חישוביות - סיכום הרצאות

מרצה ד"ר ענת פסקין צ'רניאבסקי

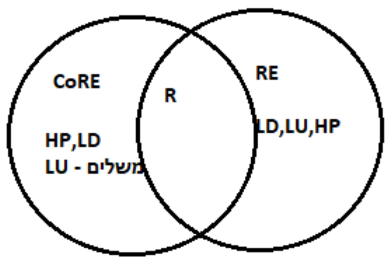
24 בינואר 2020

סיכום זה מבוסס על הרצאותיה של ענת בימי חמישי, יש לקחת בערבון מוגבל.

כמו כן יש לא מעט טעויות הקלדה, אשמח לשמוע הערות / תיקונים

בהצלחה לכולנו!!

נעם דומוביץ



תוכן עניינים

להבנתי ענת לא השלימה כמה הוכחות בסוף:

- $SAT \leq_p SAT_3$ - חלקי
- $VC \in NPC$ - חסרה הרדיקוציה מ $3SAT \leq VC$
- משפט $cook - levin$ הסבירה את רוב ההוכחה בציורים

0.1 שיעור 1

- הקורס הוא קורס שלישי בסדרת הקורסים החוקרים את הכח ("מה ניתן לחשב?") של מודלים חישוביים כאן הגענו לדיון על מחשב כללי (הכי רלוונטי ל"עולם האמיתי")
- הקורס הוא מתמטי־תאורטי באופיו.
- הציון: 80% בחינה, 20% מטלות (5% למטלה) - הגשה בזוגות.
- ספר הקורס: Michel Sipser, introduction to the Thoery of Compatition (סימונים מעט שונים)
  - סיכומי הרצאות מהטכניון (מעט $typos$)
- שעות קבלה: יום ג' 11 , ב 56.0.49 מייל: $anatpc@ariel.ac.il$

איפה אנחנו עומדים בהבנת מודלים חישוביים?

- קורס 1 - אוט' 1 - דן במודל של DFA - מחשב עם זכרון סופי, הראו שהוא תופס שפות רגולריות.
  - ההצדקה ללמוד אותו: פשוט ידוע עליו הרבה (גם שימושי במידה מסוימת)
  - ראינו שאי־דטרמינזם לא עוזר להגדיל את כח החישוב
- קורס 2 - אוט' 2 - דן במודל של אוטומט מחסנית א"ד - הראנו שהוא תופס שפות חסרות הקשר.
  - הצדקה ללמוד אותו: עדיין פשוט, ויודעים להוכיח לא מעט, וגם שימושי בשפות תכנות (דח"ה מתארים את רב שפות התכנות)
- קורס 3 - תורת החישוביות - נדון במודל של מחשב כללי. ננסה להבין את שתי השאלות הבסיסיות הבאות:
  1. מה ניתן חשב ע"י מחשב? (אילו בעיות) - חלק ראשון של הקורס
  2. מבין הבעיות שניתן לפתור, אלו בעיות ניתן לפתור ביעילות? - כאן ידוע הרבה פחות. יש שאלות גדולות. עדיין יש כאן תיאורה מועילה.

כדי לענות על 1. נצטרך קודם להגדיר במדויק (פורמלית) מהו מחשב. מה נרצה מהגדרה כזו?
ההגדרה צריכה לקיים:

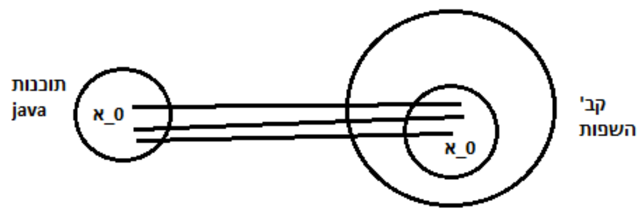
- להיות תואמת להבנה של חישוב אלגוריתמים

  בפרט צריך לאפשר לפתור כל בעיה שאנחנו אינטואיטיבית יודעים לכתוב לה קוד/אלגוריתם
  - לדוגמה $st - connectivty$ (האם קיים מסלול בין קודקודים s, t בגרף) - פותרים ע" BFS
- להיות פשוטה ופורמלית
- להיות חזקה - במובן שכל מודל סביר שניתן לממש (מבחינת חוקי הפיזיקה) לא יכול לפתור בעיה שהמודל שלנו לא יכול לפתור.

כרגע נעבוד עם מודל לא פורמלי שתופס את המבנה שלנו עד כח של חישוב כללי - מחשב (נגיד של $laptop$) שמריץ $java$, הבעיה עם מודל כזה היא שבפועל הזכרון של כל מחשב אמיתי הוא סופי, ולכן תיאורטית מודל כזה הוא DFA. לכן נניח שהזכרון שלנו **אין סופי**. זו הנחה סבירה במידה מסוימת, כי אם הזכרון נגמר באמצע החישוב אפשר תיאורטית לקנות עוד זכרון. לעומת זאת תוכנות אינסופיות זה לא סביר - כיצד צריך לכתוב את הקוד מראש. למעשה עבדנו עם זכרון (מחסנית) אינסופי כבר באוט' 2 .

כעת אשר לענות על השאלה האם קיימות בעיות (נחשוב על שפות מעל $\sum^*$) שלא ניתן לפתור ע"י מחשב?

כן. ניתן להראות "בקלות" משקולי ספירה: תוכנות נתנות לתיאור ע"י מחרוזות סופיות מעל א"ב סופי $\sum$, כלומר קב' התוכנות היא תת קב' של $\sum^*$, שהיא בת - מניה (עוצמה, $\aleph_0$ קב' השפות מעל $\sum^*$ היא בעצם $P\left(\sum^*\right)$ שעצמתה גדולה מ $\aleph_0$, ממשפט קנטור. בגלל שלכל תוכנה יש לכל היותר שפה אחת שהיא מקבלת (הערה: תוכנה כמו $while(true)$ לא מקבלת אף קלט), "רב" השפות ישארו בלי תוכנה שמחשבת אותן. זה נסיון לכל מודל חישובי "סביר" (בפרט שהתוכנות הן באורך סופי)



עדיין, הטיעון לא מספק לנו דוגמאות לבעיות לא פתירות ע"י מחשב. בהמשך הקורס נראה בעיות כאלה ונפתח טכניקות להוכיח שבעיה נתונה אכן לא פתירה:

ספוילר: נשתמש בהרחבה של לכסון (שראיתם שמוש בו במשפט קנטור), דוגמאות (ללא הוכחה כרגע) לשפות שלא נתנות לקבלה ע"י מחשב:

נדגיש: $\sum$ סופית ונדבר רק על שפות של מחרוזות סופיות מעל $\sum$ כמו באוט' .

1. בעיית העצירה: בהנתן קלט $(\langle code\rangle, \langle x\rangle)$ קלט לקוד צריך להחליט האם התוכנות המתוארות ע"י $\langle code\rangle$ עוצרת בריצתה על x או לא.

   אינטואיטבית, הבעיה היא שאם ננסה להריץ את הקוד על x פקודה-פקודה, אם הוא עוצר, גם אנחנו נעצור ונגיד "עוצר" - זה טוב. אבל אם הקוד לא עוצר אז גם ההרצה שלנו לא תעצור ואלג' שלנו לא יעצור (במקום להגיד "לא עוצר").

   זו לא הוכחה כי אולי יש אלג' יותר טוב שכן פותר את הבעיה. בהמשך נוכיח שאכן לא קיים אלג' כזה.

2. שקילות של תוכנות: בהנתן שתי תוכנות מחשב (למשל $java$) $\langle code1\rangle, \langle code2\rangle$ האם הן מחשבות את אותה השפה?

   (א) בעיית השקילות של שני אוט' מחסנית א"ד לא נתנת לחישוב במחשב

   הערה: בעית השקילות של DFA-ים היא כן ניתן לפתרון ע"י מחשב (בגדול מסתכם בהסתמכות על הגרף של אוט' מסויים שנקבע ע"י שני האוטומטים, ובדיקת $s-t$ קשירות בין q_0 ל q_f)

3. בהנתן קב' סופית של מטריצות ריבועיות מעל השלמים $\{A_1, ...A_n\}$ האם קיימת מכפלה סופית שלהם הנותנת את מטריצת האפס? כלומר מכפלה מהצורה $A_1^7A_2^8A_5^{14}$..... (כלומר ניתן להכפיל כל מטריצה יותר מפעם אחת)

נעבור להגדרה של המודל המדוייק שנעבוד איתו בקורס - **מכונת טיורינג.**

הוצע ע"י $Alan\ Turing$ בסביבות 1940 זה אכן מודל פורמלי ומאוד פשוט לתיאור (ולניתוח) של מחשב. זה המודל שאומץ ע"י רב החוקרים במדעי המחשב התאורטיים וגם ברב האוניברסיטאות באולם בקורס חישוביות. קיימים מודלים שקולים שגם חוקרים (לפעמים), למשל $\lambda - calculus$ (הוצע בסביבות 1930) שהוא הבסיס לשפות תכנות פונקציוליות. מודל שקול אחר (נראה בתרגול 2) הוא מודל RAM - בדיוק הרחבה של מחשב "רגיל" שמריץ גרסה של אבסמלי עם זכרון אינסופי. זו עדות מסויימת לכך שמכונת טיורינג היא חזקה במובן שרצינו. אכן כבר טיורינג שיער שזה המצב:

תיזת צרץ' טיורינג: כל מחשב "סביר" (למימוש והרצה ב"עולם הפיסקלי שלנו") לא מסוגל לפתור בעיה שמכונת טירוינג לא מסוגלת לפתור.

זו מעין "הנחת עבודה" זה לא משפט כי למשל "סביר" לא מוגדר היטב. גם אם היינו מגדירים הכל במדויק, לא ברור (לאף אחד) כיצד להוכיח אבל כאמור, יש עדויות לא רעות לכך. מצד שני, בחישוב יעיל נראה שתזת צ'רץ־טיורינג (לחישוב יעיל) לא עובדת

־עדין אין הוכחה אבל יש עדויות. ספציפית המודל של חישוב קוונטי נראה יותר חזק מבחינת חישוב יעיל (זמן פטינומי) למשל ידוע אלגוריתם קוונטי יעיל לפירוק מספרים לא ידוע אלג' קלאסי ומאמינים שאין) שכיוון שמתסמכים על הנחות קשיי חישוב של פירוק מס' בקריפטוגרפיה למשל, עובדים על לבסס הצפנות על הנחות קושי אחרות, שקשות גם למחשב קוונטי (בתקווה).

מכונת טיורינג ־ תאור (כמעט) פורמלי

מכונת טיוריג מוגדרת ע"י שביעיה:

- Q ־ קב' סופית של מצבים
- q_0 מצב התחלתי
- $\sum$ א"ב הקלט
- Γ א"ב העבודה
- $\flat$ מצבים סופיים
- F קב' של מצבים סופיים (תפקיד שונה מאוטומט מחסנית(
- δ פונקציית מעברים

$\flat$־ים (מסמן לנו בסרט האינסופי לימין את סוף הקלט)

נזכר שבאוטומט מחסנית לא היה צריך את זה כי בכל רגע דרשנו לתת את הפלט הנכון עבור הרישא שקראנו.

<u>מהלך החישוב:</u> מוגדר ע"י δ. בכל צעד מבצעים "מהלך" לפי δ . δ מוגדרת כך $\delta\left(Q\backslash F\right)\times\Gamma\rightarrow Q\times\Gamma\times\left\{ L,S,R\right\}$

- Q מצב נוכחי
- Γ ־ תו המוצבע ע"י הראש
- $Q\times\Gamma$ = תו שכותבים במקום המוצבע ע"י הראש $\times$ מצב חדש
- לאן הראש זז בסוף הצעד: $(Left,Stay,Right)$ ־ $left$ קורה בסוף הצעד.

<u>סיום החישוב:</u> אם מגיעים למצב השייך ל F החישוב מיד עוצר והפלט שלו (על הקלט $\times$ שלו) מוגדר כתוכן הסרט משמאל לראש.

אם לא עוצרים אף פעם , אז הפלט על אותו קלט לא מוגדר.

כלומר הפונקציה שמ"ט M מחשבת , $\delta_{M}:\sum^{*}\rightarrow\Gamma^{*}$ היא פונקציה חלקית (מותר ש δ_{M} לא תהיה מוגדרת על קלטים מסוימים)

דוגמה כיצד נפלוט ε :

$$\downarrow$$

$$*\quad..\quad..$$

נעצור כאשר הראש בתו הראשון

כדי לפלוט את $11\flat$ צריך לעצור בתא ה־4 ותוכן הסרט יהיה:

$$1\quad1\quad\flat\quad*\quad..$$

דוגמה: נחשב את הפונקציה המוגדרת כך: $\begin{Bmatrix}\delta:\left\{ 0,1\right\} ^{*}\rightarrow\left\{ 0,1\right\} ^{*}\\ f(x)=0x \end{Bmatrix}$

רעיון הבניה: נכתוב 0 בהתחלה ונזוז ימינה כאשר את התו שדרסנו, נזכור באמצעות מצב:

פורמלית: $M=\left(Q=\left\{ q_{0},q_{1},q_{f}\right\} ,\sum=\left\{ 0,1\right\} ,\Gamma=\left\{ 0,1,\flat\right\} ,\flat,F=\left\{ q_{f}\right\} ,\delta,q_{0}\right)$

q_0 ־ זוכר $0+$ מצב התחלתי , q_1 זוכר 1 , q_f ־ מצב סופי

$Q,F\backslash\delta$	0	1	$\flat$
q_0	$(q_0,0,R)$	$(q_1,0,R)$	$(q_f,0,R)$
q_1	$(q_0,1,R)$	$(q_1,1,R)$	$(q_f,1,R)$

0.2 הרצאה 2

בשעור הקודם הגדרנו מודל של מחשב שנעבוד איתו במלך הסמסטר . ההגדרה היתה יחסית פשוטה ופורמלית בנוסף קיווינו שמודל הזה במובן שכל מודל סביר לא מאפשר לחשב פונק' שלא ניתנות לחישוב במודל שלנו של מכונת טירוניג. (תזת צרץ' טירוינג). התיזה היא מעין "הנחת עבודה" שאין לנו הוכחה עבורה ,אבל ישנן עדויות די משכנעות לנכונות שלה. כל מודל חישוב שהוצע אי פעם (בעיקר מאז 1930) לא היה חזק מממ"ט:

- מודל RAM (עם זכרון אינסופי) - תראו בתרגול שהוא שקול
- $cakukus$ (עליו מבוססות שפתות תכנות פונקציונליות כמו ml , $haskell$) - כנראה יותר חזק אם מגבילים לחישוב יעיל (זמן פולינומי) של פונקציות.

בשעור היום נדון יותר לעומק בשקילות מודלים. לפני כן נשלים את הדוגמה של מימוש הפונק' $f(x)=xx$ באמצעות מכונת טירונג

<u>דוגמה</u>: נתבונן בפונקציה $f:\sum^*\to\sum^*$ המוגדרת ע"י $f(x)=xx$ (לכל $x\in\sum^*$)
רעיון הבניה: נלך הלוך־חזור, ובכל פעם נוסיף בצד ימין (במקום ה $\flat$) הראשון את התו הבא להעתקה)

$$\begin{array}{|c|l}\hline x & \flat \ \dots \ \flat \\ \hline\end{array}$$
$$\downarrow$$
$$\begin{array}{|c|c|l}\hline x_1 \ x_2' \ \dots \ x_n' & x_1 & \flat \ \dots \\ \hline\end{array}$$
$$\downarrow$$
$$\begin{array}{|c|c|l}\hline x_1 \ x_2 \ x_3' \ \dots \ x_n' & x_1'' \ x_2'' & \flat \ \dots \\ \hline\end{array}$$

נשים '־ים על האותיות להעתקה ו " על האותיות ימין בכל פעם נוריד את ה ' מהאות שכבר מטופלת וככה נדע למצוא את האות הבאה להעתקה. (הערה: כמו תמיד זה מימוש אחד, ויתכנו שיפורים/מימושים אחרים)
נעבור למימוש הרעיון. נבנה מכונה $M=(Q,q_{0,},\sum\Gamma,F,\flat,\delta)$

$\Gamma=\sum'\cup\sum''\cup\{\flat\}$ כאשר

$\sum'=\{\sigma'|\sigma\in\sum\}$ ו - $\sum''=\{\sigma''|\sigma\in\sum\}$ - נניח בה"כ ש$\sum$ לא מכילה גרסאות מהצורה a או a'' של תויים $a\in\sum$

$Q=\{q_0,q_1,q_2,q_f\}\cup\{q_\sigma|\sigma'\in\sum\}$ כמו כן

q_0 מצב התחלתי וגם אחראי על גילוי התו הבא לכתיבה

$q_f,$ אחראי על הליכה ימינה עד להעתקת האות הבאה להעתיק (שאותה q_0 זוכר) + תג ראשון

$F=\{q_f\}$

<u>δ:</u>

$Q,F\backslash\delta$	$a\in\sum$	$a'\in\sum$	$a''\in\sum$	$\flat$
q_0	(q_a,a,R)[1]	(q_a,a,R)	(q_2,a'',S)[4]	$(q_f,\flat,S)$
$(b\in\sum)\ q_b$	(q_b,a',R)[2]	(q_b,a',R)	(q_b,a'',R)	(q,b'',L)
q_1	(q_0,a,R)[3]	(q_1,a',L)	(q_1,a'',L)	dont care[6]
q_2	dont care	dont care	(q_2,a,R)	$(q_f,\flat,S)$

1 - גילוי אות לכתיבה בפעם הראשונה . בפרט לא נשים עליה ' כי היא כבר בתהליך העתקה.

2 - זו בהכרח הפעם הראשונה שמעתיקים ולכן צריך לתייג את שאר הקלט

3 - גילינו את האות הכי ימנית שכבר טופלה, נעבור ימינה כדי לאפשר לq_0 לטפל באות הבאה לטיפול

4 - מעביר שליטה למצב המנקה "ים לקראת סיום

5 - δ : רק אם $x=\varepsilon$ ואז נרצה מיד לפלוט ε

6 - צריך לכתוב משהו לנכונות סינטקטית

שקילות מודלים :

במודל של מכונת טיורינג שהגדרנו עשינו הרבה החלטות שרירותיות (כמו תמיד), שאפשר יהיה לבנות ולקבל מודל בעל אותו כח חישובי (שקול). לדוגמה: אפשרי היה לותר על ה S ועדיין לקבל מודל שקול.

כיצד נוכיח שקילות מודלים?

כמו שעשינו באוטומטים, בצורה קונסטריקטיבית. כלומר נראה שלכל M במודל I , קיימת M' המחשבים את אותה פונק' במודל II , וכן לכל M במודל II קיימת M' המחשבת את אותה פונק' ב I , בד"כ הדרך לבנות M'שקולה כזו היא לסמלץ את M צעד־צעד, ובסוף כל צעד "מסומלץ" נהיה בקונפיגורציה במצב כללי המחקה את הקונפיגורציה של M בתום אותו צעד.

לדוגמה, עבור המודל M'_{TM} לא S , נראה כך:

<u>כיון קל:</u> תהי M מ"ט במודל M'_{TM}, ונבנה מ"ט שקולה במודל M_{TM} , $M'=M$, כאן M במודל החדש היא תמיד מ"ט חוקית במודל המקורי ולכן ברור ש $M'=M$ שקולה ל M

<u>כיוון יותר מענין:</u> תהי M מ"ט רגילה ונבנה M' שקולה במודל M'_{TM} . רעיון הבניה הוא להגדיר M' הזהה ל M , פרט לכניסות מהצורה $\delta(q,a)=(p,b,S)$. אותן נבנה לקב' שורות δ' שיראה כך:

$$\delta'(q,a)=(P_R,b,R)$$
$$\forall c\in\Gamma\ \delta'(P_R,C)=(P,c,L)$$

כלומר נזוז ימינה ואז שמאלה.

נשים לב שאם היינו הולכים קודם שמאלה ואז ימינה היינו עלולים לקבל סימולציה לא נכונה במקרה של S בקצה השמאלי של הסרט.

- בצעד הראשון M' היתה נשארת במקום לפול מהסרט
- בצעד השני היתה זזה ימינה ומוצאת את עצמה באחד ימינה מהמקום שהתכוונו

נעבור להגדרה פורמלית של מודל חישובי.

<u>הגדרה:</u> מודל חישובי הוא מיפוי

$\mathbb{M}:\sum^*\to\delta:\sum_1^*\to\sum_2^*$ קב' הפונקציות (חלקיות)

בה"כ נניח תמיד ש $\sum=\{0,1\}$

ייצוג של "תוכנית" במודל בתור מחרוזת (סופית)

זו הגדרה כללית מאוד של מודל חישובי שמגדירה התאמה בין תוכנות במודל לפונקציות שהן מחשבות. הגדרת מודל לא מציבה כל הגבלה על צורת ההתאמה. אבל בד"כ ההתאמה תהיה באמצעות תיאור מדוייק כיצד "מריצים" את התכונה.

0.3 שיעור 3 ־ יום חמישי $14/11/19$

<u>תכנית להיום:</u>

- נסיים הוכחות שקילות של מודל מ"ט דו־סרטית למ"ט רגילה (כהבנה, נתן הגדרה פורמלית של מ"ט ומספר מודגים שימושיים)
- נדבר על מ"ט אוניברסלית כזו שמקבלת מ"ט אחרת $\langle M\rangle$ שעבורה, מריצה את M על x ואם עצרה, מחזירה את הפלט $f(x)$
- נגדיר מחלקות של שפות, לפי "קושי חישובי " שנתעניין בהן במהלך הקורס: R, RE

שקילות של מ"ט דו־סרטית למודל הרגיל:

<u>סקיצת הוכחה:</u>

כיון קל בהנתן מ"ט חד־סרטית M, נבנה מ"ט M' שקולה במודל הדו־סרטי. אינטואיטיבית, מ"ט חד־סרטית היא מקרה פרטי של דו־סרטית, והיינו רוצים לקחת $M' = M$.

פורמלית זה לא נכון (למשל לפונקציה δ בשני המודלים יש תחום תחום ותווך שונה, או במילים אחרות במ"ט דו־סרטית וחייבים לציין מה עושים בשני הסרטים). לכן נקח מימוש של M' שבו בסרט ה I נפעל בדיוק כמו M ובסרט ה II לא נעשה כלום.

כלומר לכל שורה $\delta\left(q,a\right)=\left(p,\flat,m\right)$ (ב M) נוסיף ל δ' : $\forall c\in\Gamma\ \delta'\left(q,a,c\right)=\left(p,\flat,\smiley,m,s\right)$ ($\smiley,S$ לא באמת חשוב)

<u>תזכורת ־ מ"ט דו סרטית:</u>

$$\delta : Q\backslash F\times\Gamma\times\Gamma\to Q\times\Gamma\times\Gamma\times\{L,R,S\}$$

- $Q\backslash F$ ־ מצב משותף
- Γ ־ תו בסרט ב I
- Γ ־ תו בסרט הII
- $\Gamma\times\Gamma$ ־ מה כותבים
- $\{L,R,S\}$ ־ לאן זזים
- בסקיצה ־ אתחול יראה כך:

$\downarrow$

x	$\flat$

$\downarrow$

$\flat$

- פלט: תכן הסרט ה I משמאל לראש ברגע העצירה.

לפני שנוכיח את הכוון המעניין, נעשה אתנחתא ונגדיר מספר מושגים שימושיים (לכל אורך הקורס)

<u>הגדרה 3.1 :(קונפיגורציה)</u>: בהנתן מ"ט M , קונפיגורציה (מצב רגעי) של M היא שלשה $C=(q,i,\alpha)$ המתארת (באופן מלא) את המצב הכולל של מכונה אחרי מספר (סופי) של צעדי ריצה:

- כאן q הוא המצב הנוכחי.
- i ־ התא שבו נמצא הראש הקורא/כותב.

- α ־ תוכן הסרט, כיוון שמדובר בריצה מס' סופי של צעדים על קלט כלשהו x מספיק לשמור רישא סופית של α

כמוסכמה נשמור $\varphi = max\left(|x|, T\right)$ תוים, כאשר T הוא התו הימני ביותר שבו M ביקרה. אם, בצורה כזו מבוטח שמימין ל $\alpha\left[x\right]$ יש רק $\flat$ ים ואין צורך לשמור אתם במפורש.

<u>הגדרה 3.2 (צעד חישוב של מ"ט):</u> תהי M מ"ט ו C_1, C_2 קונפיגורציה שלה. נאמר ש C_2 עוקבת ל C_1 אם M יכולה לעבור מ C_1 ל C_2 בצעד חישוב יחיד. כלומר C_1, C_2 מהצורה:

$$C_1 = \left(q, i, \alpha\right), C_2 = \left(p, i', \beta\right)$$

כאשר: $\delta\left(q, \alpha\left[i\right]\right) = \left(p, b, m\right)$ וגם:

$$i' = \begin{cases} i & m = S \\ i+1 & m = R \\ max\left(i-1, 1\right) & m = L \end{cases}$$

max בשביל לטפל ב'נפילה' מהסרט

β היא מחרוזת באורך $|\alpha|$ בד"כ, ומקיימת $\beta\left[i\right] = b$, $\beta\left[j\right] = \alpha\left[j\right]$, $i \neq j, j \leq |a|$.

בנוסף (לא הבד"כ) אם I' גדול מ $max\left(|x|, T\right)$ (התו הימני ביותר שבקרנו) ־ אז β יהיה באורך $|a| + 1$ i' ו $\beta\left[i+1\right] = \flat$.

בעצם הגדרנו מחדש, בצורה מדויקת וקצרה כיצד מתבצע צעד חישוב של M , נסמן זאת כך: $C_1 \vdash C_2$

<u>הערה:</u> לכל קונפ' $C = \left(q, i, \alpha\right)$ של M שבה $q \notin F$ (כלומר קונפ' שאינה סופית) יש קונפ' עוקבת אחת ויחידה.

<u>הגדרה 3.3 (הפונקציה f_M שמ"ט מחשבת):</u> תהי M מ"ט. ויהיה $x \in \sum^*$ קלט עבורה. נסמן ב $C_x = \left(q_0, 1, x\right)$ את הקונפ' ההתחלתית של ריצת M על x (במקרה ו$x = \varepsilon$ אז $C_x = \left(q_0, 1, \flat\right)$) מתואר ע"י סדרת הקונפיגורציות (היחידה) מהצורה:

$$C_x = C_1, C_2, ...$$

כאשר לכל $i > 1$, $C_{i+1} \vdash_M C_i$, $f_M(x)$ תוגדר כך:

$$f_M(x) = \begin{cases} a\left[1, ..., i-1\right] & \text{A finite sequence with configartion of: } \left(q, i, \alpha\right) \\ \perp\text{(undifined)} & \text{else (infinite)} \end{cases}$$

נחזור להוכחה שלנו לגבי שקילות בין דו־סרטי לחד־סרטי בכוון המעניין.

כלומר בהנתן מ"ט דו־סרטית M נבנה מ"ט חד סרטית M' . M' תבצע סימולציה צעד־צעד של M , כך שבסוף סימולציה של צעד מסוים הקונפי' של M' "תקביל" לקונפ' של M . נציע פתרון מסוים (כמובן שקיימים פתרונות אחרים שעובדים):

<u>רעיון הפתרון:</u> נרצה לשמור על קונפ' של ריצת M בתור קנופ' של ריצה M' שלנו. בפרט נשמור את תכן הסרטים אחד אחר השני מופרדים ב \$ ים (ורק את החלק הרלוונטי שאחריו בוודאות יש $\flat$ים בדומה ליצוג של צעד α בקונפ')

$$\alpha_1 \$_1 \alpha_2 \$_2$$

$\alpha_1 \alpha_2$ ־ תוכן הסרט הראשון/השני בהתאמה

כיצד נשמור את מקומי הראשים? יהיה נח לסמן אתם ע"י # ליד התו בתא: לדוגמה

$$\downarrow$$

a	b	c	$\flat$	$\$_2$

ייצוג בתור $ab^{\#}c\flat\$_2$.

המצב q יישמור (איכשהו) חלק מהמצב שלנו. כדי לסמלץ צעד חישוב של M נבצע סריקה משמאל לימין. בדרך ימינה (עד ה # השניה) נאסוף את המידע על התווים המוצבעים ע"י הראשים. ובדרך שמאלה נעדכן את הקונפ' (את מקום הראש, את תוכן הסרט בתא המוצבע, ולבסוף את המצב)

- נתחיל מקופנ' שבה:

$$x_1^{\#}, x_2, ..., x_n, \$_1\flat^{\#}\$_2$$

נדאג לזה , נכתוב שגרה קצרה שהמצב ההתחלתי q_0 שלנו יבצע) נעביר שליטה ל q_0 של M .

- בדרך ימינה נאסוף את התוים ב # ים :

$$q_0 \underset{\text{first I}}{\Rightarrow} q_0^a \underset{\text{first II}}{\Rightarrow} q_0^{a,b}$$

(במעבר ה I $a = x_1$, $b = \flat$)

- במעבר שמאלה , # ית הראשנה שנתקל בה נעבור מ $q_0^{a,b}$ ל $\tilde{p}^{a,b}$ כאשר ב M

$$\delta\left(q,a,b\right) = \left(p,c,d,m_1,m_2\right)$$

ונזיז את # בהתאם ל m_2 ונכתוב (בתא עם ה # המקורית) את d . בפעם הבאה שנתקל ב # נעדכן לפי c_1m_1ונעבור למצב p .

- סקיצה:

$$q : \underset{\xleftarrow{\#\text{ update }\#}}{\xrightarrow{\#^a\text{data gathering }\#^b}}$$

הערה: אמנם מ"ט דו־סרטית שקולה למ"ט חד־סרטית, אך המודל הדו־סרטי נותן שיפור ביעילות. לדוגמה האלג' שראינו ל $f(x) = xx$ במ"ט רגילה דרש $\Omega\left(n^2\right)$ צעדים. אפשר להראות שזה אופטימלי, ואלו במ"ט ד־סרטית אפשר לחשב את $f(x)$ ב $O(n)$ צעדים.

0.3.1 הפונקציה האוניברסלית ומ"ט אוניברסלית

בקורס הרבה פעמים נרצה להריץ מ"ט על קלט מסויים (כאשר שניהם נתונים כקלט) ולראות מה יצא.

פורמלית הפנוקציונליות שרוצים לחשב כאן היא הפונ' האוניברסלית U המוגדרת כך:

$$U\left(\langle M\rangle, \langle X\rangle\right) = \begin{cases} f_M(X) & \text{valid syntax } \textbf{and} \text{ M stop on x} \\ \perp & \text{else} \end{cases}$$

M - קידוד של מ"ט , X - קידוק של קלט עבור M

מ"ט M מוגדרת נקראת מ"ט אוניברסלית אם $U = f_M$. (בפרט , על קלטים עליהם U אינה מוגדרת, נרצה ש M לא תעצור).

קימות מ"ט אוניברסליות.

למה צריך קידודים של M ו x ולא M, x ישירות?

- לגבי החלק M בקלט זה ברור, כי M שביעיה ונרצה לקודד אותה איכשהו בתור מחרוזת, כי מ"ט M_u (אם נצליח לבנות M_U כזו) מקבלת כקלט רק מחרוזות.

- אבל למה צריך לקודד את x ו$f_M(x)$? אלה כבר מחרוזות!

הבעיה היא שלכל מ"ט M בקלט יש $\sum$ משלה, ולכן $x \in \sum^*$ על פני כל הקלטים האפשרים, x שייך לקב' של מחרוזות מעל א"ב אינסופי ואילו ל M_U יש א"ב סופי. כנ"ל לגבי Γ והפלט. לכן M_u תקבל קידוד של הקלט בא"ב שלה $(\sum_u)$ ותפלוט קידוד של הפלט $\langle f_M(x) \rangle \in \Gamma_U^*$. נקבע בה"כ Γ_u קבוע

$\sum_U = \{0,1\}$. הקידודים יהיו מספיק פשוטים כדי שהמשתמש ידע לקודד את x "ולפענח" את $\langle f_M(x) \rangle$ מהא"ב שלו לא"ב של M_u בחזרה.

נקבע את הקידוד שנעבוד איתו: (באופן מדוייק: קידוד של מכונה הוא מיפוי ממכונות למחרוזות בינאריות. כנ"ל קידוד של קלט $x \in \sum^*$ הוא מיפוי מ $\sum^*$ למחרוזות בינריות $\sum_u^*$).

הקידוד של $x \in \sum^*$ כלשהו ייקבע כך: בה"כ נתייחס לתוים של $\sum$ בתור המספרים $1, 2, .. |\sum|$ לדוגמה נתייחס ל $X = abk$ בתור $1, 2, 11$. נקודד את $x = x_1, ..., x_n$

תו - תו בפורמט "אונארי" כלומר $x_1, ..., x_n$ יקודד כ

$$\overset{x_1}{1..1}0\overset{x_1}{1..1}0....0\overset{x_n}{1..1}0$$

מכונת טיורינג M תקודד קח: בשביל להקל על עצמנו נניח

$$M = \left(\underbrace{Q}_{[l]} \underbrace{\sum}_{[n]}, \underbrace{\Gamma}_{[m]_{m>n}}, \flat = m, q_0 = 1, F = \{2,3\}, \delta \right)$$

נקודד את M ע"י :

$$\overset{l}{1}0\overset{n}{1}0\overset{m}{1}0 \langle \delta \rangle \text{ - } \delta \text{ קידוד של}$$

נקודד את δ בגישה ה'אונארית" גם: בתור רשימת כנסות $\delta(q,a) = (p,b,d)$ יקודד בתור :

$$\overset{q}{1}0\overset{a}{1}0\overset{p}{1}0\overset{b}{1}0\overset{d}{1}00$$
$$\overset{q'}{1}0\overset{a'}{1}0\overset{p'}{1}0\overset{b'}{1}0\overset{d'}{1}0$$
$$\vdots$$

סה"כ $(l-2) \cdot m$ שורות בסדר לקס' של הכניסות, למשל $\langle M \rangle, \langle X \rangle$ יהיה שרשור של הקדודים של M ו X .

שאלה: מתי קלט $\langle M \rangle, \langle X \rangle$ לא תקין?

- למשל אם $x \in \sum^*$, כלומר יש בו תו שגודלו המספרי חורג מ l .
- דוגמה נוספת כאשר M לא חוקי למשל ב $\langle \delta \rangle$ חסרות כניסות.

כל אלה (ועוד) אפשר לבדוק ע"י M_u

כיצד נבנה M_U (אכן אפשר!) :

1. על קלט $\langle X \rangle, \langle M \rangle$, M_U תבדוק תקינות , ואם נכשל תכנס ללופ אינסופי (לא תעצור).
2. אחרת - תסמלץ את ריצת M על x צעד- צעד, כיצד?

לצורך נוחות נשתמש במ"ט דו־סרטית:

סרט 1 ישמור את הקונפ' הונכחית של x , סרט 2 ישמור את הקידוד של M ו $\langle M \rangle$.

מבנה הקנופ' יהיה דומה למה שראינו בסימולציה של מ"ט דו־סרטת לדוגמה:

$$\overset{\alpha_1}{1}\overset{\alpha_2}{1}....*\overset{q}{1}\#\overset{\alpha_n}{1}*$$

כוכביות לציין את מקום הראש

כדי לבצע צעד נחפש מה לעשות ע"י חיפוש של $\overset{q}{10}\overset{\alpha_n}{1}$ בתוך $\langle\delta\rangle$ בסרט 2 .

0.3.2 מ"ט לקבלת שפות

מעכשיו והלאה נניח שהמ"ט שלנו היא בעלת שני מצבים סופיים $F=\{q_{acc,}q_{rej}\}$, ו $\sum=\{0,1\}$ אלא אם נאמר אחרת.

נוסיף למ"ט $feature$ של קבלת שפות

<u>הגדרה: 3.4</u> נאמר שמ"ט מקבלת מילה $x\in\sum^*$, אם M בריצתה על x עוצרת במצב q_{acc}

<u>הגדרה 3.5 (השפה של מ"ט)</u> בהנתן מ"ט M נגדיר:

$$L(M)=\left\{x\in\sum^*|\text{m accept x }\right\}$$

השפה של M

נאמר ש M מכריעה את $L(M)$ אם M תמיד עוצרת על כל קלט).

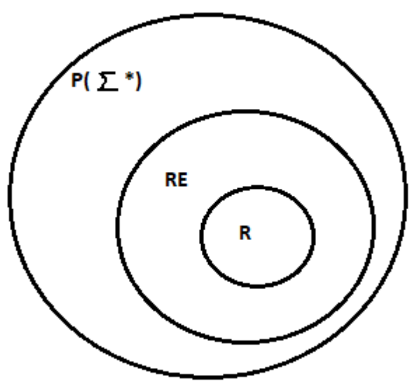
<u>הגדרה 3.6 :</u>

$$R\triangleq\left\{L\in P\left(\sum^*\right)|\text{exist Turing machine for L(M)=L }\right\}$$

<u>הגדרה 3.7:</u> קב' כל השפות שנתן לקבל ע"י מ"ט תקרא RE כלומר :

$$RE\triangleq\left\{L\in P\left(\sum^*\right)|\text{exist Turing machine M accepted L}\right\}$$

תמונת עולם



- R - שפות סופיות , שפות רגולרית , שפות ח"ה , קשיורת בגרפים, פירוק מספרים
- RE - יש דברים - "רב השפות" הן כאן משיקולי עוצמה

0.4 שיעור 4 - חמישי 21/11/19

היום נחקור יותר לעומק את המחלקות R,RE נתחיל בחקר של תכונות סגור

0.4.1 טענה 4.1 : R סגורה למשלים

הוכחה:

תהי $L\in R$. נראה $\sum \in R$ מהגדרת R , קיימת מ"ט M המכריעה את L . נבנה מ"ט M' עבור $\bar{L}$ באופן הבא:

- $M'(x)$ מריצה את M על x ועונה הפוך.
  - כלומר אם M עצרה ב q_{acc} , נעצור ב q_{rej}
  - ואם עצרה ב q_{rej} נעצור ב q_{acc}.

ניתן לעשות זאת בדומה לבניה של מ"ט אוניברסלית שראינו בהרצאה הקודמת. נשים לב שכאן M היא מ"ט קבועה ושם היתה חלק מהקלט M' פשוט את $\langle M\rangle$ (חלק מ δ , בעזרת מצבים, תכתוב את $\langle M\rangle$ לסרט בנוסף ל $\langle x\rangle$ בתחילת החישוב ,ואז תוכל לרוץ בדומה למכונה האוניברסלית).

נשים לב שכאן אפשר גם (פתרון אחר) להפוך בין q_{acc} ל q_{reg} כדי לקבל M' כלומר δ' תוגדר בדיוק כמו δ פרט לכך שכניסות מהצורה $\delta\left(q,a\right)=\left(q_{acc},b,m\right)$ ימופו לכניסות מהצורה $\delta'\left(q,a\right)=\left(q_{rej},b,m\right)$ ובאופן דומה q_{rej} יוחלף ב q_{acc}. אבל בד"כ לא יהיה אפשרי לבצע שינויי פשוט ברמת δ ונסתפק בפתרונות בסגנון המכונה האוניברסלית , שנתאר במילים נשאר להראות ש M'שבנינו (בפתרון הראשון) אכן מכריע את $\bar{L}$.

נבדוק את שני המקרים:

- עבור $M\colon x\in\bar{L}$ דוחה את x (כי $x\notin L$) . לכן בסימולציה של M על x ע"י M' , M' תעצור, תזהה דחיה ואז תענה הפוך, כלומר q_{acc} - כנדרש
- עבור $x\notin\bar{L}\Leftarrow x\in L\Leftarrow M$ מריצה על x תקבל $\Leftarrow M'$ תסיים את הרצת הסימלוציה על M על x , תזהה קבלה ותענה הפוך, כלומר q_{rej}, כנדרש . בפרט M' תמיד עוצרת, ולכן מכריעה את $\bar{L}$.

ההוכחה לא תעבוד כדי להוכיח סגירות של RE למשלים, והיום גם נוכיח שאינה סגורה למשלים, הבעיה היא שהיפוך בין q_{acc} ל q_{rej} ב M שמקבלת אבל לא בהכרח מכריעה את L ועבור מילים $x\notin L$ עליהן גם M' לא תעצור עליה, ולא נצליח להפוך את הסטטוס קבלה/ אי קבלה שהלן ו $L\left(M'\right)\neq L(\bar{M})$.

0.4.2 טענה 4.2: R סגורה לאיחוד.

הוכחה:

- יהיו $L_1,L_2\in R$. נרה ש $L_1\cup L_2\in R$ ע"י בניית מ"ט $M_{1,2}$ המכריעה את $L_1\cup L_2$.
- יהיו M_1,M_2 המכירעות את L_1,L_2 בהתאמה :

1. $M_{1,2}(x)$: מריצה את M_1 על x שומרת את התוצאה במשתנה $out_1=\left\{true_{q_{acc}},false_{q_{rej}}\right\}$
2. $M_{1,2}(x)$: מריצה את M_2 על x שומרת את התוצאה במשתנה $out_2=\left\{true_{q_{acc}},false_{q_{rej}}\right\}$
3. עוצרת ב q_{acc} אם $out_1\vee out_2=T$, אחרת עוצרת ב q_{rej}

- נראה שM_{12} מכריע את $L_1\cup L_2$

– אם $x\in L_1\cup L_2$: M_1 או M_2 תעצור ותקבל (או שתיהן) $\Leftarrow$ נגיע לשלב 3 וה OR שיחושב יהיה $T\Leftarrow M_{12}$ תקבל, כנדרש $\left(x\in L\left(M_{12}\right)\right)$

– אם $x\notin L_1\cup L_2\Leftarrow M_1$ ו M_2 יעצרו ב $q_{rej}\Leftarrow$ נגיע לשלב 3 הוא OR שיחושב $F\Leftarrow M_{12}$ תדחה את x, כנדרש

0.4.3 טענה 4.3 : RE סגורה לאיחוד

<u>הוכחה:</u>

ניסיון I : נבנה M_{12} כמו בהוכחה עבור R . הבעיה כאן היא במקרה ש $x \in L_2 \backslash L_1$. במקרה זה $x \in L_1 \cup L_2$, ולכן המכונה M_{12} צריכה לקבל את x (בפרט לעצור) ו M_{12} שבנינו תתקע בשלב 1 ולא תעצור לעולם.

נסיון II : (עובד) הרעיון הוא להריץ את M_1, M_2 (המכריעות את L_1, L_2 בתאמה ב"מקביל")

כיצד נעשה זאת? נריץ את המכונות על x לסירוגין . בצעד ראשון נריץ צעד 1 של M_1 , בצעד 2 צעד 1 של M_2 , בצעד 3, צעד שני של M_1 וכו'

<u>פורמלית:</u>

1. $M_{12}(x)$ תריץ את M_1, M_2 במקביל על x
2. אם אחת מהן עצרה ב q_{acc} , מיד נעצור ונקבל
3. אם שתיהן עצרו ב q_{rej} , נעצור ונדחה.

נשאר להוכיח את נכונות הבניה. $x \in L\left(M_1\right)$ או $x \in L\left(M_2\right)$ (מנכונות M_1, M_2)

- אם $x \in L_1 \cup L_2$ לפחות אחת מהמכונות עוצרת על x ב q_{acc} נסמן ב i את הצעד שבו היא עוצרת. מהבניה M_{12} תזהה עצירה זו בסימולציה של הצעד הנ"ל בצעד סימולציה $2i$ לכל היותר. במקרה זה היא גם תעצור ותקבל. גם אם המ"ט השניה עצרה לפני כן, זה לא ישנה את התוצאה ל q_{rej}, כי M_{12} לא עוצרת ב $q-rej$ אלא אם שתי המכונות עצרו ב q_{rej}
- אם $x \notin L_1 \cup L_2 \Leftarrow x \notin L_1 = L\left(M_1\right)$ וגם $x \notin L_2 = L\left(M_2\right) \Leftarrow$ אף אחת מהמכונות לא תעצור ב $q_{acc} \Leftarrow M_{12}$ לא תעצור ב q_{acc} (אלא תעצור ב q_{rej} אם שתיהן עוצרות או לא תעצור מש"ל

נשים לב שהגדרת RE קיימת "חולשה" אסמטרית: אומנם מכונה המקבלת שפה L אף פעם לא טועה, אבל היא יכולה לא לעצור (רק) אם $x \notin L$. (אם $x \in L$ חייבת לעצור ב q_{acc}) טבעי להסתכל על המקה האסימטרי בכיוון ההפוך. נגדיר מחלקה מתאימה:

0.4.4 הגדרה: $coRE = \left\{L \subseteq \sum^{*} \mid \bar{L} \in RE\right\}$

ניתן גם הגדרה שקולה שתופסת ישרות את האינטואציה שרצינו להעביר:

0.4.5 הגדרה II של $coRE$

נאמר $L \in P\left(\sum^{*}\right)$ שייכת לקבוצת המילים $CoRE$ אם קיימת מ"ט M שדוחה כל $x \in \bar{L}$, ועבור $x \in L$ מקבלת או לא עוצרת .

שימו לב ש $L(M)$ לא בכרח שווה ל L רק מוכלת ב L , נראה שההגדרות שקולות

0.4.6 טענה 4.6 : הגדרות 4.5 , 4.4 $CoRE$ שקולות

$4.5 \Rightarrow 4.4$:

תהי $L \in CoRE$ לפי הגדרה 4.4, נראה ש $L \in CoRE$ לפי הגדרה 4.5 .

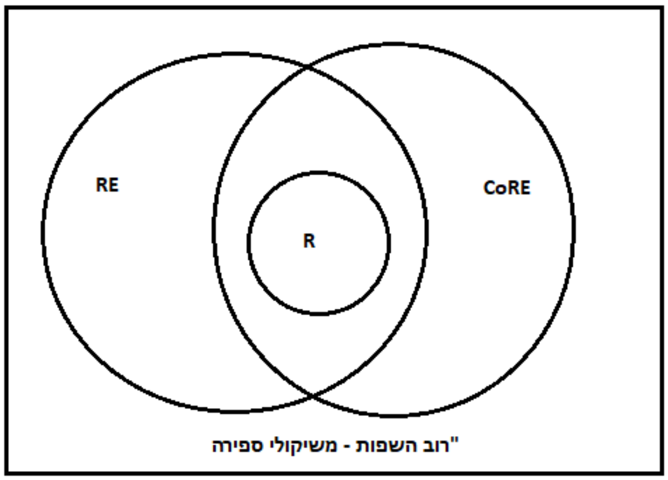
- <u>בניה:</u> תהי M מ"ט המקבלת את $\bar{L}$. נבנה $\bar{M}$ מתאימה עבור L שתספק את הגדרה 4.5 . $\bar{M}(x)$ - תריץ את M על x ותענה הפוך. (אם לא עוצרת אז לא תעצור)
- <u>נכונות:</u>
  - אם $x \in L \Leftarrow x \notin \bar{L}$, ולכן M או תעצור ב q_{rej} או לא תעצור $\Leftarrow \bar{M}$ או תעצור ב q_{acc} או לא תעצור, כנדרש
  - אם $x \notin L \Leftarrow x \in \bar{L}$, מהגדרת $RE \Leftarrow M$ בריצתה על x עוצרת ב q_{acc} , מהבניה $\Leftarrow$ ב q_{rej} , כנדרש.

4.5 $\Leftarrow$ 4.4 :

תהי $L \in CoRE$ לפי הגדרה 4.5, נראה ש $L \in CoRE$ לפי הגדרה 4.4 .

- בניה: תהי M מ"ט המובטחת עבור L נבנה מ"ט $\bar{M}$ המקבלת את $\bar{L}$, ונסיק ש $L \in CoRE$ לפי הגדרה 4.4 - $\bar{M}(x)$: מריצה את M על x ועונה הפוך
- נכונות:
  - אם $x \in L$ $\Leftarrow$ M תעצור ב q_{acc} או לא תעצור $\Leftarrow$ $\bar{M}$ תעצור ב q_{rej}או לא תעצור $\Leftarrow$ $x \notin L\left(\bar{M}\right)$, כנדרש.
  - אם $x \notin L$, מהגדרת 4.5 $\Leftarrow$ M עוצרת ב q_{rej} , מהבניה $\Leftarrow$ $\bar{M}$ עוצרת ב q_{acc} $\Leftarrow$ $x \in L\left(\bar{M}\right)$, כנדרש.

תמונת עולם



0.4.7 טענה: $R = RE \cap CoRE$

הוכחה: נראה הכלה דו כיוונית.

<u>כיוון קל - $R \subseteq RE \cap CoRE$</u>

זה מתקיים כי מ"ט M המכריעה את L ובפרט מקבלת את L ולכן $L \in RE$ וגם מקיימת את הדרישה של הגדרה 4.5 לשייכת ל $CoRE$ (אף פעם לא טועה ומותר לה לא לעצור רק אם $x \in L$ אבל במקרה שלנו היא תמיד תעצור, בפרט עבור כל $x \in \bar{L}$

<u>כיוון מעניין $RE \cap CoRE \subseteq R$</u>

תהי $L \in RE \cap CoRE$. לכן קיימת מ"ט M_1, M_2 .בהתאמה כך ש:

1. כל M תמיד צודקת לגבי שייכות של x ל L במידה ועוצרת
2. אם $x \in L$, M_1עוצרת ב q_{acc}
3. (מהגדרה 4.5) אם $x \notin L$, M_2 עוצרת ב q_{rej}

נבנה M שמכריעה את L באופן הבא:

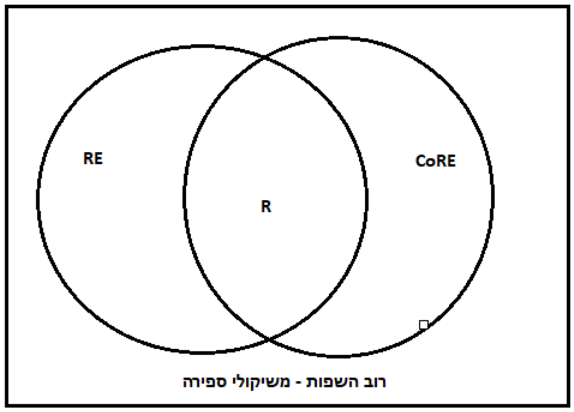
$M(x)$: מריצה את M_1 ו M_2 על x במקביל. (כמו שעשינו בהוכחה ש RE סגורות לאיחוד) ברגע שאחת מהן עצרה מיד עוצרת ועונה כמוה

נכונות הבניה:

אם $x \in L$ $\Leftarrow$ בהכרח , M עוצרת על x ב q_{acc} $\Leftarrow$ יתכן אחד משני מקרים:

1. M_1 היא הראשונה שעצרה בצעד $\geq$ מהצעד שבו M_2 עוצרת אם בכלל) ,אז מהבניה M תעצור ב q_{acc} , כמוה . כנדרש.
2. M_2 עוצרת ראשונה . אבל מתכונה 1 , היא גם תעצור ב q_{acc} ו M תענה כמוה, כנדרש.
3. $x \notin L \Leftarrow$ באופן דומה לטיעון במקרה הקודם, מובטח ש M_2 תעצור ב q_{rej} וגם אם M_1 עוצרת לפניה, היא תעצור ב q_{rej} ו M תעצור ב q_{rej} , כנדרש.

לכן, נוכל לעדכן את מפת העולם:



0.4.8 הגדרה: שפות הלכסון

$$L_D = \{\langle M\rangle |\text{M accept } \langle M\rangle\}$$

מוסכמה:

לשם פשטות, נשנה את הקידוד של מ"ט שראינו בהרצאה 3 , ונחליט שכל מחרוזת מקודדת מ"ט כלשהי:

- אם הקידוד חוקי לפי הפורמט שהגדרנו אז יקודד מ"ט כמו קודם.
- ואם לא יקודד את המכונה M_{stam}:שמיד עוצרת ומקבלת:

$$\forall a\ \delta(q_0) = (q_{acc}, q, s)\,,\ \sum = \{0,1\}\,, \Gamma = \{0,1,\flat\}$$

הקידוד ה"קנוני" שהגדרנו בהרצאה הקודמת.

0.4.9 טענה: $L_D \in RE\backslash R$

<u>הוכחה:</u>

שייכות ל RE - נבנה מ"ט המקבלת את L_D , (אבל לא תכריע אותה)

$M_D(\langle M\rangle)$:

תבדוק "חוקיות" של $\langle M\rangle$. אם לא חוקי מיד תעצור ותקבל (אז $\langle M\rangle$ הוא קידוד של M_{stam} וזו מקבלת הכל, בפרט את $\langle M_{stam}\rangle$)

אחרת, : תריץ את M על $\langle M\rangle$, ותענה כמוה.

<u>נכונות</u>: בבית

אי שייכות ל R **:** נראה ש $\bar{L}_D$ אינה ב RE ומכך נסיק בקלות ש: $L_D \in R$. כיצד נוכיח?

נוכיח באמצעות טעון לכסון.

נתבונן בטבלה הבאה:

	0	1	1	0	$\dots$			
$M\backslash\sum^*$	$\langle M_1\rangle$	$\langle M_2\rangle$		$\langle M_i\rangle$			$\langle M_j\rangle$	
$\langle M_1\rangle$	**0**	1	1					
$\langle M_2\rangle^2$	1	**0**	1	1	0	0	1	$\dots$
			_					
$\langle M_i\rangle$				_		0^1		
					_			
						_		
$\langle M_j\rangle$								

1 . כל משבצת מופיע 1 אם M_i מקבלת את $\langle M_j\rangle$ ו 0 אחרת

2. בשורה $\langle M_2\rangle$ ישנו וקטור של 0ים ו1ים כך שה1ים מופיעים במקומות המתאימים למילים ב $L\left(M_2\right)$

נתבונן בשפה $\bar{L}_D$ (מופיע מעל השורה הראשונה) ־ זוהי שורה אינסופית כלשהי של 0ים ו1ים. האבחנה הקריטית היא ששורה זו אינה שווה לאף אחת מהשורות בטבלה שהן רשימת כל השפות ב RE , ואז נסיק שאינה RE .

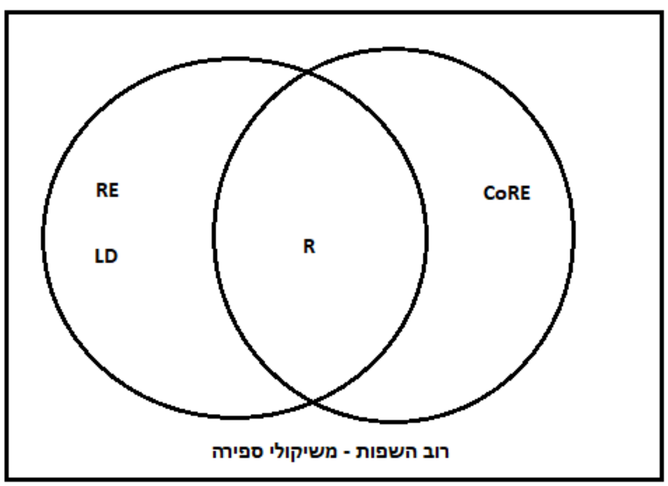
מדוע? עבור השורה של $\langle M_i\rangle$כלשהי, $\bar{L}_D$ שלנו אינה מסכימה עם M_i לגבי הקלט $\langle M_i\rangle$:

- אם M_i מקבלת את $\langle M_i\rangle$, אז $\langle M_i\rangle \in L_D$ (כתוב 1 בכניסה (i,i)) , אבל מהגדרת $\bar{L}_D$, $\langle M_i\rangle \notin \bar{L}_D$
- אם M_i לא מקבלת את $\langle M_i\rangle$, אז $\langle M_i\rangle \in L_D$ וכתוב 1 בשורה של $\bar{L}_D$ לעומת 0 ב (i,i)

$\square$

כעת נראה ש $L_D \notin R$:

נניח בשלילה שכן, אז מסגירות של R למשלים גם $\bar{L}_D$ הייתה ב R ואז היינו מקבלים $\bar{L}_D \in R \subseteq RE$, בסתירה למה שהוכחנו.



0.5 שיעור 5 ־ 28/11/19

נזכר בתמונת העולם משיעור שעבר (לעיל) , נזכר שהוכחנו ש L_D אינה שייכת ל R בלכסון.

נזכר שבהוכחה בלכסון ש $\bar{L}_S$ אינה ב RE) בעצם בנינו את $\bar{L}_D$ ע"י הסתכלות בטבלת כל השפות ב RE והיפוך האלכסון של הטבלה (זו טבלה עם $\aleph_0$ שורות ו $\aleph_0$ עמודות שמואנדקסות ע"י קידודי מכונות). לא ברור כיצד להשתמש בגישה כזו עבור שפה "חדשה" שלא אנחנו בנינו במיוחד ואנחנו חושדים שהיא קשה (לא ב R . לא ב RE / לא $CoRE$) .

היום נלמד טכניקה כזו שמאפשרת בהרבה מקרים להוכיח "קושי" של שפה. הרעיון הכללי הוא לקחת L_2 שפה שידוע לנו (למשל, הוכחנו בשיטה אחרת , למשל בלכסון) שידוע שלנו שהיא קשה ולהראות עבור שפה חדשה L_1 שהיא קשה על סמך העובדה ש L_2קשה. נתחיל מדוגמת שימוש בגישה, ואחר כך נפרמל אותה.

0.5.1 **הגדרה:** 5.1 **השפה האוניברסלית:**

$$L_U = \{\langle M\rangle, \langle X\rangle \mid \text{קידוד חוקי סינטקטית וגם } \langle M\rangle, \langle X\rangle \text{ מקבלת את } x\ M\}$$

כאשר: $\langle M\rangle$ קידוד של מ"ט, $\langle X\rangle$ קידוד של קלט עבור M

הבעיה יכולה להיות רק ב $\langle X\rangle$: בתאמה לא"ב של M וכו' (מבחינתנו כל מחרוזת היא קידוד חוקי של מ"ט)

0.5.2 **טענה** 5.2 : $L_u \in RE\backslash R$

<u>הוכחה</u>: **נראה קודם ש** $L_u \in RE$ (סטנדרטי) . נבנה מ"ט M_u המקבלת את L_u

$M_u(y)$:

1. אם y אינה מהווה קידוד חוקי $(\langle M\rangle, \langle X\rangle)$ של זוג מכונה וקלט עבורה, נעצור ונדחה. תקינות סינטקטית כזו, אפשרית לבדיקה (למשל צריך לוודא התאמה בין $\sum$ של M של x וכו'...)
2. נריץ את M על x (בדומה למכונה האונבריסלית שבינינו בהרצאה 3 נבצע סימולציה צעד־צעד)

   אם M קיבלה, נקבל ואם דחתה, נדחה.

<u>נכונות</u>

אם $y \in L_u = (\langle M\rangle, \langle X\rangle)$: M מקבלת את x (מהבניה) $\Leftarrow$ L_u תזהה שהקלט תקין ב 1 וב 2 הסימולציה תעצור ותזהה קבלה. מהבניה של M_U, גם M_U תעצור ותקבל

אם $y \notin L_u$ ישנם שלושה מקרים:

1. y אינו קידוד תקין. M_U תזהה זאת בשלב 1 ותדחה $\Leftarrow$ $y \notin L\left(M_U\right)$, כנדרש.
2. M בריצה על x (והקידוד חוקי) דוחה. במקרה זה , M_U תעצור בשלב 2 (בסימולציה), ותזהה דחיה ותדחה גם $\Leftarrow$ $y \notin L\left(M_U\right)$,כנדרש,
3. M לא עוצרת על x (הקידוד חוקי) . M_U בסימולציה בשלב 2 לא תסיים את הסימולציה (על כל צעד של M בריצה על M_U מסמלצת את הצעד, כיוון שיש ∞ צעדים יהיו ∞ צעדי סימולציה), ולא תעצור (על y) $\Leftarrow$ $y \notin L\left(M_U\right)$, כנדרש.

נראה ש $L_u \notin R$

איך? נשתמש בעובדה ש $L_D \notin R$ (הוכחנו בשבוע שעבר): נניח בשלילה ש $L_U \notin R$ נשתמש במכונה המובטחת M_U (שמכריעה את L_U , לפי ההנחה בשלילה) כדי לבנות מ"ט M_D שתכריע את L_D . אבל זו תהיה סתירה כי $L_D \notin R$ ולכן אין מכונה כזו.

כיצד M_D תוגדר?

$M_D\left(\langle M\rangle\right)$:

- נריץ את M_U על הקלט $x = (\langle M\rangle, \langle M\rangle)$ (האם המכונה מקבלת את הקידוד של עצמה) ונענה כמוה (אינטואיטבית , כי M_U יודעת לענות בדיוק על השאלה האם מכונה עוצרת על קלט נתון, ואותנו מעניינת התשובה לגבי המכונה M והקלט $\langle M\rangle$) .
- כיצד נעשה זאת: בדומה למימוש של המכונה האוניברסלית. בפרט, הקידוד של M_u (שמהמכונה האוניברסלית צריכה) יגיע מ δ של M_D. כלומר נרשום את $\underbrace{(\langle M\rangle, \langle M\rangle)}_{input}, \underbrace{\langle M_u\rangle}_{machine}$ בתור קלט לפרוצדורה שתריץ את המכונה האוניברסלית.

  הערה לגבי הקלט: קל לייצר (להכפיל את $\langle M\rangle$ זה אפשרי ב RAM , אפילו ראינו מימוש במ"ט בהרצאה 2)

נשאר להראות ש M_Uמכריעה את L_D :

$\langle M\rangle \in L_D \Leftarrow M$ מקבלת את $\langle M\rangle \Leftarrow$ בסימולציה של M_u על קלט $(\langle M\rangle , \langle M\rangle)$ M_u תקבל ולכן גם M_D תקבל את (את $\langle M\rangle$) $\Leftarrow$ $\langle M\rangle \in L(M_D)$ כנדרש.

$\langle M\rangle \notin L_D \Leftarrow M$ אינה מקבלת את $\langle M\rangle$ (נתעלם מחוקיות קידוד) $\Leftarrow$(מההנחה ש M_U מכריעה את L_U) של M_U דוחה את $(\langle M\rangle , \langle M\rangle) \Leftarrow M_D$ דוחה את $\langle M\rangle$, ובפרט עוצרת על $\langle M\rangle$

כלומר לסיכום, M_D תמיד עוצרת עם התשובה הנכונה לגבי שייכות של $\langle M\rangle$ ל $L_D \Leftarrow M_D$ מכריעה את L_D וזו סתירה. לכן M_U שמכריעה את L_U אינה קיימת, ולכן $L_U \in R$

$\square$

הערה: בדומה ללכסון, הטכניקה שהשתמשנו בה כאן הייתה עובדת גם עבור מודלים אחרים "עבירים" לא רק עבור מ"ט). הינו צריכים רק שלכל מכונה יהיה קידוד, ושתהיה דרך להריץ את המכונה על קלטים גם הנתן הקידוד (היה עובד גם ב RAM , מחשב קוונטי, וכו'), נעבור להכללה של השיטה.

רידוקציות

נסטה מהנושא לרגע ונדבר על חישוב יעיל. בבעיות שרוצים לפתור בעולם האמיתי הרבה פעמים זה מועיל להשתמש באלגוריתם (פונקציה בפייתון) שפותר בעיה A , כדי לפתור יותר ביעילות / בקלות בעיה B אחרת.

נאמר שאלגוריתם שפותר את B ע"י קריאות לתוכנה שפותרת את A בתור קופסא שחורה (מקבלת קלטים ומוציאה פלטים) נקרא רדוקציה. בדכ בעולם האמיתי משתמשים ברדוקציות כדי לבנות אלג' ל B. אולם (כמו שנראה בחלק שני של הקורס, הן גם שמושיות כדי להראות דווקא שבעיה מסויימת היא קשה לפתרון)

נראה דוגמה לשימוש בכיוון ה"חיובי" של רדקוציות (כדי לבנות אלגוריתמים)

דוגמה: נגדיר בעיות שקשרות לפירוק מספרים לגורמים.

<u>בעיה A :</u>

על קלט $x \in \mathbb{N}^+$ יש לפלוט מחלק לא טריוויאלי $y \in \mathbb{N}^+$ $(y \neq 1, x)$ קטן ביותר של x , או לפלוט "לא קיים" אם אין כזה (x ראשוני).

<u>בעיה B :</u>

על קלט (x, k) , זו מספרים ב $\mathbb{N}^+$, כאשר $k \leq x$ נפלוט 1 אם קיים לx מחלק לא טריוויאלי $k <' y$, אחרת נפלוט 0.

בשתי הבעיות לא ידוע אלג' יעיל (כזה שרץ בזמן פולינומי בקלט). לדוגמה האלג' הנאיבי שבודק התחלקות בכל y עד לשרש רץ בזמן $\Omega\left(2^{\frac{n}{2}}\right)$. אבל כן ניתן לבנות רידוקציה יעילה מ A ל B ולהיפך.

<u>רידוקציה מ B ל A :</u>

על קלט (x, k) , האלג' M_B^A (פתור את B בעזרת A)

- קורא ל $A(x)$ ומקבל y או לא קיים,
  - אם קיבל לא קיים - יחזיר 0
  - אחרת ישווה את $\frac{x}{y}$ ל k . יחזיר 1 אם"ם $k < \frac{x}{y}$ (יוצא שזה המחלק הכי גדול הלא טריוויאלי)

הרדוקציה M_B^A אכן יעילה , מבצעים קריאה אחת ל A + חלוקה של שני מספרים בני n ביטים - $O(n^2)$ פעולות על ביטים. סה"כ יעיל (פולינומי בקלט)

<u>רדוקציה מ A ל B :</u>

חיפוש בינארי.

1. נקרא ל $B\left(x, k=1\right)$:
   - אם החזיר 0 - נחזיר לא קיים

- אחרת נאתחל $a = 2$, $b = x - 1$

2. נעבוד באיטרציות. בכל איטרציה נשאל $B\left(X, \frac{a+b}{2}\right)$
   - אם החזיר 1 , נעדכן את $a \leftarrow \frac{a+b}{2}$,
   - אחרת נעדכן את $B \rightarrow \frac{a+b}{2}$

3. נעצור כאשר $b - a \leq 3$ ואז נחפש ידנית (נבצע חלקות של x ב $a, a+1, ..., b$)

ביצענו $O(n) = O(\log x)$ קריאות ל B + עוד עבודה $O\left(n^2\right)$ בחלוקות. סה"כ הרדקוציה יעילה.

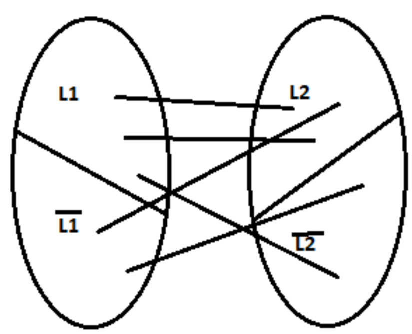
נחזור לענייננו

אנחנו ברדוקציות (בד"כ) בכיון השלילי, בדומה למה שעשינו בהוכחה ש $L_U \in R$ בעזרת L_D שידוע שאינה ב R .
כדי שרדוקציות יהיו שימושיות גם כדי להוכיח אי שייכות ל RE או $CoRE$, ולא רק ל R נגבית את צורת הרדוקציה. בפרט בניגוד לרדוקציה מ A ל B שראינו שמבצעת מספר קריאות לאלגוריתם למכונה עבור הבעיה שאליה עושים רידוקציה, אז אנחנו נאשר רק קריאה אחת. נפרמל את המושג של רדוקציות שנשמתמש בו.

0.5.3 הגדרה פונקציית רידוקציה:

בהנתן זוג שפות $L_1, L_2 \subseteq \sum^*$ נאמר שפונקציה $f : \sum^* \rightarrow \sum^*$ היא פונקציית רידוקציה בין זוג השפות אם היא מקיימת:

1. f מלאה
2. f ניתנת לחישוב (בפרט M_f המחשבת אותה מייד עוצרת)
3. תקפה : $f(x) \in L_2 \iff x \in L_1 \quad \forall x \in \sum^*$,

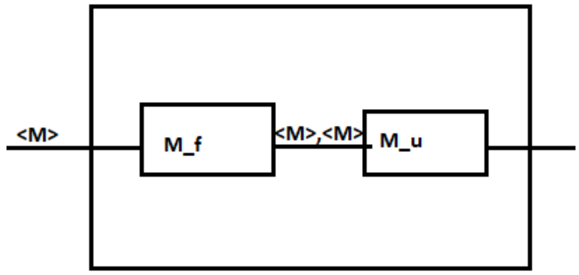


הערה: אין דרישות נוספות כגון חח"ע מ f :
אם קיימת פונקצית רדוקציה מ L_1 ל L_2 כלומר ש L_1 נתנת לרדוקציה ל L_2 , ונסמן ב $L_1 \leq L_2$.ננסח מחדש את ההוכחה של $L_U \in R$ "על סמן $L_D \in R$ " במונחי פונקציית רידוקציה . בהמשך נכליל את ההוכחה למשפט רדקוציה). בעצם, בלי לומר זאת במפורש, הראינו ש $L_D \leq L_U$. מה הייתה פונקציית הרידוקציה?
$f\left(\langle M\rangle\right) \rightarrow \left(\langle M\rangle, \langle M\rangle\right)$
נראה שהיא אכן מקיימת את שלושת התכונות של פונק' רדוקציה מ L_D ל L_n)

1. f מלאה (כל מחרוזת נתן לפרש כקידוד של מכונה) בפרט ההגדרה של f לא דורשת אפילו את זה.
2. f נתנת לחישוב - קל לקחת מחרוזת ולשכפל אותה

3. תקפה: $\langle M\rangle \in L_D \Rightarrow M$ מקבלת את $\langle M\rangle \Rightarrow f(\langle M\rangle) = (\langle M\rangle, \langle M\rangle) \in L_U$
ואם: $\langle M\rangle \notin L_D \Rightarrow M$ לא מקבלת את $\langle M\rangle \Rightarrow f(\langle M\rangle) = (\langle M\rangle, \langle M\rangle) \notin L_U$

מה הוכחה עשתה? בנתה M_D מתוך M_U (דמיונית שהנחנו בשלילה את קיומה) בעזרת f . כיצד? נסמן ב M_f מ"ט המחשבת את f . בנינו את M_D כ:



כיוון ש M_f , וM_u תמיד עוצרות ו M_f תקפה, התשובה של M_u לגבי שייכות $f(\langle M\rangle)$ ל L_u זהה להאם $\langle M\rangle$ שייכת ל L_D כלומר M_D הזו מכריעה את L_D . וזו סתירה. נפרמל את הטיעון ככה שהבניה של M_D תלך לתוך ההוכחה של משפט רדוקציה שננסח. בעזרת המשפט יהיה מספיק לבנות רק את פונק' הרדוקציה הנדרשת.

0.5.4 משפט הרדוקציה (נוסח ישיר)

יהיו $L_1, L_2 \subseteq \sum^*$ כך ש $L_1 \leq L_2$ אזי:

1. אם $L_2 \in R$ אז $L_1 \in R$
2. אם $L_2 \in RE$ אז $L_1 \in RE$
3. אם $L_2 \in CoRE$ אז $L_1 \in CoRE$

0.6 שיעור 6 - 5/12/19 - יעל סבתו

(חצי שעה ראשונה מקארין)

רדוקציות

תזכורות:

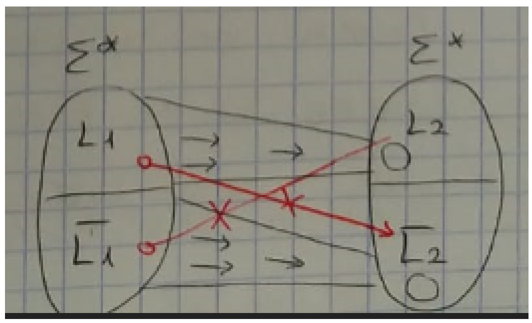
- מכונה M שמקבלת שפה L:
  - $\forall x \in L \ M(x) = L$
  - $M(x) =$ לא מוגדר (לולאה) או $M(x) = 0$ $\forall x \notin L$
- מכונה M שמכריעה שפה L :
  - $\forall x \in L \ M(x) = 1$
  - $\forall x \notin L \ M(x) = 0$
- R = קבוצת השפות שקיימת מ"ט **שמכריעה** את L
- RE = קבוצת השפות שקיימת מ"ט **שמקבלת** את L
- $coRE$ = (יודעת **לדחות** אבל לא יודעת לקבל) $\{L|\bar{L} \in RE\}$

עד עכשיו ראינו הוכחות של שייכות למחלקה $(R\backslash RE)$, והוכחנו על ידי תאור מכונה. כעת נרצה להוכיח גם על שפות מסוימות שאינן במחלקה מסוימת:

- $L_D = \{\langle M\rangle\,|\text{M accept }\langle M\rangle\} \notin R$
- $L_U = \{\langle M,x\rangle\,|\text{M accept x}\} \in RE\wedge\notin R$

הגדרה: יהיו $L_1, L_2 \in \sum^*$ נאמר ש $L_1 \leq L_2$ ־ במילים: L_1 נתנת **לרידוקציה** ל L_2 ־ אם קיימת פנו' $f : \sum^* \to \sum^*$ המקיימת:

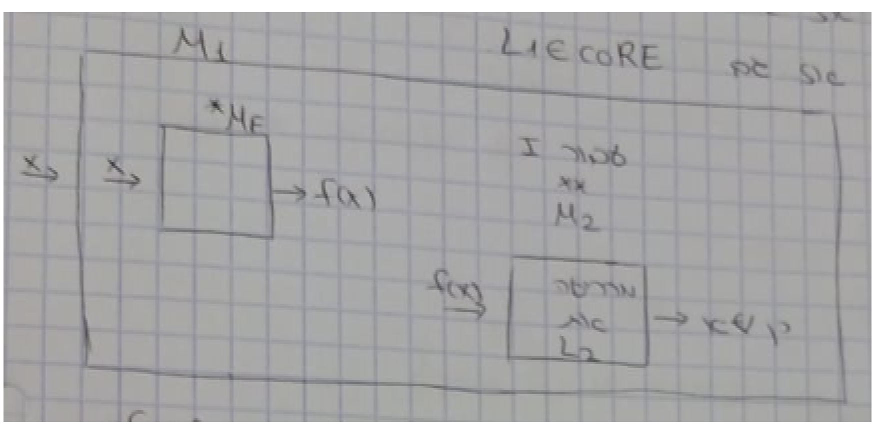
1. f מלאה
2. f ניתנת לחישוב
3. f תקפה: $\forall x\in\sum^* : x\in L_1 \iff f(x)\in L_2$



משפט הרידוקציה: אם $L_1 \leq L_2$ אז:

1. אם $L_2 \overset{*}{\in} R$ אז גם $L_1 \in R$
2. אם $L_2 \overset{**}{\in} RE$ אז גם $L_1 \in RE$
3. אם $L_2 \in CoRE$ אז גם $L_1 \in CoRE$

רעיון ההוכחה:



הערה: בזכות התקפות, התשובה לשאלה אם $x \in L_1$ זהה לתשובה לשאלה אם $f(x)\in L_2$

הוכחה ־ עבור 1

- נניח ש $L_1 \leq L_2$ וגם $L_2 \in R$ אז קיימת מכונה M_2המכריעה את L_2

- וכן קיימת מ"ט M_f המחשבת את פונ' הרדוקציה בין L_1 ל L_2
- נתאר מכונה מכריעה M_1 עבור L_1
- M_1 על קלט $x \in \sum^*$
  - תריץ את M_f על x ותקבל $f(x)$
  - תריץ את M_2 ותקבל על $f(x)$ ותענה כמוה
- מכיוון שפונ' הרדוקציה מלאה וניתנת לחישוב החישוב של M_1 לא יתקע בשלב 1 ,
- ומכיון שפונ' תקפה התשובה של M_2 על השייכות של $f(x)$ ל L_2 זהה לתשובה הדרושה עבור השייכות של x ל L_1

ניסוח שקול (ושימושי) למשפט הרדוקציה: אם $L_1 \leq L_2$ אז:

- אם $L_1 \notin R$ אז גם $L_2 \notin R$
- אם $L_1 \notin RE$ אז גם $L_2 \notin RE$
- אם $L_1 \notin coRE$ אז גם $L_2 \notin CooRE$

תכונות של רדוקציה

- אם $L_1 \leq L_2$ אז גם $\bar{L}_1 \leq \bar{L}_2$ - (תוצאה ישירה של תקפות הרידוקציה)
- היחס $\leq$ הוא רפלקסיבי: לכל שפה $L \subseteq \sum^*$, $L \leq L$ - הוכחה לפי פונ' הזהות.
- היחס $\leq$ טרנזיטיבי: אם $L_1 \leq L_2$ וגם $L_2 \leq L_3$ אז: $L_1 \leq L_3$ - הוכחה הרכבת פונקציות
- היחס $\leq$ אנטי־סימטרי: אם $L_1 \leq L_2$ אז לא בהכרח ש: $L_2 \leq L_1$

דוגמה:

$L_1 = \sum^*$, $L_2 = HP$ אז $\sum^* \leq HP$ כי נגדיר $f(x) = \langle M_{stam}, x \rangle$

נזכר בהגדרה: $HP = \{\langle M, x \rangle \,|\text{M stop on x }\} \in RE \backslash R$

אבל $HP \not\leq \sum^*$ לא נכון!

דוגמה:

נתונה השפה: $L = \{\langle M \rangle \mid \ |L\left(M\right)| \geq 2\}$

- $L \notin R \leftarrow$ נראה ברידוקציה
- $L \in RE \leftarrow$ רמז צריך הרצה מבוקרת. (מתארים מכונה <u>מקבלת</u> שעושה הרצה מבוקרת על סדור לקסיקוגרפי של $\sum^*$)

נוכיח ש $L \notin R$ ע"י רידוקציה

בדרך כלל ניקח HP, L_U :

- $HP = \{\langle M, x \rangle \,|\text{M halts on x }\}$
- $L_U = \{\langle M, x \rangle \,|\text{M accept x }\}$

אז נראה ש: $HP \leq L$ לכן נסמן: $f\left(\langle M, x \rangle\right) = \langle M' \rangle$

כך ש M' על קלט $y \in \sum^*$:

1. הרץ את M על x
2. קבל (את y)

<u>f מלאה וניתנת לחישוב</u>: כי ראינו שאפשר לקודד מ"ט למחרוזת

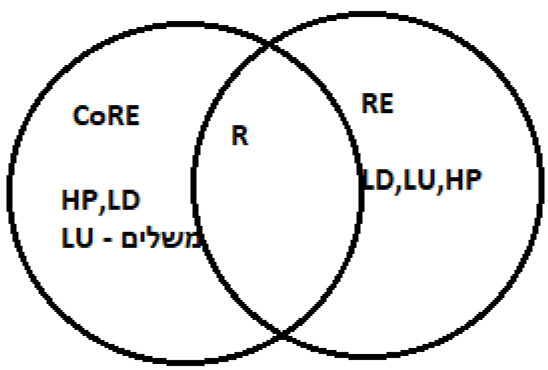
<u>f תקפה:</u>

נראה ש: $x \in L_1 \Rightarrow f(x) \in L_2$ וגם $x \notin L_1 \Rightarrow f(x) \notin L_2$:

- $\langle M,x \rangle \in HP \Leftarrow M$ עוצרת על $x \Leftarrow M'$ מקבלת כל $y \in \sum^* \Leftarrow L(M') = \sum^* \Leftarrow 2 \leq \infty = \left|\sum^*\right| \Leftarrow \langle M' \rangle \in L$
- $\langle M,x \rangle \notin HP \Leftarrow M$ לא עוצרת על $x \Leftarrow M'$ תמיד תכנס ללואה $\Leftarrow L(M) = \phi \Leftarrow |\phi| = 0 < 2 \Leftarrow \langle M' \rangle \notin L$

לסיכום , הראנו $HP \leq L$ כעת על פי משפט הרידקוציה מכיוון ש $HP \notin R$ גם $L \notin R$

נזכר במפת עולם:



דוגמה:

נרצה להראות דוגמה לשפה במשלים של $RE \cup CoRE$

מועמדת:

$$L_{=3} = \{\langle M \rangle \,|\, |L(M)| = 3\} \notin RE \cup coRE$$

בשביל להוכיח נצטרך 2 רדוקציות:

1. $HP \leq L_{=3}$ נקבל $L_{=3} \notin R$, $L_{=3} \notin coRE$
2. $\overline{HP} \leq L_{=3}$ נקבל $L_{=3} \notin RE$

<u>הוכחה 1 : $HP \leq L_{=3}$</u>

נגדיר $f(\langle M,x \rangle) = \langle M^* \rangle$, כך ש M^* על קלט $y \in \sum^*$

1. מריצה את M על x
2. אם $y \in \{0,1,01\}$ קבל
   אחרת - דחה.

<u>f מלאה וניתנת לחישוב</u> - כי ראינו שניתן לקודד מ"ט

<u>f תקפה:</u>

- $\langle M,x\rangle \in HP \Leftarrow M$ עוצרת על $x \Leftarrow M^*$ תקבל כל $y \in \{0,1,01\} \Leftarrow L(M^*) = \{0,1,01\} \Leftarrow |L(M^*)| = 3 \Leftarrow$ $\langle M^*\rangle \in L$
- $\langle M,x\rangle \notin HP \Leftarrow M$ לא עוצרת על $x \Leftarrow M^*$ תמיד תכנס ללואה $\Leftarrow L(M^*) = \phi \Leftarrow |L(M^*)| = 0 \neq 3 \Leftarrow \langle M^*\rangle \notin L$

<u>הוכחה 2 : $\overline{HP} \leq L_{=3}$</u>

נגדיר $f(\langle M,x\rangle) = \langle \widetilde{M}\rangle$ כך ש: $\tilde{M}$ על קלט $y \in \sum^*$

1. אם $y \in \{0,1,01\}$ קבל
2. הרץ את M על x, אם עצרה - קבל

<u>f מלאה וניתנת לחישוב</u> כי ראינו שניתן לקודד מ"ט

<u>f תקפה:</u>

- $\langle M,x\rangle \in \overline{HP} \Leftarrow M$ לא עוצרת על $x \Leftarrow \tilde{M}$ תקבל רק את $y \in \{0,1,01\}$(משלב 1) $\Leftarrow \left|L\left(\tilde{M}\right)\right| = 3 \Leftarrow \left\langle \tilde{M}\right\rangle \in L_{=3}$
- $\langle M,x\rangle \notin \overline{HP} \Leftarrow M$ עוצרת על $x \Leftarrow \tilde{M}$ בשלב 1 תקבל את כל $y \in \{0,1,01\}$ ובשלב 2 תקבל $\sum^* = L\left(\tilde{M}\right) \Leftarrow$ $|\sum^*| = \infty \neq 3 \Leftarrow \left\langle \tilde{M}\right\rangle \notin L_{=3}$

<u>דוגמה נוספת להוכחה מהצורה $\overline{HP} \leq L$</u>

נתונה השפה: $L_\infty = \{\langle M\rangle \,|\, |L(M) = \infty|\} \notin RE \cup coRE$

הוכחה ע"י

1. $HP \leq L_\infty$ - בבית
2. $\overline{HP} \leq L_\infty$ - קצת טריקית

<u>$\overline{HP} \leq L_\infty$</u>

נגדיר $f(\langle M,x\rangle) = \langle M_x\rangle$, כך ש M_x על קלט $y \in \sum^*$

1. הרץ את M על x למשל $|y|$ צעדים
2. אם M לא עצרה - קבל

   אחרת - דחה

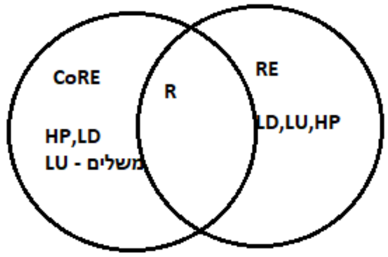
<u>f מלאה וניתנת לחישוב...</u>

<u>f תקפה:</u>

- $\langle M,x\rangle \in \overline{HP} \Leftarrow M$ לא עוצרת על $x \Leftarrow L(M_x) = \sum^* \Leftarrow \langle M_x\rangle \in L_\infty$
- $\langle M,x\rangle \notin \overline{HP} \Leftarrow M$ עוצרת על x , אחרי t כלשהו של צעדים, נפצל למקרים:
  - אם $|y| \geq t \Leftarrow M_x$ תדחה
  - אם $|y| < t \Leftarrow M_x$ תקבל
- בכל מקרה, $L(M_x) = \{y | y < t\}$ וזאת שפה סופית! $\Leftarrow \langle M_x\rangle \notin L_\infty$

0.7 שיעור 7

נזכר בתמונת העולם:



כלים שראינו עד כה להוכחת קושי של שפות:

- שיקולי ספירה (לא ניתן דוגמאות ספציפיות)
- טעון לכסון L_D לא ב R קשה להתאים לשפות גשות
- רדוקציות (משפט הרדקוציה) - אפשר להראות קושי (אי שייכות ל $R, RE, CoRE$) עבור מגוון שפות חדשות.

היום נרחיב את ארגז הכלים שלנו במשפט כללי - משפט $Rice$ שיפסק "קיצור דרך" עבור שפות מסומות, ויאפשר להמנע ממציאת רדוקציה מפורשת.

לפני כן, נראה דוגמה נוספת לרדוקציה שמושית.

0.7.1 הגדרה 7.1 : $L_{eq} = \{\langle M_1\rangle, \langle M_2\rangle | L(M_1) = L(M_2)\}$

ננסה להבין מה הסווג של השפה מבחינת שייכות ל RE ו $CoRE$. אינטואיטיבית:

<u>לגבי שייכות ל RE :</u> אפשר לנסות להריץ את M_1, M_2 במקביל. על $\sum^*$ בהרצה מבוקרת (הכי "מתוחכם" שאנחנו יודעים). אבל, גם אם עד נק' מסוימת ב $\sum^*$ התמזל מזלנו וכל המלים נדחו/התקבלו באותה הצורה בשתי המכונות (שתיהן עצרו), זה לא אומר מה יקרה בעתיד (יש עוד ∞ קלטים לבדוק בכל נקודה).

<u>לגבי שייכות ל $CoRE$:</u> כאן המצב קצת יותר קל. ברגע שנבדק מילה שעליה M_1 ו M_2 עצרו ולא הסכימו , אפשר לעצור ולדחות. אבל אם על המילה הבעייתית M_1 מקבלת ו M_2 לא עוצרת, לא נדע לזהות זאת.

נוכיח את ההשערה שלנו ע"י רדוקציה מ L_∞

תזכורת: $L_\infty = \{\langle M\rangle \,|\, |L(M)| = \infty\}$

0.7.2 טענה 7.2 : $L_{eq} \notin RE \cup CoRE$

הוכחה: נוכיח באמצעות רדוקציה $L_\infty \le L_{eq}$, שימו לב שכיוון ש $L_\infty \notin RE \cup CoRE$ נתן להוכיח את שני הדברים "בבת אחת" ספציפית מ $L_\infty \le L_{eq}$:

1. $L_{eq} \notin RE$, כי $L_\infty \notin RE$ - משפט הרדוקציה סעיף 2
2. $L_{eq} \notin CoRE$,כי $L_\infty \notin CORE$ - הרצאה קודמת - משפט הרדוקציה סעיף 3

נבנה רדוקציה מתאימה: $f(\langle M\rangle) = (\langle M_1\rangle, \langle M_2\rangle)$

כאשר: $\langle M_2\rangle = \langle M_{\sum^*}\rangle$ מ"ט שמיד עוצרת ומקבלת כל מילה (מיד מ q_0 הולכת ל q_{acc})

$M_1(y)$: מריצה את M על $\sum^*$ בהרצה מבוקרת. אם בצעד מסויים i של ההרצה המבוקרת התקבלו כבר $|y|$ מילים, נעצור ונקבל את y .

נסביר רגע בדוגמה:

* נגיד ש M מקבלת את 11 (המילה ה 7 בסדר לקסקוגרפי), בצעד ה 328 ,
* את 0 (מילה 2 בסדר הלקס' בצעד 512),
* ועוד ∞ מילים בצעדים יותר מאוחרים.
* אז M_1 על קלט $y=10$ תקבל את y בצעד 512 (כאשר מתקבלת המילה השניה) בציור:

$string \backslash step$	1	2	$\dots$					
ε_0						X		X
								$\dots$

נחזור לרידוקציה ־ נכונות:

1. f מלאה ־ כל מחרוזת ניתן לפרש כקידוד של מכוה (לפי המוסכמה שלנו של M_{stam}), לכן f אכן מוגדרת על כל קלט.
2. f נתנת לחישוב: כדי לחשב את f מבצעים פעולת קומפילציה, בפרט אין צורך להריץ מכונות על קלטים כלשהם. (זה וריאנט של המכונה האוניברסלית, ו M_1 תדאג לרשום לה את $\langle M\rangle$ בתור המכונה שמריצים ־ את כל זה לא קשה לקודד)
3. תקפות :

* עבור $\langle M\rangle \in L_\infty$: M_1 תקבל כל y , כי M מקבלת ∞ מילים, ולכן בהנתן y מסוים, תקבל $|y|$ מילים תוך מס' סופי של צעדים ברצה מבוקרת
$\Leftarrow L(M_1)=\sum^* \Leftarrow \sum^* = L\left(\sum^*\right) = L(M_2) = L(M_1) = L$, מכאן ש $f(\langle M_1\rangle) = (\langle M_1\rangle, \langle M_2\rangle) \in L_{eq}$ כנדרש.
* עבור $M \Leftarrow \langle M\rangle \notin L_\infty$ מקבלת N (מס' סופי של מילים) $\Leftarrow M_1$ תקבל אך ורק את הyים שאורכם $\geq N \Leftarrow L(M_1)$ שפה סופית $\Leftarrow \sum^* = L(M_2) \neq L(M_1) \Leftarrow (\langle M_1\rangle, \langle M_2\rangle) \notin L_{eq}$, כנדרש.

$\square$

משפט $Rice$

אינטואיטיבית, משפט $Rice$ אומר שאי אפשר לבנות $debugger$ שבהנתן קוד של תוכנה, בודק תכונה שמושית כלשהי של השפה שלה, לדוגמה:

* האם התוכנה מקבלת את "M" ?
* האם השפה של המכונה (התוכנה הזו) היא סופית? ריקה?

0.7.3 הגדרה 7.3 : תכונה של שפות ב RE

S תכונה של שפות ב RE היא תת קבוצה $S \subseteq RE$, נאמר ש S טריויאלית אםם $S \in \{RE, \phi\}$

0.7.4 משפט $Rice$ 7.4 :

תהי S תכונה של שפות ב RE , נגדיר: $L_S = \{\langle M\rangle \,|L\,(M) \in S\}$ אז:

$$L_S \in R \iff S \text{ טריויאלית}$$

<u>$\Leftarrow$ כיון קל:</u>

- תהי S תכונה טריוויאלית של שפות ב RE . נראה ש $L_S \in R$. ישנן שתי אפשרויות:

1. $S = RE$ אזי: $L_S = \{\langle M\rangle \,|L\,(M) \in RE\}$
אבל, מהגדרת RE , לכל N , $L\,(M)$ ב RE כי M היא מ"ט המקבלת את $L\,(M)$. מהמוסכמה שכל מחרוזת היא קידוד של מכונה כלשהי , $L_S = \sum^*$ אכן $\sum^* \in R$ (קל לבנות מ"ט שתקבל אותה).
2. $S = \phi$ אז $L_S = \{\langle M\rangle \,|L\,(M) \in \phi\} = \phi$, אכן $\phi \in R$ (קל לבנות מ"ט שתכריע אותה)

<u>$\Rightarrow$ כיון שני:</u>

- תהי S לא טריוויאלית נוכיח ש $L_S \notin R$ באמצעות רדוקציה שתעבוד לא S (נחלק לשני מקרים) - "משפחה" של רדוקציות.

<u>מקרה 1:</u> נניח $\phi \notin S$. נראה $HP \leq L_S$. כיון ש $HP \notin R$, משפט הרדוקציה , $L_S \notin R$

$f\,(\langle M\rangle\,,\langle x\rangle) = \langle M_x\rangle$ כאשר:

$M_x(y)$:

1. מריצה את M על x
2. (המקרה בו עברנו את המחסום) תהי $L \in S$ קיימת כזו, כי S לא טריוויאלית, כיון ש $S \subseteq RE$ קיימת לה מ"ט M_L המקבלת אותה נריץ את M_L על y ונענה כמוה

נוכיח נכונות של f :

1. f מלאה - למעשה כמו שהגדרנו אותה אינה מלאה כי לא טיפלנו במקרה $\langle\langle M\rangle\,,\langle X\rangle\rangle$לא תקין סינטקטית . אבל תמיד אפשר לטפל בזה ע"י בדיקת סינטס (קל לעשות), ואם לא עובר, פולטים M' קבוע שאינה ב L_s (במקרה הזה M_ϕ כלשהי שמיד דוחה הכל) בהמשך נתעלם מענייני תקינות סינטקטית
2. f ניתנת לחישוב - פעולת קומפילציה
3. תקפות:

- $(\langle M\rangle\,,\langle x\rangle) \in HP \Leftarrow M$ עוצרת על $x \Leftarrow$ לכל y M_x מגיעה לשלב 2 ועונה כמו $M_L \Leftarrow L\,(M_x) = L\,(M_L)$ (מבחירת $L \in S \Leftarrow L\,(M_x) \in S$ (הגדרת L_S) $\Leftarrow \langle M_x\rangle \in L_s$
- $(\langle M\rangle\,,\langle x\rangle) \notin HP \Leftarrow M$ לא עוצרת על $x \Leftarrow$ לכל קלט y , M "מתנקעת" בשלב 1 $\Leftarrow$(מהנחה) $L\,(M_x) = \phi \notin S$ $\Leftarrow \langle M_x\rangle \notin L_S$

כנדרש

<u>מקרה 2:</u> $\phi \in S$. נשים לב ש:

$$L_S = \{\langle M\rangle \,|L\,(M) \in S\} = \overline{L_{\bar{s}}} = \{\langle M\rangle \,|L\,(M) \in RE\backslash S\}$$

כי לכל $S \subseteq RE$, M מסוימת , $L\,(M) \in S$ או $L\,(M) \in RE\backslash S$

- כעת, $\phi \in S$, אז בהכרח $\phi \notin \bar{S} = RE\backslash S$.
- לכן $L_{\bar{S}}$ אינה ב R בגלל המקרה הקודם.

- בפרט $\bar{S}$ אינה טריוויאלית כי S אינה טריוויאלית
- מסגירות של R למשלים, גם $L_S = \overline{L_{\bar{S}}}$ אינה ב R (אחרת גם $L_{\bar{S}}$ הייתה ב R). מש"ל.

$\square$

דוגמאות שימוש:

דוגמה 1 : נגדיר $L_\varepsilon = \{\langle M\rangle \,|\text{M accept } \varepsilon\}$ (מקבלת את ε) , נראה ש $L_\varepsilon \in RE\backslash E$

הוכחה:

- כדי להראות שייכות ל RE , נתן לבנות מ"ט M_ε המקבלת את L_ε שעל קלט $\langle M\rangle$ מריצה את M על ε ועונה כמוה (בפרט אם M לא עוצרת, גם היא לא תעצור).
- כדי להראות $L_\varepsilon \notin R$ נשתמש במשפט $Rice$.
- נגדיר תכונה לא טריויאלית של שפות S כך ש $L_\varepsilon = L_S$ במקרה זה: $S = \{L \in RE|\varepsilon \in L\}$. כדי להשתמש ב $Rice$ צריך להראות ש S לא טריויאלית
  - $S \neq RE$: $\phi \in RE\backslash S$ לדוגמה
  - $S \neq \phi$, למשל : $\sum^*, \{\varepsilon\}, \{\varepsilon, 01101\} \in S$
- לכן מ $Rice$ נסיק ש $L_\varepsilon \notin R$

דוגמה 2 :

נגדיר $L'_\varepsilon = \{\langle M\rangle \,|\text{M doesnt accept } \varepsilon\}$ (לא מקבלת את ε)

האם ניתן להשתמש ב$Rice$ כדי להוכיח שאינה ב R (למעשה היא השפה המשלימה של L_ε ולכן אינה ב R מסגירות למשלים - אבל כאן מעניין אותנו ספציפית $Rice$). אפשר . נגדיר

$$S = \{L \in RE|\varepsilon \notin L\}$$

גם כאן S לא טריוויאלית. (אפשר להפוך סימנים בין הדוגמאות ל L_ε)

דוגמה 3 :

$$L"_\varepsilon = \{\langle M\rangle \,|\text{M reject } \varepsilon\}$$

האם ניתן להשתמש כאן ב $Rice$ כדי להראות ש $L_\varepsilon \notin R$ (זה לכשעצמו נכון נכון)? כאן זה לא אפשרי, כלומר לא קיימת S מתאימה כך ש $L"_\varepsilon = L_S$. אינטואיטיבית $L"_\varepsilon$ בודקת תכונה של M שאינה רק תכונה של $L(M)$. אפשר להראות זה במפורש: נראה זוג מכונות $\langle M_1\rangle, \langle M_2\rangle$ שאחת ב $L"_\varepsilon$ והשניה לא, אבל $L(M_1) = L(M_2)$

$M_1(y)$: מיד עוצרת ודוחה:$L(M_1) = \phi$ וגם $\langle M\rangle \in L_\varepsilon$ (דוחה את ε)

$M_2(y)$: למשל תמיד נכנסת ללואה על q_0 . $L(M_2) = \phi$ כאן $\langle M_1\rangle \notin L"_\varepsilon$ (לא דוחה את ε)

נשים לב ש L'_ε שראינו היא אינה ב RE . $Rice$ נתן לנו חלק מהאבחנה - מוכיח ש L'_ε . כדי להראות ש $L'_\varepsilon \notin RE$, נתן להסתמך על כך ש $L_\varepsilon \notin RE$ (דוקא אבחנה "חיובית")

וכיון ש $L_\varepsilon = \overline{L'_\varepsilon}$, נסיק מידית שלא יתכן $L'_\varepsilon \in RE$, אחרת היה $L'_\varepsilon \in RE \cup CoRE = R$ בסתירה ל $L_\varepsilon \notin RE$ ל $\overline{L_\varepsilon} \in R$ (סגירות R למשלים)

נראה שיטה נוספת להראות ש $L'_\varepsilon \notin RE$.

0.7.5 משפט 7.5 ל RE $Rice$:

תהי S תכונה לא טריוויאלית של שפות ב RE אז אם $\phi \in S$ אז $L_s \notin RE$

הערה: זה אפיון מלא כמו במקרה של R . ידוע אפיון, אבל לא לא נלמד אותו בקורס.
הוכחה:

- נראה ש $\overline{HP} \leq L_S$, ונסיק ש $L_S \notin RE$,כי $\overline{HP} \notin RE$, ברידוקציה
- $f(\langle M\rangle, \langle X\rangle) = \langle M_x\rangle$ כאשר $M_x(y)$:

1. מריצה את M על x
2. תהי $L \in RE\backslash S$ קיימת כזו כי S לא טריוויאלית. תהי M_L מכונה שמקבלת את L. M_xתריץ את M_L על y ותענה כמוה.

<u>נכונות הרדקוציה:</u>

- f מלאה ונתנת לחישוב - פעולת קומפילציה
- תקפות:
  - $\underline{(\langle M\rangle, \langle x\rangle) \in \overline{HP}}$ $\Leftarrow$ M לא עוצרת על x $\Leftarrow$ M_x תתקע בשלב 1 (לכל y) $\Leftarrow$(מהנחה) $L(M_x) = \phi \in S$ $\Leftarrow$ $\langle M_x\rangle \in L_S$, כנדרש.
  - $(\langle M\rangle, \langle x\rangle) \notin \overline{HP}$ $\Leftarrow$ M עוצרת על $x$$\Leftarrow$ M_x תגיע ל 2 (לכל y) ואז תענה כמו M_2 (בחירת L)$\Leftarrow$ $L(M_x) + L \notin S$ $\Leftarrow$ $\langle M_x\rangle \notin L_S$, כנדרש.

מש"ל

הערה: התנאי במשפט אכן מספיק אבל לא הכרחי. לדוגמה $L_{\sum^*} = \left\{\langle M\rangle | L(M) = \sum^*\right\}$ ($S = \left\{\sum^*\right\}$) היא מהצורה L_S, אבל S אינה מקיימת ש $\phi \in S$, ועדיין $L_{\sum^*} \notin RE$

0.8 הרצאה 8 - 19/12/12 יום חמישי

מה נעשה היום:

- נסיים לדבר על $Rice$
- נגדיר מ"ט א"ד . נראה שמוש מסוים שלה לחישוביות בעיקר היא תהיה משמעותית בחלק של סבוכיות
- נתחיל לדבר על חישוב יעיל (סבוכיות חשוב) . נגדיר את המחלקות P, NP

$Rice$ - המשך

בשעור הקודם נסחנו את משפט $Rice$ ל Re , R והוכחנו את שניהם. המשפט עבור R נתן אפיון מלא ל $L_s \in R$ עבור Re המשפט סיפק תנאי מספיק אבל לא הכרחי לאי שייכות ל RE .
מדוע התנאי לא הכרחי (אם $\phi \in S$ ת $L_s \notin RE$)
למשל $L_{\sum^*} \notin RE$, וכי מהצורה $L_{\sum^*} = L_s$,אבל $\phi \notin S$.
מהי S המתאימה כאן? $S = \left\{\sum^*\right\}$, לא מכילה את ϕ .

בשעור הקדם ראינו הוכחה לכך ש $L'_\varepsilon = \{\langle M\rangle \text{M not accept } \varepsilon\}$ - היא ב RE , ע"י שימוש ב $Rice$ ל R + אבחנה ש $\overline{L_\varepsilon}$ ב RE . $Rice$ עבור RE מאפשר להוכיח זאת ישרות .

- $L'_\varepsilon = \{\langle M\rangle | \varepsilon \notin L(M)\}$ כלומר $L'_\varepsilon = L_S$ כאשר $S=\{L\in RE | \varepsilon\notin L\}$. בפרט $\phi \notin S$. S לא טריואילי:
  - $S\neq RE$: למשל $\{\varepsilon\},\{0,\varepsilon,1^{100}\}\in RE\backslash S$
  - $S\neq\phi$: למשל $\phi,\{1,100\}\in S\cap RE$

נראה מספר דוגמאות ש $Rice$ לא שמושי:

- $L_u=\{\langle M\rangle,\langle X\rangle \text{ M accept x}\}$ - לא ניתן להשתמש ב $Rice$ כדי להראות אי שייכות ל R , כי זו שפה של קדודי נכונות + דקדוק קלט ובכלל לא קדודי מכונות.
- $L=\{\langle M_1\rangle\langle M_2\rangle \neq L(M_1)\neq L(M_2)\}$ גם כאן $Rice$ לא מתאים, כי מדובר בזוג קידודים של מכונה.
- דוגמה מעניינת : נזכיר $L_D=\{\langle M\rangle|\text{M accept }\langle M\rangle\}$, ידוע ש $L_D\notin R$ (מלכסון - שפה השונה שראינו שאינה ב R). האם אפשר להשתמש ב $Rice$ כדי להראת ש $L_D\notin R$?

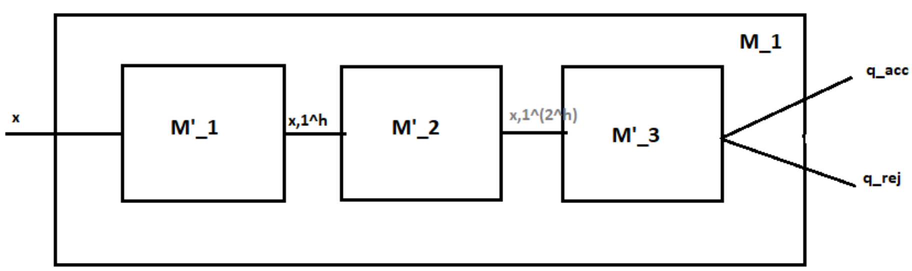
  לא, כי L_D מגדרה תכונה של מכונות שהיא מעבר לשפה שלהן. כיצד נראה זאת?

  נראה זוג מכונות $\langle M_1\rangle,\langle M_2\rangle$ כך ש $L(M_1)=L(M_2)$ אבל $\langle M_1\rangle\in L_D$ ו $\langle M_2\rangle\notin L_D$ זה יוכיח לנו שאכן L_D מגדירה תכונה של שפה של מכונה.

  רעיון הבניה:

  1. נדאג ש:
     - $L(M_1)=L(M_2)=L$
     - $\langle M_1\rangle\notin L$ ($\langle M_1\rangle\in L_D$)
     - $\langle M_2\rangle\notin L$ (M_2 לא מקבלת את $\langle M_2\rangle \Leftarrow \langle M_2\rangle\notin L_D$)
  2. למעשה נדאג ש L תהיה שפה של מ"ט בעלת $\geq K$ מצבים עבור K מספר כלשהו (די גדול)

  - נניח שהצלחנו לבנות M_1 , כך ש $L(M_1)\in L$ ו $\langle M_1\rangle\in L$ (תכיף נראה איך). אז נוכל לבנות M_2 מתאימה באופן הבא:
  - נוסיף ל M_2 $K+2$ מצבים שאינם ישיגים מ q_0 אז מס' המצבים של M_2יהיה $K <$ אבל השפה שלה היא כמו השפה של M_1 בפרט $\langle M_2\rangle\notin L$
  - נשאר למצוא K מתאים ולבנות M_1 מתאימה. כיצד נבנהM_1 כזו?
  - הקשוי הוא לדאוג ש M_1 עצמה היו לכל היותר K מצבים. נציע בניה ל M_1 : M_1 תורכב מ 3 מכונות, כך:



  - M'_1 : בכמה מצבים ניתן עשות את M'_1 ? ב $h+2$ מצבים: h מצבים כתיבת $1\cdot h$ ים , 2 להגיע לסוף x ולרשום "1"
  - M'_2 : דורש מס' קבוע מסוים a_2 של מצבים (a_2 לא תלוי ב h) המכונה מקבלת $x,1^y$ ומחזירה $x,1^{2^y}$
  - M'_3 : על קלט $x,1^z$ מקבלת אם"ם x הוא קידוד $\langle M\rangle$ של מ"ט M בעלת $\geq z$ מצבים. ניתן לממש ב a_3 מצבים, גם a_3 לא תלוי ב h
  - מה מספר המצבים של M_1 (כפונקציה של h) ?

$$h+2+a_2+a_3$$

מהי $L(M_1)$? $\{\langle M\rangle \,|\text{M has} \le 2^h \text{ states}\}$

כדי ש $\langle M_1\rangle$תהיה ב $L(M_1)$, נקבע את h להיות מספיק גדול כך ש: $h+2+a_2+a_3$ (כי 2^h גדל הרבה יותר מהר מ h) לדוגמה אם: $a_2=a_3=200$,

נדאג ש: $h=9 \Leftarrow 402 \le 2^h - h$ (או יותר) כבר עובד ואז $k=2^9=512$, מש"ל

מכונת טיורינג אי־דטרמינסטית

0.8.1 הגדרה: מכונת טיורינג אי־דטרמינסטית

מכונת טיורינג אי־דטרמינסטית תוגדר כמו מ"ט דטרמניסטית, אלא ש δ תאפשר בכל צעד בדיוק שתי בחירות במקום בחירה אחת. כלומר:

$$\delta : (Q\backslash F)\times\Gamma \to (Q\times\Gamma\times\{L,S,R\})^2$$

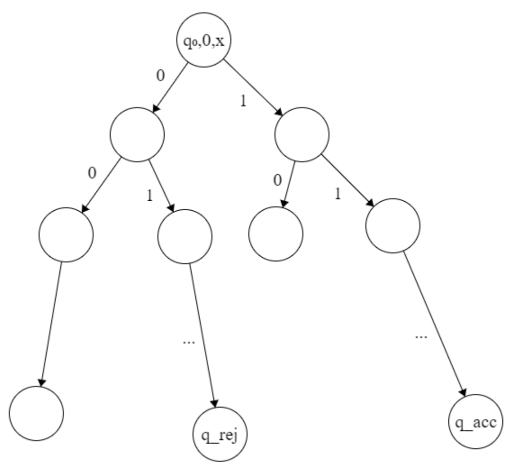
מ"ט א"ד מגדירה על קלט x עץ חישוב במקום מסלול חישוב:

כיצד נגדיר את $L(M)$ למ"ט א"ד ?

0.8.2 הגדרה : $L(M)$ למ"ט א"ד

הגדרה 8.2 : נאמר ש מ"ט א"ד M מקבלת מילה $x \iff$ קיים מסלול בעץ החישוב של M על x בו M עוצרת ב q_{acc} (יתכנו מסלולים אחרים בהם היא עוצרת ב q_{rej} או לא עוצרת)

כמו קודם , נגדיר $L(M)=\{x\in\sum^* \,|\text{M accept x}\}$



<u>הערות:</u>

1. החישוב של M על x תמיד מתנהל לפי סדרת בחירות כלשהן בδ (שמותרות) המגדירה מסלול חישוב בעץ. בפרט , בכלל לא מובטח שנגיע למסלול מקבל דוקא: לכן להריץ מ"ט א"ד לא ייתן לנו בהכרח את התשובה הנכונה.

   בפרט הריצה יכול לא לעצור אפילו עבור $x\in L$, אם היינו מממשים בחירה δ ע"י הטלות מטבע, עבור $x\in L$ מובט רק שההסתברות לקבל היא לא 0

2. המודל כאן קצת שונה מהגישה באוטומטים ששם δ יכלה לפלוט קב' סופית כלשהי של מהלכים, בפרט ϕ כאן הקבוצה היא בדיוק בגודל 2 . זה בלי הגבלת הכלליות.

נגדיר את RE_{ND} ־ כקבוצת השפות שקיימת מ"ט א"ד המקבלת את L = $\{L|\text{Exist non-detemenstic turing machine accpet L}\}$

נראה שאי־דטרמינזם בעצם לא מוסיף כח בהקשר של חשוביות

0.8.3 $RE = RE_{ND}$

הוכחה (סקיצה) - הכלה דו־כיוונית

<u>כיון קל - $RE \subseteq RE_{ND}$:</u>

- תהי M דטרמינסטית, נבנה M' א"ד המקבלת את אותה שפה ע"י "שכפול של δ
- כלומר , לכל $\delta\left(q,a\right)=\left(p,b,m\right)$ נגדיר $\delta'\left(q,a\right)=\left(\left(p,b,m\right),\left(q,b,m\right)\right)$
- לא קשה להראות ש $L\left(M'\right)=L\left(M\right)$ (עץ החישוב של δ יורכב משכפולים של מסלול החישוב של M)
- אופציה אחרת: $\delta'\left(q,a\right)=\left(\left(p,b,m\right),\left(q_{a_j},b,m\right)\right)$ - שומרים על המסלול המקורי, ומוסיף הרבה מסלולים שנתקעים

<u>כיון שני - $RE_{ND} \subseteq RE$:</u>

<u>נסיון I :</u> תהי M מ"ט א"ד .נבנה M' דטר' שקולה באופן הבא: $M'(x)$ תבצע DFS על עץ החישוב של M על x (תפתח את עץ־הקונפיגרוציות) אם זיהתה 'קבל' תקבל מייד.

זה לא עובד, כי ייתכן שנתקע במסלול אינסופי לפני שנגיע ל q_{acc} .

<u>נסיון II:</u> נסרוק את עץ החישוב של M על x ב BFS . בצעד ה i נפתח את רישות המסלולים עד אורך i . אם נזהה קבלה, נקבל.

המשפט שהוכחנו יכול להקל על הוכחות ששפה מסוימת L כן ב RE ולהחליף הרצה מבוקרת. לדוגמה: נתבונן ב L מהצורה:

$$L=\{M \mid\mid |x_1|<|x_2|<|x_3| \text{ קיימת 3 מילים } x_1,x_2,x_3 \text{ ש } M \text{ מקבלת את } x_1,x_2 \text{ ודוחה את } x_2 \text{ וגם } \}$$

$L\in RE$ נתן להוכיח זאת ע"י בניית מ"ט שמבצעת הרצה מבוקרת, אבל גם להוכיח ע"י בניית מ"ט א"ד מתאימה.

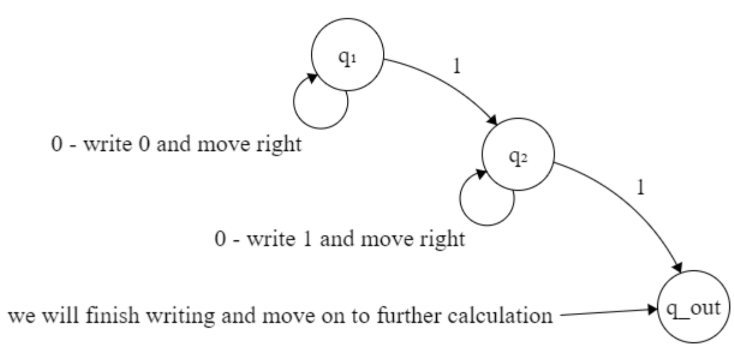
$M_L\left(\langle M\rangle\right)$:

1. תנחש מילים x_1,x_2,x_3
2. אם $|x_1|<|x_2|<|x_3|$ תריץ את M על $x_1,x_2,x3$ סדרתית.
   - אם M קבלת את x_1,x_3 ודחתה את x_2 נקבל
   - (אחרת, או נתקע אם M נתקעת על אחד הקלטים).

כעת נבהיר מהו הניחוש וכיצד ממשים אותו. נוכיח ראשית נכונות:

- אם $\langle M\rangle\in L$: קיימים x_1,x_2,x_3 מתאימים $\Leftarrow$ קיים ניחוש שמנחש את x_1,x_2,x_3 $\Leftarrow$ במסלול שבו מנחשים אותו M בהכרח תקבל $\Leftarrow$ קיים מסלול מקבל $\Leftarrow$ $\langle M\rangle\in L\left(M\right)$, כנדרש.
- אם $\langle M\rangle\notin L$: לא קיימים x_1,x_2,x_3 כנדרש $\Leftarrow$ לכל ניחוש x_1,x_2,x_3 לא נקבל (נדחה או נתקע)

כיצד נממש את תהליך הניחוש? (זה תמיד יעבוד), ננחש מחרוזת ש $0,1$ ים ובסוף נפרסר אותה בתור $x_1\underbrace{\ ,\ }x_2\underbrace{\ ,\ }x_3$ (אם יתאפשר , ויצאה לנו מחרוזת חוקית, אחרת מיד נדחה). בכל צעד נרצה להוסיף ביט $0/1$ או לעצור (סיימנו לכתוב את המחרוזת) , כלומר יש 3 אפשרויות אבל δ מרשה רק 2 . נממש ע"י שני מצבים (שגרת ניחוש):



חלק 2 - סבוכיות

תמונת עולם - $zoom-in$ על R :

מה זה יעיל מבחינתנו:

- נרצה התאמה למציאת מה שמחשבים ליעיל נרצה שיוגדר כיעיל ולהיפך.
- הרצה "נוחות מתמטית" : לדוגמה, אי־תלות במודל, תכונות סגור (אם ל L_1 יש אלג' יעיל אז גם ל $\overline{L_1}$)

אילו מדדים יענינו אותנו - זמן ריצה בעיקר, כמות זכרון שמשתמשים אקראית, וכו' .

נגדיר זמן ריצה כמס' צעדי החישוב שמ"ט מבצעת.

0.8.4 הגדרה:פונקצית זמן הריצה למ"ט דטרמינסטית

תהי M מ"ט דטרמניסטית. נגדיר את פונ' זמן הריצה שלה, tm $tm:\sum^* \to \mathbb{N}$ - פונקציה חלקית מוגדרת כך:

- $tm(x)$ - מס' הצעדים ש tm מבצעת בריצה על x אם עוצרת
- אחרת - לא מוגדר

0.8.5 הגדרה: חסם זמן ריצה לM

נאמר שפונקציה מלאה $t(n):\mathbb{N}\to\mathbb{N}$ היא חסם על זמן הריצה של M אם לכל $x\in\sum^*$ מתקיים ש$tm\left(x\right)\le t\left(|x|\right)$

0.8.6 הגדרה: M (מ"ט דטר') יעילה/פולינומית

<u>נאמר ש M (מ"ט דטר') היא יעילה או פולינומית</u> אם קיים עבורה חסם זמן ריצה $p(n)$ שהוא פולינום (בה"כ נתן להניח ש $p(n)=c{\cdot}n^d$ קבועים c,d*)

כלומר החלטנו להגדיר שמכונה היא יעילה אם היא רצה בזמן פולינומי.

האם ההגדרה הזו מתאימה למציאות?

- כן במובן של אלג'בעולם האמיתי , שנחשב ליעיל רץ בזמן פולונימי. בכיון השני לא בהכרח. למשל סבוכיות $2^{2^{2^{30}}}\cdot n^{100}$ לא שמושי אפילו לקלטים באורך 1 לעומת זאת $\left\lceil \frac{2^n}{2^{2^{2^{100}}}} \right\rceil$ או $n^{\log\log\log\log(n)}$ - לא יעיל מבחינת ההגדרה, אבל שניהם פרקטיים.

0.8.7 הגדרה: {קיימת מ"ט דטרמיסטית יעילה המכריעה את L | L } = P

{ קיימת מ"ט דטר' יעילה המחשבת את f | f }= $POLY$

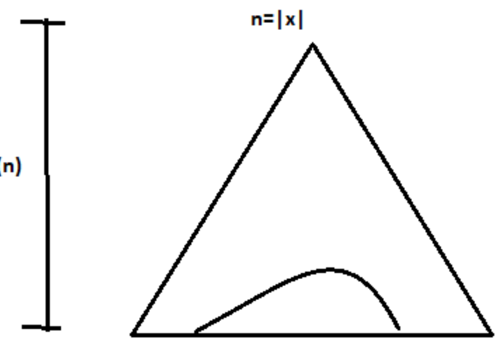
כעת נגדיר מ"ט א"ד יעילות

0.8.8 תהי M מ"ט א"ד נאמר ש $tm(x)$ היא פונק' זמן הריצה של M

נגדיר את הפונקציה $tm(x)$:

- אם M עוצרת על x בכל מסלול $\Leftarrow$ זה המקס' על פני כל המסלולים של מס' צעדים הריצה של M על x
- אחרת $\Leftarrow$ לא מוגדר

בדומה למ"ט דטר' חסם זמן ריצה $t(n)$ מוגדר כמו קודם. נאמר כמו קודם ש M א"ד יעילה אם קיים עבורה חסן זמן ריצה שהוא פולינומי



נגדיר את מחלקת השפות שקיימות להן מ"ט א"ד יעילה

0.8.9 NP = { קבוצת השפות שקיימת מ"ט א"ד יעילה המקבלת את L | $L\}$

0.9 שיעור 9 - 26/12/19

בשיעור הקודם הגדרנו, והיום נרצה להראות שבתמונת העולם:

$$P \subseteq NP \subseteq R$$

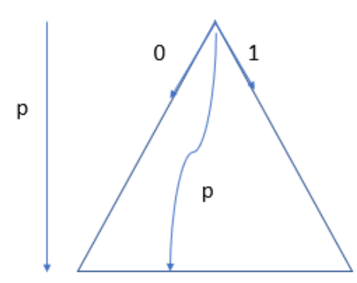
נזכיר:

P מחלקת כל השפות שקיימת עוברן מ"ט דטרמינטית פולינומית

NP - מחלקת השפות שקיימת עבורן מ"ט אי־דטרמינסטית פולינומית

0.9.1 9.1 : אכן $P \subseteq NP$, כיצד נוכיח?

בהנתן שפה L עם מ"ט M_L פולינומית המכריע אותה, נבנה מ"ט א" M' פולינומית המקבלת את אותה השפה: אפשר למשל "להכפיל" את δ . כלומר לכל כניסה $\delta(q,a) = (p,b,m)$ נגדיר $\delta'(q,a) = ((p,b,m),(p,b,m))$

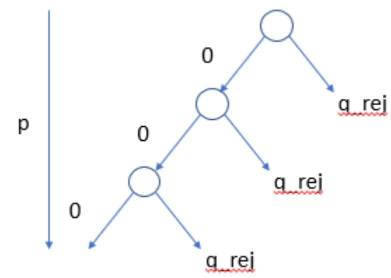


נקבל שנכונות:

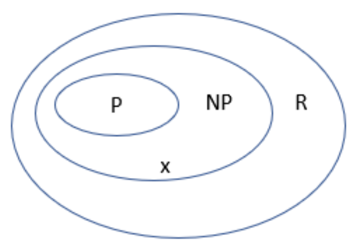
- עבור $x \in L$, M' תקבל בכל המסלולים (בפרט קיום מסלול מקבל)
- עבור $x \notin L$, M' תדחה בכל המסלולים, כנדרש.

בנוסף M' אכן יעילה עם חסם זמן ריצה זהה לזה של M

אלנטרנטיבית, אפשר היה גם להגדיר $\delta\left(q,a\right)=\left(p,b,m\right),\left(q_{rej},b,s\right)$



נחזור למפת העולם:



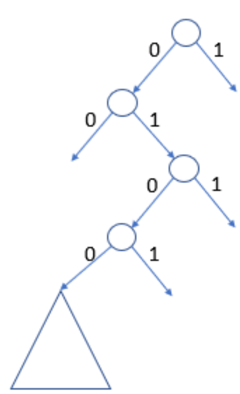
אחת השאלות הגדולות במדעי המחשב : האם קיימת שפות כמו x ? כלומר האם אי־דטרמינזם עוזר בחשוב יעיל? הרב מאמינים $P\neq NP$

0.9.2 **טענה:** 9.2 - $NP\subseteq R$

הוכחה: תהי $L\in NP$. תהי M_L מ"ט א"ד פולינומית המקבלת אותה. נבנה מ"ט דטר' M המכריעה את L .

נסרוק את עץ החישוב של M_L על x - למשל ב DFS (BFS) , אם מצאנו מסלל מקבל אז נקבל, אחרת נדחה. , מדוע זה עובד?

כל מסלול ב M_L הוא סופי אפילו (חסום ב $p\left(\left|x\right|\right)$ עבור פולינום p מסוים), לכן אין סכנה להתקע.



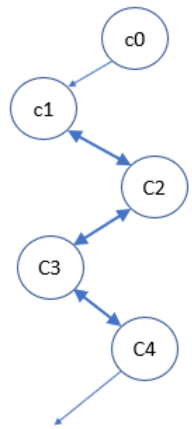
נפרט לגבי מימוש הסריקה: נשמור את רישת המסלול שאיתו עובדת כרגע:

בכל צעד נבחר האם מוסיפים 0 או 1 למסלול ולפי אלג' הסריקה של DFS על כל צעד מחשבים את הקונפיגוצריה הנוכחית (העוקבת של הקונפ' הנוכחית ששמרנו קודם).

כיצד נעדכן קונפ' ?

בדומה למ"ט אוניברסלית - לפי δ כדי לחזור אחורה נחזור לקונפ' האחרונה ששמרנו במסלול (עד כה).

כדי לחזור אחורה נחזור לקונפ' , נמחק את C_4 רק כאשר נחזור ל C_3



הדגמה חזרה אחורה.

הערה: הסימולציה הזו אינה לוקחת זמן פולינומי באופן כללי, כי אנחנו עוברים על כל עץ החישוב של M_2 על x ,במקרה הגרוע, ויש לו (במקרה הגרוע) $2^{P(n)}$ עלים. האלג' שבנינו רץ בזמן אקספוננציאלי.

מדוע NP היא בכלל מחלקה מעניינת?

הרי אם נריץ מ"ט א"ד יעילה על קלט מסוים, אז לא נוכל לדעת (במקרה שקבלנו q_{rej}) , האם $x \in L$ או $x \notin L$. יתרה מזאת, יתכן שישנו רק מסלול מקבל אחד עבור x מתוך $2^{p(n)}$ מסלולים, ואז אם נממש את המכונה ע"י הטלות מטבע, ההסת' להיות צודקים עבור x כזה קרובה מאוד ל 0 $\left(\frac{1}{2^{p(n)}}\right)$. נבין טוב יותר את המחלקה, ע"י הסתכלות אלטרנטיבית על NP (כמערכת הוכחה).

0.9.3 הגדרה 9.3 : נאמר ש $R \subseteq \sum^* \times \sum^*$ הוא יחס NP אם הוא מקיים :

1. R חסום פולינומית. קיים פולינום $p(n)$ כך ש $(x,y) \in R \Leftarrow |y| \le p(|x|)$

2. קיימת מ"ט (דטר') פולינומית המכריעה את השפה:

$$\widetilde{L}_R = \left\{(x,y) \in \sum^* \times \sum^* \mid (x,y) \in R\right\}$$

דוגמה 1ל : היחס $\left(x, x11x^2\right)$ או $\left(x, x11x^3\right)$ - הוא יחס חסום פולינומית עם חסם $P(n) = 4n + 4$ (שימו לב שהעלה בשלישית מכפילה אורך פי 3 לא בשלישית)

דוגמה 2ל: היחס שראינו בסעיף הוא גם נתן לזיהוי פולינומי (x^2 או x^3 נתן לחישוב יעיל ע"י אלג' כפל ארוך של כיתה ג' - כשמתארים אלגוריתם מספיק לתאר אלג' למודל REM כי האקסטרה חישוב במעבר למ"ט הוא פולינומי).

עבור יחס NP , R נגדיר את השפה L_R :

$$L_R = \left\{x \in \sum^* \mid \exists y \; R(x,y) = T\right\}$$

נגדיר :

$$\{L|\ L=L_R \text{ כך ש } R,NP \text{ - קיים יחס}\}=NP$$

מה האינטואיציה בהגדרה ? השפה L היא "קצת קלה" במובן שקיים רמז קצר לשייכות לL שגם ניתן לבדוק ביעילות. ואם $x\notin L$ אז אף רמז לא ישכנע את הבודק.

למשל העיסוק במתמטיקה הוא "שפה ב NP " במובן מסוים. נפרמל:

השפה :

$$\{x|*\}$$

$$nice-true-theorems=\{x|\ \left|x^5\right|\geq \text{ משפט במתמטיקה שהוא נכון וקיימת לו הוכחה באורך }\}$$

x^5 שרירותי - בגדול נרצה הוכחות קצרות אחרת אף אחד לא יכול /ירצה לוודא אותן (כולל מחשב)

קשה למצוא הוכחות משפטים שיש להם הוכחה קצרה (לכן יש מקצוע של מתמטיקאי, אבל כל אחד יכול לוודא הוכחות קצרות ביעילות באורך של x וההוכחה במבנה של הוכחות מתמטית.

הנקודה החשובה כאן היא שההוכחה קצרה , הוכחת ארוכות לא נוכל אפילו לוודא ביעילות ולכן זה לא מעניין. להוכחות קצרות זה מעניין כי לפעמים מצליחים להוכיח דברים, גם אם זה באופן כללי קשה. אם מישהו הצליח להוכיח משהו רוצים שכל אחד אחר יוכל לבדוק אותו ביעילות.

אז מהו יחס NP המתאים לשפה $nice-true-theorems$?

$$R\left(x,y\right)$$

כאשר x - משפט. y - הוכחה באורך $\left|x^5\right|$ למשפט - (כאן $p(n)=n^5$)

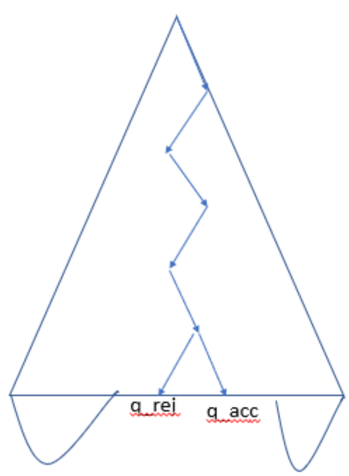
נוכיח שקילות של שתי הגדרות ל NP .

0.9.4 משפט 9.4 : הגדרות 9.3 ו 8.9 לNP שקילות

הוכחה: נראה הכלה דו־כיוונית (של המחלקות הרלוונטיות)

כיון 1 : תהי $L\in NP$ לפי הגדרה 8.9

- תהיה M מ"ט א"ד המקבלת את L בזמן החסום ע"י פולינום pm . נבנה יחס־NP R כך ש $L=L_R$
- $R(x,y)$ (אם $(x,y)\in R$ נקרא ל y "עד" (לשייכות x ל L_R)) y: הוא מסלול בעץ החישוב של M על xהמסתיים ב q_{acc} לדוגמה $y=1101101$ - מגדיר רצף בחירות של δ - מתוך 0 או 1) עד לשייכות x לשפה עבור הדוגמה להלן, אבל 110110 או 1101100



- ראשית נבדוק שאכן $L=L_R$:
  - $x\in L$: M מקבלת את $x \Leftarrow$ קיים מסלול y ב M המקבל את $x \Leftarrow$ (מהגדרת y - y המתאים למלסלול זה) קיים y כך ש $R(x,y)$
  - $x\notin L$: לא קיים מסלטל מקבל ב $M \Leftarrow$ כל y יקיים $(x,y)\notin R$
- נשאר לוודא ש R הוא אכן יחס NP :

1. חסום פולינומית: מהגדרת R , $(x,y)\in R \Leftarrow |y|\le p(|x|)$, החסם על זמן הריצה של M בכל מסלול
2. נתן לזיהי פולינומי בהנתן (x,y) , נסמלץ את ריצת M על x על המסלול המוגדר ע"י y .

- נפרט קצת מדוע החישוב אכן פולינומי - נקבל אם"ם המסלול y מסתיים ב q_{cc} :
  - הסימולציה מתבצעת בדומה למ"ט אוניברסלית. בכל מקרה, אורך המסלול שבודקים הוא $\le \min\left(\left(|y|,p(|x|)\right)\right)$ $|y|+p(|x|)$
  - במהלך ההרצה נשמור קונ' הנוכחית ונעדכן בכל פעם (תוכן סרט + מקום ומצב) , זמן החישוב:

$$\underbrace{\#\text{Calculation steps}}_{\le|y|+p(|x|)}\times\underbrace{\text{Next configuration's calculation time}}_{O(|x|+p(|x|)+y)*}$$

$*$ הסבר: בערך לינארי אורך הקונפ' , אורכה $O(..)$ כי בכל צעד חישוב הסרט גדל ב1 לכל היותר

- כלומר סה"כ מספר צעדים פולינומי בקלט $|(x,y)|$.

<u>כיון שני:</u> תהי L שפה ב NP לפי הגדרה 9.3 נראה $L\in NP$ לפי הגדרה 8.9

- יהי R יחס NP מתאים עבור L, ויהי $P(x))$ החסם מ1 ו $q(n)$ חסם זמן ריצה הריצה של אלג' המזהה את $\tilde{L}_R$ (מהגדרה $1,2$ שניהם פולינומים).
- נבנה מ"ט א"ד פולינומית עבור L :

<u>בניה:</u>

- $M(x)$: בגדול M תנחש את y , ותבדוק אם אכן $R(x,y)=True$
- נפרט:

1. M תחשב את $P(|x|)$
2. תהליך הניחוש: תקצה קטע : $\underbrace{*\_\_\_\_\_\_\_\_\_\_\_\_*}_{p(|x|)}$

   תעבור לתחילתו ותנחש מחרוזת של 0ים ו1ים שמסתיימת <u>לכל היותר</u> לפני ה $*$ השניה. (כלומר כל עוד לא הגענו ל $*$ "ננחש" ביט חדש b ונשרשר אותו, או שנעצור כאן. בכל מקרה כשעוצרים y הוא:

$$\underbrace{*\_\_\_\_\_\_\_\_|\downarrow....*}_{y}$$

3. תבדוק האם $R(x,y)=T$, ותקבל אם כן, אחרת תדחה.

<u>נכונות:</u>

1. יורכב משלושה שלבים:

(א) נמיר מ $|x|$ בעצם נתון באונארי, לבינארי. יקח בערך $O(|x|\log|x|)$ פעולות. (מעדכנים מונה באורך $\log|x|$) , נסמן את המונה ב Cnt .

(ב) חשוב הפולינום $p\left(Cnt\right)=Cnt^{d}\cdot c$ - פולינומי באורך המונה השוטף שהוא $d\left(\log\left|x\right|\right)$ בכל רגע עושים $d-1$ הכפלות כופל בקבוע. סה"כ $poly\left(\log\left|x\right|\right)$

(ג) חלק מ2 - נקצה את השטח * _ _ _ _ _ _ _ * שלמעשה דורש תרגום מבינארי חזרה לאונארי של $Cnt=p\left(\left|x\right|\right)$ בערך $p\left(\left|x\right|\right)\cdot poly\left(\log\left|x\right|\right)$. פעולות (ונוריד 1 מהמונה על כל תא שנקצה עד שנגיע ל 0)

2. הסברנו

3. לוקח זמן :

עבור הy שחישבנו:

$$q\left(\left|x\right|+\underbrace{\left|y\right|}_{\leq p\left(\left|x\right|\right)}\right)\approx O\left(q\left(\underbrace{p\left|x\right|}_{\text{A polynomal too}}\right)\right)$$

הסכום עדיין פולינומי

<u>נשאר להראות נכונות:</u>

- $x\in L$ קיים y כך ש $R\left(x,y\right)$ $\Leftarrow$ כיון ש M מאפשרת לנחש כל y שאורכו $\geq P\left(\left|x\right|\right)$, בפרט יהיה קיים מסלול (אפילו הרבה- בכלל השלבים הדטרמינסטים בחישוב אם נממש על ידי הכפלה) שבו M תנחש את ה y המתאים (אורכו $\geq p\left(\left|x\right|\right)$) כי $p\left(n\right)$ חסם הנובע מכך שR חסום פולינומית שקבענו) $\Leftarrow$ מקבלת את x $\Leftarrow$ $x\in L\left(M\right)$
- $x\in L$ $\Leftarrow$ לא קיים y מתאים $\Leftarrow$ (בניית M) M תדחה בכל המסלולים

$\square$

<u>דוגמאות לשפות ב NP</u>

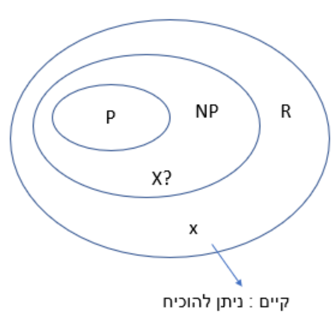
דוגמה :

$$factory=\left\{\left(x,k\right)|y\neq1 \text{ וכן } k\leq y<x \text{ המקיים } x \text{ של } y \text{ קיים מחלק } x,k\in\mathbb{N}^{+}\right\}$$

נראה ש $factory\in NP$. שימו לב ש $factoring$ הוא לא בדיוק המשלים של $Prime=\left\{x|\text{ x is prime }\right\}$

0.10 שיעור 10

תמונת עולם:



תזכורת:

$$factory=\left\{\left(x,k\right)|y\neq1 \text{ וכן } k\leq y<x \text{ המקיים } x \text{ של } y \text{ קיים מחלק } x,k\in\mathbb{N}^{+}\right\}$$

נראה ש$factory \in NP$

הוכחה:

נוכיח באמצעות בניית מ"ט א"ד יעילה.

בניה:

$M_{factoring}(n,k)$:

1. ננחש מספר y (באורך $\log_2 n \geq$ ביטים, עם אפסים מובילים). המספר הוא פשוט מחרוזת באורך n שבהמשך נפרסר כמספר.
2. נבדוק שאכן $y|n$ וכן $1 < k \leq y < n$ $(*)$. אם כן, נקבל אחרת נדחה.

נוכיח נכונות וסיבוכיות של הבניה:

<u>נכונות:</u>

- עבור $(n,k) \in factoirng$ $\Leftarrow$ קיים y המקיים את $(*)$. מהבניה, כל yשאורכו כאורך n (כלומר בפרט מכסה את כל המספרים $n \geq$ אבל רק אותם) $\Leftarrow$ קיים מסלול שבו מנחשים y מתאים והבדיקה שלו עובדת $\Leftarrow$ $(n,k) \in L\left(M_{factoring}\right)$, כנדרש.
- עבור $(n,k) \notin factoring$ $\Leftarrow$ לא קיים y המקיים את $(*)$ $\Leftarrow$ בכל מסלול y יתגלה ב2 כלא מקיים את $(*)$ $\Leftarrow$ $M_{factoing}$ $\Leftarrow$ תדחה בכל המסלולים $\Leftarrow$ $(n,k) \notin L\left(M_{factring}\right)$, כנדרש.

<u>יעילות:</u>

- נראה פולינום $p(n)$ כך שזמן הריצה בכל מסלול חסום ע"י $p(n)$:
  - ראשית, $|y| = |n| \leq |k|$ (x הקלט)
  - הבדיקה ב2 דורשת פעולת חילוק על מס' באורך $\underbrace{O(\log n)}_{|x|}$ ביטים גם פולינומי ב$|x|$ לפי פעולת חילוק שלמדנו בכיתה ג'
    - בערך $O\left(m^2\right)$ פעולות על ביטים , עבור חלוקה של שני מס' באורך m ביטים .

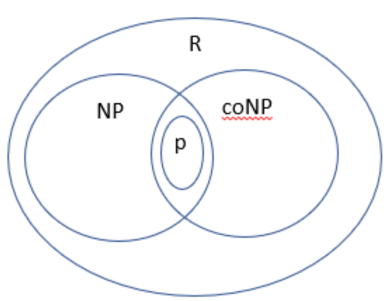
אפשר לשים לב, שבהגדרת NP ישנה אסימטריה :

- אם הקלט $x \notin L$, אכן מסלול דוחה .
- ואם $x \in L$ אז נדרש רק קיום של מסלול מקבל אבל הרבה מסלולים אחרים יכולים לדחות.

במילים אחרות, (לפי ההגדרה האלטרנטיבית) מובטח עד קצר לשייכות לשפה שנתן לבדוק ביעילות, אבל <u>לאי שייכות</u> אין (באופן כללי) עד כזה.

נגדיר את המחלקה "המקבילה" שבה יש עד $x \in \overline{L}$ שנתן לבדוק ביעילות ולא דוקא ל $x \in L$

0.10.1 הגדרה 10.1: $coNP = \left\{L | \overline{L} \in NP\right\}$



האם $NP=coNP$ היא גם שאלה פתוחה גדולה. (כלומר האם NP סגורה למשלים). למה היפוך מצבים לא עובד?
תהי M מ"ט א"ד יעילה עובר L . מהי $L\left(\overline{M}\right)$ (הופכים בין q_{acc} ל q_{rej})?

$$\overline{L}\cup * \{x\in L|x \text{ קיים מסלול הדוחה את } M\text{ב}\}$$

- ב$\overline{L}$ בוודאי יתקבל (בכל המסלולים , אפילו)
- ב$(*)$ - בהרבה מקרים, למשל במ"ט שלנו עבור $factoring$ זו כל L (יכנסו לנו עוד דברים)

לא קשה לראות ש $P\subseteq NP\cap coNP$

- תהי $L\in P$, אזי $\overline{L}\in P$ (P סגורה למשלים), נראה:
  1. $L\in NP$ ע"י בניית יחס NP מתאים.
     - למשל היחס הוא $\{(x,1)\,|x\in L\}$
     - חסום פול' - כי:
       (א) $|y|=1$ אם $(x,y)\in R$
       (ב) ניתן לזיהוי פולינומי ע"י בדיקת:
          i. $y=1$
          ii. $x\in L$ (אפשרי כי $L\in P$)
  2. $\overline{L}\in NP$ ע"י בניית יחס NP מתאים ל $\overline{L}$
     - באופן דומה נגדיר $R=\left\{(x,1)\,|x\in\overline{L}\right\}$ ....

לא ידוע האם $P=coNP\cap NP$ (מאמינים שלא) . למה לא קל לבנות מ"ט יעילה לשפה $L\in coNP\cap Np$ כלשהי?
ידוע שיש עד לשייכות ל L , ועד לאי שייכות אבל עדיין לא ברור שנתן למצוא את העד ביעילות.

ישנן שפות מעניינות שהן כן ב $NP\cap coNP$ ולא ידוע שהן ב P - למשל $factoring$. נראה שאכן $factoring\in NP\cap coNP$
$factoring\in NP$ הראנו, נותר להראות שייכות ל$coNP$.
<u>הוכחה:</u>

- נבנה יחס NP עבור $\overline{factoring}$, כיצד יראה עד y' לטענה $(n,k)\notin factroing$
  - $y'=(p_1,e_1),...,(p_1,e_k)$ (הפירוק של n לגורמים ראשונים), כאשר $p_1<p_2<...<p_t$ ראשוניים
  - מתקיים ש: $n=\prod_{i=1}^{t}p_i^{e_i}$
- $R((n,k),y')$ - צ"ל ש R אכן יחס NP עבור $\overline{factroing}$ (ענת בלבלה את הסדר בסה"כ צ"ל אלגוריתם + נכונות +יעילות):
  1. אכן $\{x|\exists yR(x,y)\}=\overline{factroing}$, כי בדקנו בעזרת (y') האם המחלק הכי גדול של n k קטן/שווה או לא (ב $\overline{factoring}$ לא)
  2. R חסום פול': (כי t קטן)
  3. נתן להזיהוי פולי' -הסברנו תוך כדי
- האלגוריתם:
  1. יבדוק שאכן $p_1<...<p_t$, וכולם ראשוניים. זה יעיל בדיקת ראשונית היא ב P (נמצא אלג' AKS רק ב 2002 שלוקח לפחות $\Omega\left(n^6\right)$ זמן)

2. בנוסף יבדוק שאכן, $n=\prod_{i=1}^{t}p_i^{e_i}$ זה יעיל כי $\sum e_i$ בפירוק אמיתי היא לכל היותר $\log_2 n = O\left(|x|\right)$ ו נבצע לכל היותר מס' כזה של כפליל על מס' באורך $O\left(|x|\right)$ (ה p_i ־ים)
3. לבסוף נבדוק האם $y=\frac{n}{p_1}$ (p_1 המחלק הכי גדול) מקיים $y|n$ ו $1<k\leq y<n$ אם אכן לא מתקיים, מחזיר "כן" אחרת מחזיר "לא"

NP היא מחלקה של שפות (מהגדרה 2) שיש הוכחה קצרה לשייכות לשפה. כלומר יש מע' הוכחה:

$$\overbrace{P_{rovier}(x)}^{\text{omnipotent}} \overset{\text{y}}{\longrightarrow} \overbrace{V_{erifier}(x)}^{\text{efficient}}$$

בודק האם $R\left(x,y\right)$

המע' מקיימת:

1. (חשוב) V יעיל
2. שלמות: לכל $x\in L$ קיימת הוכחה y שתשכנע את V
3. נאותות: לכל $x\notin L$ לא קיימת הוכחה y שתשכנע את V

מסתבר שעבור $coNP$ גם נתן לבנות מע' הוכחה עם V יעיל, שלמות ונאותות אם נרשה יותר אינטראקציה בין הצדדים (יותר מהודעה אחת והסתברות טעות קטנה בכל כוון)

כלומר לדוגמה:

- שלמות: $\forall x\in L$, V מקבל בהסתברות ≤ 0.99 כאשר מתקשר עם המוכיח.
- נאותות: $\forall x\notin L$, V דוחה בהסתברות ≤ 0.99 <u>לכל</u> מכויח שהוא P^{+}

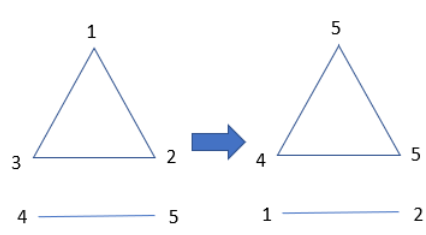
נראה דוגמה: נגדיר

$$GI=\left\{ \left(G_1,G_2\right)|G_i\text{undirected graph,}G_2\text{isomorphic to }G_2\right\}$$

תזכורת: G_2 איזומורפי ל G_1אם קיימת פרמוטציה π על V_1 כך ש:

$$G_2=\left(V_1,E_1\right)=\left\{ \left(\pi\left(u\right),\pi\left(v\right)\right)|\left\{ u,v\right\} \in E_1\right\}$$

לדוגמה:



שאלת חימום

האם $GI\in NP$? כן. העד המתאים הוא פרמוטציה π

$\Leftarrow$ $\overline{GI}\in coNP$. נבנה מע' הוכחה "מורחבת" עבור $\overline{GI}$ (למרות שלא ידוע ש $\overline{GI}\in NP$)

1. בודק ש $|V_1|=|V_2|$, אחרת מיד דוחה.

2. חוזר 7 פעמים מגריל באקראי ביט $b \leftarrow \{1,2\}$, מגריל פרמוטציה אקראית על π, V_1 מחשב $G' = \pi\left(G_2\right)$

$$P \underset{\xrightarrow{\text{guess b' for b}}}{\xleftarrow{G' = \pi\left(G_2\right)}} V$$

אם P תמיד צדק, נקבל. אחרת נדחה.

- לא קשה לראות ש V יעיל:
- <u>שלמות:</u> אם G_1, G_2 לא איזו' , אז בכל איט' G' איז' f G_b אבל לא ל G_{3-b} וכך P ,ע"י בדיקת איזו' ידע תמיד להגיד נכון מהו $b \Leftarrow v$ יקבל בהסת' 1 .
- <u>נאותות:</u> אם G_1 איז' ל G_2 , אז G' היא תמיד גרף אקראי האיזומורפי לשניהם. כלומר G' לא מכיל שום מידע לגבי b (מתפלג באופן בלתי תלוי ב b) לכן ההסת' של p לנחש את b בכל איט' היא $\frac{1}{2}$. ההסתרות לא לעטת בכל האיט' היא $\left(\frac{1}{2}\right)^7 = \frac{1}{128} < 0.01$, זו ש V יקבל למרות ש $G_1 \cong G_2$

NP - שלמות

מסתבר שקיים ב NP "גרעין קשה" של שפות כך שכל שפה L ב"גרעין" מקיימת ש $P = NP \Leftrightarrow L \in P$. כלומר מספיק להבין האם P השפה שייכת ל P עבור שפה כלשהי בגרעין, ולהסיק ממנה לגבי כל המחלקה. (זה פשוט מסוים של הבעיה). כדי להגדיר את הגרעין נזדקק למושג של רידוקציה פולינומית:

0.10.2 הגדרה 10.2:פונקציית רידוקמה פולינומית

הגדרה 10.2: יהיו L_1, L_2 שפות, נאמר שf היא פונקציית רידוקציה פולינומית מ L_1 ל L_2 אם:

1. $f \in Poly\left(y\right)$ (בפרט f מלאה)
2. תקפות: $\forall x \in \sum^* \ f(x) \in L \Leftrightarrow x \in L_1$

אם קיימת f כזו עבור L_1, L_2 אמר ש L_1 נתנת לרדוקציה פולי' ל L_2 נסמן $L_1 \leq_p L_2$

0.10.3 משפט הרדוקציה 10.3: יהיו $L_1 \leq_p L_2$. אזי $L_1 \in P \Leftarrow L_2 \in P$

כעת נגדיר את הגרעין הקשה, שהוא קב' השפות השלמות ב NP תחת רדקוציות פולינומית.

0.10.4 הגדרת NPC

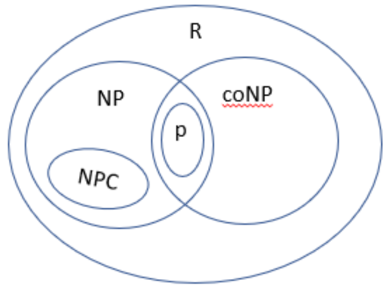
המחלקה NPC היא קב' כל השפות L המקיימות:

1. $L \in NP$
2. L שלמה ב NP ,כלומר $\forall L' \in NP$, $L' \leq_p L$

0.11 הרצאה 11

תזכורת : NPC אם היא קב' השפות המקיימות:

1. $L \in NP$
2. $L \in NPH$. כלומר (הגדרת NPH): $L' \leq_P L$ לכל $L' \in NP$



הגדרנו "גרעין קשה" של NP תכף נראה שלכל $L \in NPC$ (בפרט נראה $NPC \neq \phi$) $L \in P \Leftrightarrow NP = P$ המושג מגדיר באופן בלתי תלי במקביל ע"י $cook$ ו $levin$.

מדוע המושג של NPC הוא מעניין?

כאמור , מאפשר לפשט את הבעיה של האם $P = NP$ (מחלקות של שפות) לשאלה האם $L \in P$ עובר שפה מסוימת

* (יותר פרקטי): מספק "מצב קושי" עבור שפות "חדשות" שאנחנו פוגשים (בד"כ הן היהיו ב NP) הן קשות. כלומר אם נוכיח שבעיה (שפה) L שנרצה לפתור היא בNPC , נדע לא לנסות לבנות עבור אלג' פולינומי. נצטרך לחפש פתרונות חלופיים. למשל אלג' קירוב או יוריסטיות (אלג שלא עובד לכל הקלטים וגם בד"כ לא יודעים לנתח מתי הוא עובד, אבל בפועל הוא עובד בזמן סביר להרבה קלטים שימושיים בעולם ה"אמיתי"

המשך דיון ברדוקציות פולינומיות.

0.11.1 תכונות $\leq_P$

אבחנה היחס $\leq_P$ (יחס בינארי המוגדר על קב' השפות $P\left(\sum^*\right)$ מקיים:

1. רפלקסיבי : לכל $L \in P\left(\sum^*\right)$ מתקיים $L \leq_P L$ - ע"י פונ' הזהות $f(x) = x$שהיא כמובן ב $POLY$
2. לא סימטרי:
   (א) נקח:$L_2 = HP, L_1 = \sum^*$:
   - אז $L_1 \leq_p L_2$ (נמפה הכל למילה קבוע ב HP)
   - בכיון השני $HP \not\leq_p \sum^*$ כי אפילו $HP \not\leq \sum^*$ - (בגלל תקפות - "אין לאן" למות מילים ב HP)

   (ב) נשים לב שכל $L \in RE$ (ובפרט כל $L \in NP$, כי $NP \leq RE$) מקיימת: $L \leq_P L_u$.

   מדוע? עבור L נתונה תהי M_L מ"ט המקבלת את L . אזי $f\left(x\right) = \left(\left\langle M_L\right\rangle, \left\langle x\right\rangle\right)$ היא פונק' רידוקציה מתאימה שבפרט נתנת לחישוב בזמן פולינומי

   - בכיון השני, אם נבחר $L \in R$ כלשהי, $L \leq_P L_u$ (אחרת נקבל סתירה למשפט הרדוקציה, כי ידוע ש $L_u \notin R$)
3. טרנזיטבית:

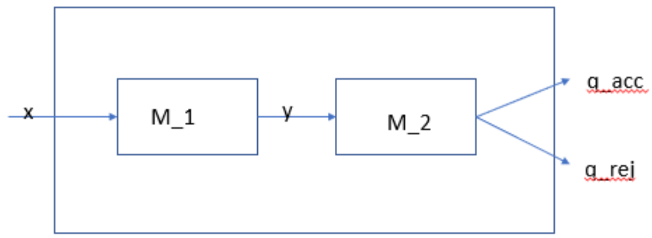
- אכן אם, $L_1 \leq_p L_2$, וגם $L_3 \leq_p L_3$ אז $L_1 \leq_p L_3$: נרכיב את פונק' הרדוקציה בין 1 ל 2 לזו בין 2 ל 3. בפרט הפונצקיה החדשה נתנת לחישוב (בזמן) פולינומי (החסם הוא הרכבה של החסמים הפולי' לשתי פונק' שהוא גם פולינום)

המשפט המרכזי בהקשר של NPC הוא משפט הרדוקציה לפונ' רדקוציה פונילנומיות. נזכיר:

משפט הרדוקציה 10.3: **יהיו** $L_1, L_2 \subseteq \sum^*$ **כך ש** $L_1 \leq_p L_2$ **. אזי** $L_2 \in P \Leftarrow L_1 \in P$

הוכחה(דוגמה להוכה של משפט הרדקוציה המקורי):

נבנה מ"ט פול' יעילה עבור L_1 :



M_1 מ"ט יעילה (דטר) עבור f $(f \in POLY)$

נתוח הבניה:

- נכונות:
  - תקפות f: $x \in L_1 \underbrace{\Leftrightarrow}_{f} f(x) \in L_2 \underbrace{\Leftrightarrow}_{M_1} M_1$ מקבלת את x
- יעילות: יהיו p_1, p_f פולינומים החוסמים את זמן הריצה של M_2, M_f בהתאמה.
  - אז M_1 רצה בזמן $\geq \underbrace{p_2\left(p_f\left(|x|\right)\right)}_{\text{because: } |y| \leq p_f(|x|)} + \underbrace{p_f\left(|x|\right)}_{\text{f(x) - calculation time}}$

נוכיח שאכן מספיק להכריע את הסטטוס של בעיה כלשהי ב NPC (מבחינת שייכות ל P) כדי להכריע האם $P = NP$

0.11.2 משפט 11.2 : **תהי** $L \in NPC$ **אז** $L \in P \Leftrightarrow P = NP$

הוכחה

כיון 1 :

- נניח $L \in P$ מהגדרת NPC (חלק 2)
- לכל $L' \in NP$ $L' \leq_p L$
- עכשיו ממשפט הרידוקציה $L' \in P \Leftarrow P = NP$ (כי L' שפה כלשהי ב NP)

כיון 2 :

- נניח ש $P = NP$, אז מהגדרת NPC (חלק 1) $L \in NP = P$ -

לאן נמשיך מכאן בקורס?

נראה עבור בעיות שונות שהן NPC . הבעיות יגיעו מכל מיני תחומים: גרפים, לוגיקה,בעיות אלגבריות. זה מעניין כי נפתח אינטואיציה כיצד נראות בעיות קשות (במובן של NPC) . בעתיד, כשנתקל בבעיה חדשה ונרצה לקבל עדות לקושי שלה ונצליח להוכיח שהיא ב NPC זו תהיה עדות "הכי טובה שאפשר" לכך שהבעיה לא ב P (לפחות לא ננסה לפתח אלג' יעיל)

כיצד נוכיח שהבעיה היא NPC ?

1. ישירות (נראה עכשיו)
2. ע"י רידוקציה מבעיה שכבר ידועה ב NPC

0.11.3 טענה 11.3 **: תהי** $L\in NPC$ **ותהי** $L'\in NP$ **. אזי אם** $L\le_p L'$ **אז** $L'\in NPC$

הוכחה ־נראה ש L'מקיימת את שתי הדרישות

1. $L'\in NP$ (נתון)

2. $L'\in NPH$.כלומר, מקיימת $L\le_p L'$, וגם לכל $L"\in NP$

$L"\le_p L$ (כי $l\in NPC$ ובפרט $L\in NPH$) כעת מטרנזיטביות של $\le_p$, $L"\le_p L'$ כלומר $L'\in NPH$ (כי $L"$ שפה כלשהי ב NP)

0.11.4 הגדרה 11.4 **: השפה** $BH-Bounded\ Halting$ **:**

$$BH=\{\langle M\rangle,1^p,\langle x\rangle \mid M \text{ מ"ט א"ד קיים ב } M \text{ מסלול מקבל בריצתה על } x \text{ שאורכו } \le l\}$$

0.11.5 משפט 11.5 **:** $BH\in NPC$

הוכחה:

רעיון ההוכחה (לפחות לחלק של NPH) דומה לשלמות של L_u ב RE שראינו . בפרט לא מאוד מפתיע שהבעיה ב NPC (נובע כמעט מיידית מהגדרת NP)

פורמלית:

יש להראות ע"פ הגדרת NPC :

1. <u>$BH\in NP$</u>

נבנה א"ד פולינומית מתאימה:

$M_{BH}\left(\langle M\rangle,1^P,\langle x\rangle\right)$:

1. תנחש מסלול חשוב בעץ של M על x שאורכו $\le l$ (ראינו איך עושים זאת)

   כלומר המחרוזת y שמנחשים היא מהצורה $y=y_1,y_2,...y_t$, כאשר y_i מגדיר אם בוחרים אפשרות 0 או 1 ב δ בצעד i

2. מריצה את M על x במסלול המוגדר ע"י y אם המסלול מסתיים בדיוק כך שy מסתיים, ומסתיים ב q_{acc} נקבל ,אחרת נדחה

נכונות הבניה:

<u>נכונות :</u>

$\left(\langle M\rangle,1^P,\langle x\rangle\right)\in BH$ $\iff$ ב M קיים מסלול y באורך $t\le l$ המקבלת את x $\iff$ (מהבניה) M_{BH} תנחש מסלול y מקבל כזה $\iff$ $\left(\langle M\rangle,1^P,\langle x\rangle\right)\in L\left(M_{BH}\right)$

<u>יעילות:</u>

- שלב 1 : ניחוש y : $O\left(l\right)$ צעדים
- שלב 2 (הרצה) : המכונה האוניברסלית שבנינו על קבלט $\langle M\rangle,\langle x'\rangle$ אכן רצה בזמן פולינומי ב $|M|$ ומספר צעדי הסימולציה ו $|x|$. (שמירה ועדכון קונפיגרוציה נוכחית.

  כאן מס' צעדי הסימולציה , היא $\le l \Leftarrow$ פולינומי בקלט שאורכו $|\langle x\rangle|+l+|\langle M\rangle|$

2. <u>$BH\in NPH$</u>

תהי $L\in NP$ נגדיר רדקוציה פול'י מ L ל BH :

$f(x)$ פולטת: $\left(\langle M_L\rangle,1^{P_L(|x|)},\langle x\rangle\right)$, כאשר M_L מ"ט א"ד פולינומית (קיימת כי $L\in NP$), ו $P_L\left(|x|\right)$ חסם המובטח עבור M_L

נראה ש f אכן פונק' רידקוציה פלונומית תקפה מ L ל BH :

<u>2. תקפות</u> ־ ישירות מהגדרת NP וחסם זמן ריצה עבור מכונה א"ד פולינומית:

$x \in L \iff$ קיים ב M_L מסלול המקבל את x $\overset{1}{\iff} \left(\langle M_L\rangle, 1^{P_L(|x|)}, \langle x\rangle\right)$

1. כיון ש P_L חסם על אורך כל מסלול, מסלול מקבל, אם , קיים אורכו $\leq P_L(x)$

1. <u>$\delta \in POLY$:</u> נבנה מ"ט פול' עבור M_f, f
$M_f(x)$:

1. כתיבת $\langle M_L\rangle$ - זמן קבוע $O(1)$. לא תלוי ב $|x|$.
2. חישוב זמן כתיבת x , לנארי ב $|x|$ (מקודדים אות־אות)
3. חישוב $P_L(|x|)$:
   (א) נחשב את x בבינארי, בעצם נמיר את x מאונארי לבינארי , דורש בערך $O(n \log n)$ זמן
   (ב) נחשב את הפולינום $P_L(b)$ - דורש מס' קבוע של כפלים על מספרים בגודל $O(\log|x|)$.
   - לדוגמה אם $p = 17 \cdot n^4$: נבצע שלושה כפלים כדי לחשב n^4 נצבור בכל פעם את המכפלה עד כדי n, n^2, n^3 ואז נכפול ב 17 $\Leftarrow$ זמן $poly(\log|x|)$ - נשתמש באלג' בית לכפל. תהי c התוצאה
   (ג) נמיר את c חזרה לאונארי = זמן $O(c \log c)$ שזה פולינומי ב $|x|$ בערך $P_L|x|$.

סה"כ פולנומי ב $|x|$

הערה: למה חשוב ב BH לייצג את החסם l על מס' הצעדים באונארי? למה לא בבינארי?

אם l היה נתון בבינארי אז M_{BH} שבינינו לא היתה בהכרח יעילה. למה?

M_{BH} מבצעת לפחות l צעדים באחד המסלולים (שבו מנחשים y באורך l) , בייצוג בינארי הקלט היה נראה כך $\langle\langle M\rangle, l, \langle x\rangle\rangle$ כאשר אם l בבינארי , הוא דורש $O(\log l)$ ביטים, זמן הריצה שלנו היה במקרה הגרוע אקספו' בקלט (לא מובטח ש $\langle M\rangle, \langle x\rangle$ מספיק ארוכים יחסית ל l)

בשיעורים הבאים נגדיר כמה שפות מתחומים שונים שהן יותר "טבעיות" מ BH נוראה שכל אחת היא NPC ע"י שרשרת הרדוקציות הבאה:

$$BH \to SAT \to 3SAT \to VC \to \begin{matrix} HS \\ SC \\ O1P \\ Subset\ sum \end{matrix}$$

SAT :

תזכורת של אלגברה בוליאנית

פסוק לוגי : $p(x_1, ..., x_n)$ מוגדר מעל משתנים בוליאנים $X = (x_1, ..., x_n)$ השמה למשתנים $(\phi(X))$ היא פונק' שממפה x_i לערך T או F . פסוקים רקורסיבית:

<u>בסיס:</u>

$\varphi(X) = x_i$ או $\varphi(X) = \overline{x_i}$ נקרא ליטרל, לדוגמה: $\varphi(x_1, x_2, x_3) = \overline{x_3}$

<u>"הרכבה" :</u>

- בהנתן $\varphi_1(x)$, $\varphi_2(x)$ ניתן להגדיר $\varphi(x) = \varphi_1(x) \wedge \varphi_2(x)$
- בהנתן $\varphi_1(x), \varphi_2(x)$ ניתן להגדיר $\varphi(x) = \varphi_1(x) \vee \varphi_2(x)$

הצבה של השמה לפסוק $\varphi(x)$:

- אם $\varphi(x) = l_i$ (ליטרל)
  - אם $l_i = x_i$ אז $\varphi(x) = T$ אסם $\phi(x_i) = T$
  - אם $l_i = \overline{x_i}$ אז $\varphi(x) = T$ אסם $\phi(x_i) = F$
- אם $\varphi(x) = \varphi_1(x) \wedge \varphi_2(x)$
  - אז $\varphi(\phi) = T$ אסם $\varphi_1(\phi) = T$ וגם $\varphi_2(\phi) = T$
- אם $\varphi(x) = \varphi_1(x) \vee \varphi_2(x)$
  - אז $\varphi(\phi) = T$ אסם $\varphi_1(\phi) = T$ או $\varphi_2(\phi) = T$

0.11.6 הגדרה 11.6 : פסוק לוגי φ הוא בצורת CNF

הגדרה 11.6 : נאמר שפסוק לוגי φ הוא בצורת CNF אם: $\varphi(x) = \bigwedge_{i=1}^{m} C_i$, ו C_i היא פסוקית CNF כלומר מהצורה $C_i = \bigvee_{j=1}^{h_i} l_i$

לדוגמה $P(x_1, ..., x_5) = (\vee..\vee) \wedge (\vee..\vee) \wedge (\vee..\vee)$

השפה SAT :

$$SAT = \{ \varphi(x) | \varphi(\phi) = T \text{ כך ש } , \phi \text{ הוא פסוק } CNF \text{ שקימת לו השמה מספקת כלומר } \varphi \}$$

לדוגמה: עבור הפסוק מהדוגמה, ההשמה:

$$\phi(x_3) = F \qquad \phi(x_4) = T \qquad \phi(x_2) = T \qquad \phi(x_1) = T \qquad \phi(x_5) = T$$

היא השמה מספקת לכן $\varphi(x)$ הנ"ל בSAT

נגדיר $K - cnf$ כפסוק CNF שבכל פסוקית יש בדיוק k ליטרלים :

$$3SAT = \{\varphi(x) | \text{ הוא פסוק } 3 - cnf \text{ ספיק } \varphi(x)\}$$

0.12 הרצאה 12 - 16/01/20

תוכנית עבודה (חלקית) :

$$BH \to SAT \to 3SAT \to VC \to \begin{matrix} HS \\ SC \\ O1P \\ Subset\ sum \end{matrix}$$

נוכיח שהשפה הבאות הן NPC

תזכורת: משפט: בהנחה ש $P \neq NP$, כל שפה $L \in NPC$ אינה בP (מדד קושי מסוים)

נתחיל מהגדרת השפות

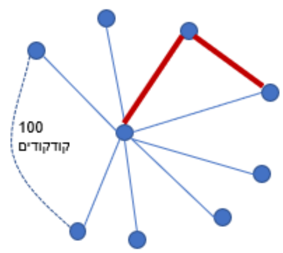
0.12.1 הגדרה 12.1 השפה $Vertex\ cover$

$$VC = \{(\langle G \rangle, k)\ |\ U \text{ לא מכוון וקיימת } G=(V,x)\}$$

$VC = \{(\langle G\rangle ,k)\ |$ U לפחות אחד הקצוות שייך ל $e\in E$, כך שלכל קשת k בגודל $U\subseteq V$ לא מכוון וקיימת $G=(V,x)\}$

$\langle G\rangle$ הוא קידוד של גרף למשל כמטריצת שכנויות

- לדוגמה הגרף המלא על n קודקודים פורש כיסוי בצמתים בגודל $n-1$ $n-2$ לא יספיקו , כי הקשת $e=\{v,u\}$ כך ש v,u לא בכיסוי לא מכוסה
- בגרף כוכב הבא (תוספת של קשת), עם 100 קודקודים - מספיקות 2 קשתות



0.12.2 הגדרה 12.2 השפה $Hitting\ set\ \ (HS)$

$HS=\{N,K,C_1,...,C_t|$ $\forall i\ C_i\cap U\neq\phi$ כך ש k בגודל $U\subseteq[n]$ של U וקיית תת $[n]$ מוכלת ב C_i כל $\}$

0.12.3 הגדרה 12.3 השפה $Set\ cover$

$SC=\{N,K,C_1,...,C_t|$ $\bigcup_{i\in I} C_i=[n]$ כך ש k בגודל $I\subseteq[t]$ קיימת $\}$

דוגמה :

$$\begin{array}{cccc} n=10 & k=3 & c_1=\{1,5,4,3\} & \\ c_2=\{2,8\} & c_2=\{7,5,6\} & c_4=\{10\} & c_5=\{9,8\} \end{array}$$

כאן אי אפשר כי חייבים לקחת יותר מ3 קבוצות

הערה: האלג' הנאיבי ל SC בודק כל תת קב' $I\subseteq[t]$, בגודל k ומקבל אם מצא SC יש $\binom{t}{k}$ קב' כאלה לבדוק לדוגמה עבור $t=n^3$ $k=\frac{n}{2}$ זה לא פולינומי בקלט
כאמור נראה ש $SC, HS\in NPC$ ע"י רידוקציה מ VC

0.12.4 משפט 12.4 $VC\in NPC$

<u>הוכחה</u>

1. 1. $VC\in NP$ מ"ט א"ד פולינומית עבור VC - $M_{VC}(\langle G\rangle ,k)$ תנחש קב' $U\subseteq V$ בגודל k , ותבדוק שהיא מהווה VC (כל הקשתות מכוסות על ידה)

   לא קשה לראות שהמכונה אכן פולינומית (בפרט $|U|$ שמנחשים היא באורך $k\log n$ ביטים , לדוגמה והבדיקה גם ניתנת לבצוע בזמן פולינומי בגרף)

2. 2. $VC\in NPH$ - אחר ההפסקה

0.12.5 **משפט** 12.5 $HS \in NPC$

<u>הוכחה:</u>

1. $HS \in NP$: מ"ט א"ד פולי' מתאימה תחשב U ותבדוק שאכן $\forall i\ U \cup C_i \neq \phi$
2. $HS \in NPH$: נראה ע"י רדוקציה מ VC ממשפט 12.4 , נקבל ש $HS \in NPH$ (כי $VC \in NPH$)

רעיון הרדוקציה:

כאן , נשים לב ש VC הוא מקרה פרטי של HS : קיים מיפוי מאוד פשוט מקבל עבור VC לקלט עבור HS :

$f(\langle G\rangle, x)$: מזהה בין $V = \{v_1, ..., v_n\}$ ל $[n]$ (למשל נזהה בין v_i למספר i)

$k = k'$

נזהה בין E לבין הקב' $C_1, ..., C_t$ כל $e_i = \{v_{i1}, v_{i2}\}$ תמופה ל $C_i = \{i1, i2\}$

נפלוט :

$$\left(n, k, C_1, ..., C_{t=|E|}\right) \text{ שקבלנו}$$

נכונות הרידוקציה f :

<u>f פולינומית:</u> עוברים על הגרף, סופרים קודקודים ומעתיקים את מס' הקודקודים שיצא , את k ,ו את הקשתות (תוך כדי מיפוי למספרים התאימים) - זה פולינומי (ב $|\langle G\rangle, k|$)

<u>תקפות f:</u>

הרעיון כאן הוא (תמיד) להראות כיצד עד עבור x , מתרגם לעד עבור $f(x)$ ולהיפך (נרחיב עוד מעט)

כיון $\Leftarrow$:

יהיה $(\langle G\rangle, k) \in VC \Leftarrow$ קיימת $\{V_{i1}, ..., v_{ik}\}$ המהווה VC עבור $G \Leftarrow$ קיימת $\{i1, ..., ik\}$ המכסה את כל הקב' C_i ,שכן אם U הוא VC אז מתקיים בעצם $\forall e \in E\ e \cap U \neq \phi$, ולכן לכל C_i שהתקבל מe_i כלשהו $C_i \cap U' = \phi \Leftarrow f(\langle G\rangle, k) \in HS$

כיוון$\Rightarrow$:

בשונה מהחלק הראשון בקורס לא נוכיח: שאם $x \notin L_1$ אז $f(x) \notin L_2$, אלא נוכיח טענה שקולה שלפיה אם $f(x) \in L_2$ אז $x \in L_1$

במקרה שלנו:

נניח ש $f(\langle G\rangle, k) \in HS \Leftarrow$ קיימת קב' $U \in [n]$, $U = \{i_1, i_2, ..., i_k\}$ כך ש $\forall i\ U' \cap C_i \neq \phi \Leftarrow$ (בניה בפרט $k = k'$) קיים VC $U = \{v_{i1}, ..., v_{ik}\}$ עבור $G \Leftarrow (\langle G\rangle, k) \in VC$ כנדרש.

הערה: הכיון ההפוך $HS \leq_P VC$ יותר מסובך, כי VC הוא מקרה פרטי, רידוקציה אפשרית:

$$HS \leq_P SAT_P \leq_p 3SAT \leq_p VC$$

0.12.6 **משפט** 12.6 $SC \in NPC$

<u>הוכחה:</u>

1. $SC \in NP$. ננחש $i \subseteq [t]$ בודק k , ובודקים האם $\bigcup_{i \in I} C_i = [n]$
2. $SC \in NPH$: נראה $VC \leq_p SC$, ונסיק $SC \in NPH$ על סמך משפט 12.4

<u>רעין הבניה</u>

העבודה שמחפשים k קב' רומזת שנרצה $k = k'$ ונרצה אישהו לזהות בין הקודקודים לבין הקבוצות $C_1, ...C_t$ ובין E ל $[n]$

כיצד נזהה בין הצמתים לקבוצות? לכל קודקוד v נתאים לו את קב' הקשתות $E_v = \{e | v \in e\}$, כלומר הקשתות היוצאות מv . נסכם:

$f\left(\langle G\rangle ,k'\right)=\left(n=\left|E\right|,k=k',C_1,...,C_{t=\left|V\right|}\right)$ כאשר $C_i=E_{v_i}$

נכונות הרדוקציה f:

<u>פולינומית</u> : סופרים קשתות, ואז עוברים על הצמתים ולכל צומת מחשבים את E_v

<u>תקפות:</u>

$\Leftarrow$

נניח $\left(\langle G\rangle ,k'\right)\in VC$ ⇐ קיים VC $U=\{V_{i1},...,V_{ik'}\}$ ⇐קיים SC $I=\{i1,...,ik\}$ כאשר $k=k'$ (מהבניה) כך ש

$\bigcup\limits_{i\in I}C_i\overset{*}{=}\bigcup\limits_{i\in I}E_{v_i}\overset{vc}{=}\!\!\rightarrow[|E|]$ מתרגם חזרה ל

$*$ עד כדי מספור שמות

$\Rightarrow$

נניח $f\left(\langle G\rangle ,k'\right)\in SC$ ⇐ קיימת קב' $I\subseteq[n]$ $=\{i_1,...,i_k\}$ כך ש $\bigcup\limits_{i\in I}C_i=[n]$ ⇐ (מהבניה) קיימת קב' קדקודים $U=\{v_{i1},....,v_{ik'=k}\}$ המהווה VC עבור E (כי מבחירת I , היא מכסה את כל הקשתות , כי כל C_{i_j} היא למעשה $E_{v_{i_j}}$ -קב' הקשתות היצאות מ V_{i_j} מש"ל

0.12.7 משפט 12.7 $3SAT\in NPC$

<u>הוכחה:</u>

1. $3SAT\in NP$, ננחש השמה ϕ ונבדוק שהיא מספקת ונקבל את $\varphi\left(x\right)$ אם"ם ϕ אכן מספקת
2. נראה על ידי $SAT\leq_p 3SAT$ - נראה בשבוע הבא, נקבל ש $3SAT\in NPH$

הערה: זה שימושי לפשט את SAT ל $3SAT$ (כשפה NPC) כדי להקל על רדוקציות לפשות חדשות בגלל המבנה היותר פשוט של $3SAT$

<u>רעיון הרידוקציה:</u>

עבור קלט $\varphi\left(x_1,..,x_n\right)$ לרדוקציה $\varphi\left(x\right)=\bigwedge\limits_{i=1}^{m}C_i$ נפלוט $\varphi'\left(x,y\right)$ מהצורה $\varphi\left(x,y\right)=\bigwedge\limits_{i=1}^{m}C_j'$ כאשר C_i' יהיה פסוק $3CNF$.

שימו לב שלמרות ש C_i' הוא לא בהכרח פסוקית בודדת ל $\bigwedge\limits_{i=1}^{m}C_i'$ יהיה מבנה של $3CNF$

לדוגמה, נגדיר ש:

$$\begin{gathered}C_1'=\left(x_1\vee\overline{x_3}\vee y_1\right)\wedge\left(x_{17}\vee\overline{y_1}\vee\overline{x_3}\right)\\ C_2'=\left(x_2\vee x_3\vee x_7\right)\wedge\left(x_1\vee x_1\vee\overline{x_8}\right)\\ than:\\ C_{1'}'\wedge C_2'=\left(x_1\vee\overline{x_3}\vee y_1\right)\wedge...\wedge\left(x_1\vee\overline{x_3}\vee y_1\right)\end{gathered}$$

כיצד נבצע את ההתאמה בין C_i ל C_i' נחלק למקרים:

1. $C_i=l_1\vee l_2\vee l_3$ ⇐ 3 ליטרלים ניקח $C_1'=C_i$
2. $C_i=l_1\vee l_2$ ⇐ 2 ליטרלים , ניקח $C_i'=\left(l_1\vee l_2\vee l_2\right)$
3. $C_i=l_i$ ⇐ 1 ליטרלים , נכפיל כמו ב2
4. $C_i=\bigvee\limits_{j=1}^{t}l_{i,j}$ כאשר $t\geq u$. נתאים

$$C_i'=\left(l_{i_1}\vee l_{i_2}\vee y_{i_1}\right)\wedge\left(\overline{y_{i_1}}\vee l_{i_2}\vee y_{i_2}\right)\wedge\left(\overline{y_{i_2}}\vee l_{i_4}\vee y_{i_3}\right)\wedge....\wedge\wedge\left(\overline{y_{i_{t-3}}}\vee l_{i_{t-1}}\vee y_{i_t}\right)$$

בראשון והאחרון יש שתי משתנים מקוריים ומשתנה 1 חדש

באמצעים יש את המשתנה שהוספנו בשלילה , ומשתנה חדש נוסף

לדוגמה:

$$C_{14}=x_1\vee\overline{x_{14}}\vee x_{12}\vee\overline{x_3}\vee x_2$$

הופך ל:

$$C'_{14}=(x_1\vee\overline{x_{14}}\vee y_{14,1})\wedge(\overline{y_{14,1}}\vee x_{12}\vee y_{14,2})\wedge(\overline{y_{14,2}}\vee\overline{x_3}\vee x_2)$$

נשים לב ש C'_i לא שקול לוגית ל C_i (אפילו סט המשתנים עליו הם מוגדרים שונה - אם נסתכל על C_i כמוגדר מעל (x,y) עדיין לא תהיה שקילות לוגית)

נתוח $f(\varphi(X))$ **:**

<u>פולינומית</u>

כל פסוקית הופכת לפסוקית שאורכו פי 3 מהפסוקית המקורית (מבחינת מס' ליטרלים) וגם קל לחשב את C'_i . בערך לינארי ב$|\varphi(x))|$על מכונת RAM (כולל הקצאת ה $y_{i,j}$ ים)

<u>תקפות</u>

$\Leftarrow$

נניח ש $\varphi(x_1,..,x_n)\in SAT$ ותהי $\phi(x)$ השמה מספקת עבור φ , נבנה השמה $\phi'(x,y)$ המספקת את φ'

- נגדיר $\phi'(x)=\phi(x)$ ונקוה שנצליח להשלים השמה ל yים שסה"כ ϕ' תספק את φ' (אכן נצליח)
- לכל C'_i , נשלים את ההשמה לϕ' ל $y_{i,j}$ כך ש C'_i תסתפק. אם נצליח, אז גם $\varphi=\bigwedge C'_i$ יסתפק ע"י ϕ' . נחלק למקרים:

1. C'_i לא מכיל $y_{i'j}$ים במקרה זה היא שווה ל C_i . כיון ש $\phi'(x)=\phi(x)$, ו ϕ מספקת את φ' אז ϕ מספקת גם את C_i כי $\varphi=\bigwedge C_i$) ולכן גם את C'_i
2. אם $C_{i'}$ מכיל $y_{i,j}$ ים : נשים לב שקיים ליטרל $l_{i,j}=x_{r_i}$ המסתפק ע"י ϕ בגלל השמה של ל x , כי ϕ מספקת כל פסוקית C_i , וכדי שC_i תסתפק לפחות ליטרל אחד שלה צריך להסתפק. נחלק למקרים:

(א) $l_{i,j}$ אינו שייך לפסוקיות ה"קצה" ב C'_i כלומר

$$C'_i=...\left(\overline{y_{i,j-3}}\vee l_{ij-1}\vee\underbrace{y_{i,j-2}}_{T}\right)\wedge\left(\underbrace{\overline{y_{i,j-2}}}_{F}\vee\underbrace{l_{ij}}_{T}\vee\underbrace{y_{i,j-1}}_{F}\right)\wedge\left(\underbrace{\overline{y_{i,j-1}}}_{T}\vee l_{ij+1}\vee y_{i,j}\right)$$

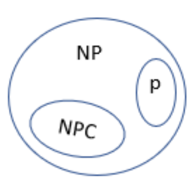
$l_{i,j}$ הוא T (כי ההשמה מספקת) נקבע $y_{i,j-1}=\overline{y_{i,j-2}}=F$, ואז ה"גל" יתפשט

כלומר ה"גל של Tים יתפשט שמאלה וימינה ויאפשר לקבוע את $y_{i,j}$ ים כך שכל הפסוקיות ב C'_i יסתפקו.

$\Rightarrow$ נניח ש $\varphi'(x,y)\in SAT$ ונוכיח ש $\varphi(x)\in 3-SAT$ -

0.13 שיעור 13 - 23/01/20

תמונת עולם בהנחה ש $P\neq NP$



$$L \notin NP \backslash NPC \cup P$$

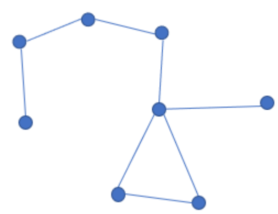
כלומר שפות שאינן הכי קשות במובן של NPC אבל גם אינו ב P (הוכחה בלכסון, תראו בסבוכיות)

<u>התמודדות עם NPC</u>

אמרנו (הוכחנו) שבהנחה ש $P \neq NP$, שזו הנחה סבירה , כל שפה $L \in NPC$ אינה ב P אבל , לפעמים נרצה לפתור בעית שהן NPC . בד"כ רוצים לא רק להגיד אם $x \in L$ (כלומר , קיים y עבורו $R(x,y)$ (יחס NP) , אלא רוצים למצוא אותו

לדוגמה , בהנתן G שרוצים להבין אם הוא ב HC בעצם רוצים למצוא HC אם קיים (למשל כדי לתכנון מסלול של שליח ב $Amazon$)

דוגמה נוספת: בהנתן רשת כבשיים , נרצה להציב בחלק מהצמתים תחנות דלק, ככה שבכל כביש ישנה תחתת דלק לפחות בקצה אחד.



נרצה להציב שכמה שפחות תחנות דלק. זו ממש הבעיה של VC ,אבל בגרסת החיפוש (בהמשך ההרצאה נראה שהיא שקולה פול' לבעיית ההכרעה)

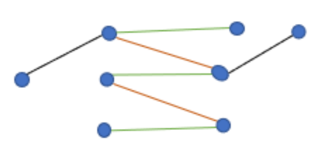
ידוע ש $VC \in NPC$ ולכן אין לבעיה אלג' פולינמי מה נעשה?

1. נוותר (פחות טוב)
2. אלג' הקירוב: ננסה למצוא אלג' יעיל שמוצא מקיום של תחנות דלק שמספרן לא בהכרח המינמלי האפשרי, אבל מובטח שהוא לא הרבה יותר גדול מהמינימלי האפשרי (נוכיח זאת)זה סביר כי נשלם קצת יותר משאבים אבל נמצא פתרון ולא נוותר

נראה אלג' קירוב יעיל אם פקטור 2 ל VC כלומר אלג' שבהנתן גרף G , מוצא $U \subseteq VC$ ב G שגדולו לכל היותר פי 2 מהמינימלי האפשרי

<u>תזכורת שידוף בגרף $(matching)$</u>

שידוד בגרף : קב' קשתות זרות בצמתים ב E .



הירוק הוא בגודל 3 קשתות. האדום הוא שידוך בגודל 2 קשתות

שידוך מקסימלי M בG הוא שדוך ב G , שלא נתן להוסיף לו קשתות.

- לדוגמה, השידוכים הירוק והאדום הם מקסימלים בגרף.

  הקשתות השחורות מהוות שדוך לא מקסימלי בגרף

- שימו לב שידוכים מקסימליים הם לא בהכרח באותו גודל. לדוגמה האדום בגדול 3 והירוק בגודל 2

עבור אלג' הקירוב שלנו , נצטרך , אלג' למציאת שדוך מקסימלי בגרף. מסתבר שקיים אלג' פולינומי לכך:

$Find\_max\_macth(G)$:

- מאתחל $M \leftarrow \phi$
- מסדר את E בסדר כלשהו $E = e_1, e_2, ..., e_m$ עובר על E לפי סדר זה , עבור קשת l_i , בודק אם נתן להוסיף אותה ל M כך ש M ישאר שידוך. אם כן מוסיף (אחרת לא מוסיף)
- פולטים את M

נתוח : קל לראות שהאלג' פולינומי

<u>נכונות:</u>

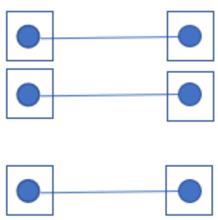
נשים לב ש M הוא תמיד שידוך (בכל שלב באלג' - אינוריאנטה) , נניח בשלילה ש M שהאלג' החזיר לא מקסימלי. אזי קיימת $e_i \in E$ שנתן להוסיף ל M כך ש M ישאר שידוך. איך זה קרה?

כאשר האלג' הסתכל על l_i , הוא החליט שלא נתן להוסיף אותו, אבל זאת סתירה (כי נתן להוסיף אותן אפלו לM , לא רק ל M' שהיא ה M בשלב i : $M' \subseteq M$) מש"ל.

אלג' קירוב ל VC :

$M_{VC}(G)$:

1. מוצא שידוך מקיסמלי ב G , M (פולינומי - הראינו אלג')
2. נקח לכיסוי את כל הקודקודים ב M : U_M



<u>נתוח:</u>

- U_M אכן VC , כי M מקסימלי ב G (תשלימו לבד את הפרטים)
- הVC הכי קטן בגרף הוא בגודל לפחות $|M|$ מס' קשתות כי חייבים לבחור קודקוד מכל קשת ב M כדי לכסות אותה (אחד לפחות)

$\Leftarrow$ $|U_M| = 2|M|$ הוא <u>לכל היותר</u> פי שניים בגודל מהכיסוי הכי קטן.

<u>הערה:</u>

ישנן בעיות ב NPC עם אלג' קירוב $1+\varepsilon$ עבור $\varepsilon > 0$ הקבוע קטן כרצוננו לדוגמה $Knapsack$ (לא בחומר)

זיהוי שקול לחיפוש

<u>זיהוי גורר חיפוש</u>

ניתן להראות שלכל שפה ב NP , אם קיים אלג' פולינומי לשפה אז קיים אלג' פולינומי המוצא עד y או אומר שאין (אם"ם $x \in L$) עבור $R_L(x,y)$ (יחס NP כלשהו עבור השפה

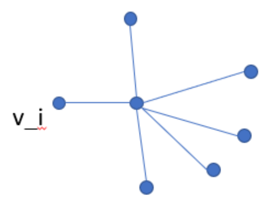
נדגים זאת על VC :

נראה שבהנתן אוב לשפה VC . נתן לבנות אלג' פולינומי שמוצא VC מתאים/מחליט שאין.

$Find^{vc}(G,k)$:

1. שואל את האוב האם $VC(G,K)=1$. אם לא נעצור ונחזיר "לא קיים"
2. נעבור רקרוסיבית:

- נאתחל $U \leftarrow \phi$ (משתנה גלובלי)
- נעבור על כל הקודקודים ב G לפי סדר מסויים. לכל קודקוד v_i ננסה לבדוק אם קיים VC בגודל k המכיל אותו. איך?



- נוריד את v_i והקשתות היוצאות מ G נקבל G' (v_i כבר מטפל בהן)
- נשאל את האוב לגבי $(G', k'=k-1)$
- ברגע שנמצא v_i כזה , נעדכן $G \leftarrow G'$, $k \leftarrow k-1$, נוסיף את v_i ל U ונחזיר ל 2 .

הערה:

<u>חיפוש גורר זיהוי</u>

הכוון השני (חיפוש גורר זיהוי) בשקילות הוא קל. אם קל למצוא VC בגודל K ב G (א להגיד שאין) , אז קל לבדוק קיום:

- נפעיל את האלג' לחיפוש ונענה לא אם אמר "אין" ו"כן" אם החזיר y .

מסקנה: מספיק להסתכל על NP במושגי שפות כמו שעשינו, כי זה שקול פולינומית לבעיות החפוש שבאמת מעניינות.

0.13.1 משפט 13.1 $cook-levin$

$SAT \in NPC$

הוכחה

$SAT \in NP$: $M_{SAT}(\varphi)$: תנחש השמה ϕ ותציב בφ . אם קיימם $\varphi(\phi)=T$ תקבל, אחרת תדחה.

$SAT \in NPH$: נתחיל מחימום.

נראה ש $SAT \leq_p VC$ כך שיתקיים :

$$\text{כך שיתקיים } \varphi(y) \text{ ספיק} \iff \text{קיימת קבוצה } U \text{ ב } VC \text{ עבור } (G,K)$$

קודם כל צריך להחליט מה המשתנים בפסוק φ רעיון פשוט (לא יעבוד עד הסוף) :

- נמדל את ה U שמחפשים (את העד שלנו) בתור סדרת משתנים: $Y=y_1,y_2,....,y_n$
- המשמעות של $y_i=T$ תהיה : הקודקוד v_i נבחר ל U.
- היינו רוצים ש φ יבטא את הטענה: " Y מתאים ל VC בגודל $\geq k$ ב G (כמסה על קשת)

- ממע' ספרתיות אחננו יודעים שכל פונקציה $f\left(y_1,...,y_n\right)$ נתן לבטא כפסוק CNF נתאים פסוקית לכל F בטבלת האמת (מפת קרנו). לכן באופן כללי ייתכנו 2^n פסוקיות וזה אקספוננציאלי ב n . אצלנו n הוא $|V|$ (בערך גודל הקלט) ולכן רידוקציה כזו לא תרוץ בזמן פולינומי.

- ננסה לבנות φ בצורה חכמה בהתשחב בבעי המסוימת של VC , נדרוש:
  - $\varphi\left(Y\right)=\varphi_{cover}\left(Y\right)\wedge\varphi_k\left(Y\right)$, $cover$ - כיסוי של כל קשת , k - Y לא מכיליל יותר מk Tים
  - כיצד נממש?
    - $\varphi_{cover}\left(Y\right)=\bigwedge\limits_{e=(v_i,v_j)\in E} y_i\vee y_j$,סה"כ $2\left|E\right|$ ליטרלים בפסוק.
    - $\varphi_k\left(Y\right)=\bigwedge\limits_{\substack{I\subseteq[n]\\ |I|=k+1}} \vee_{i\in I}\overline{y_i}$ בכל תת קב' בגדול $k+1$ קיים לפחות F אחד

    זה לא פולינומי באופן כללי. לדוגמה עבור $k=\frac{n}{2}$ יהיו $\binom{n}{\frac{n}{2}}>2^{\frac{n}{3}}$ פסוקיות. לא פולינומיות

רעיון 2 (יעבוד)

"נעזור" יותר לפסוק. כלומ נתן יצוג יותר מפורט של ההשמה ע"י מידול במשתנים ואז φ יצטרך לעבוד פחות קשה, ובתקווה יצליח להיות יותר קטן:

- נחשב על U כסדרה של קודקודים $U=u_1,u_2,...,u_k$ (למעשה k מגולם בתוך המשתנים)

- נייצג את הסדרה ע"י המשתנים: $k\cdot n\{y_{11},...y_{in}....y_{k1},...y_{kn}$ משתנים

- כאשר $y_{i,j}$ יגיד מבחינתנו שמקום ה i נשים את הקודקוד v_j .

  נרצה לדרוש $\varphi\left(y\right)$ ש y מתאר סדרה U בגודל $\geq k$ שהיא VC
  - לדוגמה $U=v_2,v_{14},v_9$ יתאים להשמה שבה : $\begin{matrix} y_{1,2}=T\\ y_{2,14}=T\\ y_{3,8}=T\end{matrix}$ ו F לכל השאר (הj מספר הקודקוד , הi אינדקס לסדר הקודקודים)

  שימו לב שלא כל השמה חזרה חזרה ל U חוקי. לדוגמה , יתכן שכל ה $y_{i,j}$ ים הם F דווקא זו לא בעיה כי כך נקבל U יותר קטן
  - דוגמה 2 : יתכן ש $y_{1,5}=y_{1,2}=T$. זו "רמאות" בעייתות כי ככה עלולים לעקוף את הדרישה שיש לכל היותר k קודקודים ב U .

- נגדיר פסוק $\varphi\left(Y\right)=\varphi_{cover}\left(Y\right)\vee\varphi_{legal}\left(Y\right)$

$$\varphi_{cover}\left(Y\right)\bigwedge_{e=\{v_i,v_j\}\in E}\left(\underbrace{y_{1,i}\vee y_{2,i}\vee....\vee y_{k,i}}_{v_i\text{ choesed for U}}\vee\underbrace{y_{1,j}\vee...\vee y_{k,j}}_{v_j\text{ choosed for U}}\right)$$

$$\varphi_{legal}\left(Y\right)=\bigwedge_{i=1}^{k}\bigwedge_{\substack{I\subseteq[n]\\ |I|=2}}\left(\underbrace{\vee_{i\in I}\overline{y_{i,j}}}_{*}\right)$$

* שני ליטרלים . בכללי , לכל קב' בגדול 2 מבין $y_{i,?}$ ים לפחות אחד הוא F

מקרה כללי $L \leq_p SAT$:

תהי $L \in NP$. תהי M_L מ"ט א"ד פולינומית עבור L . יהי $t(n)$ חסם זמן ריצה פולינומי עבורה .

רעיון:

כמו עבור VC , נדאג להתאים בין "עד" לכך ש $x \in L$ להשמות מספקות, מהו העד? המסלול y .

רעיון שלא יעבוד: נייצג אותו על ידי $Y = y_1, ..., y_i, ..., y_t$ לכל i נסמן 0 או 1 δ בצעד ה i

נדרוש φ שהמסלול מהוה חישוב מקבל בM_L בריצה על x . (עבור הרדוקציה x קבוע).

רעיון 1 :

נדרוש $\varphi(Y)$ שבעצם קיימת רישא של Y שמהווה מסלול מקבל כי לא יודעים מראש מה יהיה האורך המדוייק של y). הבעיה היא שלפסוק יש מבנה קבוע ואם y היה מייצג מסלול ולא רישא של מסלול, לא היינו יודעים מה אורכו המדויק. הבעיה היא שלא ברור מראש שנוכל לבנות פסוק CNF יעיל שבודק כזה Y . בפרט הפסוק צריך לסמלץ ריצה של המכונה M .

אז רק כדי לבצע את הסימולציה נצטרך עוד משתנים:

נייצג את הריצה של M באמצעות טבלת קונפי'

	1	....	j	....
C_t				
C_i			$C_{i,j}*$	
C_0	q_x, x_1	$x_2..$	...	$\flat$ .. $\flat$

* בד"כ תוכן הסרט בסוף צעד i , בתא j

ושורה בטבלה תהיה תוכן הסרט

בשביל להמשיך לסמלץ נרצה את מקום הראש ומצב הנוכחי. נייצג זאת ע"י הוספת q לתא עליו מצביע הראש.

כלומר אם בצעד i הראש במקום i והמצב הוא q בתא $C_{i,j}$ יהיה (q,a) - כאשר a תו נוכחי שנמצא מקום j

נמדל את הטבלה באמצעות הכנסת משתנים עבור C_0 , נחשב את הקנופ' הבאות , נובדוק שהקונ' הראשונה היא אכן C_0 המתאימה ל x והקונפ' האחרונה היא קונפ' מקבלת

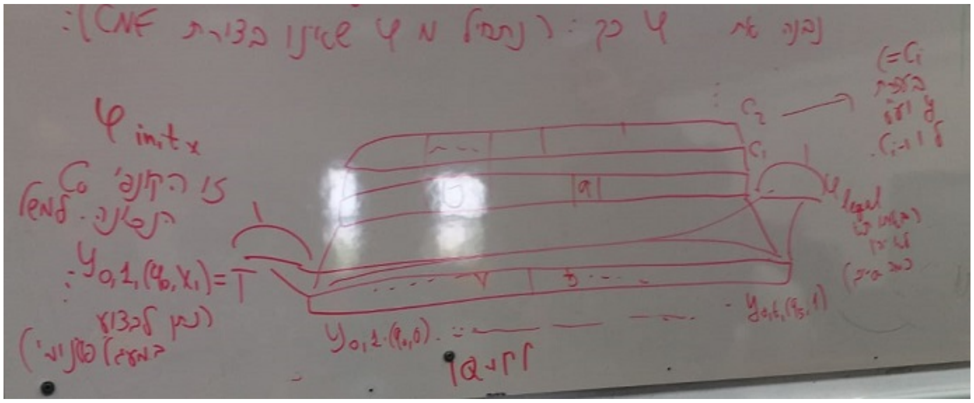
מכאן ענת עברה להסביר בציורים

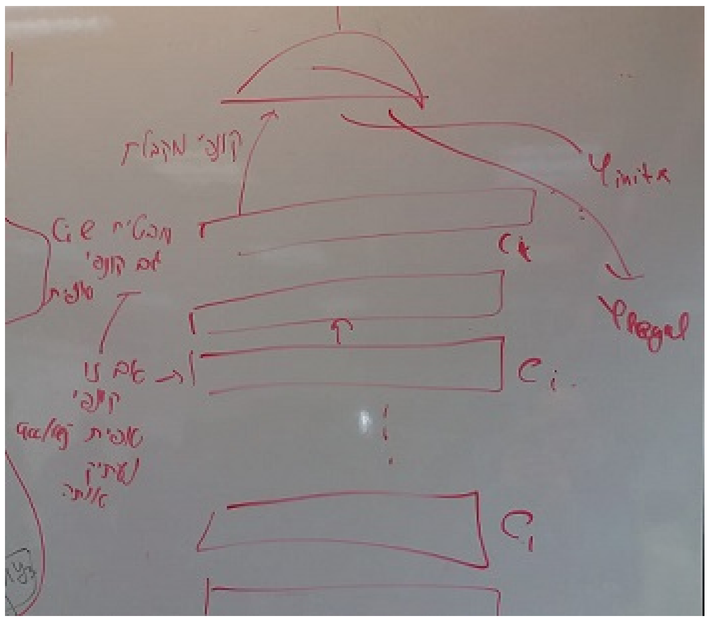
והמבין יבין...

נכניס משתנים:

$$\left\{ \underbrace{y_{0,j}, a}_{\text{conf' 0}} \right\} \begin{matrix} a \in \Gamma \cup Q \times \Gamma \\ i \leq j \leq t \end{matrix}$$

$y_{0,a} = T$ אם בקונפ' C_0 בתא j יש a , נבנה את φ כך : (נתחיל מ φ שאינו בצורת CNF :

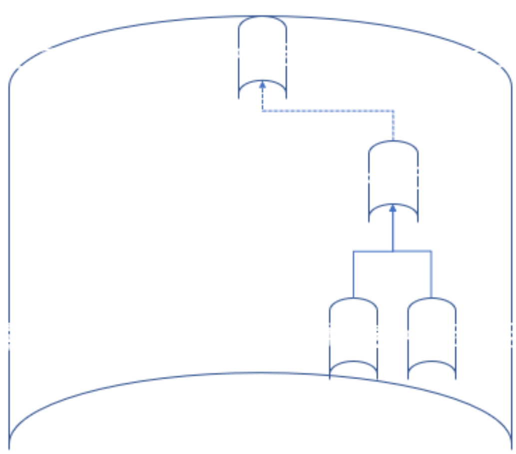




הגענו $\varphi(y,x)$ שאינו CNF אבל יעיל בקלט - תקף

צעד אחרון: נראה רדוקציה מפסוק לוגי כללי $\varphi(X)$ (Circut SAT) (ספיק) לפסוק CNF ספיק.

רעיון הרדוקציה : לעזור לפסוק CNF בבדיקה :



f תגרום ל y_iים יספקו את ביניים

$\Downarrow$

$\varphi(x,y)$ (y זה הערכי הביניים)

דוגמה: נפלוט (לכל שער y_i בפלט)$\bigwedge y_{700}$

נבצע לכל $y_3 \wedge y_4 = y_7$ שהם שלושה משתנים CNF - לכל היותר 8 פסוקיות - סופי