, אז דוחים מיד.
2. חשב את הערך $g\left(n\right)$ (g היא פונקצית זיכרון, לכן אין בעיה לבצע זאת).
3. נבצע סימולציה של M על x.
   3.א. אם M עוצרת ומקבלת אז U עוצרת ודוחה.
   3.ב. אם M עוצרת ודוחה אז U עוצרת ומקבלת.
   3.ג. אם M גולשת מ $g\left(n\right)$ זיכרון אז עוצרת ו... (זה לא משנה אם נדחה או נקבל).
   3.ד. אם M רצה יותר מ $2^{g\left(n\right)}$ צעדים, (כלומר M לא תעצור לעולם) אז עצור וקבל.

צעד 1 דורש זיכרון קבוע.
צעד 2 דורש לכל היותר $g(n)$ זיכרון מכיוון ש $g(n)$ היא פונקצית זיכרון.
צעד 3 - סימולציה של מכונה עם זיכרון $s(n)$ דורש $O(s(n))$ זיכרון, ולכן בהתבסס על 3.ג, $O(g(n))$ זיכרון מספיק לסימולציה.

כלומר קיבלנו: $L = L(U) \in DSPACE(g(n))$

נותר להראות שלא יתכן שהשפה L מתקבלת ע"י מ"ט שהזיכרון שלה הוא $f(n)$.
נניח בשלילה ש $L \in DSPACE(f(n))$. כלומר קיימת מ"ט M שמשתמשת בזיכרון $c \cdot f(n)$, יש לה לכל היותר $2^{\alpha f(n)}$ קונפיגורציות (עבור קבוע α כלשהו) והשפה שלה היא L.

יהי n מספיק גדול כך שמתקיימים התנאים הבאים:
1. $g(n) > c \cdot f(n)$. ברור שקיים כזה מכיוון ש $f(n) = o(g(n))$.
2. $g(n) > \alpha \cdot f(n)$.
3. $n > |\langle M \rangle|$.

יהי x באורך n מהצורה $1^k 0 \langle M \rangle$. (קיים x כזה בגלל דרישה מספר 3 על n).
נסתכל על הריצה של U על x:
את הבדיקה בצעד הראשון כמובן שנעבור וכך גם את החישוב שבצעד השני.
בצעד השלישי אין סיכוי שנגיע למצב 3.ג שבו M מנסה להשתמש בזיכרון גדול מ $g(n)$ מכיוון ש $g(n)$ גדול מ $c \cdot f(n)$ שהוא דרישת הזיכרון המקסימאלית של M.

אם M לא עוצרת על x אז מספר צעדיה יעבור בשלב כלשהו את $2^{\alpha \cdot f(n)}$ המקיים $2^{g(n)} > 2^{\alpha \cdot f(n)}$.
במקרה זה מתקיים ש $x \in L(U)$ אבל $x \notin L(M)$.

אם M כן עוצרת על x אז יתבצע שלב 3.א או 3.ב ובכל מקרה נקבל ש U ו M לא מסכימות על x.
לכן $L(M) \neq L$ בסתירה להנחה.

<u>שאלה:</u> האם מהעובדה $L \in C$ עבור מחלקת שפות C, נובע ש $\overline{L} \in C$?

$L \in C = DTIME(t(n))$ - התשובה חיובית - פשוט נחליף בין המצב המקבל למצב הדוחה.
$L \in C = DSPACE(s(n))$ - לפחות עבור $s(n)$ "נוחה", ניתן להניח שיש מ"ט שתמיד עוצרת, ולכן שוב אפשר להחליף בין המצב המקבל q_A למצב הדוחה q_R.
$L \in C = NTIME(t(n))$ - לא ידוע. לדוגמה, לא ידוע האם $NP = CO - NP$.

$L \in C = NSPACE\left(s(n)\right)$ - נוכיח את המשפט הבא:

משפט Immerman:

תהי $s(n)$ פונקצית זיכרון. אזי $L \in NSPACE\left(s(n)\right) \Rightarrow \bar{L} \in NSPACE\left(s(n)\right)$

במילים אחרות: $NSPACE\left(s(n)\right) = CO - NSPACE\left(s(n)\right)$.

<u>הערות:</u> השאלה הייתה פתוחה במשך 20 שנה, ונפתרה בערך בשנת 1985.

$$L \in NSPACE\left(s(n)\right) \xrightarrow{SAVITCH} L \in DSPACE\left(s^2(n)\right) \rightarrow \bar{L} \in DSPACE\left(s^2(n)\right)$$

מקרה פרטי: $con = \{(G,s,t) \mid G$ יש מסלול מכוון מ s ל t ב $\}$: $\overline{con} \in NSPACE\left(\log n\right)$

<u>הוכחה:</u> נראה אלגוריתם עבור $\overline{con}$ ואז, כדי להוכיח את המשפט עצמו, בהינתן $L \in NSPACE\left(s(n)\right)$, נשתמש במ"ט א"ד M בעלת זיכרון $s(n)$ ולכל קלט x נתבונן בגרף הקונפיגורציות של M על x (כמו שהראנו בהוכחת משפט $SAVITCH$).

$x \in \overline{con}$ אם ורק אם אין מסלול בגרף הקונפיגורציות של M על x מ C_{Start} ל C_{Accept}.

<u>צעד 1:</u> נניח בנוסף שנתון לנו מספר N שהוא מספר הצמתים "הישיגים" מ s בגרף.

<u>אלגוריתם א"ד:</u>

נחש בסדר עולה N צמתים.

לכל צומת u וודא ש u הוא לא t, וודא ע"י ניחוש מסלול מ s ל u שאכן u "ישיג".

אם כל הבדיקות עברו - נקבל, אחרת נדחה.

אם t איננו ישיג אז יש חישוב שבו מנחשים נכון את כל N הצמתים הישיגים וגם עבור כל אחד מהם, u, מנחשים את המסלול הנכון מ s ל u, אז במסלול הזה מקבלים.

אם t כן ישיג אז קיימים רק $N-1$ צמתים שונים מ t וישיגים מ s. לכן לפחות בדיקה אחת תיכשל בכל מסלול - לכן תמיד דוחים.

נגדיר: N_l = מספר הצמתים שהישיגים מ s ע"י מסלולים באורך קטן או שווה ל l.

כלומר, $N = N_n$.

ברור ש $N_0 = 1$.

נראה תהליך אינדוקטיבי שבו מחשבים את N_l מתוך N_{l-1}.

נתאר "חישוב" א"ד של N_l מתוך N_{l-1}.

"חישוב" - בכל מסלול, או שדוחים או שמחשבים ערך נכון <u>וכן</u> קיים מסלול בו מחושב הערך הנכון.

נותר לתאר את הפרוצדורה שבהינתן N_{l-1} "מחשבת" את N_l:

הרעיון: נעבור אחד-אחד על כל הצמתים בגרף. לכל אחד מהצמתים, ננחש האם ניתן להגיע אליו, ואז נבדוק את הניחוש.

האלגוריתם:

$N_l \leftarrow 0$
עבור כל צומת w בגרף, $1 \le w \le n$:
א. נחש אם w ישיגה ב l צעדים או לא.
ב. בדוק את הניחוש:
- אם הניחוש הוא "ישיגה" - נחש מסלול באורך קטן או שווה ל-l מ s אל w. אם הניחוש נכון, אז $N_l \leftarrow N_l + 1$

- אם הניחוש הוא "לא ישיגה" - נחש בזה אחר זה (בסדר עולה) את N_{l-1} הצמתים הישיגים ע"י מסלול באורך קטן או שווה ל-$l-1$. לכל אחד מהם, ודא שאכן הוא ישיג ב $l-1$ או פחות צעדים. ודא שהוא שונה מ w וגם שלא ניתן להגיע ממנו בצעד אחד אל w.
אם בדיקה כלשהי נכשלה - דחה.

<u>זיכרון:</u>
מונים - $O(\log n)$ ביטים. בכל רגע נתון צריך להחזיק את N_{l-1} ואת N_l. מחזיקים גם את l עצמו וגם מספר צמתים שכל אחד דורש $O(\log n)$ ביטים. יש להחזיק את הצומת w ועוד $O(1)$ צמתים נוספים בבדיקות הישיגות / אי ישיגות.
סה"כ - $O(\log n)$ זיכרון.

<u>מחלקות של זיכרון לוגריתמי:</u>
$DL \triangleq DSPACE(\log n) \subseteq P$ - נחשבת למחלקה של זיכרון יעיל.
$NL \triangleq NSPACE(\log n)$

לא ידוע האם $DL = P$.
האם יש דברים שניתן לחשב בזמן פולינומי וזיכרון לוגריתמי לא מספיק עבורם?

לא ידוע האם $DL = NL$. למשל, לא ידוע האם $con \in DPSACE(\log n)$.

<u>סיבוכיות זיכרון לוגריתמית:</u>

$$DL = DSPACE\left(\log n\right)$$

$$NL = NSPACE\left(\log n\right)$$

לא ידוע האם $DL = P$ וכן לא ידוע האם $DL = NL$.
בפרט לא ידוע האם $CON \in DL$.

<u>רוצים:</u> איזושהי רדוקציה "$\le$" כך ש $L_1 \in DL \Leftarrow \begin{cases} L_1 \text{"} \le \text{"} L_2 \\ L_2 \in DL \end{cases}$.
אי אפשר להשתמש ברדוקציה פולינומית $\le_p$ כי יכול להיות שהיא דורשת זיכרון פולינומי.

תזכורת (מהתרגול):
מ"ט לחישוב פונקציות (בהקשר של "סיבוכיות זיכרון") היא מ"ט רב סרטית עם
- סרט קלט לקריאה בלבד.
- סרט פלט לכתיבה בלבד.
- $k \ge 1$ סרטי עבודה.

סיבוכיות הזיכרון לוקחת בחשבון רק את סרטי העבודה.

(*) <u>עובדה:</u> אם f, g ניתנות לחישוב בסיבוכיות זיכרון לוגריתמית אז גם $g \circ f\left(x\right) \triangleq g\left(f\left(x\right)\right)$ ניתנת לחישוב בזיכרון לוגריתמי.
הבעיה היא שלא ניתן לשמור את כל $f\left(x\right)$ בשביל לחשב את $g\left(f\left(x\right)\right)$.
הפתרון הוא שבכל פעם ש M_g צריכה איזשהו ביט מהקלט שלה, אז מריצים מחדש את M_f ומחשבים את הביט הזה.

<u>הגדרה:</u> $L_1 \le_{\log}^{m} L_2$ אם קיימת פונקציה f הניתנת לחישוב בזיכרון לוגריתמי המקיימת:
$x \in L_1 \Leftrightarrow f\left(x\right) \in L_2$.

<u>אבחנות:</u>
1. $\forall L : L \le_{\log} L$
2. טרנזיטיביות: $L_1 \le_{\log} L_3 \Leftarrow \begin{cases} L_1 \le_{\log} L_2 \\ L_2 \le_{\log} L_3 \end{cases}$. זה נובע מהעובדה (*).
3. $L_1 \in DL \Leftarrow \begin{cases} L_1 \le_{\log} L_2 \\ L_2 \in DL \end{cases}$. זה נובע מהעובדה (*).

<u>הגדרה:</u> שפה L נקראת NL-שלמה אם מתקיימים שני התנאים הבאים:
א. $L \in NL$
ב. $\forall L' \in NL \quad L' \le_{\log} L$.

<u>מסקנה:</u> אם L היא NL-שלמה וגם $L \in DL$ אז $DL = NL$. זה נובע מאבחנה 3.

משפט: השפה CON היא NL-שלמה.
תזכורת: $CON = \{G,s,t \mid t$ קיים מסלול מכוון ב G מ s אל $\}$

הוכחה:
א. $CON \in NL$ - ראינו כבר - פשוט מנחשים מסלולים - בכל צעד מנחשים צומת יחיד. אין צורך לזכור יותר ממספר קבוע של צמתים והזיכרון הדרוש לשמירת כל צומת הוא לוגריתמי.
ב. תהי $L \in NL$ שפה כלשהי. נראה ש $L \le_{\log} CON$.

$L \in NL \Leftarrow$ קיימת מ"ט א"ד בעלת סיבוכיות זיכרון $s(n) = O(\log n)$ המקבלת את L:

$L = L(M)$. נניח בה"כ שלמכונה M יש קונפיגורציה מקבלת יחידה C_{Accept} ונסמן ב C_{Start} את הקונפיגורציה התחילית.
הרדוקציה המבוקשת תתאים לכל קלט x, פלט $f(x)$ שהוא מופיע עבור השפה CON כך שיתקיים $x \in L \Leftrightarrow f(x) \in CON$.

$f(x) \triangleq ($ גרף הקונפיגורציות של M על x $, C_{Start}, C_{Accept})$

נכונות: $x \in L$ אמ"מ יש ל M מסלול מקבל בחישוב על x וזה מתקיים אמ"מ יש מסלול בגרף הקונפיגורציות מ C_{Start} אל C_{Accept} וזה מתקיים אמ"מ $f(x) \in CON$.

סיבוכיות זיכרון:
- נניח לשם נוחות ש G מיוצג ע"י רשימת קשתות.
- עבור בזה אחר זה על כל זוגות הקונפיגורציות, C_1, C_2. לכל זוג כזה בדוק האם $C_1 \mid -^1_{M,x} C_2$. אם כן, כתוב את הקשת (C_1, C_2), לפלט.
- הזיכרון הדרוש: לשמירת הקונפיגורציות $O(\log n)$.
  לבדיקת $C_1 \mid -^1_{M,x} C_2$ גם $O(\log n)$.
  סה"כ $O(\log n)$.

סיכום ביניים: $DL \subseteq NL \subseteq P \subseteq PSPACE$
בנוסף, ידוע $DL \neq PSPACE$ (ע"פ משפט ההיררכיה).
לכן לפחות אחת משלוש ההכלות הנ"ל היא הכלה אמיתית.

<u>אובות - Oracles:</u>

רדוקציות: ראינו רדוקציות $\le$, $\le_p$, $\le_{\log}$. ליתר דיוק: $\le^m$, $\le_p^m$, $\le_{\log}^m$.

<u>מוטיבציה:</u>

1. אם $L_1 \in P \Leftarrow \begin{cases} L_1 \text{"} \le \text{"} L_2 \\ L_2 \in P \end{cases}$ - הרעיון הוא שעל מנת לבדוק האם $x \in L_1$ נחשב את $f(x)$, וניתן אותו כקלט למ"ט M_2 המחשבת את L_2, ונענה כמוה.

<u>הגדרה:</u> תהי A שפה. מ"ט M עם אוב לשפה A היא מ"ט עם 2 סרטים מיוחדים:
- סרט שאלות - סרט לכתיבה בלבד.
- סרט תשובות - סרט לקריאה בלבד.

ו- 2 מצבים מיוחדים: $q_?, q_!$.

M עובדת כמו מ"ט רגילה עם התוספת שכאשר M נכנסת למצב $q_?$ ו w הוא תוכן סרט השאלות, אז בצעד הבא קורים הדברים הבאים:
- w נמחק מסרט השאלות.
- על סרט התשובות נרשמת התשובה $A(w)$. התשובה היא 1 אם $w \in A$ ו 0 אם $w \notin A$.
- המכונה עוברת למצב $q_!$.

<u>הגדרה:</u> $L(M^A)$ היא אוסף המילים ש M עם אוב A מקבלת.
- סיבוכיות זמן - כרגיל (נזכור כי תשובה מתקבלת תוך צעד אחד).
- סיבוכיות זיכרון - כרגיל (לא לוקחים בחשבון את הסרטים המיוחדים).
- הרחבה למ"ט א"ד עם אוב A כרגיל **(האוב דטרמיניסטי גם במקרה זה)**.
- אם נקבע את M ונחליף את A ב B אז סביר ש $L(M^A) \ne L(M^B)$ ושסיבוכיות המכונה משתנה.

<u>הגדרה:</u>
1. P^A - אוסף כל השפות הניתנות לזיהוי ע"י מ"ט פולינומית דטרמיניסטית עם אורקל (אוב) ל A.
2. NP^A - אוסף כל השפות הניתנות לזיהוי ע"י מ"ט פולינומית אי-דטרמיניסטית עם אורקל (אוב) ל A.
3. $PSPACE^A$ - אוסף כל השפות הניתנות לזיהוי ע"י מ"ט דטר' בעלת זיכרון פולינומי עם אורקל (אוב) ל A.
4. $DTIME^A(n^5)$ - אוסף כל השפות הניתנות לזיהוי ע"י מ"ט דטר' בעלת זמן n^5.

<u>מוטיבציה:</u>

1. רדוקציות טיורינג:

$L_1 \le_p^T L_2$ אם קיימת מ"ט M המקבלת אורקל ל L_2 כך ש $L_1 = L(M^{L_2})$ ו M רצה בזמן פולינומי בהינתן L_2.

<u>טענה:</u> $L_1 \in P \Leftarrow \begin{cases} L_1 \le_p^T L_2 \\ L_2 \in P \end{cases}$ (הוכחה בסימולציה).

<u>הערה:</u> $\overline{SAT} \le_p^T SAT$ - בהינתן פסוק, ניתן אותו לאורקל בלי שינוי, ונענה הפוך ממנו.

$NP = CO - NP \Leftrightarrow \overline{SAT} \le_p SAT$

2. טכניקות "ניתנות לייחוס" - Relativized.
אולי: אם ניתן להוכיח $P^A = NP^A$ או $P^A \neq NP^A$ אז ניתן להסיק אותו הדבר על P מול NP.
(זה לא נכון, כפי שנראה בתרגול - קיימים A,B כך ש $P^A = NP^A$ וגם $P^B \neq NP^B$).

דוגמה: לכסון.
אם נוסיף אוב כלשהו A להוכחת משפט ההיררכיה ,שמתבצעת באמצעות לכסון, התוצאה לא תשתנה, כלומר משפט ההיררכיה מחזיק גם אם מוסיפים אוב A למכונות. מ

לכן לכסון זאת טכניקה ניתנת לייחוס.

מסקנה: לא ניתן להוכיח ע"י לכסון $P \neq NP$ (כי היה נובע ש $\forall A \ P^A \neq NP^A$ וזה לא נכון).

3. ההיררכיה הפולינומית.
נסתכל על השפות הנמצאות בין P לבין $PSPACE$ (אם יש כאלה בכלל, שהרי לא ידוע האם $P \subset PSPACE$ או $P = PSPACE$).

P^{SAT} - כל מה שאפשר לעשות ע"י מ"ט הרצות בזמן פולינומי ע"י אורקל היודע לחשב את SAT.
$\overline{SAT}, SAT \in P^{SAT}$.
$NP \subseteq P^{SAT}$ - בהינתן $L \in NP$ ורדוקציה f , $L \leq_p SAT$, על קלט x , חשב $\varphi = f(x)$ ושאל את האורקל האם φ הוא פסוק ספיק, וענה כמוהו.
$CO - NP \subseteq P^{SAT}$.
$P^{SAT} = P^{3COL}$ - אפשר לשים במקום SAT כל שפה אחרת NP -שלמה. נוכל לסמן: $P^{NP} \triangleq P^{SAT}$
$P^{SAT} \subseteq PSPACE$ - אפשר לסמלץ את האורקל של SAT בזיכרון פולינומי.

הגדרה: ההיררכיה הפולינומית:
נגדיר 3 היררכיות: $\Delta_0^P, \Delta_1^P, \Delta_2^P, \ldots$ $\Sigma_0^P, \Sigma_1^P, \Sigma_2^P, \ldots$ $\Pi_0^P, \Pi_1^P, \Pi_2^P, \ldots$
(בהמשך לא נרשום Δ_0^P אלא Δ_0 (נזניח את ה P).

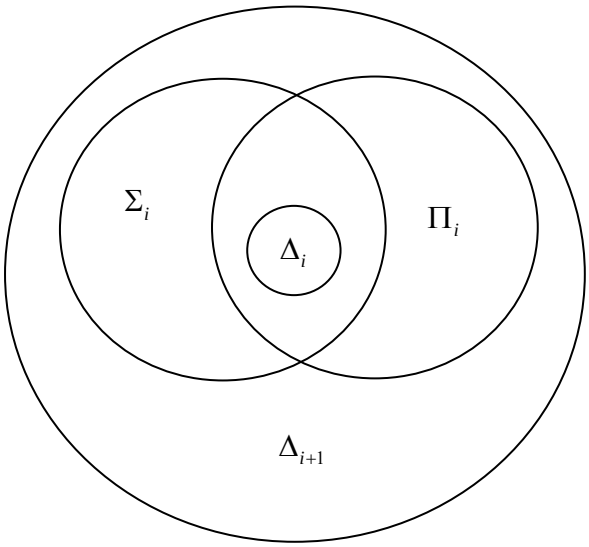
רמה 0 של ההיררכיה: $\Delta_0^P = \Sigma_0^P = \Pi_0^P = P$
רמה i+1 של ההיררכיה: $\Delta_{i+1} \triangleq P^{\Sigma_i}$, $\Sigma_{i+1} = NP^{\Sigma_i}$, $\Pi_{i+1} = CO - NP^{\Sigma_i}$.

דוגמאות:
$\Delta_1^P = P^{\Sigma_0} = P^P = P$
$\Sigma_1^P = NP^{\Sigma_0} = NP^P = NP$
$\Pi_1 = CO - NP$
$\Delta_2 = P^{NP}$ (ראינו קודם)

אבחנות:
1. $L \in \Sigma_i \Leftrightarrow \overline{L} \in \Pi_i$ - נשתמש במ"ט של L ונהפוך לה את המצב המקבל והמצב הדוחה.
2. $\Delta_i \subseteq \Sigma_i, \Pi_i$ - זאת מכיוון שמ"ט דטר' היא מקרה פרטי של מ"ט א"ד.
3. $\Sigma_i \subseteq \Sigma_{i+1}$ - פשוט לוקחים את הקלט, נותנים לאורקל ועונים כמוהו.
4. $\Pi_i \subseteq \Pi_{i+1}$.
5. $\Delta_i \subseteq \Delta_{i+1}$.

6. $\Sigma_i, \Pi_i \subseteq \Delta_{i+1}$

באופן ציורי:



<u>אבחנה:</u> $\Delta_i, \Sigma_i, \Pi_i \subseteq PSPACE$.
<u>ההוכחה:</u> באינדוקציה על i.

עבור המחלקה NP אנחנו מכירים את SAT: $\exists x R(x)$.
עבור המחלקה $CO-NP$ אנחנו מכירים את $\overline{SAT}$: $\forall x R'(x)$.
עבור המחלקה $PSPACE$ אנחנו מכירים את $TQBF$: $\forall x \exists y \forall z ... R(x, y, z, ...)$.
בהמשך נראה שעבור המחלקה Σ_2 מתאימה השפה $\exists x \forall y S(S, X)$.
עבור המחלקה Π_2 מתאימה השפה $\forall x \exists y T(x, y)$.

<u>תזכורת:</u>

- מ"ט עם אורקל / אוב. מסומן ע"י M^A.
- ההיררכיה הפולינומית:

רמה 0: $\Delta_0, \Sigma_0, \Pi_0 = P$.

רמה $i+1$:
$$\Delta_{i+1} = P^{\Sigma_i}$$
$$\Sigma_{i+1} = NP^{\Sigma_i}$$
$$\Pi_{i+1} = CO - NP^{\Sigma_i}$$

<u>תכונות בסיסיות של ההיררכיה הפולינומית:</u>

$$L \in \Sigma_i \Leftrightarrow \overline{L} \in \Pi_i$$

$$\ldots \subseteq \Delta_i \subseteq \begin{Bmatrix} \Sigma_i \\ \Pi_i \end{Bmatrix} \subseteq \Delta_{i+1} \subseteq \begin{Bmatrix} \Sigma_{i+1} \\ \Pi_{i+1} \end{Bmatrix} \subseteq \ldots$$

<u>דוגמה:</u>

בעיית המינימיזציה של נוסחאות CNF.

נתון: נוסחת CNF, φ.

מבוקש: נוסחת CNF, φ' השקולה ל φ הקצרה ביותר. כלומר $\forall x \in \{0,1\}^n : \varphi(x) = \varphi'(x)$.

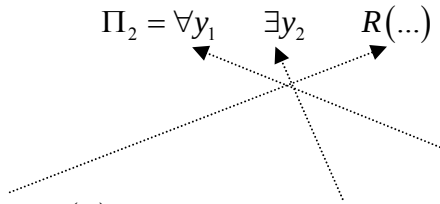
הבעיה הזאת יותר קשה מ SAT ויותר קשה מ $\overline{SAT}$: אם נרצה לפתור את בעיית SAT, כאשר נתון לנו פסוק φ - נמצא את φ' ואם הוא שונה מ $FALSE$ אז φ ספיק ואחרת הוא לא ספיק.

השפה המתאימה: $MIN = \{\varphi \mid \varphi$ הוא CNF מינימאלי$\}$
כלומר לא קיים פסוק φ' שקול ל φ וקצר יותר.

לא ידוע אם $MIN \in P, NP, CO-NP$.

<u>טענה</u>: $MIN \in \Pi_2 = CO\_NP^{NP}$.

ראינו שאפשר להסתכל על המחלקות באופן הבא:
$$NP = \exists y R(x,y)$$
$$CO-NP = \forall y R(x,y)$$



$$\Sigma_{17} = \exists y_1 \forall y_2 \ldots \exists y_{17} R(\ldots)$$
$$\Pi_{30} = \forall y_1 \ldots \exists y_{30} R(\ldots)$$

<u>אבחנה:</u> $\varphi \in MIN \Leftrightarrow$ **לכל** φ' קצרה מ φ **קיימת** השמה x כך ש $\varphi(x) \neq \varphi'(x)$.

<u>הוכחה:</u> נגדיר שפה $L_{\neq} \triangleq \{\varphi, \varphi' \mid \varphi, \varphi'$ הן נוסחאות CNF שאינן שקולות$\}$
$L_{\neq} \in NP$ - מנחשים השמה x ומוודאים ש $\varphi(x) \neq \varphi'(x)$.

נתאר מ"ט $M^{L_{\neq}}$ פולינומית וקו-אי-דטרמיניסטית.

(קו-אי-דטרמיניסטי - קלט ששייך לשפה היא מקבלת בכל המסלולים, וקלט שלא שייך לשפה היא דוחה לפחות באחד מהמסלולים).

$M^{L_{\neq}}$ על קלט φ:
1. נחש φ' קצר מ φ.
2. שאל את האורקל על φ,φ' וענה כמוהו.

<u>מתקיים:</u> הניחוש לוקח זמן פולינומי ושאלת האורקל לוקחת זמן קבוע. לכן $M^{L_{\neq}}$ היא פולינומית.

אם $\varphi \in MIN$ אז לא קיימת φ' קצרה מ φ ושקולה לה לכן לכל ניחוש φ' קצר מ φ האורקל מקבל ולכן גם המכונה $M^{L_{\neq}}$ מקבלת.
אם $\varphi \notin MIN$ אז קיימת φ' קצרה ושקולה לה, ולכן במסלול שבו $M^{L_{\neq}}$ מנחשת את φ', האורקל דוחה, ולכן גם המכונה $M^{L_{\neq}}$ דוחה במסלול הזה.

<u>טענה</u>: לכל i, המחלקה Σ_i סגורה לאיחוד. כלומר אם $L_1, L_2 \in \Sigma_i$ אז גם $L_1 \cup L_2 \in \Sigma_i$.
<u>הוכחה:</u> באינדוקציה על i.
<u>בסיס:</u> $i = 0$, לכן $\Sigma_i = \Sigma_0 = P$ וכידוע P סגורה לאיחוד.
<u>צעד:</u> נתונות $L_1, L_2 \in \Sigma_{i+1}$ וצריך להוכיח ש $L_1 \cup L_2 \in \Sigma_{i+1}$.
הנחת האינדוקציה: $L_a, L_b \in \Sigma_i \Rightarrow L_a \cup L_b \in \Sigma_i$.

ע"פ ההגדרה, קיימת מ"ט אי-דטר' פולינומית $M_1^{A_1}$ עבור L_1,
ע"פ ההגדרה, קיימת מ"ט אי-דטר' פולינומית $M_2^{A_2}$ עבור L_2,
עבור $A_1, A_2 \in \Sigma_i$.
נגדיר: $A = \{0x \mid x \in A_1\} \cup \{1y \mid y \in A_2\}$.
ברור ש $\{0x \mid x \in A_1\}, \{1y \mid y \in A_2\} \in \Sigma_i$ - פשוט משמיטים את האות הראשונה ומריצים את המ"ט של A_1 או A_2 בהתאמה.
<u>אבחנה:</u> $A \in \Sigma_i$ כאיחוד של שתי שפות ב Σ_i, ע"פ הנחת האינדוקציה.

נתאר מ"ט א"ד פולינומית M^A עבור $L = L_1 \cup L_2$ על קלט w:
- המכונה מנחשת $i \in \{1,2\}$ שעבורו $w \in L_i$.
- מריצה את המכונה המתאימה ($M_1^{A_1}$ או $M_2^{A_2}$) על w, כאשר אם M_1 פונה לאורקל A_1 עם שאלה x, אז M מסמלצת זאת ע"י שאלת $0x$ לאורקל A, ובאופן דומה אם M_2 פונה לאורקל A_2 עם שאלה y, אז M מסמלצת זאת ע"י שאלת $1y$ לאורקל A.

<u>מתקיים:</u> M^A פולינומית (כי $M_1^{A_1}, M_2^{A_2}$ כאלה) וברור ש $L(M^A) = L_1 \cup L_2$.
מש"ל.

ראינו ש: $\ldots \subseteq \begin{Bmatrix}\Sigma_i \\ \Pi_i\end{Bmatrix} \subseteq \Delta_{i+1} \subseteq \begin{Bmatrix}\Sigma_{i+1} \\ \Pi_{i+1}\end{Bmatrix} \subseteq \ldots$

נשאלת השאלה האם מדובר בהכלות אמיתיות או בשוויון.

מסתבר שאם באיזשהו מקום באמצע, יש שוויון, אז מכאן ואילך יש תמיד שוויון.

<u>משפט:</u> אם קיים $i \geq 1$ עבורו $\Sigma_i = \Pi_i$ אז "ההיררכיה קורסת לרמה ה- i", כלומר לכל $j \geq i$ מתקיים $\Sigma_j = \Sigma_i$ (ואז גם $\Pi_j = \Sigma_i$ וגם $\Delta_{j+1} = \Sigma_i$).

<u>מסקנות:</u>

אם $NP = CO\_NP$ אז ההיררכיה קורסת לרמה 1.

אם $P = NP$ אז ההיררכיה קורסת לרמה 1.

<u>הוכחת המשפט:</u> מספיק להוכיח ש $\Sigma_i = \Pi_i$ אז $\Sigma_{i+1} = \Sigma_i$.

לכל j גדול יותר נקבל ש $NP^{\Sigma_i} = \Sigma_{i+1} = \Sigma_i$ $\underset{\text{אינדוקציה}}{\Rightarrow}$ $\Sigma_j = NP^{\Sigma_{j-1}}$.

תהי $L \in \Sigma_{i+1}$. נראה ש $L \in \Sigma_i$.

כלומר המטרה היא להראות מ"ט $\hat{M}^A$ כך ש $A \in \Sigma_{i-1}$ ו $L\left(\hat{M}^A\right) = L$.

לכן קיימת מ"ט M^B א"ד פולינומית ששפתה L, $L \in \Sigma_{i+1}$ ו $B \in \Sigma_i$.

ולכן $B \in \Sigma_i$ ולכן קיימת מ"ט $M_1^{A_1}$ א"ד פולינומית ששפתה $B \in NP^{\Sigma_{i-1}}$.

היינו רוצים לסמלץ את שאלת האורקל B באמצעות $M_1^{A_1}$ אבל הבעיה היא ש $M_1^{A_1}$ היא א"ד.

ולכן $\overline{B} \in \Pi_i$ לכן ע"פ ההנחה $B \in \Sigma_i$.

לכן קיימת מ"ט $M_2^{A_2}$ א"ד ששפתה $\overline{B}$.

<u>נגדיר:</u> $A = \{0x \mid x \in A_1\} \cup \{1y \mid y \in A_2\}$ וע"פ הטענה הקודמת נקבל ש $A \in \Sigma_{i-1}$.

נתאר $\hat{M}^A$ עבור שפה L:

$\hat{M}^A$ על קלט w:

מסמלצת את M^B על הקלט w. כאשר M^B שואלת את האורקל B שאלה z, אז במקביל נריץ את $M_1^{A_1}$ על z ואת $M_2^{A_2}$ על z (משתמשים באורקל A בשביל לסמלץ את האורקלים A_1, A_2)

ואם $M_1^{A_1}$ מקבלת את z אז בוודאות $z \in B$.

אם $M_2^{A_2}$ מקבלת את z אז בוודאות $z \notin B$.

אם שתיהן דוחות אז דוחים.

אם הסימולציה של M^B מסתיימת אז עונים כמוה.

<u>יעילות:</u> $\hat{M}^A$ היא פולינומית: היא מסמלצת את M^B שהיא פולינומית.

על כל אחת מהשאלות z (יש לכל היותר מספר פולינומי של שאלות וכל אחת - אורכה פולינומי).

מפעילים 2 מכונות פולינומיות. לכן סה"כ - הכל פולינומי.

<u>נכונות:</u>

אם $w \in L$ אז ל M^B יש מסלול מקבל על w. יתר על כן, לכל z מתקיים ש $z \in B$ או ש $z \in \overline{B}$ ולכן יש למכונה המתאימה ($M_1^{A_1}$ או $M_2^{A_2}$) מסלול שמקבל את z, ולכן יש מסלול בו $\hat{M}^A$ מקבלת.

אם $w \notin L$ אז או שדוחים כבר באחת הסימולציות של שאלה כלשהי z, או שלכל z יש לנו את התשובה הנכונה - במקרה כזה, כיוון ש M^B דוחה בכל מסלול אז גם $\hat{M}^A$ תדחה.
מש"ל.

ניתן להגדיר מחלקת איחוד: $PH \triangleq \bigcup_{i\geq 0}\Sigma_i = \bigcup_{i\geq 0}\Pi_i = \bigcup_{i\geq 0}\Delta_i$
אזי מתקיים: $P \subseteq NP \subseteq PH \subseteq PSPACE$.
את ההכלה $PH \subseteq PSPACE$ אפשר להראות באמצעות כמתים, או באינדוקציה על i:
$\Sigma_0 = P \subseteq PSAPCE$ ו $\Sigma_i = NP^{\Sigma_{i-1}}$ ולכן ע"פ הנחת האינדוקציה $\Sigma_{i-1} \subseteq PSPACE$, ולכן אפשר לבצע סימולציה.

טענה: אם $PH = PSPACE$ אז ההיררכיה קורסת.
הוכחה: אילו $PH = PSPACE$ אז קיים k כך ש $TQBF \in \Sigma_k$.
לכן באמצעות שימוש ברדוקציות ובשלמות נקבל $PH \subseteq \Sigma_k$.

מימד VC - (VC Dimension)

נתונה מטריצה בינארית M בגודל $n \times n$ (או לחילופין אוסף מחרוזות S שמתארות את השורות).
אומרים ש M (או S) **מנתצת** (Shatters) קבוצת עמודות $B \subseteq [n]$ אם ההטלה של M על העמודות ב B מכילה את כל $2^{|B|}$ הצירופים האפשריים (כ"א לפחות פעם אחת).

מעוניינים למצוא מהו ה B הגדול ביותר שאפשר לנתץ במטריצה M.
כלומר, מימד VC של מטריצה M (אוסף וקטורים S) הוא גודל ה B הגדול ביותר שהיא מנתצת.

דוגמה: מטריצה M_1 שב- $\log n$ העמודות הראשונות כל צירוף מופיע פעם אחת בדיוק, ובשאר העמודות יש אפסים. נקבל ש $VC(M_1) = \log n$.
אבחנה: אם M היא מגודל $n \times n$ אז $VCD(M) \leq \log n$.

דוגמה: מטריצת היחידה M_2. $VCD(M_2) = 1$.

מה הסיבוכיות של חישוב VCD?
תשובה 1: M נתונה במפורש.
$L_1 = \{M, k \mid VCD(M) \geq k\}$

$L_1 \in NP$ - הסבר: אם $k \geq \log m$ - דחה.
אחרת, נחש קבוצה B בגודל k וודא שכל 2^k צירופים קיימים. אם כן, קבל, אחרת דחה.

האם L_1 היא NP-שלמה? לא. גודל העד הוא $\log^2 m$ - מנחשים קבוצה B בת לכל היותר $\log m$ איברים, והייצוג של כל איבר הוא $\log m$. השפה הזאת שלמה במחלקה שנקראת $LOGNP$.

$L_1 \in DTIME\left(POLY \cdot 2^{|\text{עד}|}\right) = DTIME\left(n^{\log n}\right)$.

תשובה 2: M מיוצגת ע"י מעגל $C(M)$, שבהינתן i, j מחשב בזמן פולינומי את $M(i, j)$.

$$L_2 = \{C(M), k \mid VCD(M) \geq k\}$$

$L_2 \in \Sigma_3$ (L_2 היא Σ_3-שלמה).

הסבר: $C, k \in L_2 \Leftrightarrow \exists B(|B| = k) \forall \alpha (\alpha \in 0,1)^{|B|} \exists i (1 \leq i \leq m) C(i, B) = \alpha$

כלומר: קיים ווקטור B בגודל k כך שלכל וקטור בינארי α באורך $|B|$, קיימת שורה i ב M כך ש $C(i, B) = \alpha$.

חישוב הסתברותי

נתעניין בחישוב הסתברותי **בזמן פולינומי**, או חישוב הסתברותי **בזיכרון לוגריתמי**.

על מה ההסתברות? ההסתברות היא על מסלולי החישוב של מכונת טיורינג או על מסלולי החישוב של האלגוריתם.
הגדרה לא פורמאלית: רוצים שלכל קלט, האלגוריתם (המכונה) יצליח בהסתברות גבוהה על פני מסלולי החישוב שלו, להגיע לתשובה הנכונה.

(לכן, לדוגמה, אלגוריתם לזיהוי מספרים ראשוניים, אשר על קלט n תמיד עונה שהוא לא ראשוני, הוא לא אלגוריתם הסתברותי מוצלח. אומנם ההסתברות שהוא יצליח גבוהה מאוד, מכיוון שרוב המספרים אינם ראשוניים, אבל על כל קלט ראשוני האלגוריתם **תמיד** טועה).

דוגמה:
נתון פולינום $Q(x_1,...x_n)$ ב n משתנים מדרגה d מעל שדה $GF(p)$ - המספרים מ 0 עד $p-1$ עם פעולות מודולו p.

מה זה דרגה?
פולינום בצורה קנונית: $Q(x_1,x_2,x_3)=x_1^2\cdot x_3+7x_1\cdot x_2^5...$, כלומר מוצג כסכום מכפלות.
לכל מכפלה מוגדרת הדרגה שלה, שהיא סכום החזקות.
בדוגמה שלנו: $d(x_1^2\cdot x_3)=3$, $d(7x_1\cdot x_2^5)=6$
דרגת פולינום Q היא סכום הדרגות המקסימאלי במכפלה כלשהי בייצוג קאנונים.
בדוגמה שלנו: $d(Q)=\max\{3,6\}=6$.

השאלה: האם $Q\equiv 0$ - כלומר האם זהו פולינום השווה זהותית לאפס.

מוטיבציה:
1. זהות בין פולינומים. נתונים פולינומים Q_1,Q_2 ורוצים לדעת האם $Q_1\equiv Q_2$.
אם היינו יודעים לפתור את הבעיה הקודמת (האם $Q\equiv 0$) אז פשוט היינו בודקים האם $Q_1-Q_2\equiv 0$.
2. אלגוריתם MVV למציאת שידוך בגרף. האלגוריתם מסתכל על הגרף בצורה של מטריצה ובכל תא שם משתנה התלוי בשאלה האם יש קשת או אין קשת.
השאלה היא קיים שידוך בגרף שקולה לשאלה האם הדטרמיננטה של המטריצה המתאימה לגרף שונה מאפס. מכיוון שדטרמיננטה היא פולינום מדרגה n, הרי שהבעיה שקולה לבעיה הקודמת, $Q\equiv 0$.
(שידוך - אוסף של קשתות בגרף הזרות בצמתים)

נחזור לשאלה $Q\equiv 0$.
איך נתון הקלט? אם הוא היה נתון בצורה הקנונית, הבעיה הייתה טריוויאלית.
לכן הקלט לא בהכרח נתון בצורה קנונית.
נקבל את הקלט כ"קופסה שחורה": כל ייצוג שבו ניתן לחשב את $Q(x)$ ביעילות, לכל x.
דוגמאות:
- $\det(A_G)$ - נתונה מטריצה, והפולינום הוא הדטרמיננטה שלה.
- פולינום מפורש, אבל לא בצורה קנונית. לדוגמה: $(x_1+1)(x_2+1)\cdot...\cdot(x_n+1)$.
אין בעיה לחשב בשיטה הנ"ל את ערך הפולינום בכל נקודה ביעילות, אבל העברתו לצורה הקנונית תיקח זמן אקספוננציאלי.

בעצם, זה אומר שיש לנו אוב ל Q.
לכן האלגוריתם שלנו יהיה תלוי ב n (מספר המשתנים), ב d (דרגת הפולינום) וב p (גודל השדה).

<u>הערות:</u>
- קל בצורה אי דטרמיניסטית לבדוק ש $Q \neq 0$ - פשוט מנחשים השמה למשתנים ובודקים האם התוצאה היא אפס. לכן הבעיה שייכת ל $CO-NP$.
- כקופסה שחורה, "קשה" לבדוק באופן דטרמיניסטי האם $Q \neq 0$. זאת מכיוון שקיימים פולינומים שערכם בכל נקודה הוא אפס, פרט לנקודה אחת. למשל $Q = x_1 \cdot x_2 \cdot ... \cdot x_n$ בשדה מודולו 2. בכל נקודה פרט ל $(1,1,...,1)$ ערכו הוא אפס.

<u>למה (הלמה של שוורץ):</u>
יהי Q פולינום כנ"ל (ממעלה d, n משתנים, מעל $GF(p)$).
אם $Q \neq 0$ אז מספר ההשמות $\vec{x}$ עבורן $Q(\vec{x}) = 0$ קטן או שווה ל $d \cdot p^{n-1}$.
(נשים לב שאם מדובר בפולינום עם משתנה יחיד, $n=1$, אז נקבל את המשפט היסודי של האלגברה, שאומר שלכל פולינום מדרגה d יש לכל היותר d שורשים).

<u>אלגוריתם:</u>
"נתון" Q כנ"ל כקופסה שחורה.
האלגוריתם:
בחר בהתפלגות אחידה השמה $\vec{x} \in (GF(p))^n$.
חשב את $Q(\vec{x})$.
אם התקבל $Q(\vec{x}) \equiv 0$ אז ענה ש $Q \equiv 0$ ואחרת ענה $Q \neq 0$.

<u>סיבוכיות זמן:</u>
- בחירת $\vec{x}$: מייצרים $n \cdot \lceil \log p \rceil$ ביטים אקראיים - לוקח זמן פולינומי.
- חישוב $Q(\vec{x})$ ע"י קופסה שחורה ולכן יעיל.

לכן הכל פולינומי ב n וב $\log p$.

<u>"נכונות":</u>
אם $Q \equiv 0$ אז האלגוריתם תמיד עונה תשובה נכונה.
אם $Q \neq 0$: האלגוריתם יטעה אם יבחר השמה $\vec{x}$ שבה $Q(\vec{x}) = 0$. ההסתברות שזה יקרה שווה למספר ההשמות שבהן הפולינום מתאפס חלקי מספר ההשמות הכללי האפשרי. זאת מכיוון שבוחרים השמה באופן אקראי בהתפלגות אחידה.
ע"פ הלמה של שוורץ, נקבל ש $P(mistake) = \frac{d \cdot p^{n-1}}{p^n} = \frac{d}{p}$.
לכן אם $p > 2d$ אז ההסתברות לשגיאה קטנה מחצי.

<u>הוכחת הלמה של שוורץ:</u>
באינדוקציה על מספר המשתנים, n.
<u>בסיס:</u> $n=1$. ע"פ המשפט היסודי של אלגברה יש לפולינום מסדר 1, לכל היותר d שורשים כאשר d הוא דרגת הפולינום.
<u>צעד:</u> $n>1$. נחשוב על הפולינום Q בייצוג הקנוני (סכום של מכפלות).

מקרה א': x_n לא משתתף ב Q. במקרה זה, Q הוא פולינום בעל $n-1$ משתנים. לכן ע"פ הנחת האינדוקציה יש לו לכל היותר $d \cdot p^{n-2}$ שורשים. כל p האפשרויות לתת ערך ל x_n עדין מאפסות את Q ולכן בסה"כ יש לפולינום לכל היותר $d \cdot p^{n-2} \cdot p = d \cdot p^{n-1}$ שורשים.

מקרה ב': x_n כן מופיע ב Q.
נסמן ב k את החזקה הגבוהה ביותר של x_n.
נוכל לרשום את Q באופן הבא: $Q(x_1,...,x_n) = \sum_{i=1}^{k} x_n^i g_i(x_1,...,x_{n-1})$ כאשר g_i הוא פולינום בעל $n-1$ משתנים מדרגה לכל היותר $d-i$. (אם הייתה לו דרגה יותר גבוהה מ $d-i$ אז לאחר שהיינו מכפילים בו את x_n^i היינו מקבלים פולינום מדרגה גבוהה מ d בסתירה לכך שהדרגה הכללית של הפולינום היא d).
נחסום מלמעלה את מספר ההשמות ל $x_1,...,x_n$ המאפסות את Q. נחלק אותן ל 2 סוגים:
א. השמות כנ"ל שעבורן g_k (הפולינום האחרון) מתאפס ע"י $x_1,...,x_{n-1}$.
ב. השמות כנ"ל שעבורן g_k לא מתאפס.

א. מהגדרת k, מתקיים ש g_k הוא לא זהותית אפס. g_k הוא פולינום ב $n-1$ משתנים מדרגה $d-k$ ו $g_k \neq 0$ ולכן ע"פ הנחת האינדוקציה, יש לו לכל היותר $(d-k) \cdot p^{n-2}$ השמות המאפסות אותו.
לכן בוודאי שיש לכל היותר $p \cdot (d-k) \cdot p^{n-2}$ השמות כנ"ל המאפסות את Q וגם את g_k.
סה"כ: $(d-k) \cdot p^{n-1}$.

ב. נתבונן ב (לכל היותר) p^{n-1} ההשמות ל $n-1$ המשתנים הראשונים, $x_1,...,x_{n-1}$ אשר אינן מאפסות את g_k. כל השמה כזאת משרה פולינום במשתנה יחיד $Q'(x_n)$, שאיננו זהותית אפס, (מכיוון ש g_k לא מתאפס עבורן).
הפולינום הנ"ל הוא מדרגה k ובעל משתנה יחיד, כלומר $n=1$.
לכן ע"פ הנחת האינדוקציה יש לו כל היותר k השמות שמאפסות אותו.
לכן, סה"כ מספר ההשמות מסוג ב' (מאפסות את Q ולא את g_k) קטן או שווה ל $k \cdot p^{n-1}$.

סה"כ א' + ב', לכל היותר: $(d-k) \cdot p^{n-1} + k \cdot p^{n-1} = d \cdot p^{n-1}$.

<u>הגדרות:</u>
מ"ט הסתברותית מוגדרת בדומה למ"ט אי-דטרמיניסטית, כאשר בכל צעד המכונה (אם היא במצב לא סופי) יכולה לבחור באחת משתי אפשרויות (לאו דווקא שונות, כמו בחירות לנציגי סמסטר), וכל אחת מהן נבחרת בהסתברות בדיוק חצי באופן בלתי תלוי.

<u>הערות:</u> הגדרות אלטרנטיביות רבות (שקולות):
1. נתון סרט אקראיות נוסף למכונה המכיל ביטים אקראיים לקריאה בלבד.
2. למכונה יש בכל צעד 3 אפשרויות לבחור מביניהן, כאשר לכל אחת מהן יש הסתברות שליש להיבחר.

<u>סימון:</u> $P_M(x)$ - ההסתברות שהמכונה M מקבלת את x.
ההסתברות של מסלול בעל t צעדים להיבחר, היא 2^{-t}.

<u>מחלקות סיבוכיות:</u>

PP - אוסף השפות הניתנות לזיהוי ע"י מ"ט הסתברותית פולינומית המקיימת:

לכל $x \notin L$ מתקיים $P_M(x) \le \frac{1}{2}$ ולכל $x \in L$, $P_M(x) > \frac{1}{2}$.

BPP - אוסף השפות הניתנות לזיהוי ע"י מ"ט הסתברותית פולינומית המקיימת:

לכל $x \notin L$ מתקיים $P_M(x) \le \frac{1}{3}$ ולכל $x \in L$, $P_M(x) \ge \frac{2}{3}$.

RP - אוסף השפות הניתנות לזיהוי ע"י מ"ט הסתברותית פולינומית המקיימת:

לכל $x \notin L$ מתקיים $P_M(x) = 0$ ולכל $x \in L$, $P_M(x) \ge \frac{2}{3}$.

<u>הגברה: (Amplification)</u>

בהינתן מכונת RP, M, ופרמטר k נבנה מ"ט M_k שעובדת באופן הבא:

מריצה את M, k פעמים באופן בלתי תלוי, ומקבלת אם ורק אם M קיבלה לפחות פעם אחת.

אם $x \notin L$ אז $P_{M_k}(x) = 0$.

אם $x \in L$ אז $P_{M_k}(x) \ge 1 - \left(\frac{1}{3}\right)^k$.

<u>הערה:</u> אם M הייתה כזאת:

$$P_M(x) = 0 \Leftarrow x \notin L$$

$$P_M(x) \ge \frac{1}{n^{100}} \Leftarrow x \in L$$

זה עדין היה שקול ל RP: פשוט צריך להריץ מספיק פעמים (למשל n^{101}) את M.

בהינתן מכונת BPP, M, ופרמטר k נבנה מ"ט M_k שעובדת באופן הבא:

מריצה את M, k פעמים באופן בלתי תלוי, ומחליטה ע"פ הרוב.

<u>אי שוויון צ'רנוף</u>: נניח $x_1,...,x_k$ משתנים מקריים $0-1$, ולכל i: $\Pr(x_i = 1) = p \le \frac{1}{2}$.

(במקרה זה $E(x_i) = p$, $E\left(\sum_{i=1}^{k} x_i\right) = p \cdot k$).

ההסתברות שהמרחק בין הממוצע לבין התוחלת יהיה גדול מ δ - $\Pr\left(\left|\frac{\sum_{i=1}^{k} x_i}{k} - p\right| \ge \delta\right) \le 2 \cdot e^{\frac{-\delta^2 k}{2p(1-p)}}$.

ניתוח של M_k:

נגדיר משתנה מקרי: x_i עבור $1 \le i \le k$.

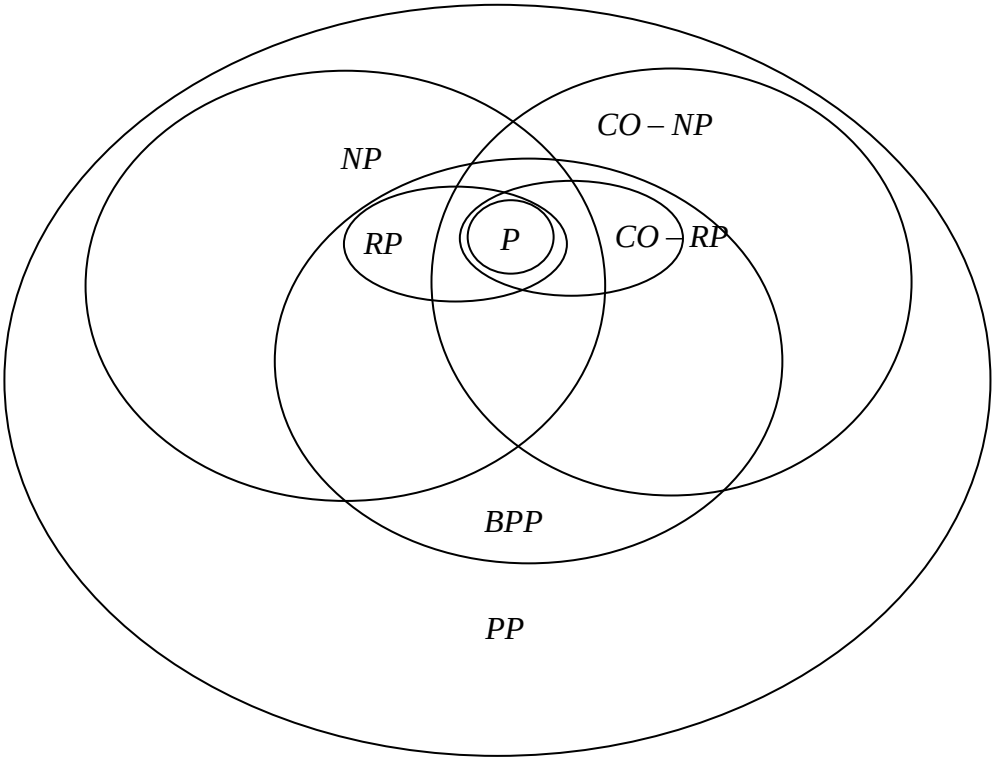
$x_i = 1$ אם בהרצה ה i של M, המכונה טועה ו $x_i = 0$ אם היא עונה נכון.

הסיכוי של המכונה לטעות בכל הרצה הוא לכל היותר שליש. כלומר: $\Pr(x_i = 1) \le \frac{1}{3} = p \le \frac{1}{2}$.

M_k טועה אם M טועה ברוב המקרים, כלומר, אם: $\sum_{i=1}^{k} x_i \geq \frac{k}{2}$, כלומר: $\left|\frac{\sum x_i}{k} - \frac{1}{3}\right| \geq \frac{1}{6}$.

על פי אי שוויון צ'רנוף: $\Pr(mistake) \leq \Pr\left(\left|\frac{\sum x_i}{k} - \frac{1}{3}\right| \geq \frac{1}{6}\right) = 2^{-\Theta(k)}$

<u>תמונת העולם:</u>



הסבר למה PP מכילה את NP:
בהינתן שפה L ב NP עם מ"ט M א"ד פולינומית מתאימה, מכונה M' תגריל מספר 0 או 1. אם יצא 1 אז היא תקבל, אחרת היא תריץ את המכונה M ותענה כמוה.
אם $x \notin L$ אז בהסתברות חצי, המכונה שלנו תחליט להריץ את M ולענות כמוה (כלומר לדחות) ולכן במקרה כזה $P_{M'}(x) \leq \frac{1}{2}$.
אם $x \in L$ אז בהסתברות חצי, המכונה שלנו תחליט לקבל מיד, ובעוד הסתברות קטנה (אבל לא אפס), היא תחליט כן להריץ את M וגם תצליח להגריל את המסלול המקבל של M. לכן $P_M(x) > \frac{1}{2}$.
לכן $L \in PP$ ולכן $NP \subseteq PP$.

בהמשך נראה ש $BPP \subseteq \Sigma_2 \cap \Pi_2$.

<u>תזכורת:</u>
דיברנו על חישוב הסתברותי:
מכונת טיורינג שבכל צעד וצעד יכולה להטיל מטבע ולבחור לאיזה מסלול לפנות על פי תוצאת המטבע.
לחלופין, למכונה יש סרט אקראיות לקריאה בלבד, שלפני ריצתה מאותחל באופן אקראי "משמיים".

המכונה מקבלת את x אם בהסתברות "גבוה" היא עוצרת עליו במצב מקבל.

הגדרנו את המחלקות הרלוונטיות:

RP - מחלקת הדברים שאפשר לעשות בזמן פולינומי באופן שכל קלט שבשפה מתקבל בהסתברות גבוהה, וכל קלט שלא בשפה לא מתקבל אף פעם.
מתקיים: $P \subseteq RP \subseteq NP$.

BPP - ההגדרה הסימטרית - המכונה צודקת על כל x בהסתברות גבוה. קלט שבשפה צריך בדרך כלל להתקבל וקלט שאינו בשפה צריך בדרך כלל להידחות.
מתקיים: $P \subseteq RP \subseteq BPP$.

האם $NP \subseteq BPP$?
האם $BPP \subseteq NP$?
התשובה לא ידועה.

ההכלה $NP \subseteq BPP$ כנראה לא נכונה. אם היא הייתה נכונה, זה היה אומר שקיים אלגוריתם הסתברותי דטרמיניסטי יעיל ל SAT.

הרבה אנשים מאמינים ש $BPP = P$ ואם זה נכון אז כמובן ש $BPP \subseteq NP$.
הבעיה היא שמכונות BPP רשאיות שיהיה להן מסלול מקבל גם עבור קלטים שהן דוחות בעוד שבמכונות NP מצב כזה אסור.

<u>משפט:</u> $BPP \subseteq \Sigma_2^P \cap \Pi_2^P$.
<u>הוכחה:</u> מספיק להוכיח ש $BPP \subseteq \Sigma_2^P$ (אם $L \in BPP$ אז $\bar{L} \in BPP$. אם $\bar{L} \in \Sigma_2^P$ אז $L \in \Pi_2^P$).
$L \in BPP \Leftarrow$ קיימת מ"ט הסתברותית M שרצה בזמן n^c ולכל קלט טועה בהסתברות $\frac{1}{2^n}$.
(תזכורת: ראינו שאפשר להוריד את השגיאה מ $\frac{1}{3}$ ל $\frac{1}{2^k}$ ע"י $O(k)$ הרצות והליכה לפי הרוב. לכן מספיק להריץ את המכונה המקורית $O(n)$ פעמים בשביל להשיג הסתברות כזאת לטעות).

<u>סימון:</u> $M(x,s)$:
x הוא הקלט ו s הוא סרט האקראיות באורך $l \triangleq n^c$.
אזי $M(x,s)$ הוא פרדיקט שמחזיר $Accept / Reject$ לפי תשובת המכונה M על קלט x וסרט אקראיות s.
אבחנה: $M(x,s)$ ניתן לחישוב בזמן פולינומי ע"י סימולציה.

<u>אינטואיציה:</u>

נניח שננסה להוכיח: $x \in L \Leftrightarrow \exists s M(x,s) = Accept$.

אם זה היה נכון, אז היינו מקבלים ש $L \in NP$, ע"פ ההגדרה $NP = \exists P$.

נניח שהיינו מרחיבים את זה להוכחה כזאת: $x \in L \Leftrightarrow \exists s_1 s_2 ... s_l \forall_{1 \le i \le l} < (x, s_i) = Accept$.

גם אם זה היה נכון, אז היינו מקבלים ש $L \in NP$.

<u>סימונים:</u>

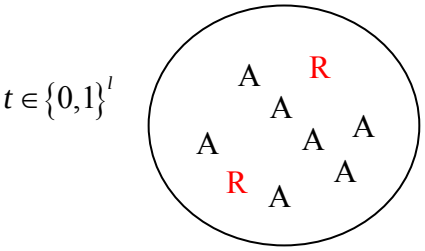
$t \in \{0,1\}^l$ - מחרוזת בינארית באורך l.

$s \oplus t$ - עוברים ביט ביט, ומחשבים xor.

$s \to s \oplus t$ - העתקה חד חד ערכית ועל - פרמוטציה של $\{0,1\}^l$.

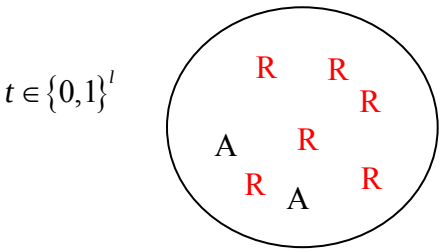
<u>למה:</u> $L = \left\{ x \mid \exists s_1 s_2 ... s_l \in \{0,1\}^l \; \forall t \in \{0,1\}^l \; \exists_{1 \le i \le l} M(x, s_i \oplus t) = Accept \right\}$

אם $x \in L$ אז עבור רוב ה $t \in \{0,1\}^l$, מתקיים $M(x,t) = A$.



כלומר, קיימת קבוצת מחרוזות, שבכל פעם שהיא תוזז, עדין מישהי מהן תיפול על $Accept$.

אם $x \notin L$ אז עבור רוב ה $t \in \{0,1\}^l$ מתקיים $M(x,t) = R$.



כלומר, לכל קבוצת מחרוזות, קיים צורה להזזה שלה, כך שכולן יפלו על $Reject$.

<u>מדוע הלמה גוררת את המשפט?</u>

א. לפי ההגדרה האלטרנטיבית של ההיררכיה הפולינומית: (פרדיקט פולינומי) $\Sigma_2^P = \exists\forall$.
מהלמה נובע שהשפה L היא מהצורה (פרדיקט פולינומי) $\forall\exists$. (ה $\exists$ הפנימי רץ על מספר מ 1 עד l ולכן אפשר לחשב אותו בזמן פולינומי).

ב. $A = \left\{ x, s_1, ..., s_l \mid \exists t \forall_{1 \le i \le l} : M\left(x, s_i \oplus t\right) = R \right\}$. נקבל ש $A \in NP$ - פשוט מנחשים את t ובודקים (רצים על כל ה i-ים האפשריים בזמן פולינומי).

נראה ש $L \in NP^A$ ונקבל ש $L \in \Sigma_2^P$.

$L \in NP^A$: על קלט x:

- נחש $s_1, ..., s_l$.
- וודא (ע"י שאילתא לאורקל A) ש $\left(x, s_1, ..., s_l\right) \notin A$.

<u>השיטה ההסתברותית:</u> מחפשים $y \in Y$ עם תכונה P עבור Y סופי.
אם נראה שבבחירה אקראית (בפילוג אחיד) של $y \in Y$, $\Pr(y \text{ מקיים את } P) > 0$
אזי קיים y המקיים את P.
זאת מכיוון ש:
[מספר ה y-ים שמקיימים את P חלקי מספרים ה y ים ב Y] $= \Pr(y \text{ מקיים את } P)$

<u>הוכחת הלמה:</u>

$x \notin L$: צריך להוכיח שלכל בחירה של $s_1, ..., s_l$ קיים t כך ש $\forall_i M\left(x, s_i \oplus t\right) = R$.
נקבע את $s_1, ..., s_l$ ונוכיח שקיים t כזה.
נוכיח זאת ע"י השיטה ההסתברותית. נוכיח שבהסתברות גדולה מ 0, t שיבחר יקים זאת.
לחילופין, ההסתברות ש t רע קטנה ממש מ 1.

מהי ההסתברות ש t רע?

$$\Pr_t(t \text{ רע}) = \Pr_t\left(\exists_i M\left(x, s_i \oplus t\right) = A\right) \underset{\substack{\text{Union} \\ \text{bound}}}{\le} \sum_{i=1}^{l} \underbrace{\Pr_t\left(M\left(x, s_i \oplus t\right) = A\right)}_{\le \frac{1}{2^n}} \le \frac{n^c}{2^n}$$

עבור n מספיק גדול מתקיים $\frac{n^c}{2^n} < 1$.
כלומר $\Pr_t(t \text{ רע}) < 1$.

$x \in L$: צריך להראות שקיימת סדרה $\vec{s} = s_1, ..., s_l$ כך שלכל t מתקיים $\exists_{1 \le i \le l} M\left(x, s_i \oplus t\right) = A$.
(בשיטה ההסתברותית): נבחר סדרה $\vec{s}$ באקראי ונוכיח שההסתברות ש $\vec{s}$ טובה גדולה מ 0, או לחילופין שההסתברות ש $\vec{s}$ רעה קטנה מ 1.

$$\Pr_{\vec{s}}(\vec{s} \text{ רעה}) = \Pr_{\vec{s}}\left(\exists t \forall_{1 \le i \le l} M\left(x, s_i \oplus t\right) = R\right) \underset{\substack{\text{Union} \\ \text{Bound}}}{\le} \sum_{t \in \{0,1\}^l} \Pr_{\vec{s}}\left(\forall_{1 \le i \le l} M\left(x, s_i \oplus t\right) = R\right)$$

בגלל אי התלות בין הבחירות של s_1 עד s_l נוכל להחליף את ה"לכל" במכפלת ההסתברויות:

$$\sum_{t \in \{0,1\}^l} \Pr_{\vec{s}}\left(\forall_{1 \le i \le l} M\left(x, s_i \oplus t\right) = R\right) = \sum_{t \in \{0,1\}^l} \prod_{i=1}^{l} \underbrace{\Pr_{\vec{s}}\left(M\left(x, s_i \oplus t\right) = R\right)}_{\le 2^{-n}} = \cdot 2^{\left(n^c\right)} \underbrace{\left(\left(2^{-n}\right)^{n^c}\right)}_{2^{\left(-n^{c+1}\right)}} < 1$$

כלומר $\Pr_{\vec{s}}(\vec{s} \text{ רעה}) < 1$.

הוכחות אינטראקטיביות

"הוכחות קלאסיות" - תהליך שבו אנחנו משתכנעים בנכונותן של טענות.
- נתונה טענה שאותה רוצים להוכיח.
- החלק הקשה: מציאת ההוכחה ע"י המוכיח (יסומן ב P עבור Prover).
- ווידא ההוכחה ע"י המוודא (יסומן ב V עבור Verifier).
- רוצים: שלמות (אם משהו נכון, יש דרך להוכיח אותו) ונאותות (לא ניתן להוכיח טענות שקריות).

נרצה בנוסף שהמוודא יהיה יעיל. כלומר יהיה קל לבדוק את ההוכחה.

הוכחות NP: לדוגמה, להוכיח ש $\varphi \in SAT$.
המוכיח ימצא (אפילו בזמן לא יעיל) השמה מספקת ל φ וישלח אותה למוודא.
המוודא יבדוק זאת בזמן פולינומי וישתכנע בנכונות, כלומר יקבל.

אם $L \in NP$ אז קיים יחס R_L חסום פולינומית וניתן לזיהוי בזמן פולינומי כך ש:
$x \in L \Leftrightarrow \exists y \left(x, y\right) \in R_L$.

לכן אם P רוצה להוכיח ש $x \in L$ הוא ימצא y כך ש $\left(x, y\right) \in R_L$ וישלח את הזוג $\left(x, y\right)$ ל V, ו V יבדוק האם $\left(x, y\right) \in R_L$ בזמן פולינומי, ואם כן, אז הוא יקבל את x.

אם ל L מערכת הוכחה כזאת.
$L \in NP$ ע"י הגדרת יחס R_L המתאים.
$\left(x, y\right) \in R_L \Leftrightarrow$ (במערכת ההוכחה על הטענה $x \in L$ כאשר P שולח את y, V מקבל)

נרצה להוסיף אינטראקטיביות, כלומר שההוכחה לא תתבצע רק ע"י שני צעדים. בנוסף, נוסיף אפשרות לטעות בהסתברות נמוכה.

דוגמה:
נתונים שני גרפים $G_1 = \left(V_1, E_1\right)$, $G_2 = \left(V_2, E_2\right)$ נקראים איזומורפיים אם קיימת העתקה π חח"ע ועל, $\pi : V_1 \to V_2$ המקיימת: $\left(a, b\right) \in E_1 \Leftrightarrow \left(\pi\left(a\right), \pi\left(b\right)\right) \in E_2$.
כלומר ההבדל בין הגרפים הוא רק בשמות הצמתים.

לדוגמה, שני הגרפים הבאים איזומורפיים:

זהו יחס שקילות, ומסמנים $G_1 \equiv G_2$.
נגדיר שפות:
$GI \triangleq \left\{G_1, G_2 \mid G_1 \equiv G_2\right\}$ - השפה שייכת ל NP
(פשוט מנחשים העתקה π ומוודאים שהתנאי מתקיים).

$GNI \triangleq \left\{G_1, G_2 \mid G_1 \neq G_2\right\}$ - השפה שייכת ל $CoNP$.

איך ניתן להוכיח ששני גרפים אינם איזומורפיים?
נניח ש P טוען: $(G_1,G_2)\in GNI$.
הפרוטוקול:
1. V בוחר באופן אקראי בהתפלגות אחידה $b\in_R\{1,2\}$.
2. V בוחר פרמוטציה אקראית על הצמתים $\pi\in_R S_n$.
3. V מחשב גרף $H=\pi(G_b)$ ושולח אותו ל P.
4. P מוצא β כך ש $H\equiv G_\beta$ ושולח את β ל V.
5. V מקבל אם ורק אם $\beta=b$.

<u>מתקיים:</u>
- המוודא עובד בזמן פולינומי הסתברותי.
- <u>שלמות:</u> אם $(G_1,G_2)\in GNI$ אז $H\equiv G_b$ ו $H\neq G_{\bar{b}}$ ולכן $\beta=b$ ועל כן תמיד V יקבל.
- <u>נאותות:</u> אם $G_1\equiv G_2$ אז $H\equiv G_b$ וגם $H\equiv G_{\bar{b}}$. בדיקה מראה ש

$\Pr(G_b\mid H)=PR(G_{\bar{b}}\mid H)$ ולכן לכל H: $\Pr(\text{V משכנע את P}\mid H)\le\frac{1}{2}$ ולכן

$\Pr(\text{P משכנע את V})\le\frac{1}{2}$.

<u>הגדרה:</u> פרוטוקול (P,V) ל 2 משתתפים P (מוכיח) ו V (מוודא) הוא מערכת הוכחה עבור שפה L אם מתקיים:
<u>שלמות:</u> לכל $x\in L$, $\Pr(\text{V מקבל את } x \text{ ב } (P,V))=1$.

<u>נאותות:</u> לכל $x\notin L$ ולכל P', $\Pr(\text{V מקבלת את } x \text{ ב } (P',V))\le\frac{1}{3}$.

V הוא הסתברותי פולינומי.
P,P' הם דטרמיניסטיים.

<u>הערות:</u>
משתתף ≡ מכונת טיורינג אינטרקרטיבית.
אם V דטרמיניסטי, אין צורך באינטראקטיביות. (P יכול לסמלץ את V) ומקבלים את NP.

אפשר היה לקלקל קצת את השלמות, כלומר $\Pr(\text{V מקבל את } x \text{ ב } (P,V))<1$ אבל זה לא היה מוסיף כוח.
אפשר היה להוסיף הסתברות ל P אולם גם זה לא היה מוסיף כוח.

הוכחות אינטראקטיביות - המשך

תזכורת: הוכחות אינטראקטיביות הן הוכחות כמו הוכחות NP עם תוספת של **אינטראקציה** ותוספת של **הסתברות** בפעולת המוודא.

מחלקת השפות שיש להן הוכחות אינטראקטיביות נקראת IP:
כלומר: IP = {L | L שפה שיש לה מערכת הוכחה אינטראקטיבית}.

האם $\overline{SAT} \in IP$ - האם יש דרך באמצעות הוכחה אינטראקטיבית לשכנע מוודא שפסוק איננו ספיק?

<u>משפט</u> (הוכחה המקורית של עדי שמיר ממכון ויצמן):
$IP = PSPACE$ (ומכאן אכן מתקיים $\overline{SAT} \in IP$ שכן $\overline{SAT} \in PSPACE$).

בהמשך הקורס נראה שפה $PERM \in IP$ וממנה מגיעים למסקנה שההיררכיה הפולינומית מוכלת ב IP: $PH \subseteq IP$.

<u>הוכחת המשפט:</u>
הכיוון $IP \subseteq PSPACE$ (פירוט נוסף בתרגול): נחשוב על עץ שמתאר את כל המסלולים בחישוב. בכל רגע נתון, צריך לזכור מסלול אחד, שאורכו פולינומי.
כמובן שאורך ההודעות של המוודא הן פולינומיות, כי הוא עובד בזמן יעיל.
אנחנו לא יודעים מה המוכיח עושה, אבל בוודאות ההודעות שלו הן באורך פולינומי (אחרת המוודא לא יוכל לקרוא אותן).

הכיוון $IP \supseteq PSPACE$:
מספיק שנראה שלשפה $TQBF$ יש הוכחה אינטראקטיבית, מכיוון ש $TQBF$ היא שפה $PSPACE$ שלמה.

<u>אריתמטיזציה:</u> נסמן $T = 1$, $F = 0$.
אם φ נוסחה בעלת משתנים חופשיים $x_1,...,x_k$ אז נמיר אותה לפולינום $\hat{\varphi}$ עם משתנים $x_1,...,x_k$.
פולינום זה יהיה מעל שדה $GF(p)$ (חיבור וכפל מודולו p).
נרצה שיתקיים: לכל השמה $\vec{v} \in \{0,1\}^k$ יתקיים $\hat{\varphi}(\vec{v}) = \varphi(\vec{v})$.

<u>המשפט היסודי של האלגברה:</u>
2 פולינומים **שונים** מדרגה d במשתנה אחד, מסכימים לכל היותר ב d מקומות.

אריתמטיזציה של נוסחאות ללא כמתים:

הפולינום $\hat{\varphi}(\vec{x})$ המתאים לנוסחה $\varphi(\vec{x})$	הנוסחה $\varphi(\vec{x})$
1	T
0	F
x_i	x_i
$1-\hat{\beta}(\vec{x})$	$\neg\beta(\vec{x})$ עבור נוסחה $\beta(\vec{x})$
$\hat{\beta}\cdot\hat{\gamma}$	$\beta \wedge \gamma$
$1-(1-\hat{\beta})(1-\hat{\gamma})$	$\beta \vee \gamma$

<u>מתקיים</u> (ההוכחה באינדוקציה):

1. לכל $\vec{v}\in\{0,1\}^k$: $\hat{\alpha}(\vec{v})=\alpha(\vec{v})$.
2. דרגת הפולינומים חסומה ע"י אורך הנוסחאות שהם מייצגים: $\deg(\hat{\alpha})\le size(\alpha)$.
3. בהינתן פסוק α קל לבנות בזמן פולינומי את הפולינום $\hat{\alpha}$.
4. בהינתן $\hat{\alpha}$ והשמה $\vec{v}$ קל לחשב בזמן פולינומי את ערך הפולינום: $\hat{\alpha}(\vec{v})$.

(כלומר בניית הפולינום וחישוב ערכו בהשמה כלשהי ניתנים לביצוע ע"י המוודא).

אריתמטיזציה של נוסחאות עם כמתים ומשתנים חופשיים:

$\varphi(x_1,...,x_n)=Q_1y_1...Q_my_m\alpha(y_1,...,y_m,x_1,...,x_n)$ כאשר $Q_i\in\{\forall,\exists\}$, α פסוק CNF.

<u>סימון</u>:

אם $P(x_1,...,x_n)$ הוא פולינום, אז נסמן: $\forall x_1P(x_1,...,x_n)\triangleq P(0,x_2,...,x_n)\cdot P(1,x_2,...,x_n)$

באופן דומה: $\exists x_1P(x_1,...,x_n)\triangleq 1-(1-P(0,x_2,...,x_n))\cdot(1-P(1,x_2,...,x_n))$.

אלו הם פולינומים בעלי $n-1$ משתנים.

הנוסחה $\varphi(\vec{x})$	הפולינום $\hat{\varphi}(\vec{x})$ המתאים לנוסחה $\varphi(\vec{x})$
$\psi(x_2,...,x_n)=Qx_1\varphi(x_1,...,x_n)$ עבור $Q\in\{\forall,\exists\}$	$\hat{\psi}(x_1,...,x_n)=Qx_1\hat{\varphi}(x_1,...,x_n)$

<u>מתקיים</u>: (ההוכחה באינדוקציה)

לכל השמה $\vec{v}\in\{0,1\}^k$, ערך הפולינום שווה לערך הנוסחה הלוגית: $\hat{\varphi}(\vec{v})=\varphi(\vec{v})$.

בפרט, אם אין משתנים חופשיים $(n=0)$:

אם $\varphi=T$ אז $\hat{\varphi}=1$.

אם $\varphi=F$ אז $\hat{\varphi}=0$.

<u>בעיות</u>:

עם כל הוספת כמת, אורך הפולינום מוכפל פי 2. (המוכיח ידאג לפתור את הבעיה)

עם כל הוספת כמת, דרגת הפולינום מוכפלת פי 2.

<u>אבחנה</u>:

לכל מספר $v\in\{0,1\}$ מתקיים $v^k=v$.

לכן, נסמן: **החלף כל מופע של** x_1^k **ב** x_1 **(לינאריזציה)** $Rx_1P(x_1,...,x_n)$.

כלומר: $Rx_1P(x_1,...,x_n)=x_1\cdot P(1,x_2,...,x_n)+(1-x_1)\cdot P(0,x_2,...,x_n)$.

<u>מתקיים</u>: $Rx_1P(x_1,...,x_n)$ - דרגתו ב x_1 היא 1 וערכו בכל השמה $\vec{v}\in\{0,1\}^n$ זהה לערכו של P.

לכן נוכל להוריד את הדרגה של פולינום המקור בעל n משתנים, לדרגה n בלי לפגוע בערך ההשמות $\vec{v}\in\{0,1\}^n$.

נתונה נוסחת QBF: $\psi=Q_1x_1Q_2x_2Q_3...Q_nx_n\alpha(x_1,...,x_n)$.

נשנה את הנוסחה ל: $\psi=Q_1x_1Rx_1Q_2x_2Rx_1Rx_2Q_3...Q_nx_nRx_1Rx_2...Rx_n\alpha(x_1,...,x_n)$

כלומר נגדיר פולינום $p_0=\hat{\alpha}(x_1,...,x_n)$ וממנו באמצעות הוספת Rx_n נקבל את הפולינום p_1.

אח"כ נוסיף את Rx_{n-1} ונקבל את הפולינום p_2 . נמשיך כך עד שנגיע ל Rx_1 ונקבל ממנו את הפולינום p_n .

אח"כ נוסיף את $Q_n x_n$ ונקבל את p_{n+1} ואחריו שוב נוסיף Rx_{n-1} וכן הלאה...

סה"כ, נקבל $O\left(n^2\right)$ פולינומים.

נסמן את הפולינום האחרון ב p_l .

<u>מבנה הפרוטוקול:</u>

- בתחילת הפרוטוקול המוכיח P צריך לשכנע את המוודא V ש $p_l = 1$.

- <u>איטרציה:</u>

בתחילת האיטרציה P צריך לשכנע את V ש "$p_j\left(\vec{u}\right) = a$" $(*)$

בסוף האיטרציה P צריך לשכנע את V ש "$p_{j-1}\left(\vec{v}\right) = b$" $(**)$

באופן שמתקיים: (שלמות) אם $(*)$ נכון אז גם $(**)$ נכון

וכן יתקיים: אם $(*)$ לא נכון, אז בהסתברות $\geq 1-\varepsilon$ גם $(**)$ לא נכון או ש V דוחה במהלך האיטרציה.

- <u>בסיס</u>: צריך להוכיח ש "$p_0\left(\vec{u}\right) = a$" ואת זה V יכול לבצע בעצמו ע"י חישוב p_0 וחישוב $p_0\left(\vec{u}\right)$.

<u>מסקנה:</u> (בהנחה שניתן לבצע את האיטרציות כמובטח):

<u>שלמות:</u> אם $\psi \in TQBF$ אז $p_l = 1$ ואז כל הטענות שצריך יהיה להוכיח בדרך הן נכונות. בפרט הטענה לגבי p_0 נכונה ולכן Vישתכנע בכך ש $\psi \in TQBF$ ויקבל.

<u>נאותות:</u> אם $\psi \notin TQBF$ אז $p_l \neq 1$ נובא (לפי נאותות האיטרציה) שבהסתברות $\geq 1-l\cdot\varepsilon$ או שהטענה על p_0 נכונה או ש V דוחה קודם.

לכל בהסתברות $\geq 1-l\cdot\varepsilon$, V ידחה את הקלט ψ.

<u>איטרציה</u>: מבנה הפולינום הוא: $p_j\left(...\right) = \left\{R/\exists/\forall\right\} p_{j-1}\left(...\right)$

נחלק למקרים:

$\forall$: $p_j\left(x_2,...,x_n\right) = \forall x_1 p_{j-1}\left(x_1,x_2,...,x_n\right) = p_{j-1}\left(0,x_2,...,x_n\right)\cdot p_{j-1}\left(1,x_2,...,x_n\right)$.

בתחילת האיטרציה צריך להוכיח "$p_j\left(u_2,...,u_n\right) = a$". כלומר נתונים $u_2,...,u_n,a$.

נגדיר פולינום $g\left(x_1\right)$ באופן הבא: $g\left(x_1\right) \triangleq p_{j-1}\left(x_1,u_2,...,u_n\right)$

המוכיח P יחשב את $g\left(x_1\right)$ וישלח אותו למוודא V, כלומר ישלח את רשימת המקדמים שאורכה כדרגת x_1 ב g אשר שווה לדרגת x_1 ב p_{j-1} .

<u>בדיקת תקינות:</u>

המוודא יחשב את $g\left(0\right)\cdot g\left(1\right)$ ויוודא שהוא אכן מקבל את a . זה אמור להתקיים מכיוון ש:

$$g\left(0\right)\cdot g\left(1\right) = p_{j-1}\left(0,u_2,...,u_n\right)\cdot p_{j-1}\left(1,u_2,...,u_n\right) = \forall x_1 p_{j-1}\left(x_1,u_2,...,u_n\right) = p_j\left(u_2,...,u_n\right)$$

אם התוצאה איננה a, הוא ידחה.

אם התוצאה היא כן a אז המוודא בוחר באקראי $u_1 \in_R GF\left(p\right)$ ומבקש שיוכיחו לו ש:

$p_{j-1}\left(u_1,u_2,...,u_n\right) = g\left(u_1\right)$. נסמן $g\left(u_1\right) = b$.

<u>שלמות</u>: אם אכן $p_j\left(u_2,...,u_n\right) = a$ אז מהבניה אכן נובע ש $p_{j-1}\left(u_1,...,u_n\right) = b$.

<u>נאותות:</u> נניח ש $p_j(u_2,...,u_n) \neq a$ ונניח שבדיקת התקינות עברה בשלום.
נסמן: g - הפולינום המוגדר בפרוטוקול.
g' - הפולינום שהמוכיח הרמאי P' שלח בפועל.
בדיקת התקינות עברה, כלומר $g'(0)\cdot g'(1) = a$.
מכיוון ש $p_j(u_2,...,u_n) \neq a$: $g(0)\cdot g(1) \neq a$.
לכן $g' \neq g$.

לכן g, g' מסכימים על $\varepsilon = \frac{\deg}{p}$ מתוך איברי השדה.
אם בחרנו (בהסתברות $\leq 1-\varepsilon$) u_1 כזה ש $g(u_1) \neq g'(u_1)$ אז גם באיטרציה הבאה הטענה שיש להוכיח שגויה. $p_{j-1}(u_1,u_2,...,u_n) = g'(u_1) \neq g(u_1)$.

<u>פרמטרים אפשריים:</u>
$l = O(n^2)$ - מספר הפולינומים.
$\forall j \deg(p_j) \leq size(\alpha) \leq size(\psi) = N$ כאשר α הוא פסוק ה CNF המרכיב את ψ.
$p \in [N^{10}, N^{11}]$ מספר ראשוני - גודל השדה.
$\varepsilon \leq \frac{1}{N^9}$.

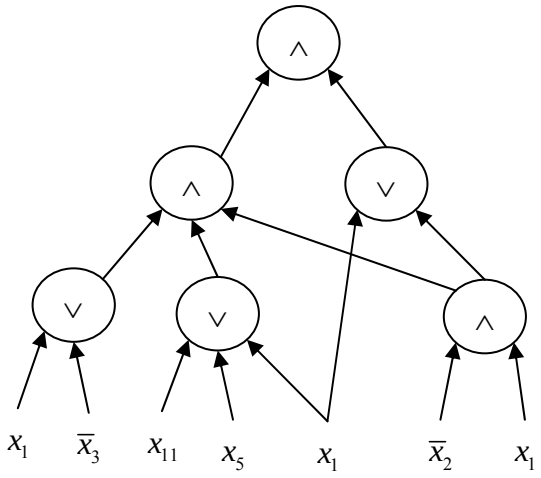
לכן הנאותות מתקיימת בהסתברות $\leq 1 - l\cdot\varepsilon$ כלומר בהסתברות $\leq \frac{1}{N^7}$.

<u>הערות:</u>
- השלמות מתקבלת בהסתברות 1.
- IP סגורה למשלים (כי $PSPACE$ סגורה למשלים).
- מספיק שכוח המוכיח יהיה $PSPACE$.

- יכולנו גם לשנות את הפרוטוקול כך שהמוודא יבצע את בדיקות התקינות רק בסיום הפרוטוקול ובמשך האיטרציות, רק ישלח ערכים אקראיים.
לכן ניתן להמיר את הפרוטוקול למשחק AM (זו ההגדרה האלטרנטיבית ל IP).

מעגלים בוליאניים

מדובר בגרפים הכוללים פעולות $\vee$ (OR) ו $\wedge$ (AND) ומשתנים או שלילה של משתנים (ליטרלים) בעלים.



<u>הגדרות</u>: מעגל בוליאני C_n על קלטים $x_1,...,x_n$ הוא גרף מכוון חסר מעגלים (DAG) המקיים את התנאים הבאים:

א. צמתים בעלי דרגת כניסה 0 נקראים צמתי **קלט** (או עלים) והם מסומנים ע"י ליטרלים מבין $x_1,...,x_n,\overline{x}_1,...,\overline{x}_n$.

ב. כל צומת פנימי מסומן ע"י אופרטור לוגי מבין $\{\vee,\wedge\}$ ונקרא **שער**.

ג. צומת בעל דרגת יציאה 0 נקרא צומת **פלט**. לרוב יהיה צומת יחיד כזה.

מעגל c_n מחשב פונקציה שמוגדרת באופן אינדוקטיבי מהעלים / שערי הקלט אל שערי הפלט.

כלומר, כל צומת בעגל מחשב פונקציה:

צומת קלט המסומן ע"י ליטרל l מחשב את הפונקציה $f(x_1,...,x_n)=l$.

צומת פנימי ש t הצמתים הנכנסים אליו מחשבים פונקציות $f_1,...,f_t$ והוא עצמו מסומן ע"י אופרטור $\otimes\in\{\vee,\wedge\}$ מחשב את הפונקציה $f(\vec{x})=\otimes_{i=1}^{t}f_i(\vec{x})$.

<u>מדדי סיבוכיות של מעגל:</u>

עומק - המרחק המקסימאלי בין צומת קלט לצומת פלט.

גודל - מספר הקשתות בגרף C_n.

<u>מוטיבציה:</u>

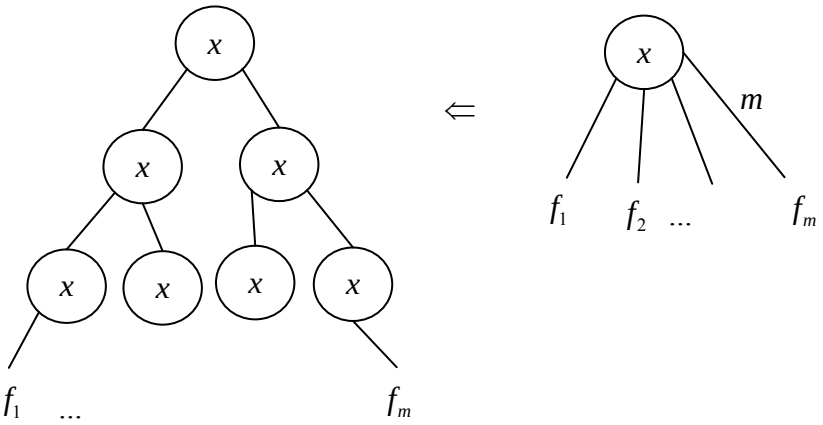
1. חומרה - השיטה הבסיסית לחישוב פונקציות מתבצעת באמצעות מעגלים בחומרה.
2. מקביליות - ניתן לחלק את חישוב הפונקציות לפעולות מקביליות - בכל שלב לחשב במקביל את כל הפונקציות הנמצאות באותה רמה בגרף. במקרה זה אנחנו זקוקים מספר מעבדים בסדר גודל של **גודל** המעגל. זמן מקבילי שקול **לעומק** המעגל.
3. קשר למחלקות / מדדי סיבוכיות אחרים. נראה את הקשר בין מה שאפשר לעשות במכונת טיורינג למה שאפשר לעשות באמצעות מעגלים בוליאניים.
4. מעגלים הוא מודל "קונקרטי" - בהינתן מעגל, קל לחשב את מדדי הסיבוכיות שלו.

וריאנטים של ההגדרה:

- אפשר להרשות להכניס למעגל קבועים $\{0,1\}$. זה לא כל כך מועיל כי אפשר לממש אותם. לדוגמה ניתן לממש ע"י $0 = x_1 \wedge \overline{x}_1$ ו $1 = x_1 \vee \overline{x}_1$.

- הגבלה על דרגת הכניסה $(fan-in)$, למשל דרגת כניסה 2.

עומק הגרף גדל פי $\log m$.



הגודל גדל פי 2.

- שערי NOT - במחיר של פקטור 2 בגודל וללא שינוי בעומק ניתן להיפטר משערי NOT. במקרה זה אין שינוי בעומק.

הרעיון: לכל שער המחשב פונקציה f נדאג לחשב גם את הפונקציה $\overline{f}$. באופן אינדוקטיבי"

בעלים: אם השער מסומן ע"י l נוסיף שער המסומן ע"י $\overline{l}$.

בצמתים פנימיים: נניח שהצומת מסומן $\wedge$ והשערים הנכנסים אליו מחשבים $f_1,...,f_t$.

$f = \wedge_{i=1}^{t} f_i$ אז $\overline{f} = \vee_{i=1}^{t} \overline{f}_i$ כאשר מהאינדוקציה, לכל $i \in \{1,...,t\}$, $\overline{f}_i$ כבר חושב.

- אפשר להוסיף גם שערים אחרים, למשל XOR: $g \oplus n \equiv \left(g \wedge \overline{h}\right) \vee \left(\overline{g} \wedge h\right)$

דוגמה:

$CVAL \triangleq \{(C_n, \vec{x}_n) \mid C_n(\vec{x}_n) = 1$ } C_n מעגל בוליאני בעל n קלטים ו $\vec{x}_n$ השמה למעגל והפלט הוא

טענה 1: $CVAL \in P$.

נחשב את ערך כל השערים במעגל. לשערי הקלט ישירות מההשמה $\vec{x}_n$. לכל שער פנימי שכבר חושבו כל השערים הנכנסים אליו - ע"פ הסימון $\wedge, \vee$ נחשב את ערכו.

זמן: פולינומי באורך הקלט.

זיכרון: ליניארי במספר הצמתים ב C_n.

האם אפשר לבצע זאת בסיבוכיות יעילה יותר?

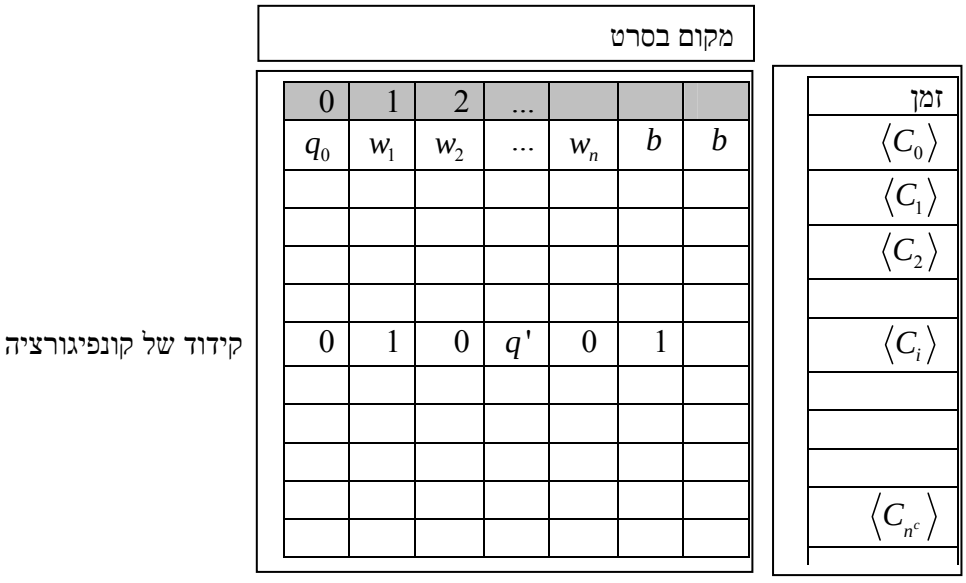
הגדרה:
שפה L היא שפה P-שלמה אם:
1. $L \in P$.
2. לכל $L' \in P$ מתקיים $L' \leq_{\log} L$ כאשר $L' \leq_{\log} L$ היא רדוקציית *Log Space*.

מסקנה:
אם L היא P-שלמה וכן $P \in DL$ אז $DL = P$.
אם L היא P-שלמה וכן $P \in NL$ אז $NL = P$.

משפט Ladner (1975):
$CVAL = \{C, \vec{x} \mid C(\vec{x}) = 1\}$ היא P-שלמה.

הוכחה:
1. $CVAL \in P$. כבר ראינו.
2. תהי $L \in P$. נראה ש $L \leq_{\log} CVAL$. $L \in P$
ולכן קיימת מ"ט M שרצה בזמן n^c ומקבלת את L.
נראה רדוקציה $f(\vec{x}) = (C_n, \vec{x})$.

נסתכל על הקונפיגורציות של M_L (כמו במשפט *cook* - הרצאה 12 בחישוביות)
טבלת חישוב של M_L על w:



שערים: $W_{t,i,a}$, עבור $0 \leq t, i \leq n^c$, $a \in \Gamma \cup Q$ (תו או מצב).
אינטרפרטציה: השער מחזיר 1 אם ורק אם בזמן t במקום i כתוב a.
שער פלט i: $W_{out} = \vee_{i=0}^{n^c} W_{n^c, i, q_A}$
שערים שמתאימים לשורה $t = 0$:
$W_{0,0,q_0} = 1$
$W_{0,i,b} = 1$ עבור $n+1 \leq i \leq n^c$ - בכל המקומות שמעבר לקלט יש בהתחלה b.
$W_{0,i,1} = x_i$ עבור $1 \leq i \leq n$.
$W_{0,i,0} = \overline{x}_n$.
כל שאר השערים עבור $t = 0$ מקבלים 0.

שערי הביניים:
$W_{t,i,a}$ תלוי ב $O(1)$ שערים.
בפרט, בשערים: $W_{t-1,j,b}$ כאשר $j \in \{i-1,i,i+1\}$ ו $b \in \Gamma \cup Q$.

החיווט המדויק תלוי רק ב M ולכן קבוע.

נכונות: האינטרפרטציה הנ"ל מתקיימת לכל t,i,a וההוכחה באינדוקציה על t.
זיכרון: לולאה על t,i,a (בזיכרון לוגריתמי) ולכל ערכים כנ"ל נחייט את $W_{t,i,a}$ כנדרש.

הערה:
לכל פונקציה $f:\{0,1\}^n \to \{0,1\}$ קיים מעגל המחשב אותה. בפרט מעגל שגודלו לכל היותר $O\left(n \cdot 2^n\right)$. (עושים זאת באמצעות DNF / CNF).

הגדרות:
- מעגל C_n על n משתנים $x_1,...,x_n$ מקבל את $L(C_n) \subseteq \{0,1\}^n$ כך ש $L(C_n) \triangleq \{\vec{x} \mid C_n(\vec{x})=1\}$.
- משפחה (לא יוניפורמית) של מעגלים היא אוסף של מעגלים, $\{C_n\}_{n\geq 0}$ המקבלת שפה $L = \bigcup_{n\geq 0} L(C_n)$.

הגדרה: $P / Poly$ - אוסף כל השפות שיש עבורן משפחת מעגלים (לא יוניפורמית) מגודל פולינומי.
כלומר קיים פולינום $s(n)=n^c$ אשר חוסם את גודלו של המעגל C_n במשפחה.

טענה: $P \subseteq P / Poly$.
הוכחה: נשתמש בהוכחה של משפט $Ladner$ ונבנה סדרת מעגלים מתאימה.
- לכל אורך n ההוכחה בנתה את אותו המעגל C_n (כלומר, ללא תלות בהשמה למשתנים $x_1,...,x_n$).
- אם M במשפט רצה $T(n)$ אז גודלו של C_n הוא $O\left(T^2(n)\right)$.

טענה: $BPP \subseteq P / Poly$.
הוכחה: בהינתן $L \in BPP$, תהי M מ"ט מתאימה. בפרט היא רצה בזמן n^c ולכל קלט x טועה בהסתברות קטנה מ $\frac{1}{2^n}$.
רעיון: נקבע את n ונראה איך לבנות את C_n (נטפל בכל אורך לחוד).
נוכיח שקיים קלט הסתברותי R_n ש $M\left(x,R_n\right)$ מוציאה את התשובה הנכונה לכל $x \in \{0,1\}^n$.
מכאן נקבל כמו בטענה ש $P \subseteq P / Poly$ שקיים מעגל בגודל פולינומי שמחשב את $M\left(x,R_n\right)$ כי R_n הוא כבר קבוע. (הגודל של R_n הוא לכל היותר n^c)

נסמן: $COINS_n : \{0,1\}^{n^c}$.
$\forall x \in \{0,1\}^n$ - $BAD_x = \{R \in COINS_n \mid M(x,R) \text{ is a mistake}\}$ - כלומר BAD_x הוא אוסף כל המחרוזות האקראיות R_n אשר עבורן המכונה M מחזירה תשובה שגויה.

$$BAD_n = \bigcup_{x\in\{0,1\}^n} BAD_x$$

נקבל ש: $\frac{|BAD_x|}{|COINS_n|} < \frac{1}{2^n}$ - כי המכונה טועה בהסתברות קטנה מ $\frac{1}{2^n}$.

$$\frac{|BAD_n|}{|COINS_n|} = \frac{\left|\bigcup_x BAD_x\right|}{|COINS_n|} < 2^n \cdot \frac{1}{2^n} = 1$$ לכן

לכן קיימת $R_n \in COINS_n \setminus BAD_n$ וזו המחרוזת המבוקשת.
נשים לב שרק הוכחנו שקיימת משפחת מעגלים מתאימה לשפה, אבל לא ניתן להסיק מכאן איך למצוא את המשפחה הזאת.

<u>הגדרה:</u> $\Sigma = \{0,1\}$

נאמר שפה L היא אונרית אם $L \subseteq \{1\}^*$.

<u>אבחנה 1:</u> קיימות שפות אונריות שאינן ב RE. לדוגמה:
$L = \{1^n \mid L(M_n = \Sigma^*)\}$ כאשר M_n היא המכונה ה n בספירה כלשהי.

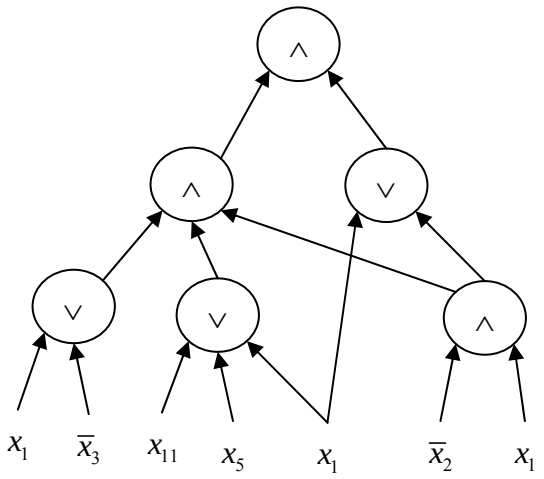
<u>אבחנה 2:</u> כל שפה אונרית היא ב $P/Poly$.
הסבר: לכל אורך n המעגל C_n המתאים הוא אחד מהבאים:
א. אם $1^n \notin L$ אז C_n צריך להוציא 0 על כל קלט מאורך n ולכן המעגל $x_1 \wedge \overline{x}_1$ יעשה את העבודה.
ב. אם $1^n \in L$ אז C_n צריך להוציא 1 רק על 1^n ולכן המעגל יהיה $\wedge_{i=1}^n x_i$.
בכל מקרה, גודל המעגל פולינומי.

<u>מסקנה משתי האבחנות:</u> קיימות ב $P/Poly$ שפות שאינן ב RE.

<u>משפט</u> (Kamp-Lipton):
אם $NP \subseteq P/Poly$ אז ההיררכיה הפולינומית קורסת.
מסקנה "סבירה": ל SAT אין מעגלים פולינומים.

<u>הגדרה:</u>
משפחה יוניפורמית של מעגלים - כמו משפחה לא יוניפורמית של מעגלים, אלא שקיימת מכונת טיורינג אשר מייצרת את המעגלים הללו.

<u>תזכורת:</u>
דיברנו על מעגלים בוליאניים:



הגדרנו **משפחת מעגלים**: $\{C_n\}_{n\geq 0}$ - מעגל אחד לכל אורך קלט.
הגדרנו את המחלקה $P/Poly$ - אוסף כל השפות שיש עבורן משפחת מעגלים מגודל פולינומיאלי.
ראינו ש $P \subseteq P/Poly$, $BPP \subseteq P/Poly$.
ראינו גם כמה שפות שאינן ב RE אשר נמצאות ב $P \subseteq P/Poly$.

לכן מגדירים את המושג **משפחת מעגלים אחידה**:
משפחה של מעגלים $\{C_n\}_{n\geq 0}$, נקראת $t(n)-time$ $\left[s(n)-space\right]$ אחידה אם קיימת מ"ט M שרצה זמן $t(n)$ [משתמשת בזיכרון $s(n)$] שעל קלט 1^n פולטת את C_n.

<u>אבחנה:</u>
$\log-space-unif\ \ poly-size\ circuits$ - מחלקת המעגלים שניתנים ליצר בצורה יוניפורמית ע"י מ"ט השתמשת בזיכרון לוגריתמי.
שפה זו מוכלת ב $poly-time-unif\ \ poly-size\ circuits$ - מחלקת המעגלים שניתנים ליצר בצורה יוניפורמית ע"י מ"ט הרצה זמן פולינומי.

$$\log-space-unif\ \ poly-size\ circuits \subseteq poly-time-unif\ \ poly-size\ circuits$$

זה טריוויאלי מההגדרה - כי כל מה שניתן לבצע בזיכרון לוגריתמי ניתן לבצע בזמן פולינומי.

$$poly-time-unif\ \ poly-size\ circuits \subseteq P$$

בהינתן x, נייצר את $C_{|x|}$ ונסמלץ [ראינו בהרצאה שעברה] $C_{|x|}(x)$, הכל בזמן פולינומי

$$P \subseteq poly-time-unif\ \ poly-size\ circuits$$

הוכחת משפט *Ladner* יחד עם האבחנה ש $\log-space$ מספיק.

<u>עומק מעגלים:</u>
עומק המעגל הוא המרחק הגדול ביותר משער קלט כלשהו אל שער פלט כלשהו.

<u>תזכורת:</u> כל פונקציה $f:\{0,1\}^n \to \{0,1\}$ ניתנת לחישוב ע"י מעגל מעומק 2.
פשוט נממש את ה CNF / DNF שממש את הפונקציה.
גודל המעגל יהיה אקספוננציאלי ודרגת הכניסה של הצמתים לא חסומה.

מה יקרה לעומק אם נגביל את המעגלים לגודל פולינומי?
מה יקרה לעומק אם נגביל את דרגת הכניסה של הצמתים (למשל ל 2)?

לכל פונקציה f כנ"ל קיים מעגל עם דרגת כניסה 2 (גודל אקספוננציאלי) ועומק $n+\log n$.

<u>טענה:</u> ל"רוב" הפונקציות $f:\{0,1\}^n \to \{0,1\}$ דרוש מעגל עם מספר צמתים אקספוננציאלי.
ל"רוב" הפונקציות, מעגל עם דרגת כניסת 2, עומקו $\Omega(n)$.
ההוכחה: טיעון ספירה:
מספר הפונקציות $f:\{0,1\}^n \to \{0,1\}$ הוא $2^{(2^n)}$.
מספר המעגלים עם t צמתים, יש לו לכל היותר t^2 קשתות, ולכל קשת יש שני צמתים בקצותיה.
לכן סה"כ מספר הגרפים המכוונים בעלי t צמתים חסום מלמעלה ע"י $2^{(t^2)}$.
לכל אחד מהצמתים צריך לבחור את הסוג שלו OR / AND, לכן יש להכפיל ב 2^t.
סה"כ עד כה: לכל היותר $2^{(t^2)} \cdot 2^t$.

ניקח לדוגמה $t = 2^{n/4}$. מספר המעגלים הוא $2^{2^{n/4}+2^{n/4}}$ וזה הרבה פחות מ 2^{2^n} שהוא מספר הפונקציות.

<u>בעיות פתוחות:</u>
- מצא פונקציה / שפה ב NP שדורשת מעגלים בגודל סופר פולינומי (כלומר שאינו פולינומי)?
- מצא פונקציה כנ"ל שדורשת מעגלים בגודל n^2?

מחלקות סיבוכיות (כמעט אותן הגדרות בסיבוכיות מעגלים ובסיבוכיות מקבילית):
NC^k [$non-unif$ או $\log-space-unif$ או $poly-time-unif$] היא אוסף השפות שקיימת עבורן משפחת מעגלים $\{C_n\}_{n\geq 0}$ שהיא [כנ"ל] בגודל **פולינומי,** דרגת כניסה 2 ועומק $O(\log^k n)$.
הקונבנציה בדרך כלל היא ש NC^k היא $non-unif$.

NC^0 - דרגת כניסה 2 ועומק קבוע אומר שהפלט תלוי במספר קבוע של ביטים בקלט.
לכן לרוב נדבר על NC^k מ 1 ומעלה.

נגדיר את שפות האיחוד: $[\ ]NC = \bigcup_{k\geq 1}[\ ]NC^k$ [ ]

AC, AC^k - כנ"ל, רק ללא הגבלת דרגת הכניסה.

<u>אבחנה:</u> $NC^k \subseteq AC^k$ - היא פשוט מקרה פרטי.
$AC^{k-1} \subseteq NC^k$ - ראינו בהרצאה הקודמת החלפת כל שער עם דרגת כניסה l בעץ מעומק $\log l$.
לכן $AC = NC$.

<u>דוגמה, טענה:</u>
CON - הקלט הוא גרף מכוון ושני צמתים והשאלה היא האם יש מסלול מכוון ביניהם.
נניח שהקלט A הוא מטריצת השכנויות של הגרף.
נניח שהצומת הראשון s הוא צומת מספר 1, והצומת השני, t, הוא צומת מספר 2.
נניח שמכל צומת יש קשת אל עצמו, כלומר $A[i,i]=1$. זה כמובן לא משפיע על הקשירות ולכן לא משנה את הבעיה.

נראה ש $CON \in \log-space-unif\ AC^1 \subseteq \log-space-unif\ NC^2$.

<u>הוכחה:</u>
נגדיר כפל בוליאני בין מטריצות כך ש $AND = \times$ ו $OR = +$.
בהינתן 2 מטריצות B,C מגודל $n \times n$ הכל הבוליאני מוגדר ע"י: $(B \cdot C)_{i,j} \triangleq \vee_{k=1,\ldots,n}\left(B_{i,k} \wedge C_{k,j}\right)$.

<u>אבחנה 1:</u> קיים מעגל בגודל פולינומי בעומק קבוע שעל קלט B,C פולט $B \cdot C$.
נשים לב שהמעגל פולט פלט בגודל פולינומי ולא רק ביט בודד.
עומק המעגל יהיה 2 קשתות והגודל הוא $O\left(n^3\right)$ (יש להוציא n^2 ביטי פלט, ולכל אחד דורש סדר גודל של n חוטים).

<u>הערה:</u> בפרט, בהינתן B, ניתן לחשב את $B^2 \triangleq B \cdot B$.

<u>אבחנה 2:</u>
נתבונן בסדרת המטריצות $A, A^2 = A \cdot A, A^4 = A^2 \cdot A^2, A^8, \ldots$ כאשר A היא מטריצת הקלט שלנו.
מתקיים: $\left(A^{(2^t)}\right)_{i,j} = 1$ אם ורק אם קיים בגרף מסלול באורך לכל היותר 2^t מ i אל j.

<u>"הוכחה": באינדוקציה:</u>
<u>בסיס</u>: $t = 0$: מהגדרת A והדרישה ש $A_{i,i} = 1$.

<u>צעד</u>: $A^{2^t} = A^{2^{t-1}} \cdot A^{2^{t-1}}$ - ראינו שיש מסלול מ i ל j באורך קטן או שווה ל 2^t אם ורק אם קיים k כך שקיים מסלול מ i ל k באורך קטן או שווה ל 2^{t-1} וגם קיים מסלול מ k ל j באורך קטן או שווה ל 2^{t-1}.
מהגדרת כפל בוליאני, הטענה נובעת.

<u>אבחנה 3:</u> $\left(A^{2^{\lceil \log n \rceil}}\right)_{1,2}$ הוא הפלט המבוקש.

לכן מספיק לחזור על סדרת הכפלות באמצעות מעגל זהה. נקבל שעומק המעגל הוא $O(\log n)$ והגודל הוא $O\left(n^3 \log n\right)$.
נשאר להראות יוניפורמיות: מ"ט שבונה את המעגל תשמור בזיכרון בכל זמן את החזקה, כלומר את t, ואת i, j, k.

לכן הזיכרון הכולל הוא לוגריתמי.

<u>טענה:</u> $NC^1 \subseteq \log-space-unif\ AC^1$.

<u>הוכחה:</u>

א. כבר ש ראינו $CON \in \log-space-unif\ AC^1$.

ב. CON היא NL-שלמה ביחס לרדוקציות $\log-space$.

בהינתן שפה $L \in NL$ ידוע $L \leq_{\log} CON$ באמצעות פונקציה f המקבלת קלט x ומייצרת גרף G ושני צמתים s,t כך ש $x \in L \Leftrightarrow f(x) = G_x = (G,s,t) \in CON$.

לכן המעגל שנבנה יהיה מחולק לשני חלקים. החלק הראשון יהיה מעגל לרדוקציה כדי להפוך את x ל G_x. החלק שני יהי המעגל שבודק האם G_x שייך ל CON.

ניזכר ברעיון הרדוקציה $L \leq_{\log} CON$:

על קלט x בונים את גרף הקונפיגורציות שמתאים ל x, G_x ובו רוצים לדעת אם יש מסלול מכוון מ C_{start} ל C_{Accept}.

בזיכרון לוגריתמי ניתן לבדוק האם ניתן לעבור בצעד אחד מקונפיגורציה אחת לקונפיגורציה שניה.

לכן נרצה לממש את ההמרה מ x ל G_x באמצעות מעגל.

איך נבנה את G_x?

כל קשת i,j בגרף הקונפיגורציות אומרת האם $C_i \mid -^1 C_j$.

בכמה משתנים של **הקלט** השאלה האם $C_i \mid -^1 C_j$ תלויה?

זה תלוי רק בביט אחד בקלט והוא הביט שעליו הראש מצביע בקונפיגורציה C_i. נסמן ביט זה ב x_k.

אם יש קשת גם כאשר $x_k = 0$ וגם כאשר $x_k = 1$ אז $(G_x)_{i,j} = 1$.

אם אין קשת גם כאשר $x_k = 0$ וגם כאשר $x_k = 1$ אז $(G_x)_{i,j} = 0$.

אם יש קשת כאשר $x_k = 1$ ואין קשת כאשר $x_k = 0$ אז $(G_x)_{i,j} = x_k$.

אם יש קשת כאשר $x_k = 0$ ואין קשת כאשר $x_k = 1$ אז $(G_x)_{i,j} = \overline{x}_k$.

המכונה שמחשבת את הגרף יכולה לבדוק מי מארבע האפשרויות הנ"ל מתקיימת ולחייט את המעגל בהתאם.

נקבל שהמעגל המחשב את G_x מתוך x הוא בעומק של קשת אחת בלבד.

<u>בעיות פתוחות:</u> האם $NC = P$?

האם כל דבר שאפשר לבצע ביעילות סדרתית אפשר לבצע ביעילות מקבילית?

האם $NC^1 = P$?

האם $NC^i = NC^{i+1}$ עבור $i \geq 1$?

האם $AC^i = NC^i$? האם דרגת הכניסה החסומה היא מגבלה מהותית?

מה שכן ידוע זה: $Parity\ Xor \in NC^1 \setminus AC^0$.

פשוט נייצר מעגל הבנוי מעץ בינארי כאשר בכל צומת רשום XOR ובעלים רשומים המשתנים.

כדי להמיר את ה XOR ל- AND ול- OR נצטרך להכפיל את עומק המעגל פי 2.

סה"כ נקבל שעומק המעגל הוא $2\log n$.

בהמשך נראה מדוע $Parity\ Xor \notin AC^0$.

<u>רדוקציות:</u>

רוצים: אם $L_1 \le L_2$ ו $L_2 \in AC^k$ אז $L_1 \in AC^k$, עבור $k \ge 0$. *

אם $L_1 \le L_2$ ו $L_2 \in NC^k$ אז $L_1 \in NC^k$, עבור $k \ge 1$. **

אפשרות ראשונה: רדוקציות פולינומיות, $\le_p$.

הבעיה היא שאם הרדוקציה פולינומית, לא נוכל לדעת על העומק הרבה, מלבד זה שהוא פולינומי, וזה כבר יותר מדי עמוק.

אפשרות שניה: רדוקציות לוגריתמיות, $\le_{\log}$.

כל ביט בפלט של הרדוקציה "ניתן לחישוב ב DL" ולכן בפרט בNL ולכן לפי המשפט הקודם, ניתן לחישוב ע"י מעגלים AC^1 או NC^2.

לכן, אם $L_1 \le_{\log} L_2$ ו $L_2 \in AC^k$ אז $L_1 \in AC^{\max\{k,1\}}$.

אם $L_1 \le_{\log} L_2$ ו $L_2 \in NC^k$ אז $L_1 \in AC^{\max\{k,2\}}$.

רדוקציות AC^0:

<u>שער אוב לשפה A:</u> זהו שער שמקבל קלטים $y = y_1,...,y_t$ ומוציא ביט שאומר האם $y \in A$ או לא.

אומרים ששפה L_1 ניתנת לרדוקציית AC^0 לשפה L_2 ומסמנים $L_1 \le_{AC^0} L_2$ אם קיימת משפחת מעגלים $\{C_n\}_{n\ge 0}$ שהם <u>בגודל פולינומי</u>, <u>עומק קבוע</u>, משתמשים בשערי אוב ל L_2 ומחשבים את L_1.

<u>הערות:</u>

- וריאנטים אחידים.
- מה שרצינו (* ו **) אכן מתקיים
  - * - ע"י החלפת שערי האוב במעגלי AC^k מתאימים.
  - ** - עלינו להיפטר מהדרגות הגבוהות של האוב ע"י פיצול לעץ.
    ניקח את מעגל הרדוקציה ו"נפתח" את שערי ה $\wedge, \vee$ מדרגה גבוה. אחרי הטרנספורמציה, העומק הוא $O(\log n)$. עדין על כל מסלול במעגל יש $O(1)$ שערי אוב. עכשיו נחליף כל אחד מהם במעגל NC^k.

<u>דוגמה:</u> $PARITY \le_{AC^0} MAJORITY$

$MAJORITY$ היא פונקציה שמקבלת מחרוזת של 0-ים ו 1-ים ומחזירה 1 אם ורק אם לפחות חצי מביטי הקלט הם 1.

בצירוף לטענה ש $PARITY \notin AC^0$ נקבל ש $MAJORITY \notin AC^0$ (כי אם $MAJORITY \in AC^0$ אז באמצעות רדוקציה נקבל ש $PARITY \in AC^0$).

- נחשב $PARITY$ בהנחה שי לנו שערי EX_l לכל l. EX_l הוא שער שמחזיר 1 אם יש בדיוק l 1-ים בקלט.
- מספיק לחשב $EX_1 \vee EX_3 \vee EX_5 \vee ...$ בשביל לחשב $PARITY$ כאשר כל אחד מה-EX_l מחובר לכל ביטי הקלט.
- נבנה EX_l משערי אוב ל $MAJORITY$.

- אם $L \geq \frac{n}{2}$ נבצע: $MAJ\left(x,0^{2l-n}\right) \wedge MAJ\left(\overline{x},1^{2l-n}\right)$

הקלט של $MAJ\left(x,0^{2l-n}\right)$ הוא בדיוק $2l$ כי $|x| = n$. השער הזה מחזיר 1 אם ורק אם יש לפחות l 1-ים ב x.

השער $MAJ\left(\overline{x},1^{2l-n}\right)$ בעל אותו אורך קלט, והוא מוציא 1 אם ורק אם ב $\overline{x}$ יש $n-l$ 1-ים.

וזה אומר שב x יש לכל היותר l 1-ים.

כלומר, לאחר חיבור שני השערים, נקבל 1 אם ורק אם מספר ה1-ים בקלט הוא l.

- אם $l < \frac{n}{2}$ עושים משהו דומה.

<u>תזכורת:</u> דיברנו על עומק מעגלים.
הגדרנו את המחלקות NC^k, AC^k.
NC^k [$non-unif$ או $\log-space-unif$ או $poly-time-unif$] היא אוסף השפות שקיימת עבורן משפחת מעגלים $\{C_n\}_{n\geq 0}$ שהיא [כנ"ל] בגודל **פולינומי,** דרגת כניסה 2 ועומק $O(\log^k n)$.
הקונבנציה בדרך כלל היא ש NC^k היא $non-unif$.

AC^k - אותו הדבר מלבד זה שדרגת הכניסה לא מוגבלת.

<u>משפט:</u> $XOR_n \notin AC^0$ $[FSS82, Yao84, Hastad86]$

<u>טענה:</u> כל מעגל מעומק d עבור XOR_n דורש גודל $2^{n^{\Omega(1/d)}}$ $[Smolasky89]$

רעיונות:
- לכל פונקציה במחלקה AC^0 יש תכונה מסוימת (בנוסף לתכונת השייכות ל AC^0), אשר אין אותה ל XOR_n.
- התכונה: הפונקציה "ניתנת לייצוג" ע"י פולינום מדרגה "נמוכה".
- בעיות:
  - o מעל $GF(2)$ הפונקציה דווקא כן ניתנת לייצוג: $XOR_n = \sum x_i$.
  - o $AND_n = \prod x_i$ - כלומר הפונקציה הזאת דורשת דרגה "גבוהה".

נקבע פרמטרים / סימונים:
d - עומק המעגל.
s - גודל המעגל.
נאמר שפונקציה f ניתנת לייצוג אם קיים פולינום מעל $Z_3 = \{0,1,2\}$ מדרגה $O(\log s)^d$ שמחשב את f נכון על $0.99 \cdot 2^n$ מההשמות הבוליאניות $\{0,1\}^n$.

למה Z_3? על מנת להיפטר מ $XOR_n = \sum x_i$.
אפשר לקרוא לאברי Z_3 גם בשמות $\{-1,0,1\}$.
במקרה $Z_3 = \{0,1,2\}$ נקבל ש $\oplus x_i$ ממשמ את ה $Parity$.
במקרה $Z_3 = \{-1,0,1\}$ כאשר $True = -1$ ו $False = 1$ נקבל ש $\prod x_i$ ממשמת את $Parity$.

מעל $GF(3)$ לא נראה שיש פולינום שמייצג את XOR.
(AND_n ניתן לייצוג ע"י הקבוע 0 - המעגל יענה נכון לכל הקלטים מלבד הקלט שהוא וקטור אפסים, אבל טעות אחת מותרת לנו).

<u>טענה 1:</u>
תהי f פונקציה הניתנת לחישוב ע"י מעגל מעומק d וגודל s.
$\Leftarrow$ ל f יש ייצוג פולינומי כנ"ל.
<u>הוכחה:</u>
נניח בה"כ שבמעגל יש רק שערי $\neg, \vee$ (ואין שערי $\wedge$).
לצורכי ספירת עומק וגודל, נספור רק את שערי ה $\vee$ ולכן משערי $\wedge$ ע"י דה-מורגן.

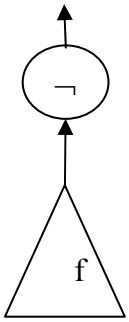
נייצג כל שער במעגל ע"י פולינום מתאים ובפרט הפולינום המתאים לשער הפלט הוא הפולינום המתאים.

שער קלט: x_i

ייוצג ע"י פולינום $p(\vec{x}) = x_i$.

שערים פנימיים:

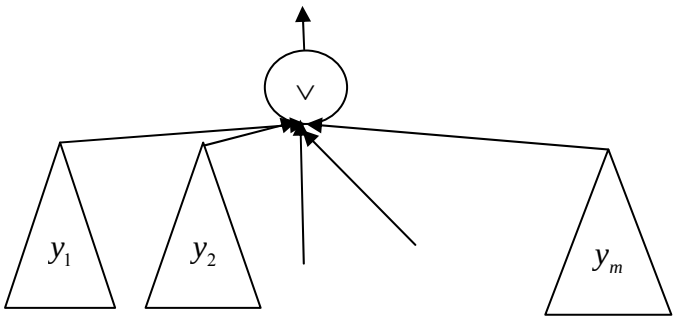
שער NOT :



אם f מיוצגת ע" הפולינום $P(\vec{x})$ אז $P'_{NOT}(\vec{x}) = 1 - P(\vec{x})$.

P'_{NOT} מייצג את הפונקציה החדשה, בכל מקום שבו P מייצג את f , ודרגתו היא כדרגת P .

שער OR :



היינו רוצים להשתמש ב $P_{Or}(\vec{y}) = 1 - \prod_{i=1}^{m}(1 - y_i)$ אבל זה יצא גדול מדי וגם השגיאה גדולה מדי.

היינו רוצים אולי להשתמש בקבוע 1: $P_{Or}(\vec{y}) = 1$.

הבעיה היא שה- y -ים אינם הקלטים המקוריים ולכן ההסתברות שהם יהיו כולם אפסים היא לא $\frac{1}{2^n}$ אלא יכולה להיות גבוהה בהרבה.

נבחר באקראי פולינום $q(\vec{y}) = \sum_{i=1}^{m} a_i y_i$ כאשר $a_i \in \{0,1,2\}$ נבחר באופן אקראי.

מה קורה אם $\vec{y} = \vec{0}$? במקרה כזה נקבל ש $q(\vec{0}) = 0$ כנדרש.

מה קורה אם $\vec{y} \neq \vec{0}$? $\Pr(q(\vec{y}) = Or(\vec{y})) = \frac{1}{3}$.

הסבר: קיים j כך ש $y_j = 1$ ולכן $q(\vec{y}) = \sum_{i \neq j} a_i y_i + a_j$.

לאחר שנחשב את $\left(\sum_{i \neq j} a_i y_i\right) \in \{0,1,2\}$ ונוסיף לו את a_j באופן אקראי בהתפלגות אחידה, נקבל $q(\vec{y}) \in \{0,1,2\}$ אשר מתפלג באופן אחיד.

לא רוצים שיצא 2. מה עושים? במקום להסתכל על $q^2(\vec{y})$.
כעת, אם $q(\vec{y})=0$ אז $q^2(\vec{y})=0$ ואם $q(\vec{y})\in\{0,1\}$ אז $q^2(\vec{y})=1$.
כעת $\Pr\left(q^2(\vec{y})=Or(\vec{y})\right)\geq\frac{2}{3}$.

<u>הבניה הסופית:</u>
נגדיר: $k=\log_3(100s)$.
נבחר באקראי בצורה שתוארה פולינומים $q_1,q_2,...,q_k$.
הפולינום שלנו יהיה $P(\vec{y})=P_{Or}\left(q_1^2(\vec{y}),q_2^2(\vec{y}),...,q_k^2(\vec{y})\right)$.
מהי דרגת הפולינום? $2k$.
מהי ההסתברות לטעות?
$\Pr\left(P(\vec{y})\neq Or(\vec{y})\right)\leq\left(\frac{1}{3}\right)^k=\frac{1}{100s}$.
נפעיל את הבניה המתוארת לכל שערי המעגל.
כלומר, ל $\vec{x}$ (השמה למעגל) נתון, ההסתברות לטעות היא לכל היותר מספר השערים כפול ההסתברות לטעות באחד מהם.
כלומר, $\frac{1}{100S}\cdot S=\frac{1}{100}$.

(הבהרה: $\vec{x}$ - השמה מקורית. $\vec{y}$ - חישובי ביניים).

לכל $\vec{x}$, ההסתברות שהפולינום המקבל טועה היא קטנה מ $\frac{1}{100}$.
מכאן קיים פולינום שטועה על לכל היותר $\frac{1}{100}$ מ 2^n ההשמות $\vec{x}$.
<u>דרגה:</u> $(2k)^d$ לכל רמה - כלומר $O(\log s)^d$.

<u>טענה 2:</u> ל XOR_n אין ייצוג ע"י פולינומים מדרגה $D=\alpha\cdot\sqrt{n}$ כאשר α הוא קבוע שיוגדר בהמשך.
<u>הוכחה:</u>
הרעיון:
נניח בשלילה שקיים ייצוג כנ"ל.
נראה שמכאן נובע ש"להרבה מאוד" פונקציות יש ייצוגים מדרגה נמוכה.
מכאן באמצעות טיעון ספירה, נראה שזה לא יתכן כי יש הרבה מאוד פונקציות ומעט מאוד ייצוגים.

נניח בשלילה שלפונקציה XOR_n יש ייצוג ע"פ פולינום $P(\vec{x})$ מדרגה D שמחשב נכון על קבוצה S שגודלה $0.99\cdot 2^n$.
נראה שלכל פונקציה $f:S\to Z_3$ יש ייצוג ע"י פולינום מדרגה נמוכה, ונקבל סתירה ע"י נימוק ספירה.

נבצע המרה:
הערכים הבוליאניים יזוזו: $\{0,1\}$ יעברו ל $\{-1,1\}$, כלומר $Fasle=1$ ו $True=-1$.
זוגיות: במקרה הקודם זה היה $\sum x_i \bmod 2$. בייצוג החדש: $\prod x_i$.

פונקצית ההמרה: $\phi:\{0,1\}\to\{1,-1\}$

$\phi(a)=1-2a$, $\phi^{-1}(b)=\frac{1-b}{2}$.

$P'(\vec{x})\triangleq\phi\left(P\left(\phi^{-1}(\vec{x})\right)\right)$ - מתייחס להשמות $\vec{x}\in\{\pm1\}^n$.

מהי דרגת הפולינום החדש? $\deg(P')=\deg(P)$.

P' מחשב נכון את $\prod x_i$ על הקבוצה $S'=\phi(S)$.

<u>טענת עזר:</u>

בהנחות שעשינו עד כה, לכל פונקציה $f:S'\to Z_3$ יש פולינום מדרגה $\frac{n}{2}+D$ שמחשב אותה נכון.

<u>הוכחת טענת העזר:</u>

תמיד קיים פולינום מולטי ליניארי (דרגה 1 בכל משתנה) מדרגה $n\ge$ אשר מחשב את f.

נסמנו ב q. $q=\sum_{A\subseteq[n]}C_A\underbrace{\prod_{i\in A}x_i}_{X_A}$

$\bar{A}$ הוא המשלים של A ב n. כלומר $\bar{A}=\{1,...,n\}\setminus A$.

נקבל ש: $X_A\cdot X_{\bar{A}}=\prod_{i=1}^{n}x_i$

לכן $X_A=\left(\prod x_i\right)\cdot X_{\bar{A}}$.

$$q=\sum_{A\subseteq[n]}C_AX_A=\sum_{\substack{A\subseteq[n]\\|A|\le\frac{n}{2}}}C_AX_A+\sum_{\substack{A\subseteq[n]\\|A|>\frac{n}{2}}}C_AX_A=\sum_{\substack{A\subseteq[n]\\|A|\le\frac{n}{2}}}C_AX_A+\sum_{\substack{A\subseteq[n]\\|A|>\frac{n}{2}}}C_A\left(\prod x_i\right)X_{\bar{A}}$$

החלפנו את כל ה A-ים הגדולים מחצי n במשלימים שלהם. שילמנו במחיר של הכפלה ב $\prod_{i=1}^{n}x_i$ אולם אנחנו יודעים שלזה קיים ייצוג של פולינום מדרגה נמוכה - $P'(\vec{x})$.

$$\sum_{\substack{A\subseteq[n]\\|A|>\frac{n}{2}}}C_A\left(\prod x_i\right)X_{\bar{A}}=\left(\prod x_i\right)\cdot\sum_{\substack{A\subseteq[n]\\|A|>\frac{n}{2}}}C_AX_{\bar{A}}=P'(\vec{x})\cdot\sum_{\substack{A\subseteq[n]\\|A|>\frac{n}{2}}}C_AX_{\bar{A}}$$

סה"כ קיבלנו: $q=\sum_{\substack{A\subseteq[n]\\|A|\le\frac{n}{2}}}C_AX_A+P'(\vec{x})\cdot\sum_{\substack{A\subseteq[n]\\|A|>\frac{n}{2}}}C_AX_{\bar{A}}$

קיבלנו פולינום שדרגתו היא $\frac{n}{2}+D$ ומחשב נכון את f לכל $\vec{x}$ ב S'.

<u>נימוק ספירה:</u> נוכיח שמספר הפונקציות $f:S'\to Z_3$ גדול ממספר הפולינומים מדרגה $\frac{n}{2}+D$.

מכאן נקבל שיש פונקציה כנ"ל שאין לה פולינום מתאים, בסתירה להנחה שלפונקציה XOR_n יש פולינום מדרגה D.

מספר הפונקציות $f:S'\to Z_3$ הוא $3^{|S'|}=3^{\left(0.99\cdot 2^n\right)}$.

מספר הפולינומים מדרגה $\frac{n}{2}+D$ מעל Z_3 הוא: $3^{\sum_{i=0}^{\frac{n}{2}+D}\binom{n}{i}}$ - בחירת ה C_A-ים.

$$\sum_{i=0}^{\frac{n}{2}+D}\binom{n}{i}=\underbrace{\sum_{i=0}^{\frac{n}{2}}\binom{n}{i}}_{=\frac{2^n}{2}}+\underbrace{\sum_{i=\frac{n}{2}+1}^{\frac{n}{2}+D}\binom{n}{i}}_{\le D\cdot\binom{n}{n/2}\le D\cdot\frac{2^n}{\sqrt{n}}=\alpha\cdot 2^n}\le\left(\frac{1}{2}+\alpha\right)2^n .$$

כלומר, לכל $\alpha\le 0.49$, מספר הפולינומים קטן ממספר הפונקציות.

<u>הוכחת המשפט:</u>

נניח שיש ל XOR_n מעגל מעומק d בעומק s.

<u>לפי טענה 1:</u> XOR_n ניתנת לייצוג ע"י פולינום מדרגה $O\left(\log s\right)^d$.

<u>לפי טענה 2:</u> כל פולינום המייצג את XOR_n, דרגתו היא לפחות $O\left(\frac{1}{4}\sqrt{n}\right)$.

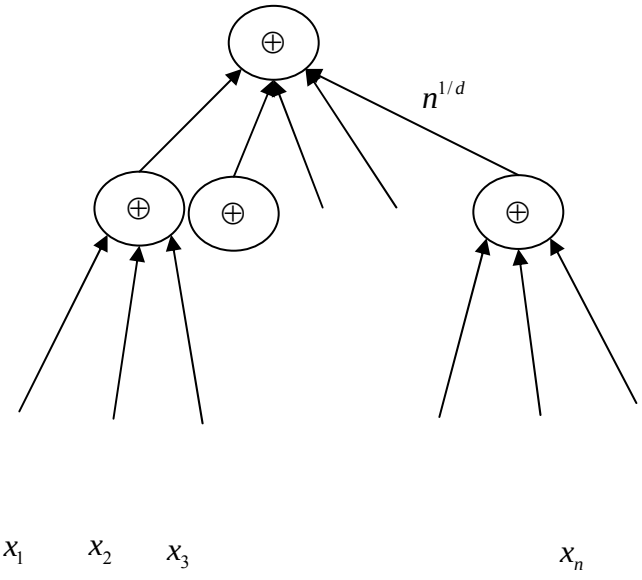
מסקנה: $\log\left(s\right)^d\ge\sqrt{n}$ (זרקנו את הקבועים).

מכאן ש $\log\left(s\right)\ge\left(\sqrt{n}\right)^{\frac{1}{d}}$.

לכן: $s=2^{\left(n^{\Omega(1/d)}\right)}$.

<u>בניה:</u> נניח שיש שערי XOR עם דרגת כניסה $n^{1/d}$.

צעד א':



צעד ב':
נחליף כל שער $\oplus$ במעגל ע"י CNF / DNF נאיבי.
עומק 2:
גודל $O\left(n\cdot 2^{n^{1/d}}\right)$

מתקבל (נאיבית) עומק $2d$.

צעד ג': בשכבות הזוגיות נחליף $\oplus$ ב AND של OR-ים ובשכבות האי זוגיות נחליף $\oplus$ ב OR של AND-ים.
כעת נוכל להצמיד שכבות ולקבל עומק d.

קושי של קירובים

דוגמה:
גודל כיסוי הצמתים המינימאלי בגרף G $\triangleq fvc(G)$.
בהנחה ש $P \neq NP$ לא ניתן לחשב את הפונקציה הזאת בזמן פולינומיאלי במכונה דטרמיניסטית.

קיים אלגוריתם שהוא 2-קירוב עבור הפונקציה fvc.
כלומר, קיים אלגוריתם פולינומי A, שלכל גרף G מתקיים $fvc(G) \le A(G) \le 2 \cdot fvc(G)$.
למעשה, האלגוריתם מוצא לא רק גודל של כיסוי שגודלו עד פי 2 מגודל הכיסוי המינימאלי, אלא גם מוצא את הכיסוי הזה עצמו.

אבחנות:
1. $f-col(G)$ - הפונקציה המוצאת את המספר המינימאלי של צבעים הנחוץ לצביעת G.
$f-col$ לא ניתנת ל C-קירוב עבור $C < \frac{4}{3}$ בהנחה ש $P \neq NP$.
הוכחה: נניח בשלילה שקיים אלגוריתם קירוב כזה A.
בפרט, A פולינומי.
אם מספר הצביעה של G הוא לכל היותר 3, אז התשובה ש $A(G)$ יוציא היא תמיד קטנה מ 4.
אם מספר הצביעה של G הוא 4 או יתר, אז התשובה של $A(G)$ היא לפחות 4.
לכן נוכל באמצעות האלגוריתם A לפתור את $3col$ (שאל את A וקבל אם $A(G) < 4$) ומכיוון ש $3col$ היא בעיה NP-שלמה, נקבל ש $P = NP$ בסתירה להנחה.

2. לא קיים אלגוריתם שמקרב α-חיבורית את הפונקציה $\omega(G)$ אשר מחזירה את גודל הקליק המקסימאלי בגרף G.
א. נניח שידוע ש $\omega(G)$ הוא כפולה של $\alpha+1$.
אזי, בהינתן אלגוריתם קירוב A כנ"ל, ניתן לחשב את $\omega(G)$.
$\omega(G) \in \{\alpha+1, 2(\alpha+1), 3(\alpha+1), ...\}$
האלגוריתם A יפלוט פלט אשר נמצא בין $i(\alpha+1), (i+1)(\alpha+1)$.
ב. קל בהינתן גרף G' לבנות גרף G כך ש $\omega(G) = (\alpha+1) \cdot \omega(G')$.
בהינתן גרף $G' = (V', E')$, נחליף כל צומת $v \in V'$ ב $\alpha+1$ צמתים $v_1, v_2, ..., v_{\alpha+1}$ עם כל הקשתות ביניהם ובנוסף אם $(u,v) \in E'$ אז לכל $i, j \in \{1, ..., \alpha+1\}$, $(u_i, v_j) \in E$.
כעת, אם האלגוריתם A יפלוט ערך הנמצא בין $(i-1)(\alpha+1)$ לבין $i(\alpha+1)$ נדע שהערך המתאים ל G' הוא i.

3. $P(x)$ - בעיית מקסימיזציה כלשהי.
נניח שקיימת רדוקציה f מ SAT ל $P(x)$ ופרמטרים $k, g \ge 1$ שמקיימת.
$$\varphi \in SAT \Rightarrow P(f(\phi)) \ge k$$
$$\varphi \notin SAT \Rightarrow P(f(\phi)) < \frac{k}{g}$$
אזי לא קיים g-קירוב עבור הבעיה $P(x)$ בהנחה ש $P \neq NP$.

נניח בשלילה שקיים אלגוריתם A קירוב כזה עבור $P(x)$.

$$A(f(\varphi)) \geq \frac{k}{g} \Leftarrow P(f(\varphi)) \geq k \Leftarrow \varphi \in SAT$$

$$A(f(\varphi)) < \frac{k}{g} \Leftarrow P(f(\varphi)) < \frac{k}{g} \Leftarrow \varphi \notin SAT$$

אזי האלגוריתם עבור SAT על קלט φ:

א. חשב את $f(\varphi)$

ב. חשב את $A(f(\varphi))$ והשווה ל $\frac{k}{g}$ - אם $A(f(\varphi)) \geq \frac{k}{g}$ קבל ואחרת דחה.

MIP: *Multi – Prover IP*

PCP: *Probablstically Checkable Proofs*

$(r(n), q(n))$ - מוודא הוא מ"ט פולינומית הסתברותית V שיש לה שני "סרטים" מיוחדים:

א. סרט אקראיות r שמכיל $r(n)$ ביטים אקראיים.

ב. "סרט" הוכחה Π.

על קלט x ואקראיות r, המוודא מחשב קבוצה בגודל $q(n)$ של ביטים מתוך Π שהוא רוצה לקרוא והוא מקבל אותם. על סמך ביטים אלה וכן x, r, הוא מקבל או דוחה.

מבחינת רעיונית, Π הוא ההוכחה לכך ש $x \in L$.

V הוא מוודא עבור שפה L:

לכל $x \in L$, קיימת הוכחה Π_x, כך ש 1=(V מקבל את x עם Π_x) $\Pr_r$.

לכל $x \notin L$, ולכל Π, $\Pr_r$(V מקבל את x עם Π) $< \frac{1}{2}$.

נשים לב שניתן לבצע הגברה (כלומר הקטנת הסיכוי לטעות) במחיר של הגדלת $r(n)$ ו $q(n)$.

מחלקת השפות שיש להן מוודא עם פרמטרים $O(r(n)), O(q(n))$ $\triangleq$ $PCP(r(n), q(n))$.

<u>אבחנות</u>:

$PCP(poly, 0) = coRP$.

$PCP(0, poly) = NP$.

כיוון: $PCP(0, poly) \subseteq NP$ - אומנם אורך ההוכחה ב PCP איננו בהכרח פולינומי אבל במקרה זה אפקטיבית הוא כן (מה שבפועל מתוך ההוכחה יקרא על ידי המוודא).

מ"ט א"ד תנחש את ההוכחה Π (כלומר תנחש רק את הקטעים שיקראו בפועל) ואז מריצה את המוודא.

$PCP(poly, poly) = EXP$

משפט ה PCP: $PCP\left(\log(n),1\right)=NP$.

"הוכחה":

כיוון $PCP\left(\log(n),1\right)\subseteq NP$

אם $L\in PCP\left(\log(n),1\right)\subseteq PCP\left(\log(n),poly\right)\subseteq NP$.

נשתמש באותה הוכחה עבור $PCP\left(0,poly\right)\subseteq NP$:

"ננחש" (האורך האפקטיבי הוא עדין פולינומי) את ההוכחה Π, ונסמלץ את V בכל $2^{c\cdot\log(n)}=n^c$ המסלולים. אם הוא מקבל בכל המסלולים אז נקבל ואחרת נדחה.

הקשר לקירובים:

M_k: (סיפוק אילוצים).

נתונים m אילוצים (פונקציות) f_i. כל אחד מהם תלוי ב k מתוך N משתנים $y_1,y_2,...,y_n$.

האילוצים נתונים בטבלת אמת.

מבוקש: איזה חלק מ m האילוצים ניתן לספק בו זמנית ע"י השמה כלשהי ל $\vec{y}$? (מספר בין 0 ל 1).

למה:

קיים k ורדוקציה פולינומית h מ SAT ל M_k שמקיימת את הדברים הבאים:

$$\varphi\in SAT\Rightarrow M_k\left(h\left(\phi\right)\right)=1$$

$$\varphi\notin SAT\Rightarrow M_k\left(h\left(\phi\right)\right)<\frac{1}{2}$$

מסקנה:

אין 2-קירוב ל M_k (בהנחה ש $P\neq NP$).

הוכחת הלמה:

$SAT\in NP$ ולכן קיים V כמובטח במשפט ה PCP. בפרט, V קורא k ביטים מההוכחה, ומשתמש ב $c\cdot\log(n)$ ביטים אקראיים.

נבנה קלט ל M_k:

$m=2^{c\cdot\log(n)}=n^c$.

$N\leq m\cdot k$ - אורך ההוכחה האפקטיבי.

$\left(y_1,...,y_N\right)$ - הביטים של ההוכחה.

$f_i=(i$ פונקציה של המשתנים שקוראים עם אקראיות $)$ - מחזיר 1 אמ"מ V מקבל את x כאשר האקראיות שלו היא i ו k הביטים שקרא הם הנ"ל.

אם $\varphi\in SAT$ אז קימת Π שגורמת ל V לקבל בהסתברות 1 ולכן קיימת השמה $\vec{y}$ שגורמת לכל ה f_i להסתפק.

אם $\varphi\notin SAT$ אז לכל Π הסתברות הקבלה קטנה מחצי ולכן לכל השמה $\vec{y}$ מספר האילוצים שמסתפק קטן מ $\frac{1}{2}m$.

$M3S$:

נתון פסוק $3CNF$ שבכל פסוקית יש בדיוק שלושה ליטרלים שונים זה מזה.
רוצים למצוא את החלק הגדול ביותר של הפסוקיות שניתן לספק בו זמנית.

מתקיים:
- $M3S$ קשה לפחות כמו $3SAT$.
- לכל פסוקית עם 3 ליטרלים שונים זה מזה, $7/8$ מההשמות מספקות אותה.
אם הפסוקית היא $(l_1 \vee l_2 \vee l_3)$ אז רק ההשמה שנותנת 0 לשלושת הליטרלים לא מספקת אותה.
לכן מטיעון ספירה נקבל שקיימת השמה שמספקת $7/8$ מהפסוקיות.
לכן אלגוריתם שתמיד מחזיר $7/8$ הוא $7/8$-קירוב ל $M3S$.

קיים קבוע ε_0 וקיימת רדוקציה h' מ SAT ל $M3S$ שמקיימת:

$$\varphi \in SAT \Rightarrow M3S\left(h'(\varphi)\right)=1$$

$$\varphi \notin SAT \Rightarrow M3S\left(h'(\varphi)\right)<\frac{1}{1+\varepsilon_0}$$

מסקנה: אין $(1+\varepsilon_0)$-קירוב עבור $M3S$ (בהנחה ש $P \neq NP$).

הוכחה:
אם $\varphi \in SAT$ אז $M_k\left(h(\varphi)\right)=1$
אם $\varphi \notin SAT$ אז $M_k\left(h(\varphi)\right)<\frac{1}{2}$.

ניקח את φ, נמיר אותו ל $h(\varphi)$ באמצעות ההוכחה הקודמת.
כעת נמיר כל f_i ב CNF מתאים בגודל $\geq 2^k$ פסוקיות.
כעת נמיר כל CNF ב $3CNF$ מתאים (סטנדרטי) (מגדיל פי k את מספר הפסוקיות).
אם $M_k\left(h(\varphi)\right)=1$ אז כל הפסוקיות ניתנות לסיפוק בו זמנית וכך גם לאחר ההפיכה ל $3CNF$.
אם $M_k\left(h(\varphi)\right)<\frac{1}{2}$ אז כל השמה מפירה לפחות חצי מהאילוצים לכן כל השמה מפירה לפחות $\frac{1}{2\cdot 2^k}$ מהפסוקיות. לאחר הפיכה ל $3CNF$, כל השמה מפירה לפחות $\frac{1}{2\cdot 2^k \cdot k}$ מהפסוקיות.

גודל הקליק המקסימאלי בגרף G $\triangleq \omega(G)$

מסתבר: הרדוקציה הסטנדרטית מ $3SAT$ לקליק, בהינתן פסוק φ עם m פסוקיות, מייצר G כך ש $\omega(G)=m\cdot M3S(\varphi)$.

את הקושי בקליק קל להגביר לכל קבוע ע"י שימוש במכפלות גרפים.

$\varphi=c_1 \wedge c_2 \wedge \ldots \wedge c_m$ כאשר $c_i=l_{i1} \vee l_{i2} \vee l_{i3}$.
קשתות: אין קשתות בתוך השלשות.
בין צמתים שבאים משלשות שונות נשים קשתות בין כל זוג ליטרלים מלבד משתנים עם שלילתם.
אם בגרף יש קליק בגודל l, אז יש דרך לספק l מהפסוקיות, ולהיפך.

לכן קיים קליק בגודל l בגרף אם ורק אם קיימת השמה שמספקת l מהפסוקיות.

בעיות ספירה

תיאור בעיית ספירה	בעיית הכרעה מקבילה	קושי בעיית הספירה
בהינתן פסוק CNF - רוצים לדעת כמה השמות מספקות אותו. הבעיה נקראת $\#SAT$ או f_{SAT}.	"קשה" (בהנחה ש $P \neq NP$) לבדוק האם פסוק CNF הוא ספיק	קשה
מספר העצים הפורשים בגרף נתון.	קל	קל
גרף דו צדדי ורוצים לדעת כמה שידוכים מושלמים יש בגרף הזה.	קל	קשה

הגדרה:

$FP \triangleq \{ f : \{0,1\}^* \to \mathbb{N} \mid$ f ניתנת לחישוב בזמן פולינומי $\}$

$\#P = \{ f : \{0,1\}^* \to \mathbb{N} \mid f(x) =$ מספר המסלולים המקבלים של M כך שלכל x על M 'ט א"ד פולי' קיימת מ"ט $\}$

דוגמה: $\#SAT \in \#P$

- בהינתן פסוק φ נחש השמה $\vec{x}$ לפסוק φ ובדוק האם $\varphi(\vec{x}) = True$ ואם כן קבל.

בעיה פתוחה: האם $FP = \#P$?

אם $P \neq NP$ אז $SAT \notin P$ ולכן $\#SAT \notin FP$. לכן $FP \neq \#P$.
האם $FP \neq \#P \Leftarrow P \neq NP$? לא ידוע.

הגדרה: פונקציה g נקראת $\#P$-שלמה אם מתקיימים התנאים הבאים:

1. $g \in \#P$.
2. $\forall f \in \#P \quad f \leq^T g$ - לכל פונקציה ב $\#P$ יש רדוקציה טיורינג (עם אורקל לחישוב פונק') ל g.

מסקנה:
אם g היא $\#P$-שלמה וגם $g \in FP$ אזי $FP = \#P$.

עובדות:
טענה: $FP \subseteq \#P$
הוכחה: אם $f \in FP$ עם מ"ט פולי' לחישוב פונקציות מתאימה M, (לכל x מחשבת $f(x)$).
נתאר מ"ט א"ד פולי' M' אשר על קלט x תנחש מנספר בעל n^c ביטים, k.
תחשב ע"י M את $f(x)$ ותקבל אם ורק אם $1 \leq k \leq f(x)$.
לכן ל M' יש בדיוק $f(x)$ מסלולים שבהם היא מקבלת ולכן $f \in \#P$.

עובדה 1:
$\#SAT$ היא בעיה $\#P$-שלמה.
"הוכחה": ראינו כבר ש $\#SAT \in \#P$. נשאר להראות ש $\forall f \in \#P \quad f \leq^T \#SAT$.
שוב וריאציה על משפט $cook$ (הרצאה 12 בחישוביות).
בהינתן מ"ט א"' פולינומית M עבור f וקלט x, נתאים להם פסוק CNF $\varphi_{M,x}$ כך שיש התאמה אחד לאחד בין המסלולים המקבלים של M על x לבין ההשמות המספקות ל $\varphi_{M,x}$.

<u>עובדה 2:</u>
קיים אלגוריתם פולינומי לספירת מספר העצים הפורשים בגרף.
הרעיון: מייצגים את הגרף ע"י מטריצה נכונה כך שהדטרמיננטה של המטריצה היא מספר העצים הפורשים בגרף.

<u>עובדה 3:</u>
חישוב מספר השידוכים המושלמים בגרף הוא בעיה $\#P$-שלמה.

$\#P \subseteq FPSPACE$ כאשר $FPSACE$ היא אוסף הפונקציות שניתנות לחישוב באמצעות מ"ט בעלת סיבוכיות מקום פולינומית.
למה? בהינתן מ"ט א"ד M המתאימה לפונקציה $f \in \#P$, פשוט נעבור על כל המסלולים המקבלים, ונספור כמה כאלו יש.

תזכורת: $\Delta_2 = P^{SAT}$
<u>משפט Toda(89)</u>: $PH \subseteq P^{\#P}$ (ידוע גם $P^{\#P} \subseteq PSPACE$)

<u>בעיית הפרמננט Permanent:</u>
A - מטריצה בגודל $n \times n$.
$DET(A) = \sum_{\sigma \in S_n} \underbrace{sign(\sigma)}_{\in\{-1,1\}} \prod_{i=1}^{n} a_{i,\sigma(i)}$ - דטרמיננטה של מטריצה - ניתנת לחישוב יעיל.

$$PER(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i,\sigma(i)}$$

<u>משפט Valiant 77:</u> $\{0,1\} - PER$ היא $\#P$-שלמה.
נראה ש $\{0,1\} - PER \in \#P$ -על קלט מטריצה A, ננחש פרמוטציה $\sigma \in S_n$ ועבורה נחשב את המכפלה $\prod_{i=1}^{n} a_{i,\sigma(i)}$ והיא תקבל רק אם יצא לה 1.
מספר המסלולים המקבלים שלה יהיה בדיוק $PER(A)$.

$PER(A) \bmod 2 = DET(A) \bmod 2$.
לכל p אחר, $PER(A) \bmod p$ היא בעיה קשה.

אינטרפרטציה קומבינטורית של פרמננט:

גרפים	**אלגברה**
גרף דו צדדי עם n צמתים בכל צד	מטריצת $\{0,1\}$ בגודל $n \times n$
שידוך מלא בגרף הנ"ל	$\sigma: \prod_{i=1}^{n} a_{i,\sigma(i)} = 1$
מספר השידוכים המלאים בגרף	$PER(A)$

נראה כי $\#SAT \le^T ... \{0,1\}-PER$ ולכן $\{0,1\}-PER$ היא $\#P$-שלמה.

<u>מבנה ההוכחה:</u>

$$\#SAT \le \#3SAT \le \{-1,0,1,2,3\}-PER \le^T UNARY-PER \le \{0,1\}-PER$$

$\{-1,0,1,2,3\}-PER \le^T UNARY-PER$:

הבעיה היא איך להיפטר מה- -1. (עבור $UNARY-PER$, הקלט אונארי והפלט בינארי).

נשתמש במשפט הצפיפות של מספרים ראשוניים: בקטע $[a,b]$ יש בערך $\frac{b}{lan(b)}-\frac{a}{lan(a)}$ מספרים ראשוניים.

נשתמש במשפט השאריות הסיני:

אם $P_1,...,P_k$ הם מספרים ראשוניים שונים זה מזה ונתונות משוואות:

$x \bmod P_1 = a_1$,

$x \bmod P_2 = a_2$,

...

$x \bmod P_k = a_k$.

אזי קיים פתרון יחיד x בתחום $\left[-\frac{1}{2}\prod_{i=1}^{k}P_i, \frac{1}{2}\prod_{i=1}^{k}P_i\right)$ וניתן למצוא אותו ביעילות.

הרעיון: מתחילים עם A שיש לה ערכים בתחום $\{-1,0,1,2,3\}$.

$-n!\cdot 3^n < PER(A) \le n!\cdot 3^n$ - המספר לא גדול מדי - ניתן לרשום אותו ב $n\log n$ ביטים.

הוא כן בעייתי כאשר מדברים על עולם אונארי, שבו לא ניתן לכתוב מספר כזה בזמן פולינומי.

נבחר $k=2n$.

נבחר k ראשוניים בתחום $\left[3n, 10n^2\right]$ (משפט הצפיפות מבטיח שיש מספיק ראשוניים בתחום הנ"ל).

לכל ראשוני P_i עבור $1 \le i \le k$ נחשב את $A_i \triangleq A \bmod p_i$ באונארי. נקבל מהאורקל את $a_i = PER(A_i) \bmod p_i$.

נחשב את $PER(A)$ מתוך $a_1,...,a_k$ ע"פ משפט השאריות הסיני.

$\prod P_i \ge (3n)^{2n} \ge 2\cdot 3^n \cdot n!$ - זה מבטיח שהפתרון שנקבל במשפט הוא הפתרון המבוקש.

זה שקול לגרף מכוון, עם משקלות, n צמתים, אפשר עם לולאות עצמיות.

זה שקול למטריצה $n \times n$ עם אותם ערכים.

כיסוי של צמתי הגרף ע"י מעגלים שקול לפרמוטציה σ כך ש $\prod_{i=1}^{n} a_{i,\sigma(i)} \ne 0$.

משקל של כיסוי הוא מכפלת המשקלות המשתתפות בכיסוי. הוא שקול ל $\prod_{i=1}^{n} a_{i,\sigma(i)}$.

סכום משקלי הכיסויים $\Leftrightarrow PER(A)$.

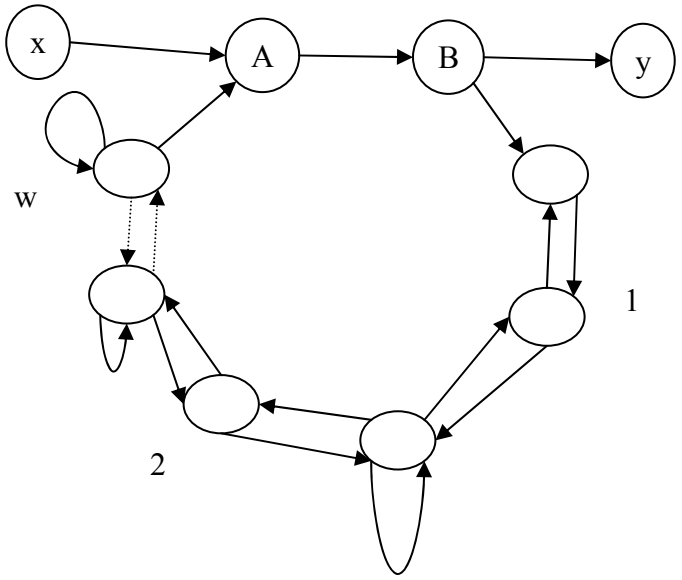
בהינתן מטריצה עם כניסות אונריות, נראה כיצד "להיפטר" מכל כניסה שערכה גדול מ 1.
למעשה, נתבונן בגרף המתאים G.
נבחר קשת $e = x \to y$ שמשקלה $w > 1$ ונראה איך ניתן להחליפה בתת גרף Ge כך שהגרף המתקבל G' יהיה בעל אותו סכום משקלי כיסויים, והוא יגדל בפקטור של $O(w)$.

כיסוי של G' מתאים לכיסוי של G אם מחוץ ל Ge הם משתמשים בדיוק באותן קשתות.
Ge מקיים את הדברים הבאים:

- גודלו $O(w)$.
- לכל כיסוי של G שאינו משתמש ב e יש כיסוי יחיד של G' שמתאים לו.
- לכל כיסוי של G שכן מכיל את e יש בדיוק w כיסויים של G' שמתאימים לו.

לכן נקבל שסכום משקלי הכיסויים ב G, G' זהה.

Ge:



מספר הצמתים הוא בערך $2w$.
מספר הקשתות הוא בערך $5w$.

x, y מכוסים ע"י מעגלים אחרים. לכן חייבים להשתמש בקשת $A \to B$ ולהמשיך מסביב למעגל.
נשתמש בדיוק בלולאה עצמית אחת.
כל כיסוי למעגלים חייב להשתמש בדיוק בלולאה עצמית אחת, ולכן יש w דרכים לעשות זאת.

ולסיום הקורס, משהו שמצאתי באינטרנט:

שְׁלִילַת מְגַלֶּה הַלּוּלָאוֹת הַמְהֻלָּל

אֵין תָּכְנִית שֶׁתֵּדַע מָה אֲחֶרֶת עוֹשָׂה.
זוֹ עֲבָדָה מוּצָקָה, וְלֹא סְתָם מְצוּצָה:
תּוּכַל עַד מָחָר אֶת הַמּח לְשָׁבֵר-
לֹא תּוּכַל לְנַבֵּא אִם תָּכְנִית תַּעֲצֹר.

נַנִּיחַ שֶׁ P הִיא שִׁיטָה שֶׁכָּזֹאת
שֶׁלְּתוֹךְ כָּל תָּכְנִית מְצִיצָה, לְגַלּוֹת
שֶׁאֵין שׁוּם לוּלָאָה אֵינְסוֹפִית מִסְתַּחֶרֶרֶת;
וְאִם אֵין שָׁם כְּלוּם- אָז 'טוֹב!' הִיא אוֹמֶרֶת.

מְזִינִים אֶת הַקוֹד וְאֶת כָּל הַנְּתוּנִים,
וְ P אָז תַּחֲקֹר בַּפְּרָטִים הַקְּטַנִּים
וּתְחַשְׁבֵן אִם הַכֹּל מִסְתַּדֵּר כָּרָאוּי
(בְּנִגּוּד לְמַצָּב לוּלְאִי לֹא רָצוּי.)

הָאֱמֶת הִיא שֶׁ P כָּזוֹ לֹא תִּתָּכֵן,
כִּי אִם תִּכָּתֵב ,P וְלִי תִּנָּתֵן,
אֶשְׁתַּמֵּשׁ בָּהּ לִצֹּר כָּשֶׁל לוֹגִי מֻצְלָח
שֶׁיְּשַׁבֵּר הֶגְיוֹנָךְ וְחוּשֶׁיךָ יְמָעֵךְ.

הַתַּכְסִיס הוּא פָּשׁוּט וְיוֹצֵא מִן הַכְּלָל.
אַגְדִּיר עוֹד תָּכְנִית, בְּשֵׁם ,Q לְמָשָׁל,
שֶׁתְּקַח כָּל תָּכְנִית, וּלְ P אָז תְּקָרֵא,
שֶׁתְּקַבֵּעַ אִם יֵשׁ בָּהּ לוּלָאָה מַמְאִירָה;

אִם יֵשׁ, אָז Q תַּדְפִּיס 'אוּף!' וְתִפָּרֵשׁ;
אַךְ אִם אֵין, אָז Q תַּחֲזֹר לָהּ לָרֹאשׁ,
וְתַתְחִיל מֵחָדָשׁ, תִּסְתּוֹבֵב בְּלִי לַחֲדֹל,
עַד יִגְוַע הַיְקוּם וְיִקָּפֵא וְיִבֹּל.

הַתָּכְנִית הַזּוֹ ,Q, לֹא תֵּשָׁאֵר יְתוֹמָה;
מַמָּזֵר שֶׁכְּמוֹתִי- אַפָּעִילָהּ עַל עַצְמָהּ!
אֵיךְ Q תִּתְנַהֵג בְּמַצָּב שֶׁכָּזֶה?
כְּשֶׁתִּקָּרֵא אֶת עַצְמָהּ- מָה בְּדִיּוּק תַּעֲשֶׂה?

אִם P תִּגָּלֶה לוּלָאָה Q -תֵּצֵא;
אַךְ P אֲמוּרָה לְרַחֵם עַל זֶה.
כָּךְ שֶׁאִם Q תֵּצֵא- אָז P תֹּאמַר 'טוֹב'!
וְ Qתֵּאָלֵץ לְהַתְחִיל שׁוּב לָסֹב!

מָה שֶׁ P לֹא תַּגִּיד Q ,יָשָׁר מְעַקֶּמֶת;
Q גּוֹרֶמֶת לְ Pלָצֵאת דֵּי מְטֻמְטֶמֶת.
כִּי אִם P צוֹדֶקֶת- יוֹצֵא שֶׁשִּׁקְרָהּ;
וְאִם מְשַׁקֶּרֶת- אֱמֶת הִיא דִּבְּרָה!

כָּזֶה פְּרָדוֹקְס אֶלֶגַנְטִי יָצָא,
פָּשׁוּט בִּגְלַל ,P הַהֶלִיךְ הַמַּמְצָא.
אִם תַּנִּיחַ שֶׁ P אֲמִתִּי- הִסְתַּבַּכְתָּ;
בְּפַח הַיּוֹקְשִׁים שֶׁטָּמַנְתִּי- נִלְכַּדְתָּ!

אָז אֵיךְ נַחֲלֵץ מִצָּרָה כֹּה סְבוּכָה?
לֹא צָרִיךְ שֶׁאַגִּיד; תְּנַחֵשׁ לְבַדְּךָ.
מַסְקָנָה הֲכָרַחִית, שֶׁבָּזֶה הָעוֹלָם,
יְצוּר אֲנָדִי כְּמוֹ -P לֹא קַיָּם.

לֹא תַּצְלִיחַ לִבְנוֹת מִין מִתְקָן שֶׁכָּזֶה
שֶׁיּוּכַל לְנַבֵּא מָה מַחְשֵׁב יַעֲשֶׂה.
זֶה בִּלְתִּי אֶפְשָׁרִי. וְלָכֵן אֲנָשִׁים
מוֹצְאִים בָּאגִים לָבַד; מַחְשֵׁבִים הֵם טִפְּשִׁים!!

<u>ספירת קונפיגורציות</u>

תהי M מ"ט דטרמיניסטית בעלת d סרטי עבודה. א"ב בגודל a, ו k מצבי בקרה.

<u>טענה:</u>

אם M רצה בזיכרון $s(n)$ אזי קיים קבוע c כך שמספר הקונפיגורציות (עם קלט באורך n) חסום ע"י $(n+2)\cdot c^{s(n)}$.

<u>הוכחה:</u>

הראש של סרט הקלט יכול להיות על $(n+2)$ מקומות (כל אחת מאותיות הקלט וה-$ שמימין וה- $ שמשמאל).

הראשים של סרטי העבודה יכולים להיות ב $s(n)$ מקומות.

תוכן כל תא יכול להיות אחת מ a האותיות, ומספר התאים יכול להיות $s(n)$.

מספר מצבי הבקרה יכול להיות k ומספר סרטי הבקרה הוא d.

לכן: $\#con \le (n+2)\left[a^{s(n)}\cdot s(n)\right]^d \cdot k \le (n+2)\cdot c^{s(n)}$

אם בנוסף $s(n)\ge \log n$ אזי קיים קבוע c_1 כך שמספר הקונפיגורציות חסום ע"י $c_1^{s(n)}$:

$(n+2)c^{s(n)} \le 2\cdot 2^{\log n} c^{s(n)} \le 2\cdot 2^{s(n)} \le (4c)^{s(n)}$.

<u>מסקנה 1:</u> אם מ"ט דטרמיניסטית משתמשת בזיכרון $s(n)\ge \log n$, רצה על קלט x באורך n יותר מ $c_1^{s(n)}$ צעדים, אזי M לא עוצרת על x ולכן $x\notin L(M)$ (קונפיגורציה כלשהי חזרה על עצמה).

<u>מסקנה 2:</u> אם מ"ט M מקבלת את L ועובדת בזיכרון $s(n)\ge \log n$, אזי קיימת מ"ט M' העובדת בזיכרון $s(n)$ ומכריעה את L (כלומר עוצרת לכל קלט).

<u>הוכחה:</u> נניח כי $s(n)$ הינה פונקצית זיכרון (ההנחה אינה נחוצה אך מפשטת). ע"פ מסקנה 1, על כל קלט $x\in L$, M רצה לכל היותר $c_1^{s(n)}$ צעדים.

נתאר פעולת M' על קלט x באורך n:

1. מחשבת את $s(n)$ על סרט נוסף. סיבוכיות זיכרון $O(s(n))$.
2. מבצעת סימולציה של M על x תוך ספירת צעדי הסימולציה בעזרת מונה על בסיס c_1.
3. אם מספר צעדי הסימולציה עובר את $c_1^{s(n)}$, כלומר המונה גדול מ $s(n)$ אז M' עוצרת ודוחה.

סיבוכיות הזיכרון: $O(s(n))$.

נכונות (הסבר שלי, לא הופיע בתרגול):

1. אם $x\in L$ אז M מקבלת את x תוך לכל היותר $c_1^{s(n)}$ צעדים, ולכן M' תקבל את x.
2. אם $x\notin L$ אז:
   a. אם M דוחה את x תוך לכל היותר $c_1^{s(n)}$ צעדים, אז M' תדחה את x על פי הסימולציה של M.
   b. אם M לא עוצרת על x אז בפרט היא לא עוצרת תוך $c_1^{s(n)}$ צעדים, אז M' תדחה את x.

<u>מסקנה 3:</u> $DSPACE\left(s\left(n\right)\right)$ סגורה למשלים עבור $s\left(n\right)\geq\log n$.

הוכחה (הסבר שלי, לא הופיע בתרגול): בהינתן שפה $L\in DSPACE\left(s\left(n\right)\right)$ עבור $s\left(n\right)\geq\log n$, קיימת לה מכונה M המקבלת את L תוך שימוש בלכל היותר $s\left(n\right)$ זיכרון. ע"פ מסקנה 2 קיימת M' **המכריעה** את L תוך שימוש ב $s\left(n\right)$ זיכרון. לכן נבנה את M'' המכריעה את $\overline{L}$ תוך שימוש ב $s\left(n\right)$ זיכרון, ע"י הרצת M' והחזרת תשובה הפוכה ממנה. לכן $\overline{L}\in DSPACE\left(s\left(n\right)\right)$.

<u>מסקנה 4:</u> אם $L\in DSPACE\left(s\left(n\right)\right)$ כאשר $s\left(n\right)\geq\log n$ אזי קיים קבוע c התלוי ב L כך ש $L\in DTIME\left(c^{s(n)}\right)$.

הוכחה (הסבר שלי, לא הופיע בתרגול): בהינתן שפה $L\in DSPACE\left(s\left(n\right)\right)$, קיימת M כך ש $L\left(M\right)=L$ ו M מקבלת את L תוך שימוש בלכל היותר $s\left(n\right)$ זיכרון. לכן מספר הקונפיגורציות שלה חסום ע"י $c^{s(n)}$, ולכן ניתן לבנות מ"ט M' שבהינתן x מריצה את M על x לכל היותר $c^{s(n)}$ צעדים. אם M עצרה אז M' עוצרת ועונה כמוה, ואחרת M' דוחה.
הזמן הדרוש ל M' הוא לכל היותר $c^{s(n)}$.
הנכונות כמו במסקנה 2.

ומכאן: $DSPACE\left(s\left(n\right)\right)\subseteq\bigcup_{c>0}DTIME\left(c^{s(n)}\right)=DTIME\left(c^{O(s(n))}\right)$.

<u>עקרון הניפוח כלפי מעלה</u>

<u>הגדרות (שלי, לא הופיעו בתרגול):</u>

$DL = \{L \mid$ זיכרון $\log|x|$ ב משתמשת לכל היותר M , ולכל קלט x כך ש $L(M) = L$ M **דטר'** {קיימת מ"ט

$NL = \{L \mid$ זיכרון $\log|x|$ ב משתמשת לכל היותר M , ולכל קלט x כך ש $L(M) = L$ M **א"ד** {קיימת מ"ט

במילים אחרות: $DL = DSPACE(\log(n))$, $NL = NSPACE(\log(n))$.

<u>דוגמה:</u> נראה שאם $DL = NL$ $(=L)$ אזי גם $NSPACE(n) = DSPACE(n)$.

מספיק להראות שאם $NL \subseteq DL$ אזי $NSPACE(n) \subseteq DSPACE(n)$.

(זאת מכיוון ש $NL \supseteq DL$ וכן $NSPACE(n) \supseteq DSPACE(n)$)

$$class_1 = NSPACE(n) \qquad class_2 = DSPACE(n)$$
$$f_1(n) = \log n \qquad f_2(n) = \log n$$
$$g(n) = 2^n \qquad \log 2^n \le n$$

תיאור סכמאטי של ההוכחה:

$L \in CLASS_1(f_1(g(n)))$ $\qquad CLASS_1 = NSPACE(n)$

שלב א': ניפוח. נראה $L' \in CLASS_1(f_1(n))$. $\qquad (NL)$

שלב ב': הנחה: $L' \in CLASS_2(f_2(n))$ $\qquad (DL)$

שלב ג': "הוצאת אוויר": $L \in CLASS_2(f_2(g(n)))$ $\qquad CLASS_2 = DSPACE$

תהי $L \in NSPACE(n)$ ותהי M_1 מ"ט א"ד שמקבלת את L בזיכרון $O(n)$.

שלב א': ניפוח. ננפח את L לשפה $L' = \left\{ x\$^{2^{|x|}} \mid x \in L \right\}$: L' . כלומר כל ה x -ים השייכים לשפה המקורית, L , ואחריהם משורש $\$$ -ים, במספר השווה ל $g(|x|)$.

נראה ש $L' \in NL$.

נתאר מ"ט א"ד M_1' המקבלת את L' בזיכרון $O(\log(n))$.

1. מוודאת שהקלט הוא מהצורה $x\$^i$. $O(1)$ זיכרון.

2. מוודאת ש $i = 2^{|x|}$ בעזרת מונה בינארי של מספר סימני ה $\$$. $O(\log(n))$ זיכרון - צריך מונה לגודל הקלט.

3. מריצה את M_1 על x תוך התעלמות מסימני $\$$ ומקבלת/דוחה כמוה. סיבוכיות הזיכרון היא O של האורך של x , שהוא בערך $\log$ של האורך של הקלט (כי אחרי ה x יש $2^{|x|}$ $\$$-ים). כלומר סה"כ $O(\log(n))$ זיכרון.

שלב ב': ע"פ ההנחה, קיימת מ"ט דטרמיניסטית M_2' שמקבלת את L' תוך שימוש ב $O(\log n)$ זיכרון.

שלב ג': "הוצאת האוויר". נשתמש ב M_2' כדי לתאר מ"ט דטרמיניסטית M_2 שמקבלת את L בזיכרון $O(n)$.

היינו רוצים להוסיף לקלט $2^{|x|}$ סימני $ ולהריץ את M_2' על המילה $x\$^{2^{|x|}}$ אולם הוספה כזו תחרוג מהזיכרון הנתון לנו שהוא $O(|x|)$.

לכן, נחזיק ראש (קורא) וירטואלי שיכיל מונה שערכו 0 אם הראש של M_2 מצביע לתוך x וערכו i אם הראש הקורא מצביע על ה "$ ה-i".

מכיוון שהערך המקסימאלי של מונה כנ"ל הוא $2^{|x|}$, החזקתו דורשת $O\left(\log\left(2^{(|x|)}\right)\right)=O(|x|)=O(n)$ זיכרון בסך הכל ולכן M_2 יכולה לקבל את L תוך שימוש ב $O(n)$ זיכרון (הסימולציה של M_2' תדרוש $O\left(\log\left(2^{|x|}\right)\right)=O(|x|)=O(n)$ זיכרון).

לכן $L \in DSPACE(n)$.

PSPACE ומשחקים

חזרה: המחלקה PSPACE

פסוק *QBF* - *Quantifaied boolean formula* $\psi = Q_1 x_1 Q_2 x_2 ... Q_n x_n \varphi(x_1,...,x_n)$ כאשר φ הוא פסוק *CNF* במשתנים $x_1,...,x_n$ ובקבועים 0,1 ו $\forall i, Q_i \in \{\forall, \exists\}$.

<u>הגדרת ערך אמת:</u>

באינדוקציה על n:

עבור $n=0$: פסוק ללא כמתים, לפי הלוגיקה של $\neg, \wedge, \vee$.

עבור $n>0$: $\psi = Qx\psi'$ עבור ψ' פסוק *QBF* שבוא x משתנה חופשי.

ערך האמת מחושב ע"פ ערך האמת של נוסחאות *QBF*: $\psi'|_{x=0}$ ו $\psi'|_{x=1}$ ומהות הכמת.

(כלומר אם $Q = \forall$ אז $\psi = \psi'|_{x=1} \wedge \psi'|_{x=0}$ ואחרת $\psi = \psi'|_{x=1} \vee \psi'|_{x=0}$).

$TQBF = \{\psi \mid \text{TRUE}$ שערך האמת שלו הוא QBF הוא פסוק $\psi\}$

דוגמה: $\exists x \forall y (\bar{x} \vee x) \wedge (y \vee \bar{y}) \in TQBF$

TQBF היא שפה PSPACE שלמה.

מבוא - משחקים

בהמשך הדיון נעסוק בקבוצה מצומצמת של משחקים בעלי שתי התכונות הבאות:

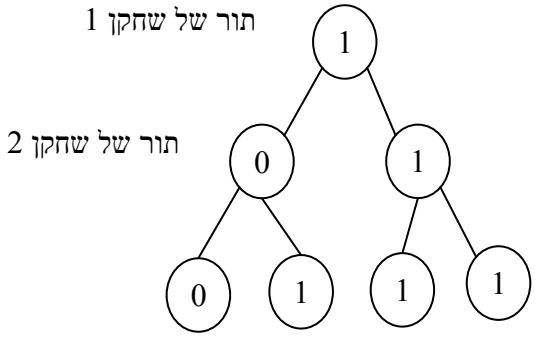
- אין תוצאת תיקו.
- מספר המהלכים האפשריים בכל תור ואורך המשחק חסומים פולינומית בגודל הייצוג של ה"לוח" לכן האסטרטגיות של השחקנים תהיינה בד"כ בגודל אקס'.

משפט פון נוימן: בכל משחק המקיים את שני התנאים שלעיל, לאחד משני השחקנים (ובדיוק לאחד) יש אסטרטגית ניצחון.

הוכחה: באינדוקציה על גובה עץ המשחק.
נתאר להלן את עקרון ההוכחה.
נקבע ערך 1 מעיד על ניצחון של שחקן 1 וערך 0 מעיד על ניצחון שחקן 2.
נסמן את כל עלי העץ ב 0,1 בהתאם למנצח.
נסמן כל צומת המתאר מצב בו תורו של שחקן 1 לשחק במקסימום מבין ערכי בניו וצומת המתאר מצב בו תורו של שחקן 2 לשחק במינימום מבין ערכי בניו.

ערך השורש (יקרא גם ערך ה MINIMAX של השורש) יעיד על השחקן שלו אסטרטגית ניצחון.

דוגמה:



דרך נוספת: ל- 1 אין אסטרטגית ניצחון = $\neg(\exists\forall\exists...\forall(1wins)) \equiv \forall\exists\forall...\exists\neg(1wins)$

וזה שקול לכך שלשחקן 2 יש אסטרטגית ניצחון.

בעיית ההכרעה:
בהינתן משחק, יש לענות על השאלה: האם יש לשחקן 1 אסטרטגית ניצחון:
$1win = \{G \mid$ משחק כך שבו ל 1 יש אסטרטגית ניצחון $G\}$
כדי להימנע מהצורך לציין בקלט את כל פרטי המשחק, מגדירים בנוסף משפחה של בעיות הכרעה דומות, כאשר כל בעיה מוגדרת מעל קבוצה אינסופית של משחקים אשר ידועה מראש. במקרה כזה הקלט יכלול רק את פרטי המשחק הספציפי (למשל, גודל הלוח, והמצב ההתחלתי), ולכן ניתן לקדדו באופן קצר יחסית.
נזכיר שסיבוכיות הבעיה נמדדת יחסית לגודל ייצוג הקלט לבעיה.

אם עבור בעיית הכרעה כזאת כל המשחקים מקיימים את התנאים שהצגנו, הבעיה שייכת ל PSPACE.
הסבר: כיוון שעומק העץ פולינומי וגודל המידע הנשמר לכל צומת הינו פולינומי, ניתן להשתמש ב DFS כדי לחשב את ערך ה MINIMAX של השורש ולהכריע את הבעיה תוך שימוש בזיכרון פולינומי.

ישנן קבוצות משחקים עבורן בעיית ההכרעה הינה PSPACE שלמה. תחת ההנחה ש $NP \subset PSPACE$ (הכלה ממש), עבור משחקים אלו, לא קיימת הוכחה "קצרה" שלשחקן מסוים יש אסטרטגית ניצחון.

דוגמאות:
דמקה - אם מכלילים ללוח $n \times n$, מגבילים את אורך המשחק לפולינום ב n ומונעים תיקו, הבעיה היא $PSPACE$ שלמה.

ניתן להתייחס ל $TQBF$ כמשחק בין שני שחקנים שייקראו $\exists, \forall$, כאשר $\exists$ מציב ערכים למשתמשים "שלו" ו $\forall$ מציב ערכים למשתנים שלו. מטרת $\exists$ היא שהנוסחה תקבע ערך אמת T תחת ההשמה המתקבלת, ומטרת $\forall$ שהנוסחה תקבל ערך אמת F.

פסוק ה QBF נמצא ב $TQBF$ אם במשחק המתאים יש ל $\exists$ אסטרטגית ניצחון.

משחק גיאוגרפיה מוכלל:
"הלוח" הוא גרף מכוון עם צומת התחלה ידוע מראש (מסומן). כל שחקן לפי תורו, מסמן את אחד השכנים של הצומת האחרון שסומן, כאשר אסור לסמן צומת שכבר סומן. מי ש"נתקע" - מפסיד.
$GG = \{(G,s) \mid$ במשחק ג"ג מוכלל עם גרף G עם צומת התחלה s, לשחקן 1 יש אסטרטגית ניצחון$\}$

<u>הוכחת PSPACE שלמות ל GG</u>
<u>טענה:</u> GG היא PSPACE שלמה.
<u>הוכחה:</u> $GG \in PSPACE$:
כל משחק מסתיים לאחר לכל היותר $|V|$ מהלכים.
בהינתן "מצב הלוח", ניתן למצוא בזמן פולינומי את כל המהלכים הבאים האפשריים, ואת מצב הלוח הנובע מהם.
נובע ש GG הוא משחק המקיים את התנאים שהצגנו ולכן $GG \in PSPACE$.

כדי להוכיח קושי, נראה רדוקציה מ $TQBF$. בהינתן נוסחת QBF, נתרגם אותה תחילה לצורה:
$\psi = \exists x_1 \forall x_2 \exists x_3 \forall x_4 ... \exists x_n (c_1 \wedge c_2 \wedge ... \wedge c_n)$.
כלומר, נדאג שהכמתים יתחלפו כנדרש ע"י הוספת משתני סרק (לכל היותר מספר משתני הסרק יהיה זהה למספר הכמתים המקורי).
ע"פ ψ נבנה גרף וצומת התחלה (ראו דוגמה בדף העזר). ניתן לראות כי הרדוקציה פולינומית.

באופן פורמאלי:

אם $\psi \in TQBF$ אזי לשחקן $\exists$ יש אפשרות להבטיח שההשמה המושרית ע"י $3n$ המהלכים הראשונים תספק את נוסחת ה CNF, ולפיכך כל בחירה של פסוקית של $\forall$ בצד ה $3n+1$ תבטיח שחקן $\exists$ יוכל להתקדם לצומת שמתאים לליטרל המסתפק ע"י ההשמה ואז $\forall$ "יתקע".
אם $\psi \notin TQBF$ אזי הנוסחה $\neg\psi$ מקבלת ערך אמת T ומכאן שלשחקן $\forall$ יש אפשרות להבטיח שההשמה המתקבלת מ $3n$ המהלכים הראשונים לא תספק את הנוסחה $c_1 \wedge ... \wedge c_n$, ולכן בצעד ה $3n+1$, שחק $\forall$ יוכל לבחור פסוקית שאינה מסתפקת ע"י ההשמה. כעת שחקן $\exists$ יבחר ליטרל אשר בהכרח לא מסתפק ע"י ההשמה.
לכן $\forall$ יוכל לבצע צעד נוסף, ולכן $\exists$ "יתקע".

משפטי היררכיה וחישוב ב log-space

משפט היררכיה ל DTIME:

תהיינה $T_1(n), T_2(n) \geq n$, כך ש- $T_2(n)$ פונקציות שעון וכן מתקיים: $\lim_{n\to\infty} \frac{T_1(n)\cdot\log(T_1(n))}{T_2(n)} = 0$.

אזי: $\mathrm{DTIME}(T_1(n)) \subset \mathrm{DTIME}(T_2(n))$ (הכלה ממש).

דוגמה לשימוש במשפט ההיררכיה ל DTIME: נוכיח כי $DTIME\left(2^n\right) \subset DTIME\left(n2^n\right)$.

$T_1(n) = 2^n$, $T_2(n) = n2^n$.

אי אפשר להשתמש במשפט כמו שהוא, כי הגבול הנ"ל הוא לא אפס:

$$\lim_{n\to\infty} \frac{2^n \cdot \log\left(2^n\right)}{n2^n} = \lim_{n\to\infty} = \frac{n2^n}{n2^n} = 1$$

הוכחה: ברור שמתקיים $DTIME\left(2^n\right) \subseteq DTIME\left(n2^n\right)$,

לכן צ"ל: $DTIME\left(2^n\right) \neq DTIME\left(n2^n\right)$.

נניח בשלילה $DTIME\left(2^n\right) \supseteq DTIME\left(n2^n\right)$.

נשתמש בעיקרון הניפוח כלפי מעלה, עם הפונקציה $n + \log n$.

נקבל:

$$DTIME\left(2^{n+\log n}\right) = DTIME\left(n2^n\right) \supseteq DTIME\left(\left(n+\log n\right)2^{n+\log n}\right) = DTIME\left(n^2 \cdot 2^n\right).$$

נקבל: $DTIME\left(2^n\right) \supseteq DTIME\left(n^2 \cdot 2^n\right)$

וזאת בסתירה למשפט ההיררכיה, מכיוון ש: $\lim_{n\to\infty} \frac{2^n \cdot n}{2^n \cdot n^2} = 0$.

טענה: $FSPACE\left(\log n\right)$ סגורה תחת הרכבה.

כלומר, נתונות שתי פונקציות שניתן לחשב בזיכרון לוגריתמי, וצ"ל להראות שאפשר לחשב את ההרכבה שלהן בזיכרון לוגריתמי.

הוכחה: תהינה $f, g \in FSPACE\left(\log n\right)$ ונראה כי גם $h(x) = g\left(f\left(x\right)\right)$ שייכת למחלקה זו.

לכאורה, ניתן לחשב את $f\left(x\right)$, לכתוב את תוצאת החישוב על אחד מסרטי העבודה ואח"כ להריץ את המכונה המחשבת את g כאשר סרט הקלט שלה יהיה הסרט עם הפלט $f\left(x\right)$.

הבעיה: תוצאת החישוב $f\left(x\right)$ עלולה להיות באורך שגדול מ $O\left(\log n\right)$ וזה יגרום לנו לחרוג מסיבוכיות הזיכרון הרצויה.

הערה: אורך הפלט של $f\left(x\right)$ לא יכול לחרוג מעבר לפולינומי כיוון שמספר הקונפיגורציות של המכונה המחשבת את $f\left(x\right)$ הינו פולינומי.

הפתרון: הרעיון הוא לא לכתוב את תוצאת $f\left(x\right)$ על סרט עבודה כלשהו, אלא כל פעם שהמכונה המחשבת את g תזדקק לביט מתוצאת $f\left(x\right)$, לחשב ביט זה מחדש, ולכתוב ביט זה על סרט העבודה.

תיאור מלא של המכונה:

נסמן: M_h - המכונה המחשבת את $h(x)$,
M_g - המכונה המחשבת את $g(x)$,
M_f - המכונה המחשבת את $f(x)$.

1. סרטי העבודה הבאים ישמשו אותנו:
- סרט עליו יישמר מספר התא עליו מצביע הראש הוירטואלי של M_g בסרט הקלט הווירטואלי שלה.
- מונה (יאותחל ל 0 כל פעם) שישמש למציאת התא שמספרו נתון בסרט הקודם.
- סרטי עבודה "רגילים" של M_g ושל M_f.
- סרט שישמש כקלט ל M_g, עליו נרשום את ביט הפלט של M_f.

2. M_h מפעילה את M_g. בכל פעם ש M_g זקוקה לתו מסוים מהקלט (מיקומו נתון ע"י הסרט הראשון), אזי M_f תופעל מחדש ובמקום לכתוב פלט כרגיל, תגדיל את המונה בסרט השני.
כאשר מונה זה יהיה שווה למונה מהסרט הראשון, ייכתב ביט הפלט בסרט המיועד לכך.
M_g תקרא ביט זה.
M_g תכתוב את הפלט שלה על סרט הפלט.

סיבוכיות זיכרון:
אם M_f עוצרת, גודל הפלט שלה בהכרח קטן מ $2^{O(\log n)}$ - כלומר פולינומי, כמספר הקונפיגורציות המקסימאלי שלה. מכאן שגודל המונים בסרט הראשון ובסרט השני הינו $O(\log n)$.
לסרטי העבודה של M_g, M_f סיבוכיות זיכרון $O(\log n)$.
(זהירות! זה ברור לגבי M_f אבל לא לגבי M_g - הקלט שלה איננו x אלא $f(x)$, ולכן הזיכרון שלה הוא לוגריתמי ביחס ל $f(x)$ ולא ביחס ל x. מכיוון שהגודל של $f(x)$ הוא פולינומי ביחס ל x, מתקיים שהזיכרון של M_g הוא לוגריתמי גם ביחס ל x).
הסרט הנוסף מחזיק ביט בודד.

האם $FSPACE(n)$ סגורה להרכבה? לא!
הוכחה: תהי M_f המכונה הבאה: על כל קלט באורך n, היא כותבת 2^n '1'-ים על סרט הפלט.
תהי $h(x) = f(f(x))$. (שימו לב שניתן לחשב את $f(x)$ בזיכרון $O(n)$).
כלומר, הפונקציה h היא סדרה באורך $2^{(2^n)}$ של '1'-ים בדיוק.
נניח בשלילה שקיימת מ"ט M_h הפועלת בזיכרון $O(n)$ ומחשבת את h.
מכאן ש M_h היא בעלת $2^{O(n)}$ קונפיגורציות. כדי לכתוב $2^{(2^n)}$ '1'-ים, על המכונה צריכה לבצע מספר כזה של צעדים לפחות. לכן עבור n גדול דיו, קיימת בהכרח קונפיגורציה שחוזרת פעמים, ולכן M_h לא עוצרת, בסתירה.

<u>תזכורת:</u> NL מוגדרת להיות $NSPACE(\log n)$.
נוכיח כי השפה הבאה שייכת ל NL.
$\overline{2-SAT} = \{\varphi \mid$ פסוק $2-CNF$ שאינו ספיק $\}$
דוגמה לפסוק $2-CNF$: $\varphi = (l_1 \vee l_2) \wedge (l_3 \vee l_4) \wedge \ldots \wedge (l_{k-1} \vee l_k)$, כאשר l_i הוא משתנה כלשהו או שלילתו.

סדרת פסוקיות תקרא **חוקית** אם היא מהצורה: $(l_i \vee l_j) - (\overline{l_j} \vee l_k) - \cdots - (\overline{l_h} \vee l_m) - (\overline{l_m} \vee l_i)$, כלומר, כל פסוקית "מתחילה" בשלילת הליטרל "האחרון" בפסוקית הקודמת, והסדרה "מתחילה" ו"מסתיימת" באותו הליטרל, שייקרא **הבסיס של הסדרה**. לדוגמא, סדרת הפסוקיות הבאה היא חוקית: $(x_1 \vee x_2) - (\overline{x_2} \vee \overline{x_4}) - (x_4 \vee x_1)$.

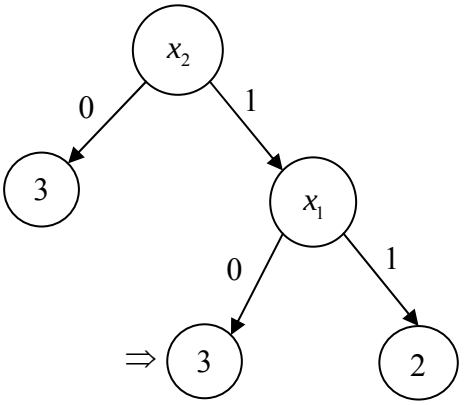
טענת עזר: פסוק 2-CNF הוא אינו ספיק אמ"מ קיימות שתי סדרות חוקיות של פסוקיותיו שהבסיסים שלהן הם ליטרל ושלילתו.
(הוכחה בדף העזר)

<u>הוכחה:</u> מנחשים את המשתנה x_i ואת שתי הסדרות כך שכל פעם, מנחשים רק את הפסוקית הבאה בסדרה (כזו שמכילה את ההיפוך של הליטרל הנוכחי). בכל רגע זוכרים רק את הליטרל הנוכחי ואת הליטרל ההתחלתי. כל ליטרל דורש $O(\log n)$ זיכרון.
אם הצלחנו לנחש שתי סדרות כנדרש - נקבל, אחרת - נדחה.
כאשר הפסוק <u>אינו</u> ספיק, וננחש את המשתנה x_i הנכון והסדרות הנכונות - נקבל (מובטח מטענת העזר, קיום של הנ"ל). אם הפסוק ספיק, כל ניחוש ייכשל, ולכן נדחה.

עצי החלטה

עצי החלטה - הגדרה:
- עץ בינארי.
- כל צומת פנימי בעץ מסומן ע"י משתנה x_i עבור $i \in \{1,...,n\}$ ויוצאות ממנו שתי קשתות. קשת אחת מסומנת ב 0 והקשת השניה מסומנת ב 1.
- כל עלה מסומן ע"י ערך פלט (בד"כ $\{0,1\}$).
- החישוב על קלט $x \in \{0,1\}^n$ מתחיל מהשורש. כאשר מגיעים לצומת המסומן ב x_i ממשיכים על הקשת שמסומנת לפי הערך בקלט של x_i.
- כשמגיעים לעלה - הסימון שלו מציין את הפלט.

דוגמה: על הקלט $x = 01$ הפלט יהיה 3.



<u>מדדי סיבוכיות:</u>

<u>לעץ T:</u>

$D(T)$ - עומק העץ (= המרחק המקסימאלי בקשתות מהשורש לעלה)

$S(T)$ - גודל העץ (= מס' העלים)

<u>לפונקציה f:</u>

$D(f) = \min_T \{D(T) \mid f$ מחשב את $T\}$

$S(f) = \min_T \{S(T) \mid f$ מחשב את $T\}$

<u>תכונות פשוטות:</u>

1. לכל $f\ \{0,1\}^n \to D$ מתקיים: $D(f) \le n$, $S(f) \le 2^n$

(פשוט נבנה עץ מלא ושלם שבתחתיתו רשומים ערכי הפונקציה כאשר כל עלה מתאים לאחת מ 2^n ההשמות, וסימונו $f(x)$).

2. לכל עץ T מתקיים $D(T) \ge \log_2 S(T)$.

דוגמה 1:

$$f\left(x_1,...,x_n\right)=parity\left(x_1,...,x_n\right)=x_1+x_2+...+x_n \bmod 2$$

טענה: $S\left(f\right)=2^n$.

מסקנה: $D\left(f\right)=n$.

הוכחה: נניח בשלילה ש $S\left(n\right)<2^n$ ויהי T העץ המחשב את f עם פחות מ 2^n עלים.

אבחנה: קיים עלה l בעומק קטן ממש מ n.
(לכל צומת יש אפס או שני בנים. אין צומת פנימי עם בן יחיד)
אם כל העלים בעומק לפחות n אזי גודל העץ לפחות 2^n.
נסתכל על עלה זה, l.
במסלול ל l בודקים לכל היותר $n-1$ משתנים. כלומר,יש משתנה x_k אותו לא בודקים.
נקבע השמה x שמגיעה לעלה l, ותהא x' השמה זהה ל x פרט להחלפת ערך ל x_k.
מתקיים: $T\left(x\right)=T\left(x'\right)$ אבל $f\left(x\right)\neq f\left(x'\right)$ ולכן T איננו עץ המחשב את f, בסתירה.

דוגמה 2:

$$f\left(x_{(1,1)},...,x_{(1,k)},...,x_{(k,1)},...,x_{(k,k)}\right)=\wedge_i\left(\vee_j\left(x_{(i,j)}\right)\right)$$

כלומר הקלט של הפונקציה הוא מטריצה ו $f\left(x\right)=1$ אם ורק אם בכל שורה במטריצה יש לפחות '1' אחד.

טענה: $D\left(f\right)=n$

הוכחה: באמצעות **טיעון יריב**.
יהי T עץ עבור f. היריב, בכל צעד מספק את הערך למשתנה ששואלים עליו בעץ. כלומר, בונים באופן הדרגתי את הקלט. הקלט המתקבל יהיה כזה שעליו העץ חייב לשאול על כל n המשתנים.
אסטרטגית היריב: אם העץ שואל על $x_{(i,j)}$ אז אם בשורה ה i נשאלו לכל היותר $k-2$ שונים אז התשובה של היריב היא 0 ואחרת (זה המשתנה האחרון בשורה), התשובה של היריב היא 1.
נשים לב כי כל עוד העץ לא בעל על n משתנים, ניתן לבחור שני המשכים שונים לקלט כך שערך הפונקציה עליהם שונה.

הערה: במקרה זה, בניגוד ל *parity*, לא כל העלים הם בעומק n בהכרח. למעשה: $S\left(f\right)<2^n$.

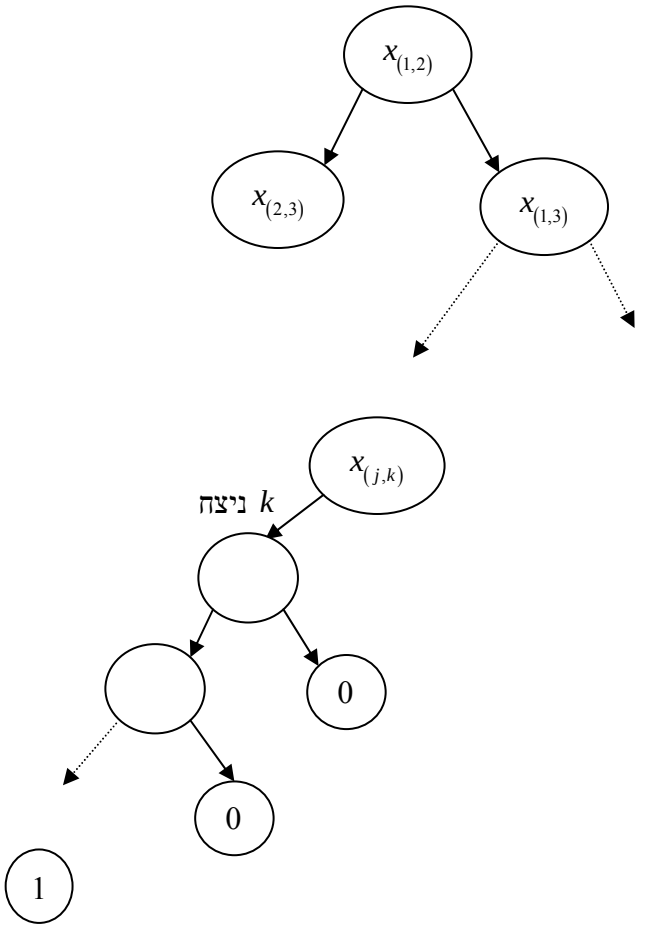
<u>דוגמה 3:</u>
לכל זוג $1 \le i < j \le k$ נגדיר משתנה $x_{(i,j)}$.
נחשוב שיש k שחקנים שמשחקים ביניהם. ערך המשתנה $x_{(i,j)}$ יהיה 1 אם i מנצח את j ו 0 אם j מנצח את i (אין תיקו).
$f\left(x_{(1,2)},\ldots,x_{(k-1,k)}\right)=1$ אם ורק אם קיים שחקן כלשהו המנצח את כל יתר השחקנים.
- הקלט אינו מוגבל, בפרט, לא מקיים טרנזיטיביות.
"המטרה" שלנו היא לחשב את f **מבלי** לשחק $n=\binom{k}{2}$ משחקים.

<u>האלגוריתם:</u>
שלב 1: ניתן ל 1 לשחק נגד 2, המנצח ישחק נגד 3, וכן הלאה, ..., המנצח ישחק נגד k (עד כה יש $k-1$ משחקים).

שלב 2: המנצח בסוף שלב 1 הוא המשתתף היחיד **שלא** הפסיד עדין. ניתן לו לשחק נגד כל מי שעדין לא שיחק נגדו. עומק החלק הזה לכל היותר $k-2$.

שלב 3: אם הוא מנצח את כולם $f=1$ ואחרת $f=0$.

<u>מימוש עץ ההחלטה:</u>



סה"כ: $D(f) \le 2k-3 = O\sqrt{n}$

מכונת טיורינג עם אוב ושאלת P=NP

תזכורת:
הגדרה: $CLASS^A$:
מחלקת השפות המתקבלת ע"י מ"ט מטיפוס $CLASS$ עם אוב ל A, עבור A - שפה נתונה.

למשל: $L \in P^{SAT}$ אם קיימת מ"ט דטר' פולינומית עם אוב ל SAT המקבלת את L.
בפרט: $NP \cup CO-NP \subseteq P^{SAT}$.
הסבר (שלי, לא הופיע בתרגול):
אם $L \in NP$ אז קיימת רדוקציה פולינומית בזמן, f, מ L ל SAT (כי SAT היא NP שלמה), כך ש $x \in L \Leftrightarrow f(x) \in SAT$. הרדוקציה הזאת מחושבת באמצעות מ"ט לחישוב פונקציות M_f הרצה זמן פולינומי ביחס לקלט שלה.
לכן, נבנה מ"ט M_L דטר' פולינומית בעלת אוב ל SAT המקבלת את L:
M_L על קלט x מחשבת באמצעות M_f את $f(x)$, רושמת אותו על סרט השאלות, ואם האוב מקבל, אז מקבלת, ואחרת דוחה.
נכונות: אם $x \in L$ אז $f(x) \in SAT$ ולכן האוב יענה תשובה חיובית, ולכן M_L תקבל את x.
אם $x \notin L$ אז $f(x) \notin SAT$ ולכן האוב יענה תשובה שלילית, ולכן M_L תדחה את x.
סיבוכיות זמן: חישוב $f(x)$ לוקח זמן פולינומי כי f היא פונקצית רדוקציה פולינומית. ההעתקה של $f(x)$ לסרט השאלות לוקחת זמן פולינומי, ובדיקת התשובה לוקחת זמן קבוע. סה"כ פולינומי.
לכן $L \in P^{SAT}$. לכן $NP \subseteq P^{SAT}$.

אם $L \in CO-NP$ אז $\bar{L} \in NP$ ולכן קיימת רדוקציה כנ"ל, $\bar{f}$ מ $\bar{L}$ ל SAT, כך ש $x \in \bar{L} \Leftrightarrow \bar{f}(x) \in SAT$. כלומר $x \in L \Leftrightarrow \bar{f}(x) \notin SAT$.
הרדוקציה הזאת מחושבת באמצעות מ"ט לחישוב פונקציות $M_{\bar{f}}$ הרצה זמן פולינומי.
לכן נבנה מ"ט M_L דטר' פולינומית בעלת אוב ל SAT המקבלת את L:
M_L על קלט x מחשבת באמצעות M_f את $\bar{f}(x)$, רושמת אותו על סרט השאלות, ואם האוב מקבל, אז דוחה ואם הוא דוחה אז מקבלת.
נכונות: אם $x \in L$ אז $\bar{f}(x) \notin SAT$ ולכן האוב ידחה ולכן M_L תקבל.
אם $x \notin L$ אז $\bar{f}(x) \in SAT$ ולכן האוב יקבל ולכן M_L תדחה.
סיבוכיות זמן: כמו קודם.
לכן $L \in P^{SAT}$. לכן $CO-NP \subseteq SAT$.
לכן $NP \cup CO-NP \subseteq P^{SAT}$.

הגדרה: $CLASS_1^{CLASS_2}$: מחלקת השפות המתקבלת ע"י מ"ט מסוג $CLASS_1$ עם אוב לשפה ב $CLASS_2$.

הסבר (שלי, לא הופיע בתרגול):
$L \in CLASS_1^{CLASS_2}$ אם קיימת שפה $L' \in CLASS_2$ כך ש $L \in CLASS_1^{L'}$.

<u>דוגמאות:</u>
$L \in P^{NP}$ אם קיימת מ"ט דטר' פולינומית עם אוב לשפה ב NP, המקבלת את L.

$P^{SAT} = P^{NP}$ כי SAT היא NP-שלמה.
הכיוון $P^{SAT} \subseteq P^{NP}$ נובע מכך ש $SAT \in NP$ ולכן ההכלה היא ע"פ הגדרה.
הכיוון $P^{SAT} \supseteq P^{NP}$ נובע מכך ש SAT היא NP-שלמה.
הסבר (שלי, לא הופיע בתרגול):
אם $L \in P^{NP}$ אז קיימת שפה $L' \in NP$ כך ש $L \in P^{L'}$. לכן קיימת מ"ט M_L, דטר', פולינומית ובעלת אוב ל L' כך ש $L(M_L) = L$.
מכיוון ש $L' \in NP$ ו SAT היא NP-שלמה, נובע שקיימת רדוקציה פולינומית f המחושבת ע"י מ"ט לחישוב פונקציות, M_f, הרצה בזמן פולינומי.
נראה כיצד לבנות מ"ט M_L^{SAT}, דטר', פולינומית, בעלת אוב ל SAT המקבלת את L.
M_L^{SAT} על קלט x:
מריצה את M_L על x, אולם מחליפה את סרט השאלות בסרט אחר. ברגע ש M_L רוצה לשאול את האוב L' על y כלשהו - מחשבת את $f(y)$, רושמת אותו על סרט התשובות, ומחזירה כתשובה ל M_L את התשובה של האוב של SAT על $f(y)$.
נכונות:
ההבדל היחיד בין M_L לבין M_L^{SAT} הוא ש M_L שואלת את האוב ל L' על ערכי y כלשהם, בעוד ש M_L^{SAT} שואלת את SAT על $f(y)$.
מכיוון ש $y \in L' \Leftrightarrow f(y) \in SAT$, הרי ששתיהן תקבלנה בדיוק את אותם קלטים.
לכן $L(M_L^{SAT}) = L$.
סיבוכיות זמן: החישוב של $f(y)$ הוא פולינומי ביחס ל y ו y פולינומי ביחס ל x, ושאר הריצה של M_L הוא פולינומי, ולכן סה"כ זמן הריצה של M_L^{SAT} הוא פולינומי.
לכן $L \in P^{SAT}$ ולכן $P^{SAT} \supseteq P^{NP}$.

ידוע: $P^P = P$ - פשוט מסמלצים את האוב.
הסבר (שלי, לא הופיע בתרגול):
הכיוון $P \subseteq P^P$ ברור. מ"ט דטר' רגילה היא פשוט מקרה פרטי של מ"ט שלא שואלת את האוב כלום.
נראה $P \supseteq P^P$.
אם $L \in P^P$ אז קיימת $L' \in P$ כך ש $L \in P^P$ ו L'. לכן קיימת מ"ט M_L דטר' פולינומית בעלת אוב L' כך ש $L(M_L) = L$.
מכיוון ש $L' \in P$, קיימת מ"ט דטר' פולינומית $M_{L'}$ כך ש $L(M_{L'}) = L'$.
נבנה מ"ט $\tilde{M}_L$ הרצה בזמן פולינומי ומקבלת את L:
M_L על קלט x:
מריצה את M_L על x ובכל פעם שהיא שואלת את האוב שלה על y כלשהו, מריצה את $M_{L'}$ על y וכך מסמלצת את התשובה שלה - אם $M_{L'}$ קיבלה אז רושמת 1 על סרט התשובות ואם היא דחתה אז רושמת 0 על סרט התשובות.
נכונות וסיבוכיות טריוויאליים.

לכן $L \in P$ ולכן $P \supseteq P^P$.

לא ידוע האם $NP^{NP} = NP$ - הבעיה היא שהאוב הוא דטרמיניסטי.
הסבר אינטואיטיבי (שלי, לא הופיע בתרגול): נסתכל על מ"ט א"ד פולינומית M^A עם אוב ל A כאשר $A \in NP$.
M^A על קלט x:
שואלת את האוב על x ועונה הפוך ממנו.
על מנת להראות ש $L(M^A) \in NP$, היינו רוצים לבצע סימולציה של האוב.
מכיוון ש $A \in NP$ קיימת מ"ט א"ד פולינומית M_A כך ש $L(M_A) = A$, כלומר, לכל קלט x מתקיים:
$x \in A \Leftarrow$ קיים ל M_A מסלול חישוב מקבל על x.
$x \notin A \Leftarrow$ כל מסלולי החישוב של M_A על x דוחים.
נסתכל על מכונה שמסמלצת את האוב:
M' על קלט x: מריצה את M_A על x ועונה הפוך ממנה.
הבעיה:
אם $x \in A$ אז האוב עונה ל M^A "כן" ולכן היא דוחה אותו בכל המסלולים. לכן $x \notin L(M^A)$.
אבל, אם $x \in A$, יתכן שקיימים ל M_A מסלולים הדוחים את x ולכן כאשר M' רצה על x, קיימים לה מסלולים שבהם M_A דוחה את x ובאותם מסלולים, M' מקבלת את x.
כלומר, $x \in L(M')$.
לכן $L(M^A) \neq L(M')$ ולכן לא הצלחנו להראות ש $L(M^A) \in NP$.

$PSPACE^{PSPACE} = PSPACE$ - פשוט מסמלצים את האוב - אותו עיקרון כמו ב $P = P^P$.

<u>טענה</u>: קיימות שפות A ו B כך ש $P^A = NP^A$ וגם $P^B \neq NP^B$.

<u>הוכחה:</u>

חלק 1: נבחר $A = TQBF$.

צריך להראות $NP^{TQBF} \subseteq P^{TQBF}$.

$NP^{TQBF} \subseteq NPSPACE$ - זאת מכיוון שהאוב שלנו הוא שפה ב $PSPACE$ ולכן אין בעיה לסמלץ אותו על סרט עבודה נוסף בזיכרון פולינומי.

$NPSPACE \subseteq PSPACE$ - נובע ממשפט $SAVITCH$.

$PSPACE \subseteq P^{TQBF}$ - מבצעים רדוקציה ל $TQBF$ - שואלים את האוב ועונים כמוהו. הרדוקציה לוקחת זמן פולינומי, והאוב עונה בזמן קבוע ולכן סה"כ הזמן פולינומי.

מסקנה: $NP^{TQBF} \subseteq P^{TQBF}$.

חלק 2: נראה B כך ש $P^B \neq NP^B$.

לכל שפה $B \subseteq \{0,1\}^*$ נגדיר: $L_B = \left\{0^n \mid B \cap \{0,1\}^n \neq \phi\right\}$ - כלומר אוסף המילים האונאריות כך שקיימות מילים ב B באותו האורך.

נבחין שלכל B מתקיים: $L_B \in NP^B$: לכל קלט 0^n מנחשים מחרוזת באורך n, שואלים את האוב עליה, ועונים כמוהו.

מטרתנו לבנות שפה B כך ש $L_B \notin P^B$.

<u>הסבר אינטואיטיבי:</u> ההוכחה מתבססת על לכסון, כלומר, נבנה את B כך שהמכונה ה i במנייה כלשהי שמ"ט דטר' פולינומיות תטעה יחסית ל L_B על מילה i בסדר כלשהו.

מהגדרת L_B ברור כי מילים אלו תהינה מהצורה 0^{n_i}.

לכן היינו רוצים לקבוע שייכות של מילים באורך n_i ל B ע"י הרצה של המכונה ה i על 0^{n_i} והחזרת "תשובה הפוכה". (נשים לב להבדל בין B ל L_B).

הבעייתיות היא שהמכונה ה i עשויה לשאול את האוב שאלות כמו "האם המילה 0^{n_i} שייכת ל B?" ולפעול בהתאם. לכן הרצת המכונה <u>עצמה</u> תלויה בשפה הנבנית.

הפתרון לבעיה המעגלית הוא בניה "חכמה" של השפה באופן הדרגתי כך שניתן יהיה להריץ את המכונה ה i (כי כל המידע הנחוץ לסימולציה של האוב כבר קיים) וכך להרחיב את השפה כנדרש מעיקרון הלכסון.

<u>פתרון פורמאלי:</u>

שלב 1: הגבלת זמן הריצה לכל מכונה במניה.

תהא $M_1, M_2, \ldots$ מניה סטנדרטית של מכונות טיורינג דטרמיניסטיות פולינומיות עם אוב, עם השינוי הבא:

למ"ט M_i נצמיד שעון שגורם לה לעצור (ולדחות) אחרי $|x|^i$ כאשר $|x|$ הוא אורך הקלט.

נבחין שלכל שפת אוב O, ולכל $L \in P^O$, קיימת M_i בסדרה כך שכאשר מצמידים לה את האוב O היא מקבלת את L (כלומר $L\left(M_i^O\right) = L$):

לכל מ"ט יש אינסוף מכונות שקולות, מהן, אחת נמצאת מספיק רחוק בסדרה, כך שהשעון לא "יכבה" אותה לפני סיום פעולתה "הטבעית".

שלב 2: הגדרת הסדרה $\{n_i\}_i$.

נגדיר באופן אינדוקטיבי סדרה עולה של מספרים טבעיים:
<u>בסיס:</u> $n_0 = 0$.
<u>צעד:</u> חישוב של n_i מתוך n_{i-1}: קבע את n_i להיות מספר טבעי m המקיים:
א. $m > n_{i-1}^{n_{i-1}}$ - זה יבטיח שלכל $i > j$, M_j על קלט 0^{n_j} לא תשאל את האוב על קלטים מאורך n_i, כי היא רצה רק $n_j^{n_j}$ צעדים. (אין לה מספיק זמן בשביל לרשום את השאלה בסרט השאלות).
ב. $2^m > m^i$ - יבטיח שקיימת מילה באורך m שעליה M_i על קלט 0^{n_i} לא תשאל את האוב.

שלב 3: הגדרת B:
נגדיר באופן אינדוקטיבי סדרה של שפות סופיות $B_0 \subseteq B_1 \subseteq \ldots$ כך שלכל B_i או ששווה ל B_{i-1} או שהתקבל ממנה ע"י הוספת מילה בודדת באורך n_i.
נגדיר $B = \bigcup_{i \geq 0} B_i$.
<u>בסיס:</u> $B_0 = \phi$.
<u>צעד:</u> חישוב B_i מתוך B_{i-1}: סמלץ את ריצת $M_i^{B_{i-1}}$ על הקלט 0^{n_i} (ניתן לעשות זאת כי B_{i-1} ידועה).
אם היא מקבלת: $B_i \leftarrow B_{i-1}$.
אם היא דוחה:
$B_i \leftarrow B_{i-1} \cup \{$המילה הראשונה לפי סדר לקסיקוגראפי באורך n_i ש $M_i^{B_{i-1}}$ לא שאלה עליה$\}$

כעת, נראה שלכל i, M_i^B אינה מסכימה עם L_B על הקלט 0^{n_i}.
<u>טענת עזר:</u> לכל i, M_i^B עונה על קלט 0^{n_i} כמו שעונה $M_i^{B_{i-1}}$.
<u>הוכחה:</u> $M_i^{B_{i-1}}$ לא שואלת את האוב על מילים באורך $\geq n_{i+1}$ (מחסימת הריצה של M_i והגדרת n_j עבור $j > i$). הגדרת n_i מבטיחה שקיימת מילה באורך זה ש $M_i^{B_{i-1}}$ לא שואלת עליה בריצתה על הקלט 0^{n_i}.
הגדרת B, מבטיחה שהמילה שהוספנו (או הוספנו) בשלב ה i היא אכן כזו. הטענה נובעת מכך שבהפרש $B \setminus B_{i-1}$ אין מילים אחרות באורך $< n_{i+1}$.
(נזכיר שאם M_i שואלת על מילים באורך $< n_i$ האוב אכן עונה בשלילה).

מטענת העזר ובניית B_i (שמבטיחה ש L_B לא תסכים עם $M_i^{B_{i-1}}$ על 0^{n_i}) - מש"ל.

ההיררכיה הפולינומית

תזכורת:
ההיררכיה הפולינומית:
רמה 0: $\Delta_0, \Sigma_0, \Pi_0 = P$.

רמה $i+1$:
$$\Delta_{i+1} = P^{\Sigma_i}$$
$$\Sigma_{i+1} = NP^{\Sigma_i}$$
$$\Pi_{i+1} = CO - NP^{\Sigma_i}$$

מחלקות מכומתות

נגדיר כמת "קיים פולינומי" שיסומן ע"י $\exists_p y$ ופירושו: "קיים פולינום P ו y באורך $P(|x|) >$", כאשר x הוא וקטור המשתנים החופשיים בתחום ההשפעה של הכמת. בדומה, מגדירים כמת "לכל פולינומי" שיסומן ע"י $\forall_p y$.

לכל מחלקת שפות C, נגדיר את המחלקה $\exists C$ באופן הבא: $L \in \exists C$ אם קיימת $L' \in C$ ופולינום P כך ש: $L = \{x \mid \exists_p y (x,y) \in L'\}$. באופן דומה מגדירים את המחלקה $\forall C$.

טענה: $NP = \exists P$

הסבר (שלי, לא הופיע בתרגול):
ראינו בקורס חישוביות את ההגדרה הבאה עבור NP:
NP הוא אוסף כל השפות L שקיים עבורן יחס דו מקומי R_L המקיים:
א. R_L חסום פולינומית: $(x,y) \in R_L \Leftarrow |y| \le q(|x|)$ כאשר q הוא פולינום.
ב. R_L ניתן לזיהוי יעיל:
קיימת מ"ט M הרצה בזמן פולינומי $p(|x|+|y|)$ ומכריעה האם $(x,y) \in R_L$.
ג. $L = \{x \mid \exists y (x,y) \in R_L\}$ - כל השמאליים ביחס שיש להם בן זוג.

כלומר R_L הוא L' וע"פ **ב'** הוא שייך ל P.
ע"פ **א'** + **ג'**, נקבל בדיוק את ההגדרה $L = \{x \mid \exists_p y (x,y) \in L'\}$.

לצורך הטענה הבאה נזכיר כי $\text{Co-C} = \{L^C | L \in C\}$ ונסמן: $\forall$ "=" $\exists$, $\neg\exists$ "=" $\forall$, $\neg\forall$ "=" $\exists$.

טענה: לכל $Q_1,\ldots,Q_k \in \{\exists,\forall\}$, מתקיים: $\text{Co-}[Q_1 \ldots Q_k C] = \neg Q_1 \neg Q_2 \ldots \neg Q_k [\text{Co-C}]$.

ההיררכיה הפולינומית – אפיון אלטרנטיבי

הגדרה אלטרנטיבית להיררכיה הפולינומית:

$\Pi_k^p = \text{Co} - \Sigma_k^p = \forall\exists\forall\exists..Q\text{P}$ $\quad\quad$ $\Sigma_k^p = \exists\forall\exists..Q\text{P}$ (כאשר יש k כמתים ו- Q כמת שנקבע בהתאם).

הוכחת האפיון האלטרנטיבי מתבצע באינדוקציה על k. הבסיס הוזכר לעיל. להלן יפורטו עקרונות הוכחת צעד האינדוקציה. ראשית, נשים לב כי מהאפיון האלטרנטיבי נובע:

(הבסיס הוא העובדה ש $\Sigma_0^P = P$ ו $\Sigma_1^P = NP = \exists P$)

- $L \in \Sigma_k^p$ אמ"מ קיימת שפה $A \in \text{P}$ (יחס k+1 מקומי), כך ש:

$L = \{x \mid \exists_{\text{p}} y_1 \; \forall_{\text{p}} y_2 \; \ldots \; Q_{\text{p}} y_k \, (x, y_1, y_2, \ldots y_k) \in A\}$ ($Q=\forall$ אם k זוגי, אחרת $Q=\exists$).

ובדומה,

- $L \in \Pi_k^p$ אמ"מ קיימת שפה $A \in \text{P}$ כך ש:

$L = \{x \mid \forall_{\text{p}} y_1 \; \exists_{\text{p}} y_2 \; \ldots \; Q_{\text{p}} y_k \, (x, y_1, y_2, \ldots y_k) \in A\}$.

כיוון ראשון: נניח שלכל $k-1$ כמתים,
$L \in \exists\forall\exists..\forall\exists QP$ גורר ש $L \in \Sigma_{k-1}^P$,
ו $L \in \forall\exists..\forall\exists QP$ גורר ש $L \in \Pi_{k-1}^P$,
ונראה שזה מתקיים גם עבור k כמתים:

תהי L שפה כך ש:
$L \in \exists\forall\exists..\forall\exists QP$ עם k כמתים.
לכן קיימת שפה $A \in \forall\exists\forall..QP$ כך ש $L = \{x \mid \exists_P y (x, y) \in A\}$.
מכיוון שעבור A יש רק $k-1$ כמתים, לפי הנחת האינדוקציה, $A \in \Pi_{k-1}^P$.

נבנה מ"ט א"ד M^A עם אוב ל A שתקבל את L.
ננחש את y ונשאל את האוב האם $(x, y) \in A$ ונענה כמוהו.
אם $x \in L$ אז ע"פ ההגדרה קיים y פולינומי ביחס ל x כך ש $(x, y) \in A$ ולכן קיים מסלול שבו M^A מנחשת את y ואז האוב עונה "כן" והיא מקבלת. לכן $x \in L(M^A)$.
אם $x \notin L$ אז לא קיים y כנ"ל, ולכן עבור כל (x, y) האוב יענה "לא" ולכן המכונה תדחה בכל המסלולים. לכן $x \notin L(M^A)$.
לכן $L(M^A) = L$ ולכן $L \in \Sigma_{k-1}^P$.

דוגמה 1:

$L = \{\exists y_1 \forall y_2 ... Q y_k \left(\left\langle M \right\rangle, 1^i \right) |$ קיימת מחרוזת y_1 כך שלכל y_2 ... M מקבלת את $\left(y_1, ..., y_k \right)$ תוך i צעדים $\}$

טענה: L היא Σ_k^P שלמה.

צריך להוכיח:

1. $L \in \Sigma_k^P$
2. לכל שפה $L' \in \Sigma_k^P$ מתקיים $L' \leq_P L$.

הוכחה:

1. $L \in \Sigma_k^P$: ראינו ש $\Sigma_k^P = \exists \forall \exists ... QP$ כאשר יש k כמתים ו Q, הכמת האחרון, נקבע בהתאם וכמובן שאת הבדיקה האם M מקבלת את $\left(y_1, ..., y_k \right)$ תוך i צעדים אפשר לבצע בזמן פולינומי.

2. קושי: תהא $L' \in \Sigma_k^P$ ויהי A היחס ה $\left(k+1 \right)$ מקומי המתאים לה.

הרדוקציה: בהינתן מחרוזת x', נבנה מחרוזת $\exists y_1 \forall y_2 ... Q y_k \left\langle M_A, 1^i \right\rangle$ כאשר הגודל של כל בלוק משתנים הוא $P\left(|x| \right)$. M_A היא מ"ט עבור A (דטר' פולינומית). x מקודד בתוך M_A כחלק קבוע של הקלט. i הוא החסם הפולינומי על מספר צעדי הריצה של M_A על הקלט המתאים.

נכונות + סיבוכיות: לוודא!

דוגמה 2:

$\exists QSAT_k = \{ \varphi | \varphi = \exists x_1 \forall x_2 ... Q x_k \Phi \left(x_1, ..., x_k \right) \}$ ו φ היא אמת לוגית $\}$

$\forall QSAT_k = \{ \varphi | \varphi = \forall x_1 \exists x_2 ... Q x_k \Phi \left(x_1, ..., x_k \right) \}$ ו φ היא אמת לוגית $\}$

משפט: לכל $k \geq 1$ מתקיים:

$\exists QSAT_k$ היא Σ_k^P-שלמה.

$\forall QSAT_k$ היא Π_k^P שלמה.

הוכחה: שייכות, מיידית.

קושי: נראה שלכל $L \in \Sigma_k^P$, k אי זוגי, $L \leq_p \exists QSAT_k$.

נסמן ב A את היחס ה $k+1$ מקומי.

$A \in P$ לכן קיימת מ"ט דטר' פולינומית שמכריעה את A. נסמנה M_A.

ממשפט $Cook$ נובע שניתן לכתוב פסוק בוליאני φ שמייצג את החישוב של M_A.

נחלק את משתני φ ל $k+2$ קבוצות.

X - תכלול את כל המשתנים המייצגים סימנים בקלט שמתאימים ל x.

(הקלט ל M_A הוא מהצורה $x, y_1, y_2, ..., y_k$).

Y_i תכלול את כל המשתנים המייצגים סימנים בקלט המתאימים ל y_i.

Z - כל יתר המשתנים.

בהינתן x, הרדוקציה תחשב את $\exists y_1 \forall y_2 ... \exists y_k, z \varphi \left(x, y, z \right)$.

(נשים לב, שהביטוי $\varphi \left(x, y, z \right)$ ספיק אם ורק אם כל הערכים המתאימים מייצגים מחרוזת בשפה שמוכרעת ע"י M_A)

לכן, אם נציב ב φ את הערכים המתאימים ל x, אזי הפסוק המתקבל מהרדוקציה יהיה ספיק אמ"מ $x \in L$.

דוגמה 3:
משפטי קריסה באפיון האלטרנטיבי.

טענה: לכל $i \geq 0$ הטענות הבאות שקולות:
1. $\Sigma_i^P = \Sigma_{i+1}^P$
2. $\Pi_i^P = \Pi_{i+1}^P$.
3. $\Sigma_i^P = \Pi_i^P$.
4. $\Sigma_i^P = \Pi_{i+1}^P$
5. $\Pi_i^P = \Sigma_{i+1}^P$.
6. $\Sigma_i^P = PH$.

$1 \Leftrightarrow 2,4 \Leftrightarrow 5,6 \Rightarrow 1$ טריוויאלי.

$6 \Rightarrow 1$: נראה שלכל $j > i$ אם $\Sigma_i^P = \Sigma_j^P$ אז גם אז גם $\Sigma_{j+1}^P = \Sigma_i^P$:

$$\Sigma_{j+1}^P = \exists \Pi_j^P \underbrace{=}_{*} \exists \Pi_i^P = \Sigma_{i+1}^P \underbrace{=}_{1} \Sigma_i^P$$

*- הנחת האינדוקציה.

$1 \Rightarrow 3$:

$$\Sigma_i^P \subseteq \Pi_{i+1}^P = Co - \Sigma_{i+1}^P = Co - \Sigma_i^P$$

$3 \Rightarrow 4$:

$$\Sigma_i^P = \Pi_i^P = \forall \Pi_i^P = \forall \Sigma_i^P = \Pi_{i+1}^P$$

$4 \Rightarrow 1$:

$$\Sigma_{i+1}^P = Co - \Sigma_{i+1}^P = Co - \Sigma_i^P = \Pi_i^P \subseteq \Pi_{i+1}^P = \Sigma_i^P$$

זיהוי פתרון יחיד ורדוקציות רנדומיות

מה ראינו עד היום?
- פרק "קלאסי"
- אובות
- חישוב הסתברותי

עד עכשיו התעסקנו בשאלה האם קיים פתרון (למשל, האם קיימת השמה שמספקת פסוק). עכשיו נתעניין בשאלה האם קיים פתרון **יחיד**.

נגדיר את השפה $UniqSAT$:
$UniqSAT = \{\varphi \mid \varphi$ היא נוסחת CNF בוליאנית בעלת השמה מספקת יחידה$\}$

<u>טענה:</u> $UniqSAT \leq_p^T SAT$.
כלומר קיימת מ"ט בעלת אוב, M^{SAT}, המכריעה את $UniqSAT$ בזמן פולינומי.

<u>הוכחה:</u> נתאר רדוקציית טיורינג מתאימה:
M^{SAT} על קלט φ :
- שאל את האוב SAT, האם $\varphi \in SAT$. אם התשובה היא לא אז דחה.
- בנה את הנוסחה: $\varphi' = \varphi(x) \wedge \varphi(y) \wedge (x \neq y)$.
נשים לב שזו ביטוי באורך פולינומי וכי φ' ספיקה אם ורק אם ל φ קיימות <u>לפחות</u> שתי השמות שונות מספקות.
- שאל את האוב האם $\varphi' \in SAT$.
אם כן -דחה. אחרת - קבל.

<u>הגדרה:</u>
נאמר ששפה L_1 ניתנת ל**רדוקציית העתקה רנדומית** (או בקיצור **רדוקציה רנדומית**) לשפה L_2 אם קיימת מ"ט מטילת מטבעות, M (המחשבת פונקציה הסתברותית $f_M(x)$) וקיים פולינום $p(x)$ כך ש

$$x \in L_1 \rightarrow \Pr\left[f_M(x) \in L_2 \right] \geq \frac{1}{p(|x|)}$$
$$x \notin L_2 \rightarrow \Pr\left[f_M(x) \in L_2 \right] = 0$$

ומסמנים $L_1 \leq_{randomized} L_2$ או בקיצור $L_1 \leq_{rand} L_2$
פונקציה f_M כנ"ל הינה רדוקציה רנדומית מ L_1 ל L_2.

<u>טענה:</u>
$SAT \leq_{rand} UniqSAT$.
נובע מהטענה שאם $UniqSAT \in RP$ אז $SAT \in RP$ ולכן (מכיוון ש SAT היא NP שלמה) יתקיים $NP \subseteq RP$.

<u>הוכחה:</u> הרעיון: הרכבה של 2 פונקציות רנדומיות.
ניקח פסוק φ כלשהו, שיש לו מספר השמות מספקות. נבצע "דילול": נייצר מ φ פסוק חדש, φ' שיש לו לכל היותר 12 השמות מספקות.
כעת ניקח את φ' ובאמצעות "בידוד" נייצר פסוק φ'', שיש לו לכל היותר השמה אחת.
נשים לב שאם ל φ אין בכלל השמות אז גם ל φ'', φ' אין בכלל השמות מספקות ולכן
$\varphi \notin SAT \Rightarrow \Pr\left(f_M(\varphi) \in UniqSAT \right) = 0$.

נראה כיצד לבצע את שלב ה"בידוד" כך ש:
א. אם φ' לא ספיקה אזי גם φ" לא ספיקה בהסתברות 1.
ב. אם ל φ' יש לכל היותר 12 השמות מספקות אזי בהסתברות קבועה ל φ" יש בדיוק השמה מספקת יחידה.

פונקצית התרגום לשלב הבידוד:
בחר $m \in \{1,...,12\}$ והחזר: $\varphi" = \varphi_m" = \left(\wedge_{i=1}^{m} \varphi'(y_i)\right) \wedge \left(\wedge_{i=1}^{m-1} y_i < y_{i+1}\right)$.
<u>הערות</u>:
- השוואה לקסיקוגראפית בין שתי מחרוזות y_i, y_{i+1} ניתנת לביטוי ע"י נוסחת CNF בגודל פולינומי.
- פולינומיות הרדוקציה ברורה.
<u>נכונות:</u>
אם φ' הינה ספיקה אזי ודאי שגם φ" הינה ספיקה.
אם ל φ' יש s השמות מספקות כאשר $s \in \{1,....,12\}$ אזי ל φ_s" יש בדיוק השמה מספקת <u>יחידה</u> שבה $y_1,...,y_s$ הן ההשמות המספקות של φ' לפי הסדר הלקסיקוגראפי.
לכן, בהסתברות $\frac{1}{12}$ ל φ" תהיה השמה מספקת יחידה.

הפתרון לשלב הדילול: *Universal Hashing*
נגריל פונקצית *hash* $h: \{0,1\}^n \to \{0,1\}^k$ ואז נגדיר:
$\varphi'_k(k) = \varphi(x) \wedge \left(h(x) = 0^k\right)$.
פונקצית ה *hash* תהיה מהצורה הבאה: $h(x) = H \cdot x$
כאשר H היא מטריצה מסדר $k \times n$ מעל $GF(2)$ (כלומר מטריצה בינארית) והכפל הוא מעל $GF(2)$.
נבחין שעבור H קבוע כלשהי, התנאי $H \cdot x = 0^k$ ניתן לייצוג ע"י נוסחת CNF בגודל פולינומי.
(ניתן להשתמש במשתני עזר שלא משפיעים על מספר ההשמות).

למשל, עבור $H = \begin{pmatrix} 0,1,0 \\ 1,1,1 \end{pmatrix}$, $x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$, התנאי המתקבל הוא:
$h(x) = H \cdot x = \left((x_2 = 0) \wedge (x_1 \oplus x_2 \oplus x_3 = 0)\right)$.

<u>הרדוקציה:</u>
1. בחר באקראי $k \in \{0,...,n-1\}$ ומטריצה H מסדר $k \times n$ מעל $GF(2)$.
2. בנה את הנוסחה: $\varphi_k'(x) = \varphi(x) \wedge \left(H \cdot x = 0^k\right)$.
3. החזרת את $\varphi_k'(x)$.

ברור כי אם φ איננה ספיקה אז גם φ' איננה ספיקה.
<u>טענה:</u> אם φ ספיקה ע"י $i \geq 12$ השמות <u>שונות</u> מ 0^n, אזי עבור k כך ש $2^{k+2} < i \leq 2^{k+3}$ (כלומר $k+3 = \lceil \log i \rceil$) מתקיים:
$\Pr($מספר ההשמות המספקות את φ_k' הוא בין 1 ל 12$) > 0.5$
<u>הערה:</u> אם $0 < i < 12$ אז בחירת $k = 0$ מקיימת את הדרישה.

<u>מסקנה:</u> אם $\varphi \in SAT$ אז בהסתברות $> \frac{1}{24n} = \frac{1}{n} \cdot \frac{1}{2} \cdot \frac{1}{12}$ ל "φ יש השמה מספקת יחידה ותנאי הרדוקציה הנדרש מתקיים.

$\frac{1}{12}$ - בחירת m בשלב הבידוד.

$\frac{1}{2}$ - מהטענה.

$\frac{1}{n}$ - בחירת k בשלב הדילול.

<u>הוכחת הטענה:</u>

תהא $A \subseteq \{0,1\}^n$ קבוצת ההשמות המספקות את φ פרט אולי ל 0^n.

נגדיר משתנה מקרי S הסופר את מספר ההשמות ב A שממופות ל $\{0,1\}^k$.

(כלומר סופר את כל ההשמות ששורדות את הדילול).

באופן פורמאלי: $S = \left| \left\{ x \in A \mid H \cdot x = 0^k \right\} \right|$.

יהי R_x משתנה אינדיקאטור המתאים להשמה x ומקבל 1 אם $H \cdot x = 0^k$ ומקבל 0 אחרת.

לכן $S = \sum_{x \in A} R_x$.

לכל השמה, $x \neq 0^n$ בדיוק חצי מהווקטורים $a \in \{0,1\}^n$ מקיימים $a^T \cdot x = 0$.

מאחר ששורות H נבחרות באופן בלתי תלוי נקבל: $E[R_x] = \Pr[H \cdot x = 0^k] = 2^{-k}$.

(תוחלת E = התוחלת של אינדיקאטור שווה לסיכוי שהוא יקבל ערך 1 ולא 0)

לכל זוג השמות שונות, $x, y \neq 0^n$, בדיוק רבע מהווקטורים מקיימים $a^T \cdot (x \mid y) = 00$,

כלומר $\Pr[a^T \cdot x = 0 \wedge a^T \cdot y = 0] = \frac{1}{4}$..

לכן $E[R_x \cdot R_y] = \Pr[H(x \mid y) = 0^k 0^k] = 4^{-k}$.

מליניאריות התוחלת: $E[S] = E\left[\sum_{x \in A} R_x\right] = \sum_{x \in A} E[R_x] = |A| \cdot 2^{-k}$.

$$VAR(S) = E[S^2] - (E[S])^2 = E\left[\left(\sum_{x \in A} R_x\right)^2\right] - 4^{-k}|A|^2$$

נחשב שונות:

$$= \sum_{\substack{x,y \in A \\ x \neq y}} E[R_x R_y] + \sum_{x \in A} E[R_x^2] - 4^{-k}|A|^2$$

$$= |A|(|A|-1) \cdot 4^{-k} + |A| \cdot 2^{-k} - |A|^2 \cdot 4^{-k}$$

$$= 2^{-k}(1 - 2^{-k})|A|$$

נראה כי אם $|A| \geq 12$ אז $\Pr[1 \leq S \leq 11] > \frac{1}{2}$.

עבור k שבחרנו מתקיים $VAR(S) < 8$ ו $4 < E[S] \leq 8$.

אי-שוויון צ'בישב: $\Pr[|x| - E[x] \geq k] \leq \frac{VAR(x)}{k^2}$.

<u>מקבלים:</u> $\Pr\left[S \notin \{1,...,11\}\right] \le \Pr\left[\left|S - E\left[S\right]\right| \ge 4\right] \le \frac{VAR(S)}{4^2} < \frac{1}{2}$.

המחלקות ZPP, RL ושרשראות מרקוב

מה ראינו עד היום?
- מחלקות "קלאסיות"
- מחלקות עם אוב
- מכונת טיורינג מטילת מטבעות.

הגדרה:
$L \in ZPP$ אם קיימת מ"ט מטילת מטבעות פולינומית M כך ש:
א. ל M יש מצב סופי נוסף: "לא יודע".
ב. M אינה טועה (כומר אינה דוחה מילה השייכת ל L ואינה מקבלת מילה שלא שייכת ל L).
ג. לכל קלט x, M עונה "לא יודע" על x בהסתברות קטנה או שווה לשליש.

טענה:
$L \in ZPP$ אמ"מ קיימת מ"ט מ"מ בעלת **תוחלת זמן ריצה** (זה לא מחייב שכל ריצה תהיה פולינומית) פולינומי שטועה בהסתברות 0.

הוכחה:
כיוון ראשון: $\Leftarrow$ נתון ש $L \in ZPP$ וצריך להראות מ"ט כנ"ל.
נריץ את המכונה עד שנקבל תשובה שונה מ"לא יודע".
ההסתברות לקבלת תשובה כזו גדולה או שווה לשני שליש בכל פעם. לכן תוחלת מספר ההרצות הנדרשות היא 1.5.
כיוון שני: $\Rightarrow$:
נריץ את המכונה למשך 3 פעמים תוחלת זמן הריצה. במקרה כזה, נקבל תשובה בהסתברות לפחות שני שליש. אם לא קיבלנו תשובה, נענה "לא יודע".

הסבר: אי שוויון מרקוב: אם x משתנה מקרי אי שלילי, אזי מתקיים: $\Pr\left(x > k \cdot E[x]\right) < \frac{1}{k}$.

טענה:
$ZPP = CoRP \cap RP$.
RP - מחלקת השפות שאפשר להכריע באמצעות מ"ט מ"מ פולינומית בעלת טעות חד צדדית.

הוכחה:
כיוון ראשון: נראה ש $ZPP \supseteq CoRP \cap RP$.
נריץ את שתי המכונות במקביל. אם אחת מהן עונה תשובה בטוחה, כלומר מכונת RP עונה "כן" או מכונת $CoRP$ עונה "לא", נענה בהתאם. אחרת נענה "לא יודע".
כיוון שני: נראה ש $ZPP \subseteq CoRP \cap RP$.
כדי להראות שייכות ל RP, נריץ את המכונה כמו שהיא, ונענה "לא" במקום "לא יודע".
עבור $CoRP$ - הפוך (כלומר נענה "כן" במקום "לא יודע").

<u>הגדרה:</u>
$L \in RL$ אם קיימת מ"ט מטילת מטבעות M שעובדת בזיכרון לוגריתמי ובזמן פולינומי כך ש:

$$x \in L \Rightarrow P_M(x) \geq \frac{1}{2}$$
$$x \notin L \Rightarrow P_M(x) = 0$$

כאשר $P_M(x)$ הוא ההסתברות ש M מקבלת את x.

שימו לב להגבלת הפולינומיות על הזמן, שנדרשת כיוון שבמ"ט מ"מ לא ניתן לזהות אי עצירה ע"י קונפיגורציה שחוזרת על עצמה או ע"י ספירת קונפיגורציות.

מתקיים: $DL \subseteq RL \subseteq NL \subseteq P$.
בהרצאה ראינו שהבעיה $STCON$ היא שלמה ב NL לרדוקציות $\log-space$. כיוון ש RL סגורה תחת רדוקציות $\log-space$, נובע שאם $STCON \in RL$ אז $NL = RL$.

<u>נגדיר:</u> $USTCON$ - הגרסה הלא מכוונת של $STCON$: בהינתן גרף **לא מכוון**, G וזוג צמתים, s,t:
$(G,s,t) \in USTCON \Leftrightarrow$ קיים ב G מסלול לא מכוון מ s ל t.

<u>טענה:</u> $USTCON \in RL$.
<u>הוכחה:</u> הילוך מקרי שמתחיל ב s הוא סדרה מקרית של צמתים שמתחילה ב s כך שבכל צעד בוחרים לעבור לאחד משכני הצומת הנוכחי בהסתברות אחידה ובאופן בלתי תלוי בבחירות הקודמות.
<u>טענה מתורת שרשראות מרקוב:</u> לכל זוג צמתים (s,t) בגרף קשיר ולא מכוון, $G=(V,E)$ (שאינו דו צדדי), תוחלת מספר הצעדים הדרושים להילוך מקרי שמתחיל ב s להגיע ל t לראשונה $\leq 2|E|\cdot|V|$.

מהטענה הנ"ל נובע:
$\Pr[$הזמן שייקח להילוך מקרי להגיע מ s ל t $> 4|V||E|] < \frac{1}{2}$ (מאי שוויון מרקוב).
נקבל את האלגוריתם הבא: בצע הילוך מקרי שתחילתו ב s למשך $4|V||E|$. אם במשך ההילוך מגיעים ל t, ענה "כן". אחרת ענה "לא".
<u>סיבוכיות</u>: זמן ריצה פולינומי.
<u>זיכרון:</u> תחזוק המונה הפולינומי דורש $O(\log n)$ והחזקת הצומת הנוכחי בזיכרון דורשת $O(\log n)$.

מדוע אי אפשר להשתמש בפיתרון דומה עבור $STCON$ (גרף מכוון)?
דוגמה:
נסתכל על גרף בעל n צמתים כאשר המסלול הקצר ביותר מ s ל t הוא מעבר על כל הצמתים, ומכל צומת יש קשת אל s.
תוחלת מספר הצעדים הדרושים להגיע ל t הוא $\Omega(2^n)$.

<u>מה ידוע עד היום?</u>
$STCON \in NL$ ולכן מ $SAVITCH$ מתקיים $STCON \in DSPACE(\log^2 n)$.
עבור $USTCON$ ידוע: $USTCON \in DL$.

<u>שרשראות מרקוב</u>

תיאור המערכת: נתונה קבוצת מצבים S ומטריצת מעברים M בגודל $|S|\times|S|$, כך ש $M_{i,j}$ היא הסתברות המעבר ממצב i למצב j. סכום כל שורה במטריצה שווה ל 1. (מטריצה סטוכסטית).

שרשרת מרקוב היא סדרה של משתנים מקריים $x_0, x_1, x_2, \ldots$ כאשר x_t מייצג את מצב המערכת בזמן t, x_{t+1} תלוי אך ורק ב- x_t וב- M, והתהליך חסר זכרון (x_{t+1} אינו תלוי ב- $x_0,x_1,x_2,..x_{t-1}$ בהינתן x_t). ניתן לייצג כל מ"מ x_t ע"י וקטור הסת' $q^{(t)}$ באורך $|S|$, כאשר: $q_i^{(t)}=\Pr[x_t=i]$. ניתן לראות שמתקיים: $q^{(t+1)}=q^{(t)}\cdot M$ (כאשר מתייחסים ל- q כוקטור שורה), ולכן בהינתן פילוג של המצב ההתחלתי, $q^{(0)}$, הפילוג לאחר הצעד ה- t נתון ע"י $q^{(t)}=q^{(0)}\cdot M^t$.

הגדרה: פילוג π נקרא **סטציונרי** אם: $\pi M=\pi$. אם מתחילים את התהליך המוגדר ע"י M עם מצב התחלתי המפולג עפ"י π, פילוג מצבים זה יישאר ללא שינוי. במונחים אלגבריים, פילוג סטציונרי הוא ו"ע שמאלי מנורמל של M המתאים לע"ע 1.

המשפט הארגודי**: לכל שרשרת מרקוב שאיננה מחזורית המתאימה לגרף סופי וקשיר היטב מתקיים:

1. קיימת הסתברות סטציונרית יחידה π, בה כל האיברים אינם אפסים.
2. תוחלת מספר הצעדים להגעה ממצב i לאותו מצב הינה: $1/\pi_i$.

הילוך מקרי על גרף*

הילוך מקרי על גרף הוא שרשרת מרקוב בה קבוצת המצבים היא קבוצת הצמתים V ומטריצת המעברים מוגדרת ע"י:

$$M_{uv}=\begin{cases}\frac{1}{d(u)} & (u,v)\in E\\ 0 & else\end{cases}.$$

משפט: יהי $G=(V,E)$ גרף קשיר, סופי, לא מכוון ושאינו דו"צ, אזי: לכל $(u,v)\in E$ מתקיים: $h_{uv}+h_{vu}\le 2|E|$, כאשר h_{uv} - תוחלת מס' הצעדים הדרושים להילוך מקרי היוצא מ- u להגיע לראשונה ל- v.

תיאור ההוכחה: נהפוך את הגרף למכוון: נחליף כל קשת שאינה מכוונת בשתי קשתות מכוונות, לשני הכיוונים. עתה, נבנה שרשרת מרקוב חדשה, ובה $2|E|$ מצבים אפשריים שהם **הקשתות** בגרף. מטריצת המעברים תוגדר ע"י: $Q_{(u,v),(v,w)}=P_{v,w}=1/d(v)$. מטריצה זו היא סטוכסטית כפולה, כלומר, סכום כל שורה וסכום כל עמודה = 1. מכאן, שההסתברות הסטציונרית המתאימה לה הינה אחידה ושווה ל- $1/2|E|$. לפי המשפט הארגודי מתקיים: תוחלת מספר הצעדים הדרוש ממעבר על קשת מסוימת, ועד מעבר נוסף על אותה הקשת (באותו כיוון) הינה $2|E|$ (עבור כל קשת בגרף).

על מנת לחשב את $h_{uv}+h_{vu}$, נניח תחילה שהגענו ל- u דרך הקשת (v,u), ועתה יש להגיע לצומת v, ולחזור ל- u דרך אותה קשת. לפי הכתוב לעיל, תוחלת מספר הצעדים הדרושה הינה $2|E|$. עתה, אם נזניח את ההנחה ההתחלתית (כיצד הגענו ל- u), התוצאה לא תשתנה, שכן התהליך חסר זכרון. ■

משפט: אם קיים מסלול בין s ו- t בגרף, תוחלת מספר הצעדים הדרושים להילוך מקרי מ- s להגיע ל- t הינה קטנה או שווה $2|E||V|$.

הוכחה: אם קיים מסלול, בהכרח קיים מסלול פשוט: $s=v_0 \rightarrow v_1 \rightarrow \ldots \rightarrow v_k=t$, כאשר $k \leq |V|$. מכאן שתוחלת מספר הצעדים הדרושים להילוך מקרי היוצא מ- s ע"מ להגיע ל- t לראשונה היא קטנה שווה לתוחלת מספר הצעדים הדרושים להילוך מקרי היוצא מ- s לבקר ב- v_1, אח"כ (לאו דווקא מיד) ב- v_2, וכך הלאה עד ל- t. מחוסר הזכרון של התהליך, מלינאריות התוחלת וכן מתוצאת המשפט הקודם, התוחלת הנ"ל הינה קטנה שווה:

$$\sum_{i=0}^{k-1} h_{v_i v_{i+1}} \leq 2|E|k \leq 2|E||V|$$

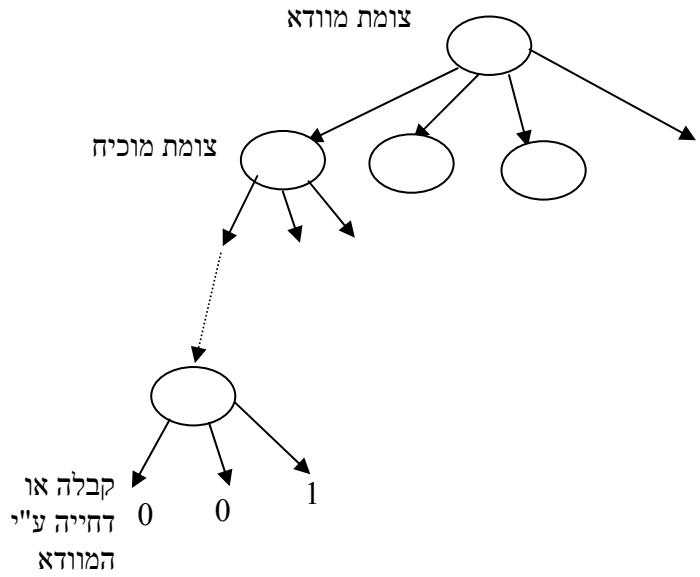
■

הוכחות אינטראקטיביות

נושאי התרגול:
- $IP \subseteq PSAPCE$
- $CoNP \subseteq IP$

משפט: $IP \subseteq PSAPCE$
הוכחה: כדי להכריע האם $x \in L$ מספיק לחשב את ההסתברות **המקסימאלית** בה מוכיח כלשהו יכול לגרום למוודא הספציפי של הפרוטוקול - לקבל.

ניתן לתאר את כל המוכיחים האפשריים (באינטראקציה עם המוודא V של הפרוטוקול) באמצעות "עץ מוכיחים":



בניו של צומת מוודא מתאימים ל $2^{P(n)}$ הודעות אפשריות הנשלחות ע"י המוודא בסיבוב המתאים, כתלות במחרוזת האקראיות r.

בניו של צומת מוכיח מתאימים ל $2^{P(n)}$ ההודעות האפשריות הנשלחות ע"י מוכיח כלשהו בשלב המתאים.

מוכיח ספציפי P, הוא תת גרף שבו לכל צומת מוכיח יש **בדיוק** בן אחד ולכל צומת מוודא מופיעים **כל** הבנים.
כדי לקבוע האם $x \in L$ מספיק לקבוע האם קיים "עץ מוכיח" (עבור מוכיח מסוים) בו משקלם ההסתברותי של העלים המסומנים ב '1' הוא לפחות $2/3$.
כלומר, האם קיים עץ מוכיח כך שמספר המחרוזות האקראיות שמובילות את המוודא לעלי '1' בעץ גדול או שווה ל $\frac{2}{3}2^{P(n)}$.

נשים לב שעומק העץ פולינומי ומכיוון שיש לכל היותר $2^{P(n)}$ בנים בכל שלב, הזיכרון הדרוש למעבר על כל העץ, לחישוב הסתברויות, הוא פולינומי.

האלגוריתם הרקורסיבי: (מחזיר את המספר המחרוזות האקראיות כנ"ל בעץ מוכיח כלשהו):
ספור (a):
אם a צומת עלה המסומן ב 1 החזר 1 ואם הוא צומת עלה המסומן ב 0 החזר 0.
אם a צומת מוכיח - החזר את הערך **המקסימאלי** של "ספור (a') עבור a' - בנים של a.
אם a צומת מוודא - החזר את הסכום **המשוקלל** (לפי התפלגות המחרוזות r) של ספור (a') על a' בנים של a.

האלגוריתם להכרעת L: על קלט x:
בנה (תוך כדי האלגוריתם, אין צורך להחזיק את כל העץ בזיכרון) את עץ המוכיחים וחשב את "ספור (שורש)".
אם הערך גדול או שווה ל $\frac{2}{3}2^{P(n)}$ - קבל. אחרת - דחה.

סיבוכיות: עומק הרקורסיה חסום ע"י $P(n)$ (עומק העץ פולינומי כי כל הפרוטוקול לוקח זמן פולינומי).
כמות המידע שיש לשמור בכל רמה היא פולינומית וכמובן שניתן לחשב ערך של עלה מתוך ייצוג המסלול בזמן ובזיכרון פולינומי.

הוכחה אינטראקטיבית עבור $\overline{3SAT}$ - נתון פסוק $3CNF$ ורוצים להוכיח שהוא איננו ספיק.

אריתמטיזציה – תזכורת כרקע להוכחה אינטראקטיבית עבור $\overline{3SAT}$

בהנתן פסוק φ ,3CNF, אשר m מספר הפסוקיות בו, תהי φ' אריתמטיזציה של φ, כלומר φ' פולינום מדרגה $3m \geq$ מעל המשתנים $x_1,\ldots,x_n$, אשר לכל השמה $x\in\{0,1\}^n$ מקבל את אותם ערכים כמו φ.

לדוגמא: הפסוקית $C=(x_1\vee\overline{x_2}\vee x_3)$ תתורגם ל: $C'=(1-(1-x_1)x_2(1-x_3))$, והפסוק $\varphi=\wedge C_i$ יתורגם ל- $\varphi'=\prod C'_i$.

לצורך ההוכחה האינטראקטיבית עבור $\overline{3SAT}$ נגדיר: $F_0=\sum_{y_1=0}^{1}\sum_{y_2=0}^{1}\ldots\sum_{y_n=0}^{1}\varphi'(y_1,\ldots,y_n)$.

שימו לב כי F_0 שווה למס' ההשמות המספקות את φ, ובפרט: $F_0\leq 2^n$.

מטרת המוכיח הינה להוכיח כי $F_0=0$. (נעבוד במודולו ראשוני $2^n<P<2^{n+1}$).

הרעיון: בכל סיבוב המוודא "יקלף" את ה Σ החיצוני ע"י הצבת ערך למשתנה המתאים. בסופו של דבר נישאר עם נוסחה חסרת Σ שערכה (קבוע) נוכל לחשב. המטרה היא לגרום למוכיח שמשקר בהתחלה (אמר ש $F_0=0$) להיתפס בהסתברות גבוהה אם הוא עובר משקר לאמת ולכן להיתפס בסוף "במקרה הרע".

<u>פורמאלית:</u>

בהינתן ערכים $(z_1,\ldots,z_{k-1})$ ל $k-1$ משתנים ראשונים, נגדיר פולינום במשתנה יחיד (המשתנה ה k):

$$F_k(z)=\sum_{y_{k+1}=0}^{1}\sum_{y_{k+2}=0}^{1}\ldots\sum_{y_n=0}^{1}\varphi'(z_1,\ldots,z_{k-1},z,y_{k+1},\ldots y_n)$$

<u>מתקיים:</u> $F_1(0)+F_1(1)=F_0$ וכן עבור ערך z_{k-1} כלשהו, $F_k(0)+F_k(1)=F_{k-1}(z_{k-1})$.

את $F_{n+1}=\varphi'(z_1,\ldots,z_n)$ (קבוע), המוודא יכול לחשב לבד בזמן פולינומי.

<u>הפרוטוקול:</u>

<u>סיבוב 1:</u> <u>המוכיח</u> שולח $2^n<P<2^{n+1}$ ראשוני וכן פולינום $G_1(z)$ מדרגה לכל היותר $3m$ ע"י שליחת מקדמיו מעל $GF(P)$ וטוען: $F_1(z)=G_1(z)$.

המוודא מוודא ש $G_1(0)+G_1(1)=0$ וכן ש P ראשוני.

הוא מגריל $z_1\in GF(P)$ ושולח אותו למוכיח.

<u>סיבוב k:</u> עבור $2\leq k\leq n$:

המוכיח שולח פולינומים $G_k(z)$ וטוען $F_k(z)=G_k(z)$.

המוודא מוודא ש $G_k(0)+G_k(1)=G_{k-1}(z_{k-1})$.

הוא מגריל $z_k\in GF(P)$ ושולח אותו למוכיח.

<u>סיבוב n+1:</u> המוכיח שולח <u>קבוע</u> G_{n+1}.

המוודא מוודא ש $G_n(z_n)=G_{n+1}=F_{n+1}$. (ניתן לעשות זאת בזמן פולינומי).

אם כן, מקבל ואחרת דוחה.

פולינומיות: ברורה.

שלמות ונאותות:

משחקי ארתור-מרלין - פרוטוקול הוכחה עם מטבעות ציבוריים

נושאי התרגול:
- מחלקות AM
- דוגמה: GNI

משחקי ארתור-מרלין הם מערכת הוכחה אינטראקטיבית בה המוודא A (ארתור) מוגבל להגרלת מחרוזות אקראיות באורך פולי' ושליחתן למוכיח M (מרלין). בסיום התקשורת, המוודא עונה ע"י חישוב פולי' דטר' התלוי בקלט ובתקשורת הקודמת עם המוכיח. למעשה, ההבדל העיקרי בין מערכת הוכחה זו ל- IP (מערכת ההוכחה שהכרנו עד כה) הוא השימוש של המוודא A במטבעות "**ציבוריים**" בניגוד למטבעות ה"**פרטיים**" בהם השתמש המוודא V עד כה.

עפ"י השחקן הראשון ומספר הסיבובים ניתן להגדיר את המחלקות הבאות:

M (= NP), A (= BPP), MA, AM, MAM, AMA, …

למשל, AM היא מחלקת השפות שניתנות להוכחה ע"י משחק בו A משחק ראשון (שולח 'אתגר אקראי' r למרלין) ואח"כ M עונה תשובה y לאתגר. לבסוף, המוודא מחשב בזמן פולי' תשובה מתוך x,y,r. באופן פורמלי: $L \in AM$ אם קיימת $A \in P$ כך ש:

$x \in L \rightarrow Pr_r[\exists y\ (x,y,r) \in A] \geq 3/4$

$x \notin L \rightarrow Pr_r[\exists y\ (x,y,r) \in A] \leq 1/4$

הקבועים $\frac{1}{4}, \frac{3}{4}$ שרירותיים, וכל זוג קבועים שאחד קטן מחצי והשני גדול מחצי יתאימו.

באופן דומה MA היא מחלקת השפות אותן ניתן להכריע בפרוטוקול הוכחה אינטראטיבי, בו מרלין מתחיל ושולח הוכחה לארתור ללא תלות בהגרלה אקראית כלשהי, ארתור מגריל מחרוזת r, ומחליט אם לקבל או לדחות כתלות ב- (x,y,r). באופן פורמלי: $L \in MA$ אם קיימת $A \in P$ כך ש:

$x \in L \rightarrow \exists y\ Pr_r[(x,y,r) \in A] \geq 3/4$

$x \notin L \rightarrow \forall y\ Pr_r[(x,y,r) \in A] \leq 1/4$

שימו לב שמתקיים: $MA \subseteq AM$.

מסתבר שהמעבר למטבעות ציבוריים **אינו** מחליש את המודל (הפרוטוקול האינטראקטיבי ל- TQBF משתמש במטבעות ציבוריים), כלומר: $AM(poly) = IP$. כמו-כן מתקיים: $AM(const) = AM = IP(2)$.

פרוטוקול AM ל GNI

תזכורת: $GNI = \{G_1, G_2 \mid G_1$ ו G_2 הם גרפים לא איזומורפיים זה לזה$\}$

ב IP השיטה הייתה שהמוודא יגריל ביט $b \in \{1,2\}$ וישלח פרמוטציה H של G_b למוכיח.

המוכיח יגלה למי H איזומורפי וע"פ זה ישלח את b.

אם הגרפים אכן לא איזומורפיים, אז למוכיח לא תהיה בעיה לגלות למי H איזומורפי ולשלוח את b בהתאם.

אם הם כן איזומורפיים, אז H איזומורפי לשניהם, ולכן בהסתברות חצי, המוכיח ישלח את הביט הלא נכון.

<u>משפט:</u> $GNI \in AM$

הרעיון: נתאים לזוג גרפים G_1, G_2 מעל n צמתים קבוצה X_{G_1,G_2} כך שאם $G_1 \neq G_2$ אז X_{G_1,G_2} "גדולה" ואחרת X_{G_1,G_2} "קטנה".

המוודא ישלח למוכיח פונקצית $HASH$ שנסמנה h, הממפה איברים של X_{G_1,G_2} לטבלה בגודל מתאים והמוכיח ינסה להראות שקיים $x \in X_{G_1,G_2}$ כך ש $h(x) = 0^k$ ע"י שליחת x כזה (אם קיים) למוודא ביחד עם הוכחה לשייכותו ל X_{G_1,G_2}.

(השפה $\{(G_1, G_2, x) \mid x \in X_{G_1,G_2}\}$ תהיה ב NP).

כיוון שבמקרה שהגרפים אינם איזומורפיים, הקבוצה X_{G_1,G_2} גדולה יותר, הסיכוי להצלחתו של המוכיח יהיה גדול יותר בהתאם.

כיצד נבנה את X_{G_1,G_2}?

הרעיון: פרמוטציות על הגרפים הקיימים.

אם הגרפים איזומורפיים, קבוצת הפרמוטציות על גרף אחד מתלכדת עם קבוצת הפרמוטציות של הגרף השני.

פורמאלית: לגרף G מעל n צמתים, נגדיר: $N_G = \{(H, \sigma) \mid H \equiv G \wedge \sigma(H) = H\}$

σ היא פרמוטציה על הצמתים.

טענת עזר שנראה בסוף: $|N_G| = n!$ לכל G.

נגדיר: $N_{G_1,G_2} = N_{G_1} \cup N_{G_2}$.

נבחין שאם $G_1 \equiv G_2$ אז $N_{G_1} = N_{G_2}$.

אם $G_1 \neq G_2$ אז $N_{G_1} \cap N_{G_2} = \phi$.

לכן: $|N_{G_1,G_2}| = \begin{cases} n! & G_1 = G_2 \\ 2n! & G_1 \neq G_2 \end{cases}$

נגדיל את היחס ל 8 (שרירותי - עדיף אפילו יותר באופן דומה), ע"י:

$X_{G_1,G_2} = (N_{G_1,G_2})^3$ ונקבל: $|X_{G_1,G_2}| = \begin{cases} (n!)^3 & G_1 = G_2 \\ 8(n!)^3 & G_1 \neq G_2 \end{cases}$

יהי m גודל הייצוג של איבר ב X_{G_1,G_2} (m פולינומי ב n).

בנוסף, נניח כי 0^m אינו ייצוג חוקי.

נקבל k כך ש $16S \leq 2^k \leq 32S$ כאשר $S = (n!)^3$.

הפרוטוקול על (G_1, G_2):

1. A שולח ל M טרנספורמציה ליניארית אקראית $k : \{0,1\}^m \to \{0,1\}^k$.
2. M שולח ל A $x \in X_{G_1,G_2}$ כך ש $h(x) = 0$ (אם קיים כזה) בצירוף 'עד' לכך ש $x \in X_{G_1,G_2}$.
3. A מוודא שאכן $x \in X_{G_1,G_2}$ וכן ש $h(x) = 0^k$. אם כן, מקבל. אחרת, דוחה.

ניתוח נכונות הפרוטוקול:

נסמן ב Rx את המאורע $h(x)=0^k$. ונסמן $T=2^k$.

בתרגול על *UniqeSAT* ראינו כי לכל $x,y\neq 0^m$ כך ש $x\neq y$ מתקיים:

$\Pr[Rx]=\frac{1}{T}$ וכן $\Pr[Rx\wedge Ry]=\frac{1}{T^2}$.

מהגדרת k מתקיים: $16S\le T\le 32S$.

מכאן שאם $G_1\equiv G_2$ אז: (ע"פ *Union bound*)

$$\Pr(\exists x: Rx)\le S\cdot\frac{1}{T}\le\frac{1}{16}$$

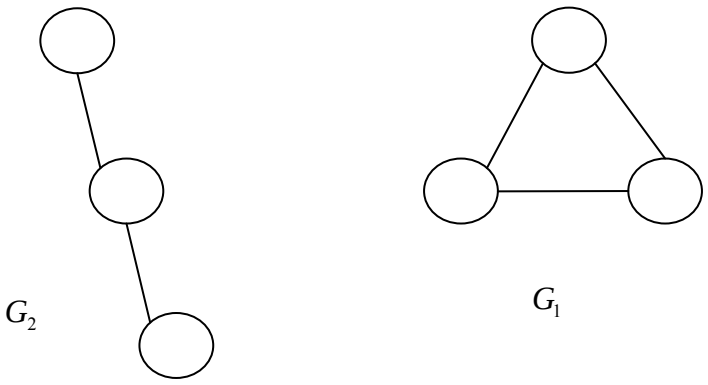
אם $G_1\neq G_2$ אז (ע"פ כלל ההכלה וההפרדה):

$$\Pr(\exists x: Rx)\ge\sum_{x\in X_{G_1,G_2}}\Pr[Rx]-\sum_{\substack{x<y\\ x,y\in X_{G_1,G_2}}}\Pr[Rx]\wedge Ry=\frac{8S}{T}-\binom{8S}{2}\cdot\frac{1}{T^2}\ge\frac{1}{4}-\binom{8S}{T}\cdot\frac{1}{(16S)^2}\ge\frac{1}{4}-\frac{1}{8}=\frac{1}{8}$$

הערה: את ההסתברויות ניתן להרחיק זו מזו באופן סטנדרטי - למשל ע"י בחירת X_{G_1,G_2} להיות חזקה גבוה יותר של N_{G_1,G_2}, חזרה על הפרוטוקול במקביל וכו'.

תזכורת: $H\equiv G$ אם ורק אם קיימת פרמוטציה Π כך שלכל זוג צמתים $v_1,v_2\in H$ מתקיים:
$(v_1,v_2)\in E_H\Leftrightarrow(\Pi(v_1),\Pi(v_2))\in E_G$.

דוגמאות:



$\left|\{H\mid H\equiv G_1\}\right|=1$ $\quad$ $\left|\{H\mid H\equiv G_2\}\right|=3$

$\left|\{\varphi\mid\varphi(G_1)\equiv G_1\}\right|=6$ $\quad$ $\left|\{\varphi\mid\varphi(G_2)\equiv G_2\}\right|=2$

מניחים שגרף ניתן לייצוג באופן יחיד, למשל ע"י רשימת הקשתות שלו לפי סדר לקסיקוגרפי.

הגרף הימני למשל ייוצג ע"י $G_1=(1,2),(1,3),(2,3)$

לכן, אם נפעיל את הפרמוטציה $\Pi=(1\ \ 2\ \ 3)$, כמו גם כל פרמוטציה אחרת על 3 צמתים, נקבל גרף זהה לחלוטין.

בהתאם, מספר הגרפים האיזומורפיים לגרף זה, הוא בדיוק 1.

<u>הוכחת טענת העזר:</u> לכל G עם n צמתים, $|N_G| = n!$

הוכחה: הקבוצה $A_G = \{\sigma \in S_n \mid \sigma(G) = G\}$ היא תת חבורה של S_n (ונקראת חבורת האוטומורפיזמים של G) - קל לוודא שקבוצה זו סגורה להרכבת פרמוטציות.

כמה גרפים שונים איזומורפיים ל G? נתבונן ביחס המוגדר ע"י $\sigma_1 \sim_G \sigma_2$ אמ"מ $\sigma_1(G) = \sigma_2(G)$.

זהו יחס שקילות אשר מחלקותיו הינן הקוסטים (השמאליים) של A_G ב S_n (שכן $\sigma_1 \sim_G \sigma_2$ אמ"מ $\sigma_2^{-1}\sigma_1 \in A_G$).

כל מחלקת שקילות מתאימה לגרף איזומורפי ל G וגודלן של כל המחלקות זהה (ושווה ל $|A_G|$).

מכאן ש $|\{H \mid H \equiv G\}| = \frac{n!}{|A_G|}$ (כמספר מחלקות השקילות של היחס).

בנוסף, לכל $H \equiv G$, H מתקיים: $|A_G| = |A_H|$.

לכן: $|N_G| = |\{H \mid H \equiv G\}| \cdot |A_G| = n!$.

מש"ל.

נוסחאות בוליאניות

הגדרה:
נוסחה בוליאנית היא מעגל בוליאני שה $fan-out$ שלו (דרגת יציאה של הצמתים) חסום ע"י 1.
$FVAL = \{ (F,x) \mid F(x)=1$, x השמה, F נוסחה$\}$

תזכורת: בהרצאה ראינו ש $CVAL$ היא P-שלמה ביחס לרדוקציות $\log-space$.

טענה: $FVAL \in DL$.

הוכחה:
נתאר אלגוריתם שמחשב את ערך הנוסחה F בזיכרון דטרמיניסטי $O(\log n)$.
נניח שלכל צומת פנימי $fan-in=2$ (דרגת כניסה) - הטיפול במקרה הכללי דומה.
הרעיון:
נבחין שניתן לסרוק את העץ הבינארי באופן "עיוור" מבלי לזכור את המסלול מהשורש.
סריקה כזו דורשת לזכור רק את מספר הצומת הנוכחי וזה דורש $O(\log n)$ זיכרון.

האלגוריתם:
בצומת u - עלה, נחשב את ערך הליטרל על ההשמה x ונחזיר תשובה לאב.
- אם מגיעים לצומת u מהאבא שלו (או ש u הוא שורש) אז יורדים לבן השמאלי.
- אם מגיעים לצומת u המסומן ב $\vee$ מהבן השמאלי עם תשובה b אז:
  אם $b=1$ חוזרים לאבא עם תשובה 1.
  אם $b=0$ יורדים לבן הימני של u.
- אם מגיעים לצומת u המסומן ב $\vee$ מהבן הימני עם תשובה b אז חוזרים לאבא של u עם תשובה b.
- אם מגיעים לצומת u המסומן ב $\wedge$ מהבן השמאלי עם תשובה b אז:
  אם $b=0$ חוזרים לאבא עם תשובה 0.
  אם $b=1$ יורדים לבן הימני של u.
- אם מגיעים לצומת u המסומן ב $\wedge$ מהבן הימני עם תשובה b אז חוזרים לאבא של u עם תשובה b.

ניתוח זיכרון:
שמירת מספר הצומת הנוכחי $O(\log n)$.
אינפורמציה הנוגעת לצומת הנוכחי (מאין באנו, הערך המוחזר) - $O(1)$.
חישוב ערך עלה $O(\log n)$ - דורש מונה כדי למצוא ערך הליטרל בהשמה x.

למה האלגוריתם הזה לא עובד עבור $CVAL$?
במעגל כללי יתכן שלצומת יהיו מספר הורים ולכן כדי לבצע DFS כנ"ל, יש לזכור את המסלול מהשורש - דבר שעשוי להצריך $\Omega(n)$ זיכרון.

הקשר בין עומק מעגל וגודל נוסחה

נעסוק רק במעגלים / נוסחאות עם $fan-in$ חסום (בפרט, 2). הסיבה - לכל פונקציה יש מעגל / נוסחה בעומק 2, עם $fan-in$ לא חסום.

תהא f פונקציה בוליאנית.

נסמן ב $L(f)$ את הגודל המינימאלי של נוסחה עבור f (עם $fan-in=2$).

נסמן ב $D(f)$ את העומק המינימאלי של מעגל עבור f (עם $fan-in=2$).

טענה: $D(f)=\Theta\left(\log\left(L(f)\right)\right)$.

הוכחה:

כיוון ראשון: $D(f)=\Omega\left(\log\left(L(f)\right)\right)$

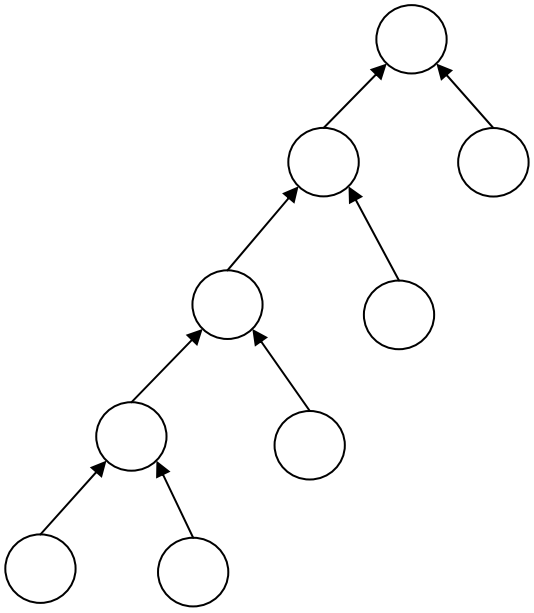
בהינתן מעגל עבור f בעומק $D(f)$ ניתן לתרגם אותו לנוסחה שקולה בעלת עומק זהה ע"י הכפלת כל תת מעגל של רכיב עם $fan-out>1$.

הנוסחה המתקבלת היא עץ בינארי בעומק $D(f)$ ולכן בעלת $O\left(2^{D(f)}\right)$ צמתים, ולכן $L(f)=O\left(2^{D(f)}\right)$ ולכן $D(f)=\Omega\left(\log\left(L(f)\right)\right)$.

כיוון שני: $D(f)=O\left(\log\left(L(f)\right)\right)$

כיוון שכל נוסחה הינה מעגל, בהינתן נוסחה φ עבור f בגודל $L(f)$ מספיק להפוך אותה לנוסחה שקולה בעומק $O\left(\log\left(L(f)\right)\right)$.

במקרה הגרוע, הנוסחה היא מהצורה הבאה:



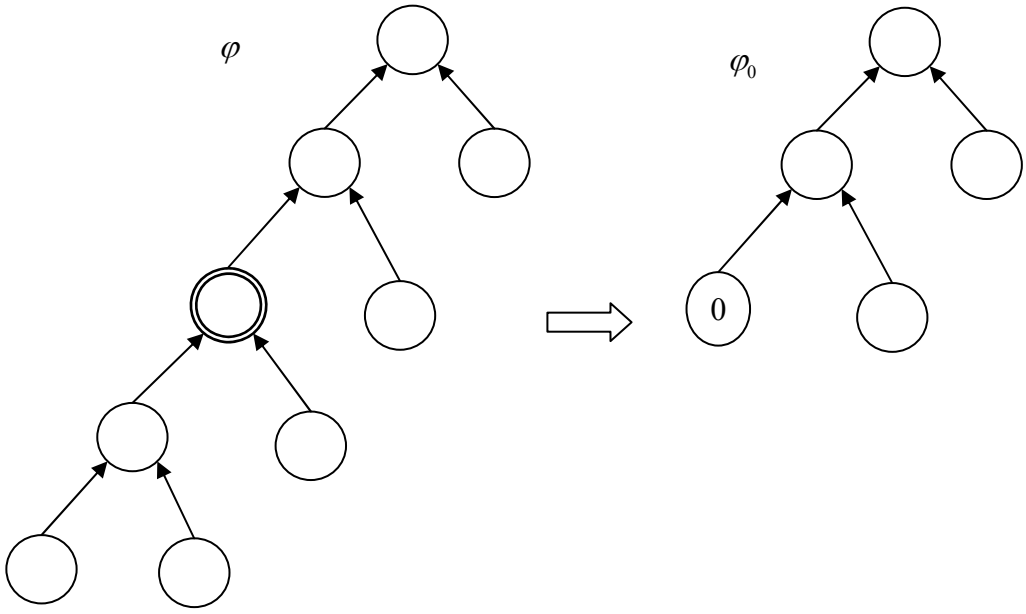
כלומר: עומק הנוסחה הוא $\frac{1}{2}L(f)$.

<u>הפתרון:</u>
נהפוך את העץ של הנוסחה φ לעץ מאוזן של נוסחה 'φ שקולה ל φ.

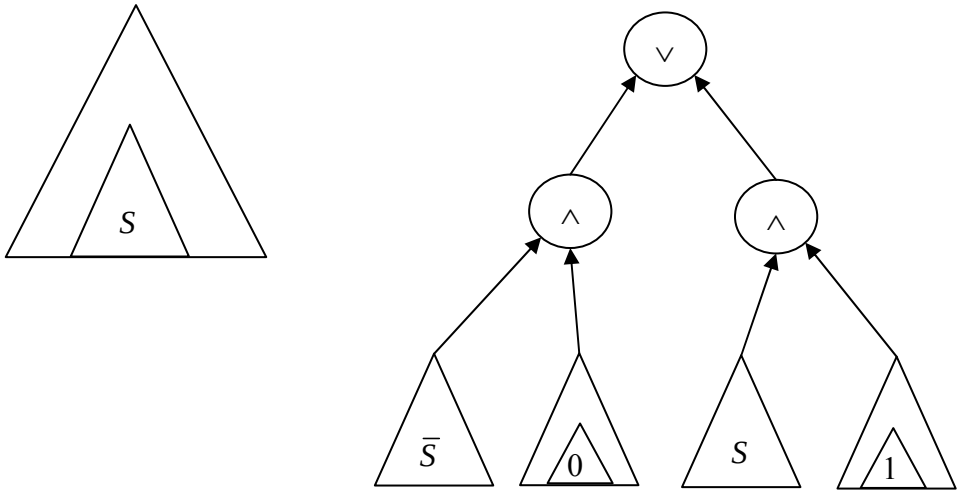
<u>נסמן:</u> $L = L(f)$.
נשים לב כי בהכרח קיים צומת בעץ כך שלתת העץ שלו יש לפחות $L/3$ צמתים ובכל אחד מתתי העצים של בניו מספר הצמתים לכל היותר $L/3$:

מתחילים מהשורש, שמספר הצמתים בתת העץ שלו גדול מ $L/3$.
אם לפחות באחד משני תתי העצים שלו מספר המצתים $\leq L/3$ אז יורדים לבן זה.
באופן רקורסיבי, עד שעוצרים.

נסמן את תת העץ של צומת כנ"ל ב S. נבחין ש $\frac{L}{3}-1 \leq |S| < \frac{2}{3}L$ (הגודל נמדד במספר הקשתות).
נסמן ב φ_0 או φ_1 המתקבלת מ φ ע"י החלפת תת העץ S בקבוע 0 או 1 בהתאמה.



נבחין ש $|\varphi_0| = |\varphi_1| = |\varphi| - |S| + 2 \leq \frac{2}{3}L + 3$.
כאשר מוסיפים 2 לצורך מימוש קבועים.
כעת, ניתן לבצע את פעולת האיזון תוך הסתמכות על השוויון: $\varphi = (\overline{S} \wedge \varphi_0) \vee (\overline{S} \wedge \varphi_1)$.



נציין כי את הרכיב $\overline{S}$ ניתן לבנות באופן דומה להוספת שערי *not* מבלי לשנות את עומק הנוסחה (כפי שראינו בהרצאה). לכן, סה"כ, הוספנו עומק קבוע (2) והקטנו את גדלי הנוסחאות בפקטור קבוע (בערך $2/3$). אם נחזור על התהליך $O(\log(L))$ פעמים, נקבל נוסחה בעומק הנדרש.

ולכן $D(f)=O(\log(L(f)))$.

<u>באופן מדויק:</u>

נגדיר:

$Depth(L)=$ {העומק המינימאלי של נוסחה שקולה ל F} max F שיש להם נוסחה בגודל L

$Depth(L)\le 2+Depth\left(\frac{2}{3}L+3\right)$.

$\forall i\le 10, Depth(i)\le i$

לכן $Depth(L)=O(\log(L))$.

כלומר, הרעיון הוא שהגודל של כל נוסחה קטן בפקטור $2/3$ וכל פעם העומק גדל ב 2.
לכן סה"כ, כיוון שבבסיס עומק כל נוסחה שגודלה לכל היותר 10, הוא גם כן לכל היותר 10, סה"כ העומק יהיה $O(\log(L))$.

<u>מסקנה:</u> $Poly-Formula=NC^1$ כאשר $Poly-Formula$ היא מחלקת השפות המתקבלות ע"י משפחת נוסחאות בגודל פולינומי ו NC^1 היא מחלקת השפות המתקבלות ע"י משפחת מעגלים בגודל פולינומי, עומק $O(\log n)$ ו $fan-in$ חסום.

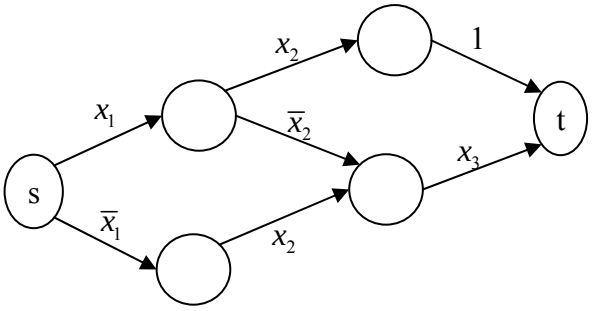
תוכניות מתפצלות

תכניות מתפצלות (Branching Programs)

הגדרה: תוכנית מתפצלת היא רביעיה $BP=(G,s,t,\phi)$, כאשר G הוא DAG, s צומת התחלה, t צומת סיום ו- ϕ פונק' סימון לקשתות המסמנת כל קשת בליטרל חיובי, ליטרל שלילי או קבוע. כל השמה x למשתנים משרה תת גרף G_x המכיל רק את הקשתות שסימוניהן מסתפקים ע"י x.

BP נקראת **דטרמינסטית** אם תחת כל השמה x, דרגת היציאה של כל צומת ב- G_x היא לכל היותר 1, אחרת BP היא א"ד. BP מקבלת את x אם בגרף G_x קיים מסלול מ- s ל- t ("מסלול מקבל").

דוגמה:



גרף זה מחשב $Majority(x_1,x_2,x_3)$: כלומר, מקבל 1 אם ורק אם לפחות 2 מהמשתנים מקבלים 1.

הגדרה: אם הגרף G של תוכנית מתפצלת BP הוא גרף שכבות (כלומר צמתיו מחולקים לשכבות, כך שכל קשת מחברת בין שכבות עוקבות), אזי **הרוחב** של BP הוא גודל השכבה הגדולה ביותר, **והאורך** הוא מספר השכבות פחות 1.

בדוגמה שלנו, הרוחב הוא 2, והאורך הוא 3.

משפט (Barrington 89): $L\in NC^1$ אמ"מ קיימות ל- L תכניות מתפצלות בעלות רוחב 5 ואורך פולי'.

תזכורת מההרצאה:
NC^k [$non-unif$ או $\log-space-unif$ או $poly-time-unif$] היא אוסף השפות שקיימת עבורן משפחת מעגלים $\{C_n\}_{n\geq 0}$ שהיא [כנ"ל] בגודל **פולינומי**, דרגת כניסה 2 ועומק $O(\log^k n)$.
הקונבנציה בדרך כלל היא ש NC^k היא $non-unif$.

<u>הוכחת המשפט:</u>

<u>כיוון 1:</u>

בהינתן שפה L עבורה קיימת משפחת תוכניות מתפצלות ברוחב 5 ובאורך פולינומי, נראה ש $L\in NC^1$

נגדיר רקורסיבית נוסחה $Path_{v_1,v_2}(\vec{x})$ המוגדרת על צמתים v_1,v_2 כך ש v_2 הוא צומת משכבה <u>מאוחרת</u> יותר מזו של v_1. הנוסחה תחזיר 1 אם ורק אם קיים מסלול מ v_1 ל v_2 על G_x.

<u>בסיס</u>: אם v_2 משכבה **עוקבת** לזו של v_2 אזי: $Path_{v_1,v_2}(\vec{x})=\begin{cases}0 & (v_1,v_2)\notin E\\ \varphi(v_1,v_2) & (v_1,v_2)\in E\end{cases}$

צעד: אם v_2 לפחות 2 שכבות אחרי v_1: $Path_{v_1,v_2}(\vec{x}) = \vee_{u \in U}\left(Path_{v_1,u}(\vec{x}) \wedge Path_{u,v_2}(\vec{x})\right)$
כאשר U היא השכבה הנמצאת בדיוק באמצע בין v_1 לבין v_2.

הנוסחה הראשית תהיה $Path_{s,t}(x)$.

<u>ניתוח</u>:
יהי D האורך (מספר השכבות) של התוכנית.
נסמן ב $Size(D)$ את גודל המעגל המתקבל לתוכנית מתפצלת באורך D.

אזי מתקיים: $Size(D) = 10 \cdot Size\left(\frac{D}{2}\right)$

(10 כיוון שבנוסחה הרקורסיבית יש 5 צמתי ביניים ($|U| \le 5$) ולכל אחד שני תנאים:
$\left(Path_{v_1,u}(x) \wedge Path_{u,v_2}(x)\right)$).

בסיס: $Size(1) = O(1)$

לכן $Size(D) = O\left(10^{\log_2 D}\right) = O\left(D^{\log_2 10}\right)$ - כמו שרצינו, פולינומי.

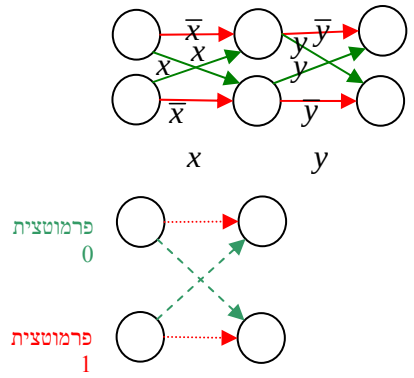
נסמן ב $Depth(D)$ את עומק המעגל המתקבל.

$Depth(D) = 2 + Depth\left(\left\lceil \frac{D}{2} \right\rceil\right) = O(\log D)$ - מוסיפים 2 בגלל שערי ה $\wedge, \vee$ שבדרך.

<u>כיוון 2</u>: נראה שממעגל בעומק d מעל שערי AND ו NOT ניתן לקבל תוכנית מתפצלת ברוחב 5 ובאורך לכל היותר 4^d.

לשם כך, נעזר בסוג מיוחד של תוכניות מתפצלות BP שנקרא **תוכניות פרמוטציה**.
<u>שלב 1</u>: מהן תוכניות פרמוטציה?
בתוכניות אלו, לכל רמה (שכבה), מותאם משתנה, וכן 2 פרמוטציות:
פרמוטצית 0
פרמוטצית 1
(הפרמוטציות שלנו תהיה מעל S_5).
כל השמה למשתנים תשרה באופן טבעי פרמוטציה מ S_5 לכל שכבה ולכן גם פרמוטציה לתוכנית ע"י הרכבת כל הפרמוטציות לפי סדר השכבות.

דוגמה:



נאמר שתוכנית פרמוטציות σ-מקבלת פונקציה בוליאנית f , $\sigma \in S_5 \setminus \{e\}$, אם לכל $\vec{x}$ כך ש $f(\vec{x})=0$ מתקבלת מהתוכנית פרמוטציות הזהות e, ולכל $\vec{x}$ כך ש $f(\vec{x})=1$, התוכנית משרה את σ.

<u>תזכורת לגבי פרמוטציות:</u>
$(1\,2)(3\,5)$ היא הפרמוטציה שבה 1 עובר ל 2 ו 2 עובר ל 1 ו 3 עובר ל 5 ו 5 עובר ל3 (ו 4 עובר לעצמו).
$(1\,2)(3\,5\,4)$ היא הפרמוטציה שבה 1 עובר ל 2, 2 עובר ל 1, 3 עובר ל 5, 5 עובר ל 4 ו 4 עובר ל 3.

<u>שלב 2:</u> למה זה טוב? (שיהיה מה לשאול במבחן?)
<u>טענת עזר:</u> מתוכנית פרמוטציה אשר σ-מקבלת את f $(\sigma \neq e)$, ניתן לקבל מתוכנית מתפצלת ל f בעלת אותן מימדים.
<u>הוכחה:</u> בוחרים i כך ש: $\sigma(i) \neq i$, קובעים את s כצומת ה i בשכבה הראשונה ואת t כצומת $\sigma(i)$ בשכבה האחרונה.
את הקשתות מסמנים בהתאם לפרמוטציות.

נכונות: מיידת מהגדרת σ-קבלה.

<u>שלב 3:</u> בניית תוכנית פרמוטציה כנדרש.
הבניה באינדוקציה על מבנה המעגל.
<u>הכנה:</u>
<u>תזכורת:</u> פרמוטציה ציקלית היא כזו שבייצוג שלה כמכפלת מעגלים זרים יש מעגל 1 שמכיל את <u>כל</u> האיברים. לדוגמה: $(1\,2\,3\,4\,5)$.
בנוסף, אם σ - τ הן ציקליות, (או במקרה הכללי, בעלות אותו מבנה מעגלים) אזי קיימת פרמוטציה γ כך ש $\tau = \gamma \cdot \sigma \cdot \gamma^{-1}$ (כלומר σ ו τ צמודות).

<u>למה:</u> תהנה σ, τ פרמוטציות ציקליות מתוכנית פרמוטציות אשר σ-מקבלת את f . ניתן לקבל תוכנית פרמוטציות מאותו גודל אשר τ-מקבלת את f .
<u>הוכחה:</u> מספיק לשנות את שמות הצמתים בשכבה הראשונה ובשכבה האחרונה בהתאם ל γ.

<u>טענת עזר:</u> לכל ליטרל חיובי x_i ולכל $\sigma \neq e$ קיימת תוכנית פרמוטציות באורך 1, אשר σ-מקבלת את $f(\vec{x}) = (x_i)$.
<u>הוכחה:</u> קובעים את σ כפרמוטציה ה 1 ואת e כפרמוטציה ה 0 של המשתנה x_i .

טענה: אם קיימת ת"פ P_1 באורך L_1 אשר σ_1-מקבלת את f_1, ות"פ P_2 באורך L_2 אשר σ_2-מקבלת את f_2 כאשר σ_1,σ_2 פרמוטציות ציקליות, אזי קיימות:

א. ת"פ P_3 באורך L_1 אשר σ_3-מקבלת את $\neg f_1$ (σ_3 ציקלית).

ב. ת"פ P_4 באורך $2(L_1+L_2)$ אשר σ_4-מקבלת את $f_1 \wedge f_2$ (σ_4 ציקלית).

הוכחת הטענה:

את P_3 ניתן לקבלת ע"י הפעלת σ^{-1} על השכנה האחרונה, וכך כל השמה שהשרתה e תשרה σ^{-1}, וכל השמה שהשרתה σ_1 תשרה $\sigma_3=\sigma_1^{-1} \leftarrow \sigma_1\sigma_1^{-1}=e$.

כדי לקבל את P_4 נגדיר את שתי הפרמוטציות הבאות:

$\sigma_1 = (1\ 2\ 3\ 4\ 5), \sigma_2 = (1\ 3\ 5\ 4\ 2)$

ניתן לוודא כי: $\sigma_4 = \sigma_1\sigma_2\sigma_1^{-1}\sigma_2^{-1} = (1\ 3\ 2\ 5\ 4)$, (אף היא ציקלית).

לכן, ת"פ כנדרש ניתן לקבל ע"י השרשור $P_1P_2P_1^{-1}P_2^{-1}$ (כאשר P_1^{-1} ו- P_2^{-1} מתקבלות מ- P_1 ו- P_2 ע"י שימוש בלמה שמוצגת בתרגול).

אם $f_1(x)\wedge f_2(x)=1$ אזי הפרמוטציה המושרית תהיה σ_4, אחרת הפרמוטציה המושרית תהיה e:

$\sigma_1 e \sigma_1^{-1} e = e\sigma_2 e\sigma_2^{-1} = eeee = e$

תרגילים ממבחנים ישנים

שאלה 1 (25 נקודות)

בכל אחד מהסעיפים הבאים נתונה רשימה של מחלקות סיבוכיות, ממוינות לפי סדר אלפבתי. לכל אחת מהרשימות (בנפרד) ציירו שני גרפים:

- גרף ההכלות - גרף מכוון אשר צמתיו הם המחלקות, ויש בו קשת מ-C_1 ל-C_2 אם אתם יודעים בוודאות (ללא הנחות כלשהן) ש-$C_1 \subseteq C_2$. אין צורך לצייר קשתות הנובעות מקשתות אחרות.
- גרף אי-השיוויונים - גרף מכוון, אשר צמתיו הם המחלקות, ויש בו קשת מ-C_1 ל-C_2 אם אתם יודעים בוודאות שקיימות שפות ב- $C_1 \setminus C_2$. אין צורך לצייר קשתות הנובעות מצרוף של קשתות אחרות והכלות אותן סימנתם בגרף הקודם.

בשאלה זו אין צורך בהוכחות.

(1) 10% $AM, BPP, MA, NP, P^{\#P}, PH, PSPACE, coRP$.

פתרון:

גרף הכלות:

$$NP \subseteq MA$$
$$coRP \subseteq BPP \subseteq MA \subseteq AM \subseteq PH \subseteq P^{\#P} \subseteq PSPACE$$

גרף אי שוויונים: שום דבר לא ידוע.

(2) 15% $\text{lu-}AC^0,\ AC^1,\ DL,\ \text{lu-}NC^1,\ NL,\ P,\ PL \triangleq DSPACE(\log^{O(1)}(n)), coRL$
(כאשר lu-C הוא קיצור של logspace-uniform-C).

פתרון:

גרף הכלות:

$$lu - AC^0 \subseteq lu - AC^1 \subseteq DL \subseteq coRL \subseteq NL \subseteq PL$$
$$NL \subseteq P$$
$$NL \subseteq AC^1$$

גרף אי שוויונים:

$NL \leftarrow PL$ (זאת מכיוון ש $NL \subseteq DSPACE\left(\log^2(n)\right)$ ו $DSPACE\left(\log^2(n)\right) \subset PL$ ע"פ משפט ההיררכיה).

$lu - AC^0 \leftarrow lu - NC^1$ (בגלל xor).

$PL, P, NL \leftarrow AC^1$ (כי AC^1 מכיל שפות שאינן ב RE).

שאלה 3 (33 נקודות)

מ"ט לבדיקת אוב לשפה L היא מכונת-אוב מטילת מטבעות פולינומית, M, כך שלכל אוב B ולכל קלט x מתקיים:

- אם $B = L$ (כלומר האוב "תקין" לחלוטין) אז M^B מקבלת את x בהסתברות 1.
- אם $B(x) \neq L(x)$ (כלומר האוב B טועה על השאלה x), אז בהסתברות לפחות $2/3$ המכונה M^B דוחה את x.

(1) 5% הוכיחו שלכל שפה $L \in P$ קיימת מ"ט לבדיקת אוב.

$L \in P$ ולכן קיימת לה מ"ט דטר' פולינומית M כך ש $L(M) = L$.

נראה מ"ט M'^B עם אוב B מתאימה:

M'^B על קלט x:

שואלת את B על x ומריצה את M על x. אם האוב עונה כמוה המכונה, אז מקבלת, ואחרת דוחה.

סיבוכיות: מיידי.

נכונות:

אם $B = L$ אז M ו B עונות תמיד אותו הדבר ולכן M' תקבל.

אם $B(x) \neq L(x)$ אז M ו B עונות אחרת ולכן M' תדחה.

(2) 8% הוכיחו שלכל שפה $L \in ZPP$ קיימת מ"ט לבדיקת אוב.

$L \in ZPP$ ולכן קיימת לה מ"ט מ"מ פולינומית M אשר בהסתברות קטנה משליש עונה "לא יודעת".

נראה מ"ט M'^B עם אוב B מתאימה:

שואלת את האוב על x ומריצה את M על x.

אם M קיבלה או דחתה את x והאוב ענה כמוה אז מקבלת.

אם M קיבלה או דחתה והאוב ענה אחרת, אז דוחה.

אם M ענתה "לא יודע" אז מקבלת.

סיבוכיות: מיידי.

נכונות:

אם $B = L$ אז או ש M ו B עונות אותו הדבר או ש M עונה שהיא לא יודעת ולכן M' תקבל תמיד.

אם $B(x) \neq L(x)$ אז אם M לא ענתה "לא יודעת" אז נדחה כי התוצאות יהיו שונות, ואחרת נקבל, אולם ההסתברות לכך קטנה משליש. כלומר ההסתברות לדחייה היא לפחות שני שליש.

(3) 20% הוכיחו כי לשפה $TQBF$ קיימת מ"ט לבדיקת אוב.

נשתמש בווריאציה על פרוטוקול ההוכחה ל $TQBF$.

M תפעל כדלקמן:
בהינתן קלט x, M תשאל את B על x. אם B ענתה "כן", אז M תבצע פרוטוקול הוכחה אינטראקטיבי על מנת להוכיח לעצמה ש $x \in L = TQBF$ (פרטים בהמשך).
אם B ענתה "לא", אז M תבצע פרוטוקול הוכחה אינטראקטיבי על מנת להוכיח לעצמה ש $x \in \bar{L}$ (גם $\bar{L}$ שפה ב $PSPACE$ כי $PSPACE$ סגורה למשלים).
פרוטוקול ההוכחה יהיה זה שראינו בהרצאה ל $TQBF$.
בכל שלב המוכיח שולח למוודא פולינום והמוודא מבצע בדיקות.
כל שהמוכיח צריך לדעת על מנת לשלוח את תשובותיו הוא הקלט, ובחירות המוודא.
המוכיח בהוכחה פעל ב $PSAPCE$ ולכן השפה:

$$L = \{(x, z_1, z_2, ..., z_k, p_k) \mid \text{בהוכחה } k \text{ הפולינומים ששולח המוכיח בשלב ה } p_k \text{, בחירות המוודא, } z_1, ..., z_k \text{ קלט, } x\}$$

היא שפה ב $PSPACE$. בפרט, קיימת לה רדוקציה ל $TQBF$.
מכאן שגם $\{y$ רישא של מילה ב L באורך $y^{1+k}\}$ $\tilde{L} = \{y\$^k \mid$ הינה ב $PSPACE$.
פרוטוקול ההוכחה יהיה כדלקמן:
בכל שלב, המוודא יבקש את הפולינום ע"י גילוי המילה המתאימה ב L בעזרת שאלות על $\tilde{L}$ (דרך $TQBF$) עד למציאת המילה הבאה.

אם $B = L$, אזי משלמות הפרוטוקול, המוכיח יצליח לשכנע את המוודא בהסתברות 1.
אחרת, עבור $B(x) \neq L(x)$, המוודא ישתכנע בהסתברות נמוכה.

שאלה 4 (33 נקודות)

מעגל אי-דטרמיניסטי C הוא מעגל בו המשתנים מתחלקים לשתי קבוצות: משתני קלט, המסומנים $x_1, \ldots, x_n$, ומשתנים אי-דטרמיניסטיים המסומנים $y_1, \ldots, y_m$. ערך המעגל מוגדר באופן הבא: $C(x) = 1$ אם קיימת השמה $y \in \{0,1\}^m$ למשתנים האי"ד, כך שערך המעגל תחת ההשמה xy הוא 1 (אחרת $C(x) = 0$). נסמן ב-$NPSC$ את מחלקת השפות אשר קיימת עבורן סדרת מעגלים א"ד (לא אחידים) בגודל פולינומי.

המחלקה $NP/poly$ מוגדרת באופן הבא: $L \in NP/poly$ אם קיימת מ"ט א"ד פולינומית M וסדרת "עצות" a_n באורך פולינומי ב-n, כך שלכל x מתקיים $x \in L$ אם"ם M מקבלת את הקלט $(x, a_{|x|})$.

(1) 5% הוכיחו או הפריכו: $NPSC \subseteq PSPACE$.

הפרכה: $P/poly \subseteq NPSC$ וראינו ש $P/poly$ מכילה שפות שאינן כריעות.

(2) 15% הוכיחו כי $NPSC = NP/poly$.

כיוון ראשון: $NPSC \subseteq NP/poly$:
נשתמש במעגל בתור העצה.

כיוון שני: $NPSC \supseteq NP/poly$:
העצות מקודדות בתוך המעגל ואי הדטרמיניזם מקודד בתוך ה y -ים.

(3) 13% הוכיחו כי $AM \subseteq NP/poly$.

תזכורת: $AM = BP(\exists(P))$ (כאשר AM היא מחלקת השפות הניתנות להוכחה ע"י משחק בעל סיבוב ארתור-מרלין אחד).

וריאציה על $BPP \subseteq P/Poly$.

<u>סיכום נושאי הקורס:</u>

- מחלקות זיכרון וזמן
- מ"ט עם אוב + היררכיה פולינומית
- חישוב הסתברותי
- פרוטוקולים ומחלקות אינטראקטיביות
- מעגלים
- PCP
- מחלקות ספירה

<u>סוגי רדוקציות:</u>

$\leq_{Log-Space}$, $\leq_P$ - הרדוקציות משתמשים בגרף הקונפיגורציה ובטבלת החישוב.

$\leq^T$ - רדוקציה באמצעות אוב.

$\leq_{rand}$ - ראינו רדוקציה רנדומית מ SAT ל $Uniq-SAT$. משתמשים במשפטי מפתח בהסתברות: צ'רנוף, מרקוב,... פונקצית $hash$, ארתמיזציה.

$\leq_{AC^0}$ - רדוקציה עבור מעגלים