

סוכם על ידי אבי שוע –

shuaavi@gmail.com

<http://www.cs.huji.ac.il/~shuaavi>

אני מקווה שהסיכומים יעזרו לכם ולעוד רבים – אבל אני חייב להעיר – יתכן, ואפילו סביר, שיש בה טעויות. **אני (ואף אחד אחר) לא לוקח אחריות** לאף ציון / בזק שיגרם בגלל הסיכום.

אבי שוע

26.2.2007

<http://www.cs.huji.ac.il/~compu>

שעת קבלה – ב' 8 – 10.

orna@cs.huji.ac.il

נהלים:

- שני בחני מגן – אמריקאיים – מי שפתר את כל תרגילי הבית ולא טיפש במיוחד יצליח לענות
 - 30.4.2007 – 15% מגן.
 - 10% - מגן – עדיין לא הוגדר.
 - על התרגילים מקבלים ציון, אבל נשמר בציונים רק 'עבר' / 'נכשל'
 - צריך לעבור בכולם למעט ב2.
- ספר : אם קונים (או מורידים) ספר – אז את "Introduction to the Theory of Computation",
..Michael Sipser, PWS Publishing Company

מה לומדים בחישוביות?

עד עכשיו למדנו:

1. Intro

a. איך להגיד למחשב מה לעשות.

2. DAST

a. מחפשים מבני נתונים יעילים.

3. אלגוריתמים

a. בעיות קשות.

ועכשיו הגענו לחישוביות

נעסוק ב:

1. האם בעיה פתירה?

2. האם היא 'קלה' או 'קשה'?

3. מה המחשבים יכולים / אינם יכולים לעשות?

b. באיזה מחיר?

הקורס יחולק ל3:

1. מודלי חישוב (אוטומטים)

2. חישוביות – מה המחשב יכול לעשות / אינו יכול לעשות

3. סיבוכיות – באיזה מחיר.

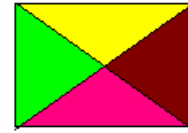
סיבוכיות

נתון גרף מכוון $G = \langle V, E \rangle$. יהו $s, t \in V$.

1. מסלול אוילר – האם יש מסלול מ s ל t העובר על כל קשת בדיוק פעם אחת?
 - a. אוילר הוכיח כי יש רק 2 קודקודים עם דרגה אי זוגית, וכל השאר בדרגה זוגית. לכן הבעיה ניתנת לפתרון בזמן לינארי.
 2. מסלול המילטוני: מסלול מ s ל t העובר בכל קודקוד בדיוק פעם אחת?
 - a. צריך לבדוק את כל האפשרויות.
- נלמד סיווג של בעיות למחלקות סיבוכיות –
- כדי שנדע לא לשאוף ליותר מדי.
 - קירובים / אלגוריתמים רנדומליים.
 - לנצל קושי (הצפנה)

חישוביות

נתון אריחים ($\equiv$ בלטות), $t_1, \dots, t_n$. כל בלטה מחולקת ל4 צבעים, כך:



אנו רוצים לדעת האם לכל n ניתן לרצף ריבוע $n \times n$ כך שאריחים שכנים יסכימו על הצבע (צבעים זהים). מותר להשתמש בכל אריח כמה פעמים שרוצים, אבל אסור לסובב.

אין אלגוריתם כזה. מכיוון שהמשפט מדבר על 'לכל n ', ולכן אי אפשר לפתור זה.

דוגמא נוספת: מחפשים תוכנית המקבלת כקלט :

1. תכנית בjava (P)

2. קלט x ל P.

ומכריעה האם היא עוצרת על x .

אי אפשר לכתוב תכנית כזו.

אנחנו נרצה לחלק את הבעיות ל'כריע' ו'לא כריע' (Decidable / Undecidable).

מספיק מבוא להפעם – נתחיל עם אוטומטים (Automata)

אוטומט הוא :

1. מודל בסיסי - "מחשב עם זכרון מצומצם"

דוגמא: יש עט. ניתן לתת לעט 6 פקודות: ON, OFF, Right, Left, Down, Up.

רוצים: מערכת שמקבלת סדרת פקודות, ומכריעה האם המילה חוקית – מייצרת Skyline, ומתחילה ב on

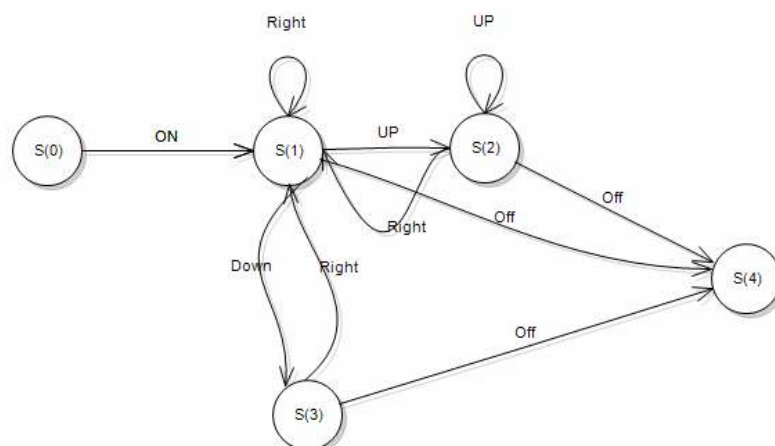
ומסתיימת ב off).

מה זה לא חוקי?

2. UD

3. DU

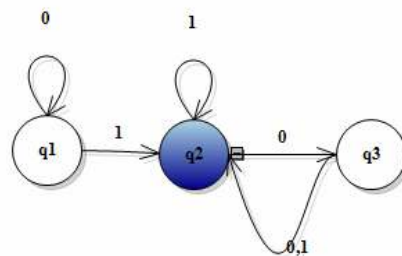
4. אסור לחזור.



הגדרות:

- אלפבית (א"ב): קבוצה סופית של אותיות, בד"כ תסומן כ $\Sigma = \{\sigma_1, \dots, \sigma_n\}$. לדוגמא: $\Sigma = \{0,1\}$, $\Sigma = \{0,1\}^4$. גם $\Sigma = \{OFF, ON, U, D, L, R\}$ הוא א"ב (מה שהשתמשנו בו קודם).
- מילה: סדרה סופית של אותיות $w = \sigma_1 \sigma_2 \dots \sigma_k$. המילה הריקה תסומן כ ϵ . נסמן $\Sigma^* = \{w : w \text{ is word over } \Sigma\}$.
- שפה $L \subseteq \Sigma^*$ - כל המילים w כך ש M מקבל את w . (מה זה M מקבל את w - ראה למטה).
- אוטומט הוא חמישייה - $M = \langle Q, \Sigma, \delta, q_0, F \rangle$.
 - Q - קבוצה סופית של מצבים.
 - Σ - א"ב
 - $\delta : Q \times \Sigma \rightarrow Q$ פונקציה פונקציית המעברים.
 - $q_0 \in Q$ מצב התחלתי.
 - $F \subseteq Q$ - מצבים סופיים. (נציירם כעיגול כפול).

לדוגמא:



כאן האוטומט הוא:

$$\Sigma = \{0,1\}, M_1 = \{\{q_1, q_2, q_3\}, \{0,1\}, \delta, q_1, \{q_2\}\}$$

δ	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_2	q_2

בהינתן מילה $w = w_1 w_2 \dots w_n$ מעל Σ אנו מגדירים ריצה של M על w כסדרת מצבים $r_0, r_1, \dots, r_n \in Q$ כך ש:

- $r_0 = q_0$ (הריצה מתחילה במצב ההתחלתי)
- לכל $0 \leq i \leq n-1$ מתקיים $r_{i+1} = \delta(r_i, w_{i+1})$. לכן ריצה של M_1 (האוטומט לעיל) על 00110 תהיה $q_1, q_1, q_2, q_2, q_2, q_3$.

נאמר שהריצה r מקבלת אם $r_n \in F$ אחרת r דוחה, והאוטומט M מקבל את w אם הריצה על w מקבלת. לדוגמא, $01 \in L(M_1)$ (האוטומט שציירנו קודם). נשים לב כי השפה של M_1 היא "יש לפחות 1 אחד, ואחרי ה1 האחרון יש מספר זוגי של אפסים".

כעת נבנה M_2 כך ש $L(M_2)$ הוא כל המילים המכילות את תת המילה 001.

בקוראנו מילה אנו יכולים להיות באחד מהמצבים הבאים:

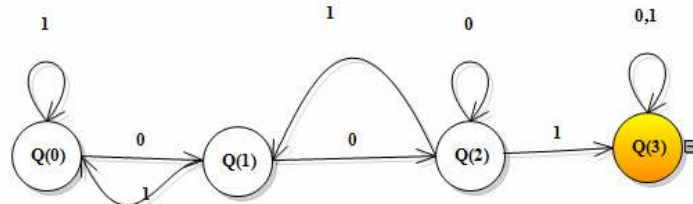
0. לפני ש001 מתחיל (בתחילת הקריאה, וכל פעם אחרי שנקרא '1' שלא בא אחרי 00).

1. קראנו 0

2. קראנו 00

3. קראנו 001

האוטומט יראה כך:



לו היינו רוצים לבנות אוטומט המקבל את כל המילים w בהם w מכילה את 0^n1 , אזי הוא יראה כהכללה של הקודם.

$$M_n = \langle \{q_0, \dots, q_{n+1}\}, \{0, 1\}, q_0, \delta, \{q_{n+1}\} \rangle$$

$$\delta(q_0, 0) = q_1$$

$$\delta(q_1, 1) = q_0$$

$$1 \leq i \leq n-1, \delta(q_i, 0) = q_{i+1}, \delta(q_i, 1) = q_0$$

$$\delta(q_n, 0) = q_n$$

$$\delta(q_n, 1) = q_{n+1}$$

$$\delta(q_{n+1}, 0) = \delta(q_{n+1}, 1) = q_{n+1}$$

כעת נבנה אוטומט לשפה $L = \{0^n1^n : \forall n \geq 0\}$. לא נצליח.

נאמר כי $L \subseteq \Sigma^*$ היא רגולרית אם קיים אוטומט סופי M כך ש $L(M) = L$ (אפשר לבנות לה אוטומט).

תכונות סגור של שפות רגולריות

$$L_1 \cup L_2 = \{w \mid w \in L_1 \vee w \in L_2\}$$

$$L_1 \cap L_2 = \{w \mid w \in L_1 \wedge w \in L_2\}$$

$$\underbrace{L_1 \cdot L_2}_{\text{concatenation}} = \{w_1 \cdot w_2 \mid w_2 \in L_2 \wedge w_1 \in L_1\} \quad \text{בהינתן } L_1, L_2 \text{ שפות מעל } \Sigma \text{ נסמן}$$

$$L^* = \{w_1 \cdot w_2 \cdot \dots \cdot w_k \mid k \geq 0, \forall i, w_i \in L\}$$

דוגמא: נניח כי $L_1 = \{1, 333\}$, $L_2 = \{22, 4444\}$, שתיהן מעל $\Sigma = \{1, 2, 3, 4\}$.

$$L_1 \cup L_2 = \{1, 333, 22, 4444\}$$

$$L_1 \cap L_2 = \emptyset$$

$$L_1 \cdot L_2 = \{122, 14444, 33322, 3334444\}$$

$$L_1^* = \{\varepsilon, 1^n, (333)^n, \dots\}$$

12.3.2007

פינת העברית : להפוך אוטומט לדטרמיניסטי – חירצון.

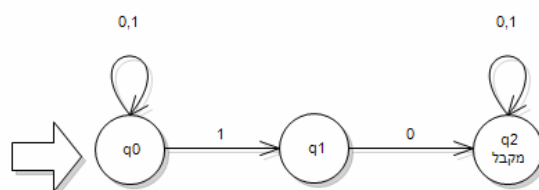
ראינו בשיעור שעבר כי ניתן להמיר אוטומט NFA ל DFA (בתהליך שנקרא Subset Construction,

$$A' = \langle 2^Q, \Sigma, \delta', Q_0, F' \rangle$$

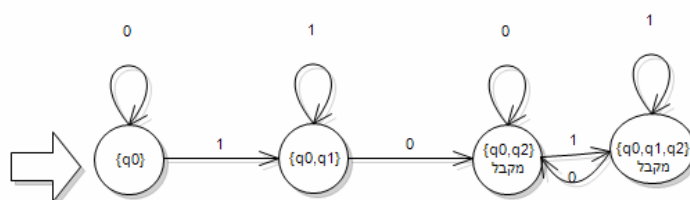
$$\delta'(S, c) = \bigcup_{s \in S} \delta(s, c) \quad \text{ע"י הגדרת}$$

$$F' = \{s \mid s \cap F \neq \emptyset\}$$

לדוגמא (אוטומט שמכיל את כל המילים שמכילות 10).



יהפך ל:



על ידי שימוש באלגוריתם הנ"ל. (נשים לב ששני המצבים הימניים הם בור מקבל, אבל די חורני – אפשר היה לצמצם אותו למצב יחיד).

נשים לב שהקפצנו את מספר המצבים באופן אקספוננציאלי. נוכיח כי אין ברירה.

משפט: יש משפחה של שפות $L_1, \dots, L_n$, כך שלכל $n \geq 1$:

1. L_n ניתנת לזיהוי ע"י NFA עם $O(n)$ מצבים.

2. ה DFA הקטן ביותר עבור L_n צריך לפחות 2^n מצבים.

נשים לב:

1. המשפט אכן גורר את החסם התחתון.

2. אם יש בניה פולימוניאלית, נפעיל אותה על המשפחה $L_1, \dots, L_n$ ומובטח שקיים $n \geq 0$ שעבורו נקבל סתירה. $\equiv$ (היה צריך משפחה – לא די בשפה אחת).

המשפחה: $\Sigma_n = \{1, 2, \dots, n, \#\}$ והשפה היא:

$$L_n = \left\{ \sigma_1 \dots \sigma_k \# \sigma_{k+1} \mid \begin{array}{l} \sigma_1 \dots \sigma_{k+1} \in \{1, \dots, n\} \\ \sigma_{k+1} \in \{\sigma_1, \dots, \sigma_k\} \end{array} \right\}$$

לדוגמא:

$$1242\#4 \in L_4$$

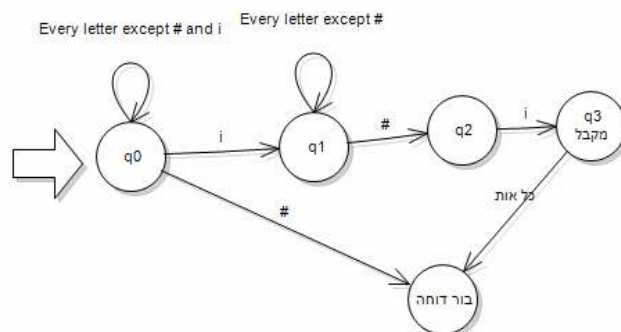
$$1222\#3 \notin L_4$$

$$13323\#1 \in L_4$$

$$1\#\#1 \notin L_4$$

ראשית, עלינו להראות NFA עם $O(n)$ מצבים עבור L_n - הוא ינחש מה תהיה האות האחרונה.

לדוגמא – מנחש ש i היא האות האחרונה –



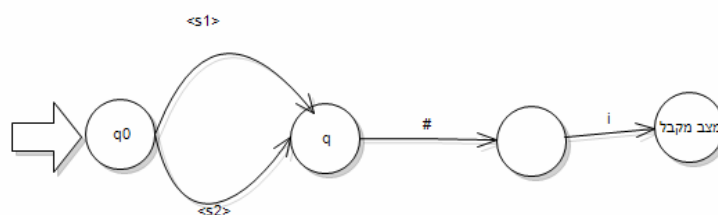
. מאחר ומספר המצבים כאן סופי,

ולא תלוי ב i , אזי אפשר ליצור אוטומט שמאחד את המקרים לכל i ב $O(n)$ מצבים.

כעת, נראה כי DFA צריך לפחות 2^n מצבים. בשלילה, יש $DFA A_n$ עם פחות מ 2^n מצבים עבור

$L_n \leftarrow$ יש שתי קבוצות $S_1 \neq S_2 \subseteq \{1, \dots, n\}$ כך ש $q = \delta(q_0, \langle s_1 \rangle) = \delta(q_0, \langle s_2 \rangle)$ (כי יש

2^n קבוצות אפשריות). המקיימות



תבי $i \in s_1 - s_2$. נתבונן ב $\delta(\delta(q, \#), i)$ - (לאן הולך A_n בקוראו $\#i$ מהמצב q).

אם $q \in F$, נקבל ש $\langle s_2 \rangle \#i \in L_n$, אבל $s_2 \#i \notin L_n$ (מבחירת i).

אם $q \notin F$, נקבל שהריצה היחידה $\langle s_1 \rangle \#i$ דוחה, בסתירה לכך שהמילה בשפה.

נחזור ל'נשים לב' - תהי $f: \mathbb{N} \rightarrow \mathbb{N}$ פונקציה פולינומיאלית שמתיימרת להיות חסם עליון לדטרמיניזציה. נתבונן ב $f(n_1), \dots$ כאשר n_i הוא מספר המצבים $3i + 2$ של NFA עבור L_i . קיים i כך ש $f(n) < 2^i$. סתירה.

ביטויים רגולריים

דוגמאות:

1. "1 במקום הלפני לפני אחרון": $(0+1)^*1(0+1)(0+1)$

2. מילים עם 1 אחד בדיוק: 0^*10^*

3. לפחות 1 אחד $0^*1(0+1)^*$

הגדרה פורמלית אינדוקטיבית: ב"ר מעל א"ב Σ

בסיס: $\emptyset, \varepsilon, a \in \Sigma$ הם ביטויים רגולריים.

צעד: אם r_1, r_2 הם ביטויים רגולריים הם ב"ר, כך גם $r_1^*, r_1 \cdot r_2, r_1 + r_2, r_1 \cup r_2$, כאשר

$$\Sigma = \{a, b\}$$

כל ביטוי רגולרי מגדיר שפה - $L(r) \subseteq \Sigma^*$. כאשר:

$$L(\emptyset) = \emptyset \quad L(\varepsilon) = \{\varepsilon\} \quad L(a) = \{a\}$$

$$L(r_1 + r_2) = L(r_1) \cup L(r_2)$$

$$L(r_1 \cdot r_2) = L(r_1) \cdot L(r_2)$$

$$L(r_1^*) = (L(r_1))^*$$

הערה:

• נהוג להשמיט את $\cdot$ - לדוגמא, כל המילים באורך זוגי - $(\Sigma\Sigma)^*$.

• סדר פעולות: $+, \cdot, ^*$ (מימין לשמאל).

דוגמאות נוספות:

• מילים שמסתיימות ב $0 - 0(0+1)^*$.

• L_n מהוכחת החסם התחתון - $i(1+2+\dots+n)^* \# i(1+2+\dots+n)^*$ $\bigcup_{i \in \{1, \dots, n\}}$

שפות לא רגולריות:

$$L = \{0^n 1^n \mid n \geq 0\}$$

$$L = \{\varepsilon, 01, 0011, \dots\}$$

טענה: אין ל L DFA.

הוכחה: בשלילה יש DFA A כך ש $L(A) = L$. יש ל A k מצבים. נתבונן ב $w = 0^k 1^k$. המילה w מתקבלת ע"י $A \Leftarrow$ יש ריצה מקבלת. נתבונן בריצה $r = r_0 \dots r_1 \dots r_k r_{k+1} \dots r_{2k}$. מעקרון שובך היונים קיימים $0 \leq i < j \leq k$ כך ש $r_i = r_j$. $q = r_i = r_j$. $\Leftarrow$ האוטומט מקבל גם את $0^{k+(j-1)} 1^k$ למרות שאינה בשפה (יש שרטוט שמבהיר, סביר שיש אצל דינה) (לא היה לי כוח להעתיק אותו)).

למת הניפוח: pumping lemma

אם L רגולרית, אז קיים $p \geq 1$ (קבוע הניפוח) כך שלכל מילה w המקיימת $|w| \geq p$, יש חלוקה של $w = x \cdot y \cdot z$ כך ש:

$$1. xy^i z \in L, \forall i \geq 0.$$

$$2. |y| > 0.$$

$$3. |xy| \leq p.$$

$$L = (0+1)^* 0(0+1): \text{דוגמא:}$$

תנאי הלמה מתקיימים:

ניקח $p = 3$ - לכל $w \in (0+1)^* 0(0+1)$ כך ש $|w| \geq 3$ נתבונן בחלוקה של w כזו - $|x| = 0$, $|y| = 1$.

מתקיים:

1. לכל $i \geq 0$, המילה $xy^i z \in L$ - כלומר - $xy^i z \in L$ זה מתקיים כי $|z| \geq 2$ (כי הניפוח לא

נוגע בשתי האותיות האחרונות).

$$|y| \geq 0 \quad .2$$

$$(|y|=1, |x|=0 \text{ כי } |xy| \leq 3 \quad .3$$

19.3.2007

איפיון שפות רגולריות

הגדרנו $\sim_L \subseteq \Sigma^* \times \Sigma^*$ כך: $x \sim_L y \Leftrightarrow xz \in L \Leftrightarrow yz \in L$ אזי $z \in \Sigma^*$ לכל $z \in \Sigma^*$ אזי $xz \in L \Leftrightarrow yz \in L$.
ראינו לדוגמא – עבור $L = (0+1)^* 0(0+1)^*$ מתקיים $11 \sim_L 111$ אבל לא מתקיים $10 \sim_L 11$,
כי $101 \in L$ אבל $111 \notin L$ ($z=1$).

בנוסף, $\neg(01 \sim_L 11)$ - כי ε זנב מפריד.

נשים לב: $\sim_L$ יחס שקילות. נוכיח:

1. רפלקסיבי – לכל $x \in \Sigma^*$ מתקיים $x \sim_L x$
2. סימטריות - $x \sim_L y \Leftrightarrow y \sim_L x$
3. טרנזיטיביות – צ"ל לכל $x, y, w \in \Sigma^*$ אם $x \sim_L y$ ו $y \sim_L w$ אז $x \sim_L w$. בשלילה –
יש z כך ש $wz \in L$ ו $xz \notin L$ (בה"כ). נתבונן ב $z \cdot y$ מכיוון ש $y \sim_L x$ אז $yz \notin L$
מכיוון ש $y \sim_L z$ אז $yz \in L$ - סתירה.

על כן, היחס $\sim_L$ מחלק את המילים מעל Σ^* למחלקות שקילות. $[w] = \{x \mid x \sim_L w\}$. נמצא את

מחלקות השקילות של $\sim_L$ עבור $L = (0+1)^* 0(0+1)^*$

1. נשים לב – לא מתקיים $0 \sim_L \varepsilon$, שכן $01 \in L$ ו $1 \notin L$. לכן יש לנו לפחות 2 מחלקות שקילות - $0, \varepsilon$.
2. בנוסף - $1 \sim_L \varepsilon$ (בגלל ה $(0+1)^*$).
3. בנוסף - 00 פותח מחלקת שקילות חדשה (הוא מקבל)
4. 01 גם מחלקת שקילות.

סה"כ מחלקות השקילות:

1. $0, 10, \dots, \Sigma^* 10$
2. $1, \varepsilon, 11, \Sigma^* 11$
3. $00, \Sigma^* 00$
4. $01, \Sigma^* 01$

משפט: [Myhill – Nerode] – מספר מחלקות השקילות של $\sim_L$ (כמו שהוגדר לעל) סופי אם"מ L רגולרית.

הוכחה: רגולרית $\Leftarrow$ מספר המחלקות סופי. יהי $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ אוטומט דטרמיניסטי עבור L .
נגדיר יחס $\sim_A \subseteq \Sigma^* \times \Sigma^*$ כך - $x \sim_A y \Leftrightarrow \delta(q_0, x) = \delta(q_0, y)$ (האוטומט מגיע לאותו מצב בקריאת (x, y)).

טענה: מספר מחלקות השקילות של $\sim_A$ גדול או שווה ממספר מחלקות השקילות של $\sim_L$.

הוכחה: יהיו x, y כך ש $x \sim_A y \Leftarrow$ לכל זנב z , $xz \in L \Leftrightarrow yz \in L$ - וזה מתקיים כי $\delta(q_0, xz) = \delta(q_0, yz)$ (מהגדרה) – ולכן $x \sim_L y$.

הגדרה: זה כמו שנגדיר יחס $\equiv_2, \equiv_4$ - ברור כי כל מה ש $\equiv_4$ הוא גם $\equiv_2$. נאמר כי $\equiv_4$ הוא refinement ('עידון') של $\equiv_2$. במקרה שלנו $\equiv_4$ הוא הדומה ל $\sim_A$, ו $\equiv_2$ הוא הדומה ל $\sim_L$.

טענה 2: מספר מחלקות השקילות של $\sim_A$ הוא מספר המצבים הישיגים ($\equiv$ יש מילה שמגיעה אליהן) של A, שהוא כמובן $|Q| \geq$.

ועל כן, מספר מחלקות השקילות של $\sim_L$ סופי.

ביוון 2: מספר מחלקות השקילות של $\sim_L$ סופי $L \Leftarrow$ רגולרית. נגדיר DFA $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ עבור L כך:

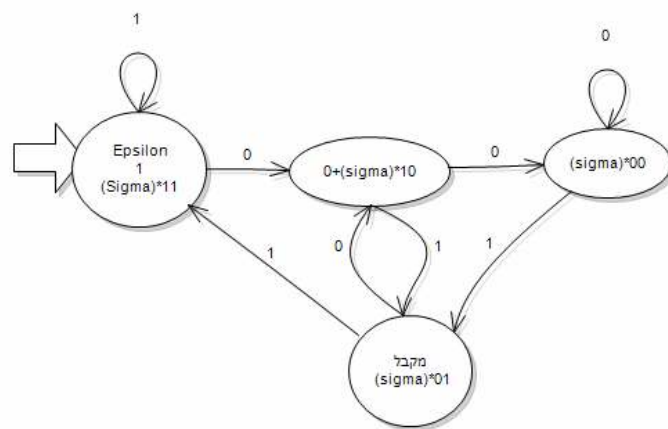
Q: מחלקות השקילות של $\sim_L$.

$\delta([x], a) = [xa]$: נשים לב שההגדרה לא תלויה בבחירת הנציג - אם $x \sim_L y$ אז $xa \sim_L ya$ (אחרת זנב מפריד z ל xa ל ya משרה זנב מפריד az ל x ו y).

$q_0 = [\varepsilon]$.

$F = \{[w] \mid w \in L\}$ - נשים לב כי מתקיים $\delta(q_0, w) = [w]$ - וניתן להוכיח מאוד בקצרה את זה באינדוקציה על $|w|$.

ניזכר ב $L = (0+1)^* 0(0+1)$ - האוטומט הוא בעצם כזה



שימושים של משפט Myhill-Nerode:

- הוכחת רגולריות / אי רגולריות של שפות
 - דוגמא: $0^n 1^n$ כך ש $n \geq 0$ אינה רגולרית
 - לכל $i \neq j \in \mathbb{N}$ לא מתקיים $0^i \sim_L 0^j$ שכן 1^i זנב מפריד $\Leftarrow$ יש אינסוף מחלקות שקילות.
- צמצום אוטומטים דטרמיניסטיים - $\sim_A$ - יחס מעל מילים Σ^* תלוי במימוש (באוטומט A). $\sim_L$ - תלוי בשפה. אפשר להגיע לשוויון.

שפות לא רגולריות - שפות חסרות הקשר – Context Free Languages

דקדוקים חסרי הקשר

$$A \rightarrow 0A1$$

$$A \rightarrow B$$

$$B \rightarrow \#$$

יש בדקדוק :

- משתנים (A, B)
- טרמינלים $(0, 1, \#)$
- חוקי גזירה $(A \rightarrow 0A1, \dots)$
- משתנה התחלתי (A) .

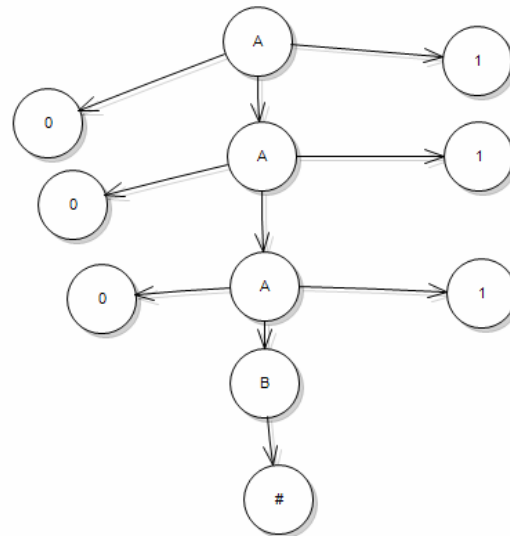
דוגמא לגזירת מילה בשפה:

$$A \xrightarrow{\text{first law}} 0A1 \xrightarrow{\text{first law}} 00A11 \xrightarrow{\text{first law}} 000A111 \xrightarrow{\text{second law}} 000B111 \xrightarrow{\text{third law}} 000\#111$$

גזירת מילה בשפה:

1. התחל מהמשתנה ההתחלתי
2. כל עוד יש במילה משתנים, מצא משתנה במילה והפעל את אחד מחוקי הגזירה הרלוונטיים.

עץ גזירה עבור $000\#111$:



CFL (Context Free Languages) – מחלקת השפות הניתנות להגדרה ע"י דקדוק חסר הקשר.

$$G = \left\langle \begin{array}{c} \underline{V} \\ \text{Variables} \end{array}, \begin{array}{c} \underline{\Sigma} \\ \text{Terminals} \end{array}, \begin{array}{c} \underline{R} \\ \text{derivation laws} \end{array}, \begin{array}{c} \underline{S} \\ \text{starting variables} \end{array} \right\rangle$$

חוק גזירה נראה כך - $V \rightarrow (V \cup \Sigma)^*$

אם $A \rightarrow w$ ו $w, u, v \in (V \cup \Sigma)^*$ אז $uAv \Rightarrow uwv$ - מייצר את uwv מ uAv .

נאמר ש $u \Rightarrow_* v$ אם יש סדרה $u_1, \dots, u_k$ עבור $k \geq 1$ כך ש $u = u_1 \Rightarrow u_2 \Rightarrow \dots \Rightarrow u_k = v$

השפה של הדקדוק G $L(G) = \{w \mid S \Rightarrow_* w \wedge w \in \Sigma^*\}$

נכתוב דקדוק עבור $(0+1)^* 0(0+1)$

$$V = \{S, A\}$$

$$\Sigma = \{0, 1\}$$

$$S \rightarrow A00$$

$$S \rightarrow A01$$

$$A \rightarrow \varepsilon \mid 0A \mid 1A$$

עוד דקדוק:

$$G = \langle \{S\}, \{a, b\}, R, S \rangle$$

$$R: S \rightarrow aSb \mid SS \mid \varepsilon$$

מה השפה? מכילה מילים כדוגמת:

abab, aaabbbb, aababb

אפשר להגיד שזו 'שפת הסוגריים המקוננים חוקית' – אם $a = '('$ ו $b = ')'$ אז היא מקבלת את כל הסוגריים המקוננים בצורה חוקית.

למה היא לא רגולרית? $L(G) \cap a^*b^* = \{a^n b^n \mid n \geq 0\}$ והצד הימני של השוויון לא רגולרי, בעוד a^*b^* רגולרי.

דקדוק עבור $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$

$$S \rightarrow A \mid B$$

$$A \rightarrow 0A1 \mid \varepsilon$$

$$B \rightarrow 1B0 \mid \varepsilon$$

26.3.2007

דקדוקים רבי משמעות – לדוגמא,

$$E \rightarrow E + E \mid E \times E \mid 0 \mid 1 \mid \dots \mid 9$$

יכול לגזור את $3 \times 5 + 2$ - ולא ברור למה הכוונה (סדר הפעולות).

$$L(G) \subseteq \Sigma^*$$

$$E \rightarrow D + E \mid D \times E$$

פתרון אחר יהיה -

$$D \rightarrow 0 \mid \dots \mid 9$$

הגדרה: דקדוק הוא **רב משמעי** אם יש מילה w עם שתי גזירות שמאליות ביותר שונות. נשים לב – יש שפות שכל דקדוק שגזור אותן הוא רב משמעי

$$L = \{0^n 1^n 2^n, n \geq 0\} \notin CFL$$

תזכורת:

$$L' = \{0^i 1^j 2^k \mid i = j \vee j = k\}$$

נתבונן ב

ברור ש $L \subseteq L'$, אבל לא ההפך.

$$L' = \{0^n 1^n 2^* \mid n \geq 0\} \cup \{0^* 1^n 2^n\}$$

נשים לב כי - ואז אפשר לגזור זאת כך –

$$S \rightarrow S_0^1 \mid S_0^2$$

$$S_0^1 \rightarrow T_0^1 R$$

$$T_0^1 \rightarrow 0 T_0^1 1 \mid \varepsilon$$

$$R \rightarrow 2 R \mid \varepsilon$$

$$S_0^2 \rightarrow L T_0^2$$

$$T_0^2 \rightarrow 1 T_0^2 2 \mid \varepsilon$$

$$L \rightarrow 0 L \mid \varepsilon$$

הדקדוק רב משמעי: יש שתי גזירות שמאליות שונות לכל מילה ב L .
אין דקדוק חד משמעי – כי דקדוק כזה 'דומה' לדקדוק עבור L שאינה חסרת הקשר.

אוטומטי מחסנית (Pushdown Automata) – **PDA**
<http://he.wikipedia.org/wiki/%D7%90%D7%95%D7%98%D7%95%D7%9E%D7%98%D7%9E%D7%97%D7%A1%D7%A0%D7%99%D7%AA>

נוסיף מחסנית – **בעומק לא חסום**. האוטומט קורא את מילת הקלט, וגם את האות בראש המחסנית. זה קרוב למחשב, עם הבדל של מחשב יש גישה אקראית, לא רק מחסנית.

נכתוב PDA עבור $0^n 1^n$.

- בקריאת 0, נדחוף 0 למחסנית.
- בקריאת 1, נשלוף מהמחסנית.

נגדיר פורמלית:

PDA – נוסף כאן אלמנט נוסף – א"ב המחסנית – נסמנו ב Γ (גאמא גדולה).
 נשים לב ש δ נהפכת לפונקציה ממש יפה:

$$A = \langle Q, \Sigma, \Gamma, \delta, Q_0, F \rangle$$

$$\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \rightarrow 2^{Q \times (\Gamma \cup \{\varepsilon\})}$$

נשים לב ש $\langle q', \gamma' \rangle \in \delta(q, \sigma, \gamma)$ משמחו שכש A במצב q, קורא σ והאות בראש מחסנית היא ε , אז A עובר למצב q' , שולף את γ , דוחף במקומה את γ' .

קונפיגורציה של A: מצב + מחסנית

$s \in \Gamma^*$ - לדוגמא $s = abc$.

אם $\langle q, s \rangle \rightarrow_{\sigma} \langle q', s' \rangle$ נאמר ש $\langle q', s' \rangle$ היא σ עוקבת ל $\langle q, s \rangle$ - כלומר – כש A במצב q ותוכן המחסנית הוא s, אז יש אפשרות לקרוא σ , לעבור ל q' ולעדכן את המחסנית ל s' (מחליפים את האות העליונה ל b).
 זה אפשרי אם s ניתן לחלוקה ל $a \cdot t$ כאשר $a \in \Gamma \cup \{\varepsilon\}$ ו $t \in \Gamma^*$ כך ש:

- $\langle q', b \rangle \in \delta(q, \sigma, a)$
- $s' = b \cdot t$

ריצה של A על מילה $w \in \Sigma^*$: סדרה של קונפיגורציות $c_0, \dots, c_n$ כך ש w ניתנת לכתיבה

כ $w_1 \dots w_m$ ו $w_i \in \Sigma \cup \{\varepsilon\}$

- $c_0 = \langle q_0, \varepsilon \rangle$ עבור $q_0 \in Q_0$.
- לכל $0 \leq i \leq m-1$, הקונפיגורציה c_{i+1} היא w_{i+1} עוקבת ל c_i .

קונפיגורציה התחלתית: $\langle q, \varepsilon \rangle$ עבור $q \in Q_0$

קונפיגורציה מקבלת: $\langle q, s \rangle$ עבור $q \in F$

ריצה מקבלת: הריצה $c_0, \dots, c_m$ **מקבלת** אם c_m היא מקבלת.

אחרי ההגדרה האינטואיטיבית הנ"ל, הגיע הזמן לתאר את $0^n 1^n$ ($\forall n \in \mathbb{N}$).

$$A = \langle Q, \Sigma, \Gamma, Q_0, \delta, F \rangle$$

$$Q = \{q_i\}_{i=1}^4, \Sigma = \{0, 1\}, \Gamma = \{0, \$\}, F = \{q_1, q_4\}$$

\$: אות שנהוגה לסימון תחתית המחסנית.

הרעיון - מ q_1 ל q_2 פשוט נשים \$ בתחתית המחסנית. ב q_2 נערים אפסים כמספר האפסים שקיבלנו. ככל ברגע שנראה 1 - נעבור ל q_3 , ונוריד 0 מהמחסנית.

נרשום את המעברים פורמלית -

$$1. \delta(q_1, \varepsilon, \varepsilon) = \{\langle q_2, \$ \rangle\} \text{ (הוספת \$ בהתחלת המחסנית)}$$

a. נאמר כי $\langle q_2, \$ \rangle$ היא עוקבת ל $\langle q_1, \varepsilon \rangle$.

$$2. \delta(q_2, 0, \varepsilon) = \{\langle q_2, 0 \rangle\} \text{ - לא משנה מה יש בראש המחסנית, אני אדחוף 0 אם קיבלתי 0.}$$

a. נאמר כי $\langle q_2, 00\$ \rangle$ היא 0 עוקבת ל $\langle q_2, 0\$ \rangle$.

$$3. \delta(q_2, 1, 0) = \{\langle q_3, \varepsilon \rangle\} \text{ - כשאני קורא 1, אני לוקח 0 מראש המחסנית, עובר ל } q_3, \text{ ולא}$$

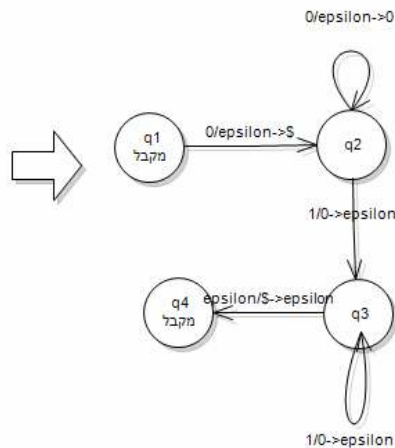
מכניס כלום במקום

$$4. \delta(q_3, 1, 0) = \{\langle q_3, \varepsilon \rangle\} \text{ - כשאני קורא 1, אני מוציא 0 מראש המחסנית, נשאר ב } q_3, \text{ ולא}$$

מכניס כלום במקום.

$$5. \delta(q_3, \varepsilon, \$) = \{\langle q_4, \varepsilon \rangle\}$$

נשים לב שלא הגדרנו $\delta(q_3, 1, \$)$ - זה מדבר על מקרים בהם יש יותר 1 מ 0.



תיאור הריצה של האוטומט על $\varepsilon 0011\varepsilon$ - אצל דינה.

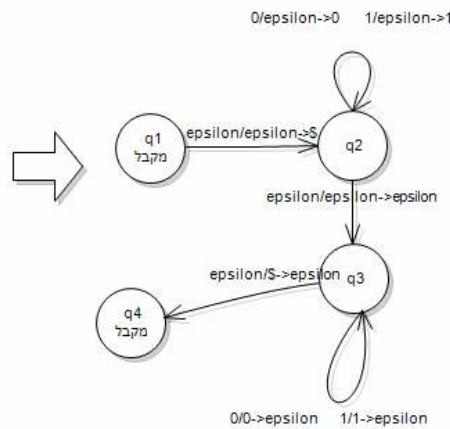
כעת נכתוב אוטומט מחסנית לפולינדרומים באורך זוגי

$$L = \{w \cdot w^R \mid w \in \Sigma^*\}$$

הרעיון - לדחוף את האותיות הנקראות למחסנית. כשנחש שהגענו לאמצע, נוציא אותיות מהמחסנית ונוודא שהן מסכימות עם אותיות הקלט.

q_2 - מה שאנו מנחשים שהוא החצי הראשון.

q_3 - קוראים את החצי השני.



הדגמת הריצה (המקבלת)

$$(q_1, \varepsilon) \xrightarrow{\varepsilon} (q_2, \$) \xrightarrow{0} (q_2, 0\$) \xrightarrow{1} (q_2, 10\$) \xrightarrow{1} (q_2, 110\$) \xrightarrow{\varepsilon} (q_3, 110\$) \xrightarrow{\varepsilon} (q_3, 110\$) \xrightarrow{1} (q_3, 10\$) \xrightarrow{1} (q_3, 0\$) \xrightarrow{0} (q_3, \$) \xrightarrow{\varepsilon} (q_4, \varepsilon)$$

נתבונן ב $L = \{ww \mid w \in \Sigma^*\}$ (מילים באורך זוגי / 0) החצי הראשון שווה לחצי השני. זוהי לא שפה חסרת הקשר.

למת הניפוח לשפות חסרות הקשר

אם L שפה חסרת הקשר אז יש $p \geq 1$ כך שלכל מילה $w \in L$ כך ש $|w| \geq p$, קיימת חלוקה $w = uvxyz$ המקיימת:

$$1. \text{ לכל } i \geq 0, uv^i xy^i z \in L$$

$$2. |vy| > 0$$

$$3. |xyz| \leq p$$

רעיון ההוכחה: נתבונן במילה w ארוכה. יש ל w עץ גזירה – יש מסלול מ S לטרמינל, ויש משתנה R שחוזר במסלול לפחות פעמיים.

נתבונן בחלוקה המושרית מהעץ – ונטען שהיא מקיימת את התנאים:

$$1. \text{ עבור } i = 0, uxz \in L - \text{ יגזור את } uxz \text{ (ציור אצל דינה).}$$

$$2. \text{ עבור } i > 1, \text{ ידוע ש } S \text{ גוזר את } uRz \text{ וכי } R \rightarrow vRy, \text{ ואפשר לבצע זאת אינדוקטיבית.}$$

28.5.2007

נכון לעכשיו, לא יהיה בוחן בהמשך הקורס.

מה כן יהיה בהמשך הקורס:

ככה"נ המבחן יזוז ב-3 שבועות קדימה.

חומר הקורס:

1. אוטומטים, שפות רגולריות, חסרות הקשר, ...
2. חישוביות
3. סיבוכיות
4. העשרה.

יום שישי שיעור השלמה בין 9:00 ל-11:00.

נדלג על ההוכחה ששפות חסרות הקשר שקולות לאוטומטי מחסנית. וכאן נסיים את החלק הראשון, של אוטומטים.

בשל השביתה, נוותר על ההעשרה. כעת נתחיל את תורת החישוביות.

תורת החישוביות

ראינו שני מודלי חישוב -

NDFA – שפות רגולריות

PDA – שפות חסרות הקשר

נגדיר מודל בשם **מכונת טיורינג** (באופן לא פורמלי – כל מה שמחשב יכול לעשות).
נגדיר שפה:

$$L = \{w\#w \mid w \in (0+1)^*\}$$

L אינה חסרת הקשר, ולא רגולרית. אבל קל לכתוב תכנית שמקבלת כקלט מילה מעל $\Sigma = \{0,1,\#\}$ ופולטת 'כן' אם המילה ב- L .

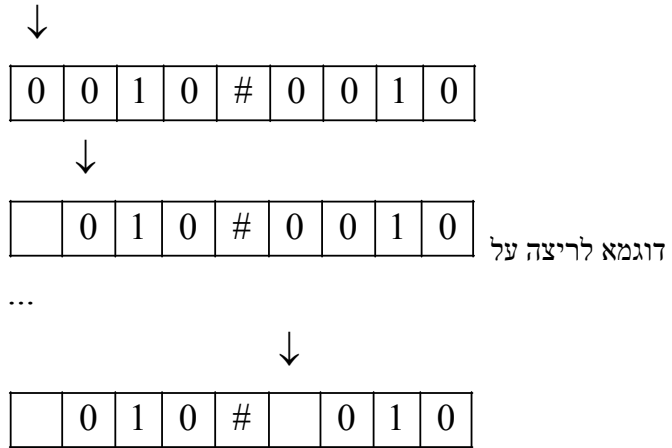
המודל: מכונת טיורינג (Allan Turing, 1936)

- סרט עבודה אינסופי
- יכולת לקרוא ולכתוב מ/על הסרט
- יכולת לזוז עם הראש הקורא ימינה ושמאלה.
- מצבי קבלה ודחיה.

בואו ונתאר באופן לא פורמלי מ"ט (מכונת טיורינג) עבור $L = \{w\#w \mid w \in (0+1)^*\}$.

ניקח את הסרט ('נייר טואלט אינסופי') שעליו כתובה המילה. מה נעשה?
 M פועלת כך:

- סורקת את הקלט ומוודה שיש # אחת בדיוק. אם לא, דוחה.
- נזגזג בין מיקומים תואמים בסרט הקלט.
 - תואמים – התא הראשון, והתא הראשון אחרי ה#
 - אחר כך: התא הראשון הלא מחוק, והתא הראשון הלא מחוק אחרי ה#.
- אם מסומנים באותה אות, מחוק אותם וממשיך.
 - אחרת, דוחה.
- כשנמחקו כל התאים משמאל ל#, בודק האם יש תאים לא מחוקים מימין. אם יש, דוחה. אחרת מקבלת.



וכך הלאה!¹

מכונת טיורינג הגדרה פורמלית:

$$M = \left\langle \underbrace{Q}_{\text{states}}, \underbrace{\Sigma}_{\text{input Alphabet}}, \underbrace{\Gamma}_{\text{Work Alphabet}}, \delta, q_0, q_{\text{accept}}, q_{\text{reject}} \right\rangle$$

Q – מצבים

Σ – א"ב קלט (אינו כולל את האות 'רווח' _)

Γ – א"ב עבודה (תמיד מתקיים $\Sigma \subseteq \Gamma$), וגם $_ \in \Gamma$ (רווח): δ

עבור מכונה דטרמיניסטית: $Q \times \Gamma \rightarrow Q \times \Gamma \times \Delta$, כאשר $\Delta \in \{L, R\}$. לדוגמא -

$\delta(q, a) = \langle q', b, L \rangle$ - כש M במצב q וקוראת a , היא עוברת למצב q' , כותבת b במקום a

וזה עם הראש הקורא תא אחד שמאלה.

עבור מכונה לא דטרמיניסטית: $\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \Delta}$

קונפיגורציה של מ"ט

1. המצב
2. תוכן הסרט
3. מיקום הראש הקורא

תאור קונפיגורציה: בהנתן מצב q ומילים $u, v \in \Gamma^*$, uqv היא קונפיגורציה שבה:

- המצב הוא q
- תוכן הסרט $u \cdot v$
- הראש הקורא מצביע על האות הראשונה ב v .

קונפיגורציה התחלתית: $q_0 w$ (על מילת קלט w).

(ניתן להתייחס לזה כ $q_0 w$ (עם רווחים עד אינסוף)).

קונפיגורציה סופית: המצב הוא q_{acc} (מקבלת) או שהמצב הוא q_{rej} (דוחה).

קונפיגורציות עוקבות:

¹ ככה"נ אצל [דינה](#) יש המשך

יהיו $\delta(q,b) = (q',c,L)$ ו $q,q' \in Q, u,v \in \Gamma^*, a,b,c \in \Gamma$ אזי $uaqbv \rightarrow uq'acv$.
 אם $\delta(q,b) = (q',c,R)$ אזי $uaqbv \rightarrow uacq'v$.

מקרה פרטי: הראש הקורא מצביע על התא השמאלי ביותר:

$qbv \rightarrow q'cv$
 הראש לא נופל, פשוט מתעלמים מההוראה שמאלה:
 $qbc \rightarrow cq'v$

M מקבלת את w אם קיימת סדרת קונפיגורציות $c_0, \dots, c_k$ כך ש c_0 התחלתית $(c_0 = q_0 w)$, c_k מקבלת, ולכל $i \geq 0$, מתקיים כי c_{i+1} עוקבת ל c_i .
 השפה של M, $L(M)$ - המילים של L מקבלת.

נאמר ש M **מזהה** את L אם $L(M) = L$.

L היא Recursively Enumerable אם קיימת מכונת טיורינג המזהה את L (Turing Recognizable).

נשים לב: יש שלושה גורלות אפשריים לריצה:

1. מגיעה לקונפיגורציה מקבלת.
2. מגיעה לקונפיגורציה דוחה.
3. **אינסופית (LOOP).**

נאמר ש M **מכריעה** את L אם M מזהה את L ובנוסף M **עוצרת על כל הקלטים**.

נאמר ששפה L היא Recursive אם קיימת מ"ט המכריעה אותה.

(נשים לב $R \subseteq RE$ באופן ברור (אם יש מכונה המכריעה, אז היא מזהה)). נראה בעתיד כי

$$R \not\subseteq RE$$

דוגמא: M שמזהה את $L = \{0^{2^n} \mid n \geq 0\}$ (מילים שמספר האפסים הוא חזקה של 2).

$$L = \{0, 00, 0000, 0^8, 0^{16}, \dots\}$$

הרעיון: נגדיר פרדיקאט מעל הטבעיים.

$$Good(k) \Leftrightarrow \exists n \in \mathbb{N}, k = 2^n$$

נשים לב כי $Good(k) \Leftrightarrow k = 1 \text{ or } good\left(\frac{k}{2}\right)$.

לכן, M תפעל כך:

- א. סרוק את הסרט משמאל לימין, מחק כל 0 שני.
- ב. אם היה בקלט בשלב א' רק 0 אחד, קבל
- ג. אם היה בקלט בשלב א' מספר אי זוגי של אפסים, דחה.
- ד. חזור עם הראש הקורא לתחילת הסרט
- ה. לך לשלב א'

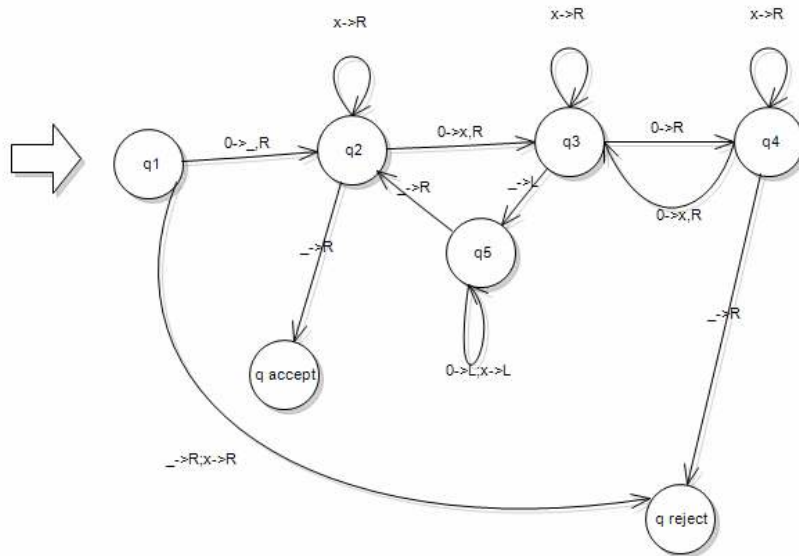
תאור פורמלי של המכונה:

$$M = \langle Q, \Sigma, \Gamma, q_1, q_{acc}, q_{rej} \rangle$$

$$\Sigma = \{0\}$$

$$\Gamma = \{0, _, x\}$$

$$Q = \{q_1, \dots, q_5, q_{acc}, q_{rej}\}$$



הערות:

q_1 - הופך את ה 0 הראשון לרווח.

q_3 - מספר האפסים עד כה זוגי.

q_4 - מספר האפסים אי זוגי עד כה.

נישאר בלולאה של q_3, q_4 עד שנגיע לרווח (מסמן את סיום המילה). q_5 יקח אותי שמאלה כל עוד יש אפסים ו x ים, עד שנגיע לרווח הראשון.

מחלקות:

RE - יש מ"ט שמזהה (יתכן ונתקע בלופ)

R - יש מ"ט שמכריעה (לא נתקע בלופ)

$co-RE$: כל השפות L כך ש $L \in RE$ - $\Sigma^* - L$

הגדרה: $\bar{L} \in co-RE \Leftrightarrow L \in RE$

משפט: $R = RE \cap co-RE$

כיוון אחד: אם $L \in R$, אזי $L \in RE$, ולכן $L \in co-RE$ (אפשר לקבל מ"ט שמקבלת משלים של שפה על ידי החלפה בין המצב המקבל והדוחה ממ"ט שמכריעה את L).
כיוון שני:

1.6.2007

התיזה של Turing Church

David Hilbert - 1900 נתן 23 בעיות פתוחות במתמטיקה. רובן הגדול כבר נפתר (נותרו 10 שעדיין פתוחות). אפשר לחפש על ידי 'Hilbert Problems' בgoogle.

הבעיה העשירית של Hilbert

הבעיה: לתאר אלגוריתם שבהנתן פולינום במספר משתנים P , יכרי האם יש ל P שורש שלם. לדוגמא – האם ל $6x^3yz^2 + 3xy^2 - x^3 - 10$ יש שורש שלם (לדוגמא $y=5, y=3, z=0$). אלגוריתם עבור הילברט – תהליך שאיתו ניתן להכריע אחרי מספר סופי של צעדים.

היום יודעים שאין אלגוריתם כזה.

התזה של Turing Church: $\Leftrightarrow$ Turing Maching. Church לא דיבר על מכונת טיורינג, אלא על משהו דומה, שנקרא $(\lambda - calculus)$.

$$L = \{ \langle p \rangle \mid p \text{ is polinom with round square} \}$$

קל לראות: $L \in RE$. (נניח שיש ל p m משתנים. נעבור על כל הערכים האפשריים ל m המשתנים. – מאחר ומדובר בוקטור של מספרים בני מניה, אין שום בעיה למצוא לוקטור הזה סידור).

הבעיה העשירית היא בעצם $L \in ? R$ בניסוח הפורמלי.

(נשים לב, עבור $m=1$, הוכיחו שהבעיה היא ב R , מאחר ואם קיים שורש x_0 , אז קיים גם שורש x_0)

המקיים $-k \frac{c_{\max}}{c_1} \leq x_0 < k \cdot \frac{c_{\max}}{c_1}$ מספר הביטויים, $c_{\max}$ מקדם מקסימלי, c_1 מקדם ראשון).

נניח שיש לנו את התזה של Turing Church (ישכנעו אותנו בתרגולים).

תאור אלגוריתם ע"י מכונת טיורינג:

1. תאור פורמלי - $M = \langle Q, \Sigma, \Gamma, \dots \rangle$ - 'זה איכסה, לא עושים'.

2. רמת המימוש – תיאור בשפה עילית של פעולת מכונת הטיורינג.

- a. מחק את הסרט
- b. כתוב על הסרט
- c. סרוק את הסרט
- d. ...

3. רמה אבסטרקטית – תאור האלגוריתם בשפה טבעית.

בעיה: אלגוריתמים רצים מעל גרפים, מטריצות, ...

מכונות טיורינג מעל שפות (א"ב Σ).

פתרון: קל לקודד. נסמן $\langle x \rangle$ כדי לסמן קידוד של העצם x .

נשים לב שאין בעיה לקודד מ"ט במ"ט.

מ"ט המכריעה האם גרף לא מכוון הוא קשור

* - G גרף לא מכוון קשיר $A = \{ \langle G \rangle \mid * \}$.

M פועלת כך: על קלט G המקודד גרף, אם מילת הקלט אינה מקודדת גרף, דוחים.

- בחר קודקוד V וסמן אותו:
- חזור עד שלא מסומנים קודקודים חדשים.
- לכל קודקוד ב G , סמן אותו אם אמ"מ יש קשת בינו לבין קודקוד מסומן.

- עבור על כל קודקודי G , אם כולם מסומנים קבל, אחרת דחה.

ענייני מימוש:

- סמן אותו – לדוגמא, אם רשמנו את הגרף כך - $(v_1, v_2) \# v_1 \dots v_2$, אפשר להחליף את האות σ הראשונה בקידוד של v (הקודקוד שאנו מסמנים), באות σ^* .
- לפי סדר, נמצא קודקוד לא מסומן, נסמן אותו כ'קודקוד נבדק' (לדוגמא - $\sigma \rightarrow \sigma$).

שפות / בעיות כריעות

בעיות מתורת האוטומטים:

1. בעיית השייכות: בהנתן A DFA, ומילה w , האם $w \in L(A)$?
2. בעיית הריקנות: בהנתן A DFA, האם $L(A) \neq \emptyset$?

שתי הבעיות ב R – בואו ונוכיח.

$$L_{DFA} = \{ \langle A \rangle, w \mid A \text{ represents DFA, and } w \in L(A) \}$$

טענה: $L_{DFA} \in R$.

M : על קלט $\langle A \rangle w$, תסמלך ריצה של A על w .

נעבוד עם 3 סרטים:

1. $\langle A \rangle w$
2. שני – מסמל את המצב
3. שלישי – מסמל את מיקום הראש הקורא על המידע.

אם הסימולציה הסתיימה במצב מקבל, קבל. אם במצב דוחה, דחה.
אם האוטומט הוא לא דטרמיניסטי:

1. אופציה א' – M תייצר אוטומט דטרמיניסטי שקול.
2. אופציה ב' – לתחזק סרט ובו המצבים שבהן A עשוי להיות בריצה כלשהיא עבור מיקום הראש הנוכחי.

אי כריעות

משפט: יש שפה $L \notin R$.

הוכחה (משיקולי ספירה) – 'למתחכמים':

קבוצה A: כל השפות מעל $\{0,1\}$ (אותה עצמה כמו תתי קבוצות של וקטור מעל $\{0,1\}$ - 2^{*0} - לא בר מניה).

קבוצה B: מכונות טיורינג המתארות שפה ב R – מ"ט $\leftarrow$ קידוד סופי $\leftarrow$ בר מניה.

על כן - $|A| < |B|$ ועל כן, יש $L \notin R$.

הוכחה על ידי דוגמא: נגדיר

$A_{TM} = \{ \langle M \rangle, w \mid M \text{ represents a Turing machine, and } M(w) = \text{accept} \}$
נטען כי $A_{TM} \notin R$. (נשים לב כי $A_{TM} \in RE$ - ניתן פשוט למסלך, ואם w מוכל ב $L(M)$, אז נעצור).

נשים לב כי נובע כי $\overline{A_{TM}} \notin RE$
 $\overline{A_{TM}} = \{ \langle M \rangle, w \mid M \text{ represents a Turing Machine, } M(W) \neq \text{acc} \}$

(נשים לב – בהשלמה, היינו צריכים לקחת גם את כל מה שהוא לא קידוד של מכונה, אבל מזה אנו מתעלמים). זה נכון גם אינטואיטיבית – איך אפשר לדעת אם מכונה לא עצרה על מילה? אולי היא עוד תעצור?

הוכחת הנשים לב: $A_{TM} \in RE$, ונראה כי $A_{TM} \notin R$. אם $\overline{A_{TM}} \in RE$ אזי $A_{TM} \in co-RE$, ואז $A_{TM} \in R$ (מההוכחה מהשיעור הקודם), בסתירה.

כעת נוכיח כי $A_{TM} \notin R$. נניח בשלילה כי $A_{TM} \in R$, ועל כן יש מ"ט H שמקבלת קלט $\langle M \rangle, w$,

$$H(\langle M \rangle, w) = \begin{cases} \text{Accept}, M(w) = \text{accept} \\ \text{Reject}, M(w) \neq \text{accept} \end{cases}$$

מ H נבנה מ"ט D , שמקבלת כקלט תאור $\langle M \rangle$ של מ"ט טיורינג ופועלת כך:

$$D(\langle M \rangle) = \begin{cases} \text{accept}, M(\langle M \rangle) = \text{accept} \\ \text{reject}, M(\langle M \rangle) \neq \text{accept} \end{cases}$$

D תפעל כך – בהינתן קלט $\langle M \rangle$, היא תריץ את $H(\langle M \rangle, \langle M \rangle)$.

מ D נבנה D' שמקבלת כקלט תאור $\langle M \rangle$ של מכונת טיורינג,

$$D'(\langle M \rangle) = \begin{cases} \text{reject}, M(\langle M \rangle) = \text{accept} \\ \text{accept}, M(\langle M \rangle) \neq \text{accept} \end{cases} \text{ ו } q_{rej} \text{ ו } q_{acc} \text{ החלפת } D \text{ מתקבלת מ } D'.$$

$$D'(\langle D' \rangle) = \begin{cases} \text{reject}, D'(\langle D' \rangle) = \text{accept} \\ \text{accept}, D'(\langle D' \rangle) \neq \text{accept} \end{cases} \text{ נתבונן בריצה של } D' \text{ על } \langle D' \rangle.$$

לסתירה.

4.6.2007

בשיעור קודם ראינו כי
 כי $A_{TM} = \{ \langle M \rangle, w \mid M \text{ represents Turing machine} \wedge w \in L(m) \}$ ראינו
 $A_{TM} \notin R$. לפרטים – שיעור קודם, או [דינה](#) (יש שרטוט יפה, שאין לי כוח להעתיק).

מכונת טיורינג אוניברסלית

מושג: מ"ט אוניברסלית – מסומנת בדרך כלל בשם U : מ"ט המקבלת כקלט מכונת טיורינג M ומילה w ומסמלצת ריצה של M על w .
 איך U פועלת? U פועלת כך – בעלת שני סרטים (ראינו שזה שקול למ"ט עם סרט אחד בתרגול).
 סרט הקלט:

1. $\langle M \rangle, w$ (סרט הקלט)

2. הקונפיגורציה הנוכחית של M על w . (סרט הסימולציה)

בהינתן $\langle M \rangle, w$ המכונה U כותבת $q_0 w$ על סרט הסימולציה. בכל שלב, U קוראת מה המצב הנוכחי q והאות a שהראש הקורא מצביע עליה, מחפשת את $\delta(q, a)$ בסרט הקלט ומעדכנת את הקונפיגורציה בהתאם.

אם M מגיעה ל q_{acc} , U מקבלת.

אם M מגיעה ל q_{rej} , U דוחה.

אם M לא עוצרת לעולם, U לא עוצרת לעולם.

נגדיר $L_5 = \{ \langle M \rangle, w \mid M \text{ stops on } w \text{ in 5 steps} \}$. ברור כי $L_5 \in R$ (שכן אפשר לסמלץ, ואם לא עצרנו תוך 5 צעדים, אז נדחה).

בעיית העצירה:

$$HALT_{TM} = \{ \langle M \rangle, w \mid M \text{ stops on } w \}$$

ברור כי $HALT_{TM} \notin R$ (שקול למה שראינו בשיעור הקודם – אם היינו יודעים להכריע בשאלה זו, היינו מכריעים את A_{TM} . זוהי רדוקציה). טריוויאלי לראות ש $HALT_{TM} \in RE$.

מהי הרדוקציה?

הרעיון – נראה ש $HALT_{TM}$ קשה לפחות כמו A_{TM} – לו היינו יודעים להכריע את $HALT_{TM}$ אז היינו מצליחים להכריע גם את A_{TM} .

משפט: $HALT_{TM} \notin R$.

הוכחה: בשלילה $HALT_{TM} \in R$, ועל כן קיימת מ"ט R המכריעה את $HALT_{TM}$. נייצר מ"ט S המכריעה את A_{TM} כך:

1. מריצה את R על $\langle M \rangle, w$. אם R דוחה (כלומר, M לא עוצרת על w - $M < w$ לא מקבלת את w), S תדחה.

2. אם R מקבלת, M עוצרת על w $S \Leftarrow w$ מריצה את M על w . מובטח שתעצור (כי R מקבלת), S תדחה / תקבל בהתאם לריצה של M על w .

מ.ש.ל.

עוד שפה שאיננו כריעה (ופותרת תרגילי בית בחצי הראשון של הקורס...)

$$Regular_{TM} = \{ \langle M \rangle \mid L(M) \text{ is regular} \}$$

טענה: $Regular_{TM} \notin R$.

הוכחה: (רדוקציה מ A_{TM} - נראה שאם זה אפשרי, A_{TM} היתה כריעה). נניח בשלילה שיש מכונה R המקבלת $\langle M \rangle$, ומחזירה כן אם $L(M)$ רגולרית, ולא אחרת.

איך נבנה R כזו את A_{TM} ? (באיזה מכונה M' (והקלט שלה) נאכיל את R כך שבאמת $L(M')$ רגולרית אמ"מ $(\langle M \rangle, w) \in A_{TM}$).

נבנה מהקלט $\langle M \rangle, w$ את M' שתפעל על קלט x כך:

1. אם x מהצורה $0^n 1^n$, M' מקבלת את x. (נשים לב – מתעלמים מהקלט $\langle M \rangle, w$)
2. אחרת, M' מריצה את M על w ומקבלת את x אמ"מ M מקבלת את w $(\langle M \rangle, w)$ הם הקלט, כזכור – ובמקרה זה, היא מתעלמת מהקלט שלה x, אבל תלויה ב $\langle M \rangle, w$. יתכן ש M' **לא תעצור**.

נשים לב:

אם M לא מקבלת את w, אז $L(M') = \{0^n 1^n \mid n \geq 1\} \notin REG$.

אם M מקבלת את w, אז $L(M') = \{(0,1)^*\} \in REG$.

כלומר, הצלחנו לייצר את A_{TM} . ☺

איך בדיוק?

כך פועלת המכונה S שמכריעה את A_{TM} :

1. מייצרת את $M'(m, w)$.
2. מריצה את R על $M'(M, w)$. אם R מקבלת $M(M')$ (רגולרית), ואז S מקבלת. אם R דוחה $(L(M'))$ (לא רגולרית), S דוחה.

בשלילה לכך ש $A_{TM} \notin R$.

פונקציה ניתנת לחישוב

הגדרה: $f: \Sigma^* \rightarrow \Sigma^*$ היא פונקציה ניתנת לחישוב (Computable Function) אם"מ קיימת מכונת

טיורינג M_f כך שלכל קלט $w \in \Sigma^*$, M_f עוצרת עם $f(w)$ על הסרט.

דוגמא: $f: x \# y \rightarrow x + y$ עבור x, y באונרי (או בינארי, או כל בסיס אחר).

נשים לב, שזו יכולה להיות $f: TuringMachine \rightarrow TuringMachine$.

$f(\langle M \rangle) = M'$ מ"ט M' כך ש M' לעולם לא מנסה לפול שמאלה מהסרט. גם זו פונקציה ניתנת לחישוב.

רדוקציות מיפוי

הגדרה: שפה $A \subseteq \Sigma^*$ ניתנת לרדוקצית מיפוי $B \subseteq \Sigma^*$ (נסמן $A \leq_m B$) אם יש פונקציה ניתנת

לחישוב $f: \Sigma^* \rightarrow \Sigma^*$ כך שלכל $w \in \Sigma^*$, $w \in A \Leftrightarrow f(w) \in B$.

רדוקציה: המרה של שאלות על חברות A לשאלות על חברות B (בד"כ כי הוכחנו כבר משהו על B).

משפט (1): אם $A \leq_m B$, $B \in R$ ו $A \in R$.

הוכחה: בהינתן M_B שמכריעה את פונקציה שניתנת לחישוב $f: \Sigma^* \rightarrow \Sigma^*$ כך שלכל $w \in \Sigma^*$

$$w \in A \Leftrightarrow f(w) \in B \text{ נבנה } M_A \text{ שמכריעה את } A.$$

על קלט w , M_A מחשבת את $f(w)$ ומריצה את M_B על $f(w)$.

משפט (2): אם $A \leq_m B$ ו $A \in R$ אז $B \notin R$ (חסם תחתון).

הוכחה: נובעת לוגית מהמשפט הקודם.

נשים לב: זהו כלי מרכזי להוכחת אי כריעות!

נפרמל את הדוגמא שראינו לאי כריעות $HALT_{TM}$.

נוכיח כי $HALT_{TM} \leq_m A_{TM}$, ועל סמך משפט 2 והעובדה ש $A_{TM} \notin R$ ינבע ש $HALT_{TM} \notin R$.

עלינו להראות פונקציה ניתנת לחישוב f כך ש $\langle M \rangle, w \in A_{TM} \Leftrightarrow \langle M \rangle, w \in HALT_{TM}$.

כלומר, זוהי $f: \langle M \rangle, w \rightarrow \langle M' \rangle, w'$.

חישוב $f: \langle M \rangle, w \rightarrow \langle M' \rangle, w'$ - בהינתן $\langle M \rangle, w$, M_f תחזיר M' ו w' כך:

חישוב M' - על קלט x :

1. M' מריצה את M על x
 2. אם M מקבלת (את x), גם M' מקבלת.
 3. אם M דוחה (את x), M' נכנסת ל loop.
- את w' נגדיר כ w .

מייצרים M' ($f(M)$) כזו ש $\langle M' \rangle, w \in HALT_{TM}$ אם $\langle M \rangle, w \in A_{TM}$.

אם $\langle M \rangle, w \in A_{TM}$ אז $\langle M' \rangle, w \in HALT_{TM}$.

אם $\langle M \rangle, w \in A_{TM}$ אז $\langle M' \rangle, w \notin HALT_{TM}$ (שלב 1 לא יסתיים, או שנגיע לשלב 3).

11.6.2007

הודעות:

ביום שישי תרגול חזרה.

- לא יהיו עוד שיעורי השלמה
- לא תהיה הרצאה בשבוע לפני המבחן (אולי יהיו תרגולי חזרה).

בעיית הריצוף

נתון $T = \{t_0, \dots, t_k\}$, $H, V \subseteq T \times T$ (מסמן איזה בלטות אפשר לשים ליד השניה), $t_0 = 3$ (מי הראשון).

לדוגמא: $T = \{1, 2, 3\}$, $H = \{(2,1), (1,2), (2,3)\}$ (בכיוון האופקי – את מי מותר לשים מעל מי) ו $V = \{(1,2), (3,2), (2,3)\}$ (מסמן את מי אפשר לשים אחד ליד השני בכיוון האנכי).

השאלה: האם לכל n אפשר לרצף בגודל $n \times n$.

נגדיר $TILE = \{\langle T, H, V, t_0 \rangle \mid \forall n \geq 1 \text{ exists } n \times n \text{ legal tile}\}$

נטען: $\overline{HALT_{TM}^\varepsilon} \leq_M TILE$ (ועל כן $TILE \notin RE$)

(נשים לב - RE - $Already\ seen$) $\overline{HALT_{TM}^\varepsilon} = \{\langle M \rangle \mid M \text{ doesn't stop on } \varepsilon\}$

כלומר עלינו להראות כי בהינתן M , ניתן לייצר T, H, V, t_0 כך ש M לא עוצרת על ε אםם

$\langle T, H, V, t_0 \rangle \in TILE$

הרעיון: נמפה בין הסרט של מכונת הטיורינג לבין הלוח.

a (q_1, R) $(q_0, -)$ *	$(q_1, -)$ (q_1, R) — *			
$(q_0, -)$ * — *	— — *	— — *	— — *	

אם $\delta(q_0, -) = (q_1, a, R)$

נשים לב ש $TILE = \{\langle T, H, V, t_0 \rangle \mid \forall n \geq 1 \text{ exists } n \times n \text{ legal tile}\}$

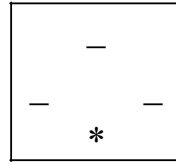
שקול ל – יש ריצוף חוקי לכל רבע המישור (ע"פ הלמה של קניג).

איך נבנה את הלוח:

1. מרצפות השורה הראשונה בלוח:

$(q_0, -)$ * — *

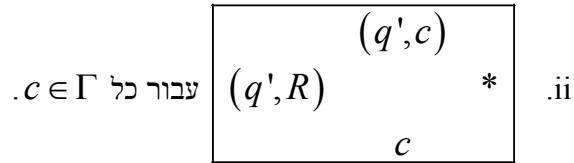
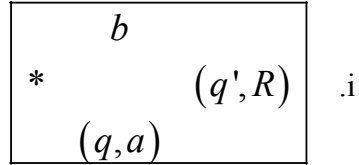
a. השמאלי ביותר -



b. כל השאר -

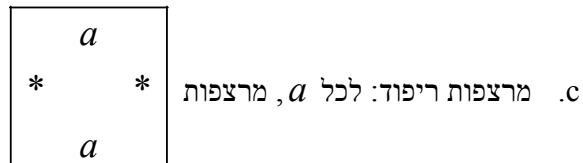
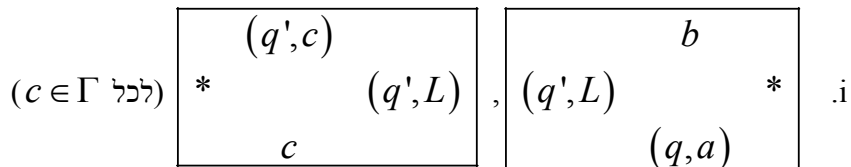
2. מרצפות המעברים:

a. 2R: לכל מעבר $\delta(q, a) = (q', b, R)$ נוסיף מרצפות:



(המעבר תרם $|\Gamma| + 1$ מרצפות חדשות)

b. 2L: לכל מעבר $\delta(q, a) = (q', b, L)$ כך ש $q \notin q_{acc}, q_{rej}$ נוסיף מרצפות:



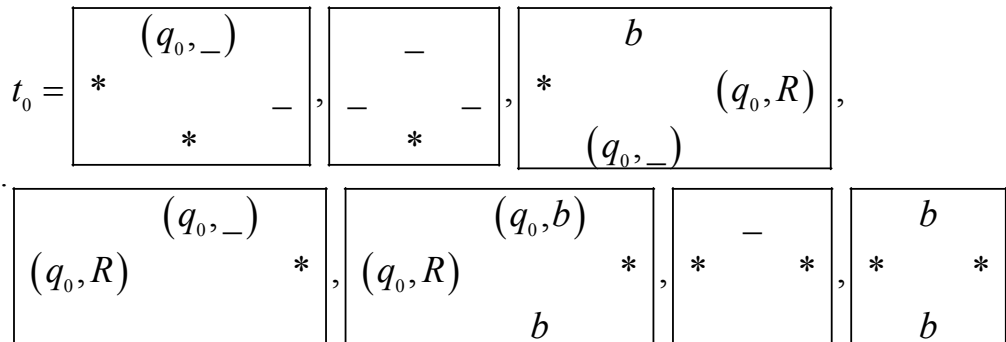
דוגמא:

$$\delta(q_0, _) = (q_0, b, R)$$

$$\delta(q_0, b) = (q_{acc}, b, R)$$

המכונה כותבת b עד אינסוף.

המרצפות:



עוד בעיה בלתי כריעה

Post correspondence problem (PCP)

נתון: אוסף אבני דומינו. כל אבן $\begin{bmatrix} a \\ ab \end{bmatrix}$. פלט: האם ניתן לארגן match? (המילה הרשומה למטה זהה לזאתי למעלה).

לדוגמא: $P = \left\{ \begin{bmatrix} b \\ ca \end{bmatrix}, \begin{bmatrix} a \\ ab \end{bmatrix}, \begin{bmatrix} ca \\ a \end{bmatrix}, \begin{bmatrix} abc \\ c \end{bmatrix} \right\}$ יש Match:

a	b	ca	a	abc
ab	ca	a	ab	c

 $\leftrightarrow$

$abcaaabc$
 $abcaaabc$

(מותרות חזרות).

טענה: $PCP = \{ \langle P \rangle \mid \text{There exists match} \} \in RE \setminus R$

סיפסר הוכיח ש $A_{TM} \leq_M PCP$

תורת הסיבוכיות

איפיון 1: REG, CFL

איפיון 2: כמה משאבים צריך כדי להכריע את השאלה?
לא נלמד (תחילת 7 sipsers) – סיבוכיות אסימפטוטית.

דוגמא: אלגוריתם להכרעת השפה $L = \{0^n 1^n \mid n \geq 0\}$

בהינתן $w \in (0+1)^*$

1. נוודא ש $w \in 0^* 1^*$
2. כל עוד אפשר, נמחק 0 ראשון ו 1 ראשון.
3. נקבל אמ"מ המחקות של ה 0ים וה 1ים הסתיימו בו זמנית.

שלב 1: $O(n)$

שלב 2: כל מחיקה $O(n)$. $\frac{n}{2}$ מחיקות $(O(n^2))$

שלב 3: $O(1)$

בסה"כ $O(n^2)$. אפשר פחות?

1. עם שני סרטים, אפשר ב $O(n)$ (מכונה עם שני סרטים, הרצה בזמן $t(n) \Leftarrow$ מכונה עם

סרט יחיד, הרצה בזמן $(t^2(n))$).

2. ב $O(n \log n)$ (הסכמה על זוגיות, חציה ב 2, חזור).

לא ניתן $o(n \log n)$. ידוע L ניתנת להכרעה ב $o(n \log n) \Leftarrow L$ רגולרית.

נגדיר $TIME(t(n)) = \{L : * \mid L \text{ ניתנת להכרעה ע"י מ"ט דטרמניסטית בעלת סרט יחיד הרצה בזמן } O(t(n))\}$.

לדוגמא: $\{0^n 1^n \mid n \geq 0\} \in TIME(n \log n)$

$NTIME(t(n)) = \{L : * \mid L \text{ ניתנת להכרעה ע"י מ"ט אי דטרמניסטית בעלת סרט יחיד הרצה בזמן } O(t(n))\}$.

נגדיר מכונת טיורינג אי דטרמניסטית. במכונה אי דטרמניסטית $\delta: Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \Delta}$. נאמר שמכונת טיורינג אי דטרמניסטית מכריעה שפה אם כל החישובים על מילים ב Σ^* עוצרים, וזמן הריצה של המכונה מוגדר כ: מספר הצעדים המקסימלי בחישוב כלשהו.

נשים לב, שריצה של מכונה אי דטרמניסטית היא עץ של קונפיגורציות.

טענה: לכל מ"ט אי דטרמניסטית הרצה בזמן $t(n)$ קיימת מכונת טיורינג דטרמניסטית שקולה הרצה בזמן $2^{t(n)}$.

המחלקות P ו NP

$P = \bigcup_{k \in \mathbb{N}} TIME(n^k)$ - כלומר, קבוצת השפות הכריעות בזמן פולינומיאלי על ידי מכונת טיורינג דטרמיניסטית.

$NP = \bigcup_{k \in \mathbb{N}} NTIME(n^k)$ - כלומר, קבוצת השפות הכריעות בזמן פולינומיאלי על ידי מכונת טיורינג אי דטרמיניסטית.

השאלה המעניינת היא $P = ? NP$

18.6.2007

ראינו $P = \bigcup_k TIME(n^k)$ (קל להכריע)

ו $NP = \bigcup_k NTIME(n^k)$ - קל לאמת.

ידוע כי $P \subseteq NP$. לא יודעים אם $NP \subseteq P$.

נדבר על בעיות NP שלמות – NPC.

נאמר כי $L \in NP - C \Leftrightarrow$ אם נמצא פתרון פולינומיאלי ל L , אזי נוכיח כי $P = NP$.

משפט: $3SAT \in P$ אז $P = NP$.

מה זה $3SAT$? בעיית הספיקות: נתונה נוסחה $\psi = (x \wedge \bar{y}) \vee (\bar{x} \wedge z)$ - רוצים לדעת אם היא ספיקה – השמה מספקת (עד קצר) היא $x=1, y=0, z=0$.

הגדרות:

- משתנה בוליאני: מקבל ערכים $\{1,0\}$
- פעולות בוליאניות - $\neg, \vee, \wedge, \text{and}$
- נוסחה בוליאנית מתקבלת ממשתנים בוליאניים ע"י הפעלת פעולות בוליאניות. לדוגמא
 $B \rightarrow 0 \mid 1 \mid Var \mid B \vee B \mid \neg B \mid B \wedge B$
 $Var \rightarrow x, y, z$

השמת אמת - $f: Var \rightarrow \{0,1\}$

בעיית הספיקות: $SAT = \{\langle \varphi \rangle \mid \varphi \text{ sfika}\}$ (ספיקה = sfika) – כלומר, יש השמה ששווה לאמת.

רדוקציה פולינומיאלית

ראינו $A \leq_m B$ - $\Leftrightarrow$ קיימת פונקציה ניתנת לחישוב $f: \Sigma^* \rightarrow \Sigma^*$ כך שלכל $w \in \Sigma^*$ מתקיים $w \in A \Leftrightarrow f(w) \in B$.

$f: \Sigma^* \rightarrow \Sigma^*$ ניתנת לחישוב בזמן פולינומיאלי אם יש מ"ט דטרמיניסטית M העוצרת בזמן פולינומיאלי ומחשבת f ($t: \mathbb{N} \rightarrow \mathbb{N}$, $t \geq 1$ פולינום, ולכל w , M עוצרת אחרי $t(|w|)$ צעדים).

הגדרה: $A \leq_p B$ - $\Leftrightarrow$ קיימת פונקציה ניתנת לחישוב **בזמן פולינומיאלי** $f: \Sigma^* \rightarrow \Sigma^*$ כך שלכל $w \in \Sigma^*$ מתקיים $w \in A \Leftrightarrow f(w) \in B$.

משפט: אם $A \leq_p B$ ו $B \in P$ אזי $A \in P$.

הוכחה: עלינו להראות מ"ט הרצה בזמן פולינומיאלי ומכריעה את A . על קלט w מחשבת את $f(w)$ ומריצה את המכונה של B על $f(w)$.

3CNF: צורה נורמלית לנוסחאות בוליאניות. לדוגמא:

$$(x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee x_4) \wedge (\dots)$$

ליטרל – משתנה או שלילתו (x_i או $\bar{x}_i$)

פסוקית - $\vee$ על מספר ליטרלים.

נוסחא ב CNF (conjunctive normal form) – היא $\wedge$ (and) של פסוקיות.

על שום מה ה-3? כי בכל פסוקית יש 3 ליטרלים.

$$3SAT = \{ \langle \varphi \rangle \mid \varphi \text{ is written at 3CNF and sfika} \}$$

ידוע: $2CNF \in P$

הגדרה: שפה L היא NP-complete (NP שלמה) אם:

1. $L \in NP$ (חסם עליון)

2. L היא $NP-hard$ (NP קשה) (חסם תחתון).

a. מה זה NP-hard : לכל בעיה $L' \in NP$ מתקיים $L' \leq_p L$.

משפט: $NP-complete$ $L \in NP$ ו $L \in P$ אז $N = NP$?

הוכחה: תהי L' בעיה ב-NP. מכיוון ש $L \leq_p L'$ (נכון כי L היא $NP-hard$ ובפרט $NP-hard$)

ו $L \in P$, אזי $L' \in P$.

תזכורת מהתרגול:

בעיית הריצוף החסום

קלט: קבוצה סופית T של אריחים $H, V \subseteq T \times T$ תנאי שכנות במאוזן ובמאונך.

יש לנו גם את $t_{init}, t_{fin} \in T$ צריך להיות הראשון, t_{fin} צריך להיות בשורה ה- n .

n נתון באונרית.

$$BT = \{ \langle T, V, H, t_{init}, t_{fin}, n \rangle \mid \text{exists legal } n \times n \text{ tile} \}$$

הבהרה:

חשוב להבין:

for $i = 1$ to n
print *

הוא אקספוננציאלי אם n נתון בבסיס 10!!!!!! למה? אם הקלט הוא 9999, הקלט הוא באורך

4, וזמן ריצה הוא 9999.

לעומת זאת, אם n נתון באונרית, זה יהיה כמובן לינארי.

מדוע $BT \in NP$? גודל העד n^2 ו n נתון באונרית $\Leftarrow$ גודל העד פולינומיאלי בקלט, ניתן לבדיקה בזמן פולינומיאלי.

בתרגול הראנו כי: $BT \in NP-hard$. בהנתן מ"מ M א"ד עם סיבוכיות זמן פולינומיאלי $p(n)$

ומילה $w \in \Sigma^*$ נייצר $\langle T, H, V, t_{init}, t_{fin} \rangle$ כך ש $\langle T, H, V, t_{init}, t_{fin}, p(n) \rangle \in BT \Leftrightarrow M \Leftrightarrow w$. מקבלת את w .

בהנתן $\langle T, H, V, t_{init}, t_{fin}, n \rangle$ אפשר לייצר בזמן פולינומיאלי נוסחת φ ב $3CNF$ כך

$$\langle T, H, V, t_{init}, t_{fin}, n \rangle \in BT \Leftrightarrow \varphi \text{ ספיקה.}$$

משתני הנוסחה: לכל $i, j \in \{1, \dots, n\}$ ואריח $t \in T$ יהיה משתנה $x_{i,j,t}$ כך ש $f(x_{i,j,t}) = 1$ שקול לכך שבהשמה המושרית מ f האריח t נמצא בקוארדינטה i, j .

שאלה: האם $x_{2,2,5} = 1$ וגם $x_{2,2,1} = 1$? לא, כי לא יכולים להיות שני אריחים באותו מיקום.

האם יתכן $x_{5,2,5} = 1$ וגם $x_{2,2,5} = 1$? כן, כי אותו אריח יכול להיות בכמה

קוארדינטות.

f צריכה לקיים:

- א. לכל מיקום מתאים לפחות אריח אחד. $\varphi_{ij} = \bigvee_{t \in T} x_{i,j,t}$
- ב. לכל מיקום מתאים רק אריח אחד $\varphi'_{i,j} = \bigwedge_{t \in T} \left(x_{i,j,t} \rightarrow \bigwedge_{t' \neq t} \overline{x_{i,j,t'}} \right)$
- ג. האריח ההתחלתי והסופי מונחים במקום $\varphi_{border} = x_{n,1,t_{fin}} \wedge x_{1,1,t_{init}}$
- ד. מתקיימים תנאי שכנות במאוזן $\theta_{i,j} = \bigvee_{(t,t') \in H} x_{i,j,t} \wedge x_{i+1,j,t'}$
- ה. מתקיימים תנאי שכנות במאונך $\theta'_{i,j} = \bigvee_{(t,t') \in V} x_{i,j,t} \wedge x_{i,j+1,t'}$

$$\varphi = \bigwedge_{i,j \in \{1, \dots, n\}} \varphi_{i,j} \wedge \varphi'_{i,j} \wedge \varphi_{border} \wedge \left(\bigwedge_{\substack{1 \leq i \leq n \\ 1 \leq j \leq n}} \theta_{i,j} \right) \wedge \left(\bigwedge_{\substack{1 \leq i \leq n \\ 1 \leq j \leq n}} \theta'_{i,j} \right)$$

ולכן הנוסחה תהיה

טענה: φ (לעיל) ספיקה אמ"מ $\langle T, V, H, t_{init}, t_{fin}, n \rangle \in BT$

הוכחה: $\Leftarrow$ תהי t השמה מספקת – מתנאים א,ב f משרה ריצוף, ומתנאים ג-ה הריצוף חוקי.

$\Rightarrow$ יהי g ריצוף $(g : \{1, \dots, n\} \times \{1, \dots, n\} \rightarrow T)$, אזי ההשמה $f : Var \rightarrow \{0, 1\}$ המושרית מ g מקיימת את φ .

כן נראה ('נקבל אינטואיציה') למה מעבר מ φ ל $3CNF$ הוא פולינומיאלי –

- א. עוברים ל CNF כללי (ללא מגבלה של 3)
- ב. עוברים מ CNF ל $3CNF$ – נניח וקיבלנו $x_1 \vee x_2 \vee x_3 \vee x_4$ – אזי נהפוך אותו ל $(x_1 \vee x_2 \vee z) \wedge (\overline{z} \vee x_3 \vee x_4)$. נשים לב שאין שקילות – אבל נקבע את z כך שיהיה את אותו ערך – כלומר, אם יש השמה מספקת במקור, תהיה השמה מספקת ביעד. עבור פסוקית עם m ליטרלים, מוסיפים $m - 3$ משתנים חדשים. לדוגמא –

$$a_1 \vee a_2 \vee \dots \vee a_m \Rightarrow (a_1 \vee a_2 \vee z_1) \wedge (\overline{z_1} \vee a_3 \vee z_2) \wedge (\overline{z_2} \vee a_4 \vee z_3) \wedge \dots \wedge (\overline{z_{m-3}} \vee a_{m-1} \vee a_m)$$

דוגמא עבור $n = 2$.

$$\left\{ \{1, 2\}, H = (1, 2), (2, 1), 1, 2, 11 \right\}$$

נגדיר $x_{1,1,a}, x_{1,1,b}, x_{1,2,a}, x_{1,2,b}, x_{2,1,a}, x_{2,1,b}, x_{2,2,a}, x_{2,2,b}$

הריצוף המתאים הוא

2	1
1	2

 אזי $x_{1,1,a} = x_{1,2,b} = x_{2,1,b} = x_{2,2,a} = 1$ וכל השאר יהיו שווים ל0.

תהיה לנו השמה מספקת (אנחנו מאמינים להוכחה).

משפט: אם $L \in NP - C$ ו $L \leq_p L'$ אז $L' \in NP - hard$.

הוכחה: צריך להראות שלכל $L'' \in NP$ מתקיים $L'' \leq_p L'$.

תהי $L'' \in NP$, ידוע $L'' \leq_p L$, ידוע גם $L \leq_p L'$. מטרגניזיביות של $\leq_p$ נובע ש $L'' \leq_p L'$.

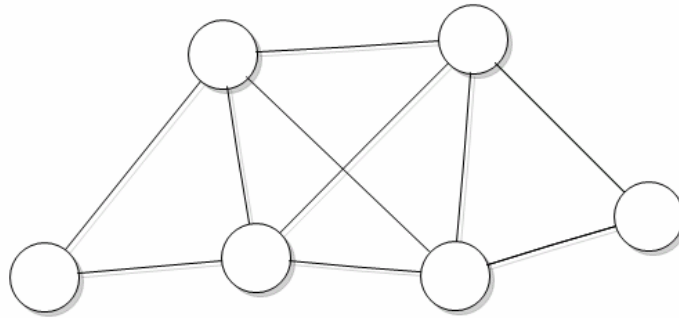
עוד בעיה NP-complete

$CLIQUE = \{ \langle G, k \rangle \mid * \}$ - G גרף לא מכוון, המכיל קליק בגודל k .

קליק $C \subseteq V$: קבוצה של קדקדים כך שיש קשת בין כל שני קדקדים בקבוצה.

k - קליק : קליק בגודל k .

דוגמא:



כאן יש 4-קליק.

נראה כי:

א. $CLIQUE \in NP$: המכונה הא"ד¹ תנחש k -קליק ותבדוק אותו.

ב. כעת נראה כי $CLIQUE \in NP-hard$. נראה $3SAT \leq_p CLIQUE$. בהנתן נוסחא

φ ב-3CNF, נבנה $\langle G, k \rangle$ כך ש φ ספיקה אם"מ יש בג k קליק.

לא לדאוג, לא נראה את ב' עכשיו (נותרו 2 דקות), זה יהיה ביום ד'.

¹ אי דטרמיניסטית.

25.6.2007

השבוע: רגיל

שבוע הבא: יש רק הרצאה ביום שני **ואין תרגולים**. יש תרגיל בית לא להגשה + פתורנו.
השבוע הבא: יש רק תרגולים – תרגולי חזרה.

סיבוכיות זכרון – Space complexity

הגדרה: בהינתן מ"ט M , העוצרת על כל קלט. סיבוכיות הזכרון של M היא פונקציה $s : \mathbb{N} \rightarrow \mathbb{N}$ כך ש $s(n)$ חסם על מספר התאים בסרט, בהם M משתמשת בריצה על קלט באורך n .

נגדיר :

$$SPACE(s(n)) = \left\{ L \mid L \text{ is decidable on deterministic Turing Machine with } s(n) \text{ Memory} \right\}$$

עבור מכונה דטרמניסטית:

1. מה עדיף, זמן $f(n)$ או שטח $f(n)$?

עדיף שטח $f(n)$, כי נוכיח $TIME(f(n)) \leq SPACE(f(n))$.

למה זה: כי בכל צעד ניתן לדלג על תא אחד בלבד.

$$SPACE(f(n)) \leq TIME(f(n) \cdot 2^{O(f(n))}) \quad 2.$$

a. למה זה נכון?

b. נשאל כמה קונפיגורציות שונות יש למ"ט הרצה בשטח $f(n)$?

$$c. \text{ נענה כי } |Q| \cdot \underbrace{f(n)}_{\substack{\text{Head} \\ \text{Position}}} \cdot \underbrace{|\Gamma|^{f(n)}}_{\substack{\text{Tape} \\ \text{Contents}}}$$

ניזכור כי חישוב עוצר אינו חוזר על אותה קונפיגורציה פעמיים (המכונה דטרמניסטית, חזרה $\Leftarrow$ לופ אינסופי).

דוגמא: SAT ניתנת לפיתרון בזכרון לינארי. בהינתן קלט $\langle \varphi \rangle$ עבור נוסחא בוליאנית φ .

1. לכל השמות אמת f למשתנים $x_1, \dots, x_m$ נשערך את φ ביחס ל f .

2. אם השתערך ל true, נקבל. אחרת, נעבור להשמה הבאה. בהשמה האחרונה, נדחה.

זכרון דרוש:

▪ השמת האמת הנוכחית (m תאים)

▪ שערוך הנוסחה לפי ההשמה הנוכחית ($O(|\varphi|)$ תאים)

נשים לב – קיבלנו כי $SAT \in NP-complete$ אבל גם $SAT \in \text{Linear space}$.

דוגמא:

הגדרה:

אוטומט A הוא ריק אם $L(A) = \emptyset$.

אוטומט A הוא אוניברסלי אם $L(A) = \Sigma^*$

בהינתן A דטרמניסטי, הכרע האם $L(A) = \Sigma^*$

$$ALL_{DFA} = \{ \langle A \rangle \mid L(A) = \Sigma^* \} \text{ - ניתנת להכרעה בזמן פולינומיאלי (PTIME).}$$

(איך נבדוק? במקום לבדוק אוניברסליות, נבדוק אם המשלים לא מקבל אף מילה – השלמה זה קל, ובדיקת ריקנות זה קל).

$$ALL_{NFA} = \{ \langle A \rangle \mid L(A) = \Sigma^* \wedge A \text{ is NFA} \}$$

כעת, נשאל את אותה שאלה על

$$\overline{ALL_{NFA}} = \{ \langle A \rangle \mid L(A) \neq \Sigma^* \wedge A \text{ is NFA} \}$$

נטען כי $ALL_{NFA} \in PSPACE$ - כלומר ניתנת להכרעה בשטח פולינומיאלי.

נשים לב – לא יודעים ש $ALL_{NFA} \in NP$ או $ALL_{NFA} \in co-NP$ (יש עד, אבל לא מובטח שהוא פולינומיאלי).

טענה 1: $L(A) \neq \Sigma^* \Leftrightarrow$ קיימת מילה w באורך $2^{|Q|}$ לכל היותר כך ש $w \notin L(A)$.

הוכחה: $\Rightarrow$ - ברור.

$\Leftarrow$ אם $L(A) \neq \Sigma^* \Leftarrow L(\bar{A}) \neq \emptyset$ ומכיוון שב $\bar{A}$ יש $2^{|Q|}$ מצבים, אזי $L(\bar{A}) \neq \emptyset$ אם יש מילה w באורך $2^{|Q|}$ כך ש $w \in L(\bar{A})$.

דוגמא להדיקות החסם: $L = \{ w \mid |w| = 0 \bmod 2, 0 \bmod 3, \dots, 0 \bmod p_i \}$ כלומר - $|w| = 0 \bmod 2 \cdot 3 \cdot \dots \cdot p_i$.
 $p_i = O(i \log i)$

טענה 2: קיימת מילה w באורך $2^{|Q|}$ כך ש $w \notin L(A) \Leftrightarrow$ קיימת סדרה של קבוצות של מצבים $s_0, \dots, s_{2^{|Q|}-1}$ שמתאימה לריצה של האוטומט הדטרמיניסטי המתקבל מ A ע"י הפעלת subset construction ויש i כך ש $S_i \cap F = \emptyset$ (הריצה של $\det(A)$ (הגרסה הדטרמיניסטית של A) על $\sigma_0, \dots, \sigma_i$ מגיעה למצב לא מקבל).

נראה מ"ט א"ד עבור $\overline{ALL_{NFA}}$ על קלט $\langle A \rangle$ המקודד NFA:

1. כתוב על הסרט את המצבים ב Q_0 .
2. אתחל מונה ל 0.
3. כל עוד המונה קטן מ $2^{|Q|}$ (Q ביטים)
 - a. אם הקבוצה S על הסרט מקיימת $S \cap F = \emptyset$, קבל (את $\langle A \rangle$) – כי מצאנו מילה שנדחית.
 - b. אחרת, נחש אות a , עדכן את הקבוצה על הסרט ל $\delta(s, a)$ והגדל מונה ב 1.
4. דחה.

סיבוכיות הזכרון:

- קבוצת מצבים - $O(|Q|)$
- מונה ($O(|Q|)$)

$$s_0 = Q_0, s_1 = \delta(s_0, \sigma_0), \dots, s_i = \delta(s_{i-1}, \sigma_{i-1}), \dots, s_i = \delta(s_{i-1}, \sigma_{i-1})$$

1. פורמלית - $s_0 = Q_0$, ולכל $i \geq 0$ קיים אות σ_i כך ש $s_{i+1} = \delta(s_i, \sigma_i)$

▪ מצביעים עבור העדכונים - $O(1)$

סה"כ $O(|Q|)$.

יש את ההוכחה הזו בסיפסר עם המון טעויות. לשים לב.

משפט Savitch

לכל $s: \mathbb{N} \rightarrow \mathbb{N}$ כך ש $s(n) \geq n$ מתקיים $(n = o(s(n)))$

$$NSPACE(s(n)) \subseteq SPACE(s^2(n))$$

(מה זה $NSPACE$ - בדיוק מה שאנו מצפים).

$$NSPACE(s(n)) = \left\{ L \mid L \text{ is decidable on ideterministic Turing Machine with } s(n) \text{ Memory} \right\}$$

הוכחה: בהינתן מ"ט א"ד עם סיבוכיות זיכרון $s(n)$ נבנה מ"ט דטרמיניסטית שקולה עם סיבוכיות זכרון

$$s^2(n)$$

הנחות על M :

1. יש קונפיגורציה מקבלת יחידה c_{acc}

2. יש קונפיגורציה התחלתית יחידה c_{init} .

3. יהי d קבוע כזה כשאין ל M יותר מ $2^{d \cdot s(n)}$ קונפיגורציות שונות.

$$a. \text{ למה יש קבוע כזה? ניזכר קיים } d \text{ כך ש } \underbrace{|Q|}_{\text{Which State}} \cdot \underbrace{s(n)}_{\text{Head Position}} \cdot \underbrace{|\Gamma|^{s(n)}}_{\text{Tape Contents}} \leq 2^{d \cdot s(n)}$$

נבנה שגרה דטרמיניסטית $reach(c_1, c_2, t)$ המכריעה האם ניתן לעבור מקונפיגורציה c_1 לקונפיגורציה c_2 ב t צעדים לכל היותר.

סיבוכיות הזמן של $reach$ תהיה $O(\log t + s(n)) \cdot \log t$.

המכונה M' תריץ את $reach(c_{init}, c_{acc}, 2^{d \cdot s(n)})$.

נשים לב:

1. לכל w , M' מקבלת את w אם M מקבלת את w .

2. סיבוכיות הזכרון של M' - $O(s^2(n)) = O(\log 2^{d \cdot s(n)} + s(n)) \cdot \log 2^{d \cdot s(n)}$

כעת, נותר להראות את השגרה $reach$:

הרעיון: נעבור על כל הקונפיגורציות c של M , ונבדוק האם $reach(c_1, c, \left\lceil \frac{t}{2} \right\rceil)$ וגם

$$reach(c, c_2, \left\lceil \frac{t}{2} \right\rceil) \text{ (למצוא קונפיגורציות באמצע).}$$

תיאור השגרה:

1. אם $t = 1$, בדוק האם $c_1 = c_2$ או שיש מעבר של צעד אחד מ c_1 ל c_2 .

2. אם $t > 1$ אז לכל קונפיגורציה c של M המשתמשת ב $s(n)$ תאים

a. הרץ את $reach\left(c_1, c, \left\lceil \frac{t}{2} \right\rceil\right)$

b. הרץ את $reach\left(c, c_2, \left\lfloor \frac{t}{2} \right\rfloor\right)$.

c. אם שתי הריצות קיבלו, קבל.
3. דחה.

(יש כאן שרטוט של עץ רקורסיה. אני מוותר).
עומק העץ יהיה $\log t = \log 2^{d \cdot s(n)}$.

המחלקה PSPACE

$$PSPACE = \bigcup_k SPACE(n^k)$$

נשים לב – ממשפט Savitch נובע כי $PSPACE = NPSPACE$.
נשים לב שנובע גם כי $co - PSPACE = co - NPSPACE$.
האם $PSPACE = co - PSPACE$? כן.

2.7.2007

ציטוט השיעור: 'כשאתה מושך דברים, הם נופלים'

הודעות חשובות:

- אין תרגולים השבוע
- בשבוע הבא – יש תרגולי חזרה

מבנה המבחן:

- יש לענות על 5 שאלות מתוך 6
- 2 אוטומטים
- 2 חישוביות
- 2 סיבוכיות

מחלקות Log Space ו NLogSpace

- נתייחס לסיבוכיות מקום
- מותר להשתמש בזכרון נוסף שהוא לוגריתמי בגודל הקלט (מלבד סרט הקלט)

נשים לב שכשדיברנו על סיבוכיות זמן לא היה טעם להתייחס לזמן תת לינארי (כי לפחות קוראים את הקלט).

מודל החישוב: יש סרט קלט, שגודלו הוא n , ויש סרט עבודה, שגודלו $o(\log n)$

דוגמא: $EQ = \{0^n 1^n \mid n \geq 0\}$. ראינו אלגוריתם שרץ בזמן $o(n \log n)$ (שרץ, מוחק 1, אז מוחק 0, וחוזר חלילה), אבל בסיבוכיות מקום $o(n)$ (כי העתקנו את הקלט).

פתרון לוגריתמי: נזכור שליצוג הערך n דרושים $\lceil \log_2 n \rceil$ ביטים. לכן, האלגוריתם סופר את מספר ה-0ים והאחדות ומשווה את המונים.
בסרט העבודה נרשום את מס' האפסים, אח"כ סימן ההפרדה ואח"כ מס' האחדים. סה"כ נשתמש ב $O(\log n)$ ביטים בסרט העבודה.

הגדרות:

המחלקה LOGSPACE (לפעמים מסומנת גם כ-L): כל השפות שיש מ"ט דטרמניסטית שמכריעה אותן תוך שימוש בזכרון נוסף $O(\log n)$ (אין חשיבות לזמן אבל ב-P).

המחלקה NLOGSPACE: כנ"ל, רק עם מכונה א"ד. מסומנת NL.

דוגמא 2: $PATH: \left\{ \langle G, s, t \rangle \mid G \text{ is graph, } s, t \text{ are vertices and there exists a path from } s \text{ to } t \right\}$

תשומת לב: אנחנו רושמים את כל מספרי הצמתים – כלומר, אם יש לי 100 צמתים, אני צריך לרשום, 1, 2, ..., 100, ולא רק 100. ככה מוודאים שגודל הקלט הוא $O(n \log n)$.

אבחנות פשוטות:

1. $PATH \in NP$ - כי בהינתן עד (מסלול מ s ל t) אפשר לוודא שאכן מסלול.
2. $PATH \in P$ - בDFS או BFS.

כעת נראה כי $PATH \in NL$.

מה אפשר לשמור ב $O(\log n)$ ביטים?
למשל:

1. אינדקס של צומת
2. באופן כללי – כל מספר קבוע של צמתים או קשתות (למה – האורך של שמירת כל צומת הוא $O(\log)$ של מספר הצמתים).

המכונה תנחש (נזכור שהיא א"ד) מסלול מ s ל t ובכל רגע תשמור רק את הצומת הנוכחי במסלול. בהתחלה תרשום בסרט העבודה את s . בכל שלב תחליף בצומת הנכחי. בנוסף, נחזיק בסרט העבודה מונה שמאותחל ל 0, וגדל ב 1 בכל איטרציה. למה? אם קיים מסלול, קיים מסלול פשוט (בלי חזרות), כלומר $n \geq$ צמתים אם באיזה שלב נכתב $\langle t \rangle$ בסרט העבודה, אז הקלט ב-PATH. אם המונה הגיע ל $|V|$, הקלט לא ב-PATH.

נוודא שמתקיים:

1. שימוש ב $O(\log n)$ מקום: כן! כי שומרים רק צומת אחד ומונה אחד שגדל עד $|V|$ לכל היותר. (למעשה גם t, n , לצורך ההשוואה)
2. קיימת ריצה מקבלת $\Leftrightarrow$ יש מסלול מ s ל t .
 - a. ברור – אם יש ריצה מקבלת, אז יש מסלול, על פי הגדרתו.
 - b. $\Rightarrow$ קיימים ניחושים שמייצגים ריצה מקבלת.

NL-Completeness

הגדרה: A שלמה ב NL אם מתקיים:

1. $A \in NL$
2. $A \in NL - hard$ - לכל שפה $A' \in NL$ קיימת רדוקציה $A' \leq_L A$

אבל איזה רדוקציה נדרוש?

L - זה אינה רדוקציה פולינומית בזמן רגילה אלא רדוקציית L-שמשמשת ב LOGSPACE.

Log Space Transducer: משרן במקום לוגריתמי. מ"ט דטרמיניסטי בעלת שלושה סרטים WriteOnly, ReadWrite, ReadOnly. בסרט RW יש $O(\log n)$ תאים. נאמר שהמכונה מחשבת פונקציה $f: \Sigma^* \rightarrow \Sigma^*$ אם לכל מילה $w \in \Sigma^*$ הכתובה בסרט ה RO M עוצרת עם פלט $f(w)$ בסרט WO.

דוגמא:

קלט: גרף עם משקולות $\langle V, E, w \rangle$, $w: E \rightarrow \mathbb{N}$

פלט: גרף לא ממשוקל $\langle V, E' \rangle$ כך ש $e \in E' \Leftrightarrow e \in E, w(e) > 7$

המשרן משתמש בשלושה סרטים – קלט (ארוך), עבודה (קצר), ופלט – ארוך. תחילה נעתיק את הצמתים לסרט הפלט, אח"כ נעבור על הקשתות. כדי להחליט אם להעתיק קשת לסרט הפלט, נכתוב את המשקל שלה על סרט העבודה ונשווה ל 7.

הערה: משקלים בעלי 4 ביטים או יותר לא יועתקו לסרט העבודה בשלמותם – כבר בביט הרביעי ניתן להסיק ש $w(e) > 7$ ולהעתיק את הקשת לסרט הפלט.

הגדרה: $A' \leq_L A$ אם קיים משרן העובד במקום לוגריתמי ומחשבת פונקציה f כך $w \in A' \Leftrightarrow f(w) \in A$.

משפט: אם $A \leq_L B$ ו $B \in L$ אז $A \in L$.

הוכחה: בהינתן M_B עבור B, נבנה M_A עבור A.

ניסיון 1: על קלט w המכונה M_A תריץ המשרן המובטח לחישוב $f(w)$ ותריץ את M_B על $f(w)$.

מה לא בסדר? עכשיו $f(w)$ נחשב חלק מהחישוב, ואינו הפלט. המשרן משתמש ב $o(\log|w|)$ מקום כדי לייצר את $f(w)$ אבל עכשיו נצטרך **כשלב ביניים** לכתוב את $f(w)$.

ניסיון 2 (מוצלח): על קלט w , המכונה תחשב בכל איטרציה את האות מתוך $f(w)$ שאותה M_B צריכה כרגע. אף פעם לא נרשום את כל $f(w)$.

נשים לב שייטכן ונריץ את המשרן המון פעמים (אפילו עבור אותו ביט), ושיהיו ביטים שנחשב מספר פעמים, אבל מותר.

מסקנה: המכונה מכריעה את A תוך שימוש ב $O(\log n)$ מקום.

שפה NL-Complete

$$PATH: \left\{ \langle G, s, t \rangle \mid G \text{ is graph, } s, t \text{ are vertices and} \right. \\ \left. \text{there exists a path from } s \text{ to } t \right\} \text{ השפה}$$

$PATH \in NL$ - ראה לעיל.

נראה $PATH \in NL - HARD$ ע"י שנראה רדוקציות $\log$ Space מכל שפה ב NL .
תהי M_A מ"ט (יתכן א"ד) המכריעה את A בשטח $O(\log n)$. בהנתן קלט w ל M_A נייצר גרף $\langle G, s, t \rangle \in PATH \Leftrightarrow$ כך ש M_A מקבלת את w .

צמתי G יהיו הקונפיגורציות של M_A בריצה על w . s תהיה הקונפיגורציה התחלתית, t המקבלת (נניח שיחידות).

איך נראית קונפיגורציה: מצב, מיקום ראש קורא, תכולת הסרט.

כמה קונפיגורציות יש? $\underbrace{|\Gamma|^{O(\log n)}}_{\text{RW tape contents}} \cdot \underbrace{n}_{\text{reading head position}} \cdot \underbrace{|Q|}_{\text{number of states}}$. לייצוג קונפיגורציה נדרש

$$\log \left(\underbrace{|Q|}_{\text{number of states}} \cdot \underbrace{n}_{\text{reading head position}} \cdot \underbrace{|\Gamma|^{O(\log n)}}_{\text{RW tape contents}} \right) \text{ תאים, כלומר } \log(c_1 \cdot n \cdot c_2^{O(\log n)}) \text{ תאים. כלומר}$$

$$\log(c_1 \cdot n) + \log(c_2^{O(\log n)}) = O(\log n)$$

המשרן: עובר בסדר לקסיגורפי על כל המילים באורך $c \log n$ ומעתיק לסרט הפלט שלו את כל המילים שמייצגות קונפיגורציה (אלו הצמתים של G).

כעת, נעבור על כל זוגות המילים האלו, ונבדוק האם הזוג מייצג שתי קונפיגורציות עוקבות. אם כן, מעתיק לסרט הפלט.

בהמשך מזהה את הקונפיגורציות ההתחלתית והמקבלת. ומעתיק גם אותן.

לסיכום: לכל $A \in NL$, $A \leq_L PATH$, לכן $PATH \in NL - COMPLETE$.

משפט: $NL \subseteq P$

הוכחה: ידוע כי $L \subseteq P$ (שיעורים קודמים). בהינתן בעיה ב NL , נוריד אותה ל $LOGSPACE$. על $PATH$ ידוע שהיא ב P , וכי הרדוקציה שראינו קודם היא פולינומית.

$$L \subseteq_{\substack{\text{don't} \\ \text{know if} =}} NL \subseteq_{\text{don't know if} =} P \subseteq \text{מה אנחנו יודעים עד כה?}$$