הקדמה והגדרות

הגדרות שראינו כבר באוטומטים ובאלגו 2. לא נרחיב יותר מדי.

- מודל חישובי – מכונה תיאורטית עם מרכיבים מסוימים: גודל זיכרון, סרט קלט, א"ב.
- כשנדבר על מודלים חישוביים, נדבר על **אלגוריתם כללי** לפתרון הבעיה.
- בעיות: בעיות הכרעה (האם תנאי מתקיים, לדוגמה האם גרף נתון הוא המילטוני) מול בעיות חיפוש (מציאת מעגל המילטוני בגרף נתון).
- קושי הבעיה – מבחינתנו, בעיה תיחשב קלה אם **קיים** אלגוריתם פולינומי שפותר אותה, וקשה אם **כל אלגוריתם** שפותר את הבעיה הוא מעריכי.
- יש בעיות שיש להן אלגוריתם הסתברותי יעיל, אבל האלגוריתם הדטרמיניסטי לא יעיל.
- בהמשך נתעניין גם בסיבוכיות **מקום** (ולא רק זמן).

מודלים שונים

בעיה שלא פתירה במודל אחד, אולי פתירה במודל אחר. דוגמה 1: השפה $L_1 \coloneqq \{0^n1^n : n \in \mathbb{N}\}$ היא לא רגולרית (למת הניפוח, אוטומטים). השפה לא מתקבלת ע"י אף אוטומט *GNFA* (אסל"ד עם מסעי אפסילון).

אם היינו יכולים לעבור על הקלט באיזה סדר שנרצה, יכולנו לפתור את הבעיה. היינו סופרים את האפסים ואחדות לפי הסדר, ובודקים שלכל 0 יש 1 מתאים.

מוצאים את ה-0 הראשון, מסמנים במקומו x. מוצאים את ה-1 הראשון, מסמנים במקומו y. חוזרים להתחלה. מוצאים את ה-0 הבא... וכו.

לדוגמה:

$$\begin{gathered}
000\ldots000111\ldots111 \\
x00\ldots000111\ldots111 \\
x00\ldots000y11\ldots111 \\
\vdots \\
xxx\ldots xx0yyy\ldots yy1 \\
xxx\ldots xxxyyy\ldots yy1 \\
xxx\ldots xxxyyy\ldots yyy
\end{gathered}$$

אם מחקנו 0 ואין יותר אחדות, או שמחקנו את ה-1 האחרון ויש עוד אפסים, אז $w \notin L_1$.

מכונת טיורינג דטרמיניסטית – *DTM*

- א"ב הקלט: א"ב סופי Σ.
- א"ב הסרט: $\Gamma \subseteq \Sigma \cup \{start, blank\}$. כלומר חלק כלשהו מא"ב הקלט, עם סימנים להתחלה וסימון למקום ריק בסרט.
- סרט הקלט – סרט אינסופי מימין.
- ראש כתיבה – קריאה. בכל צעד הוא יכול לקרוא תו אחד ולכתוב תו אחד.
- קבוצת מצבים סופית Q:
  - מצב התחלתי q_S,
  - מצב מקבל (*accepting*) q_Y,
  - מצב דוחה (*rejecting*) q_N.
- פונקציית מעברים סופית:
  - $\delta: Q' \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$
  - כאשר $Q' \coloneqq Q \setminus \{q_Y, q_N\}$
  - לכל מצב, לכל תו שקוראים, יש הגדרה יחידה למעבר. עוברים למצב, כותבים תו, זזים.
  - לפעמים במקום $\{L, R, S\}$ נכתוב $\{\leftarrow, \rightarrow, \circ\}$

נבנה *DTM* עבור דוגמה 1

ניזכר מה אנחנו רוצים שיקרה. כל פעם:

1. למצוא את ה-0 הראשון,
2. לסמן אותו ב-x,
3. למצוא את ה-1 הראשון,
4. לסמן אותו ב-y,
5. לחזור להתחלה.

חישוביות (קיץ תשפ"ו) – הרצאה 1

אם אנחנו מחפשים 0 והתו הבא הוא y, זה אומר שמחקנו את כל האפסים. אם לא נמצא עוד 1, זה אומר שיש יותר אפסים מאחדות.

אם מחקנו 1 והתו הבא הוא $blank$, אנחנו מצפים שהחלפנו את כל האפסים והאחדות ב- x,y. נחזור אחורה ונוודא את זה. אם יש עדיין אפסים, זה אומר שיש יותר אפסים מאחדות.

נגדיר מצבים:

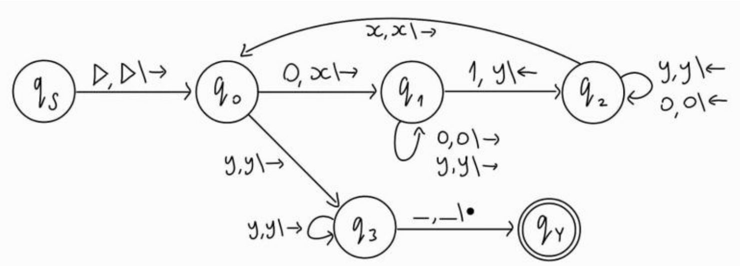
q_0 הוא המצב "מחפשים את ה-0 הבא". q_1 – מחפשים את ה-1 הבא. q_2 – מצאנו 1, כתבנו במקומו y, חוזרים להתחלה. q_3 – הגענו לסוף, נעבור עד ההתחלה ומצפים לא לראות 0 או 1.

- $(q_S, \rhd) \to (q_0, \rhd, R)$. נעבור מהמצב ההתחלתי למצב חיפוש אפסים.
- $(q_0, 0) \to (q_1, x, R)$. מצאנו 0, נכתוב x ונחפש את ה-1 הבא.
- $(q_1, 0) \to (q_1, 0, R),\ \ (q_1, y) \to (q_1, y, R)$. מחפשים 1. כל עוד יש 0 או y, נמשיך לחפש.
- $(q_1, 1) \to (q_2, y, L)$. מצאנו את ה-1 הבא. נכתוב y ונחזור להתחלה.
- $(q_2, y) \to (q_2, yL),\ \ (q_2, 0) \to (q_2, 0, L)$. כל עוד רואים y או 0, נמשיך.
- $(q_2, x) \to (q_0, x, R)$. הגענו ל-x. זה אומר שה-0 הכי שמאלי נמצא מימין.
- $(q_0, y) \to (q_3, y, R)$. חיפשנו 0 ומצאנו y, זה אומר שאין יותר אפסים. נצפה שאין יותר אחדות.
- $(q_3, y) \to (q_3, y, R)$. נמשיך לזוז ימינה.
- $(q_3, blank) \to (q_Y, blank, S)$. מצאנו את הסוף והכל תקין, נעבור למצב מקבל.

כל שילוב אחר של קלט ומצב יעבור למצב $reject$.

- אם יש יותר אפסים מאחדות (או שאין בכלל אחדות), אז תהיה פעם שאנחנו מחפשים את ה-1 הבא (q_1) ונמצא $blank$.
- אם יש יותר אחדות מאפסים, אז תהיה פעם שאנחנו ב- q_3 ונמצא 1.
- אם אין בכלל אפסים, אז ב- q_0 נמצא 1.
- לפי ההגדרה הנוכחית, מחרוזת ריקה לא מתקבלת. אפשר לגרום לכך שהיא תתקבל ע"י הוספת: $(q_0, blank) \to (q_Y, blank, S)$.

ייצוגים שונים של מכונת טיורינג - גרף וטבלה. עבור דוגמה 1:



	0	1	x	y	_
q_0	$(q_1, x, \to)$	$(q_N, 1, \bullet)$	$(q_N, x, \bullet)$	$(q_3, y, \to)$	$(q_N, \_, \bullet)$
q_1	$(q_1, 0, \to)$	$(q_2, y, \leftarrow)$	$(q_N, x, \bullet)$	$(q_1, y, \to)$	$(q_N, \_, \bullet)$
q_2	$(q_2, 0, \leftarrow)$	$(q_N, 1, \bullet)$	$(q_0, x, \to)$	$(q_2, y, \leftarrow)$	$(q_N, \_, \bullet)$
q_3	$(q_N, \_, \bullet)$	$(q_N, 1, \bullet)$	$(q_N, x, \bullet)$	$(q_3, y, \to)$	$(q_Y, \_, \bullet)$

לכל DFA אפשר לבנות מכונת טיורינג מקבילה. נתרגם את המעברים הנתונים למעברים של הפונקציה δ. מכיוון ש-DFA יודע לקרוא קלט רק משמאל לימין, כל התזוזות יהיו לימין. כל קלט של תו לא צפוי עובר ל-q_N. לדוגמה עבור השפה: $L_2 \coloneqq \{s \mid s \in \{0,1\}^*, 001 \subset s\}$:

q_0 זה המצב "מחפשים את ה-0 הראשון של 001". q_1 זה "מחפשים את ה-0 השני". q_2 זה "מחפשים את ה-1".

	0	1
q_0	$(q_1, 0, \to)$	$(q_0, 1, \to)$
q_1	$(q_2, 0, \to)$	$(q_0, 1, \to)$
q_2	$(q_0, 0, \to)$	$(q_Y, 1, \circ)$

ונוסיף גם: $(q_S, \rhd) \to (q_0, \rhd, \to),\ \ (q_2, 1) \to (q_Y, 1, \to)$.

תרגילים

<u>תרגיל 1</u>: השפה $L_1 \coloneqq \{0^n 1^n 2^n \mid n \in \mathbb{N}\}$. לא מתקבלת ע"י אוטומט מחסנית. נבנה מ"ט עבורה:

כמו בדוגמה 1, נייצר מצבים עבור חיפוש ה- 0,1,2 הבאים. מצב של חזרה להתחלה בשביל האיטרציה הבאה. ומצב של מעבר כדי לבדוק תקינות.

q_0, q_1, q_2 – מחפשים את ה-0, 1, או 2 הבא. q_3 – מצאנו את ה-2, חוזרים להתחלה. q_4 – סיימנו, בודקים שלא נשאר 0,1,2.

$$(q_S, \rhd) \to (q_0, \rhd, R), \qquad (q_3, \rhd) \to (q_0, \rhd, R), \qquad (q_4, x) \to (q_4, x, L), \qquad (q_4, \rhd) \to q_Y$$

	0	1	2	x	*Blank*
q_0	q_1, x, R			q_0, x, R	$q_4, blank, L$
q_1	$q_1, 0, R$	q_2, x, R		q_1, x, R	$q_4, blank, L$
q_2		$q_2, 1, R$	q_3, x, L	q_2, x, R	$q_4, blank, L$
q_3	$q_3, 0, L$	$q_3, 1, L$		q_3, x, L	$q_4, blank, L$

כל מצב אחר שולח ל-q_N.

<u>תרגיל 2</u>: בדיקת פלינדרום.

נרצה מצב שמחפש את התו הבא. אם הוא 0, ניכנס למצב שמחפש את התו האחרון ומצפה לראות 0. כנ"ל עבור 1.

- q_0 – מחפשים את התו הבא.
- q_1 – ראינו 0, הולכים לסוף ומצפים ל-0. q_2 – מצאנו את הסוף, נבדוק אם התו הקודם הוא 0.
- q_3 – ראינו 1, הולכים לסוף ומצפים ל-1. q_4 – מצאנו את הסוף, נבדוק אם התו הקודם הוא 1.
- q_5 – חוזרים להתחלה.

אם מחפשים את התו הבא ומצאנו ריק, סיימנו וזה פלינדרום. זה אומר שגם מחרוזת ריקה היא פלינדרום. אם רוצים לא להחשיב מחרוזת ריקה, אפשר להוסיף מצב ספציפי למעבר הראשון שבודק האם זה לא ריק. אם הולכים לסוף ומצפים ל-0 או 1 ומוצאים ריק, זה אומר שהמחרוזת באורך אי-זוגי וסיימנו את התו האחרון. אנחנו מקבל את זה כפלינדרום.

$$(q_S, \rhd) \to (q_0, \rhd, R),$$

	0	*1*	*blank*
q_0	$q_1, blank, R$	$q_3, blank, R$	q_Y
q_1	$q_1, 0, R$	$q_1, 1, R$	$q_2, blank, L$
q_2	$q_5, blank, L$		q_Y
q_3	$q_3, 0, R$	$q_3, 1, R$	$q_4, blank, L$
q_4		$q_5, blank, L$	q_Y
q_5	$q_5, 0, L$	$q_5, 1, L$	$q_0, blank, R$

כל מצב אחר שולח ל-q_N.

אלגוריתמים

תזת *Church-Turing*: *DTM* יכול לחשב כל דבר שאפשר לחשב ע"י מודל מכני. בפועל, הכוונה לכל מודל שמקיים:

- מוגדר ע"י תוכנה סופית יחידה,
- יכול לעבוד על קלט מכל גודל סופי,
- בכל מצב יש הגדרה יחידה למה קורה,
- אם המודל עוצר אחרי מספר סופי של צעדים, התשובה היא תוצאת החישוב.

דוגמאות למודלים שקולים:

- פונקציה רקורסיבית,
- מכונת טיורינג דטרמיניסטית,
- מכונות *RAM – Random Access Machine*
- תוכנות ב-*JAVA*, *C*, *FORTRAN* וכו. (בארכיטקטורה המאפשרת זיכרון אינסופי, בפוטנציאל).

מכונת טיורינג כהגדרה לאלגוריתם

מכונת טיורינג היא עוצמתית:

- מקיימת את תזת *Church-Turing*,
- ממדלת מחשב עם זיכרון אינסופי,
- עובדת על קלט בכל גודל.

אבל גם פשוטה:

- הגדרות פשוטות וברורות – נוח להוכיח דברים.
- קל לתאר מכונת טיורינג אוניברסלית שיכולה להריץ כל מכונת טיורינג אחרת. כלומר, אפשר לבנות מכונה שתקבל מכונה אחרת בתור קלט, ותבצע את הפעולות כאילו היא בעצמה אותה מכונה.
- ממדלת מחשב בעל זיכרון אינסופי, בלי לדרוש הרבה הנחות שאולי לא נוכל לקיים.

קונפיגורציה של *DTM*

תמונה רגעית (*snapshot*) של *DTM*. תוכן הסרט, המצב הנוכחי, מיקום הראש.

נכתוב $C \coloneqq u\ q\ v$, כאשר uv זה תוכן הסרט, הראש נמצא במיקום v_1, והמצב הוא q.

נאמר שקונפיגורציה אחת **מניבה** קונפיגורציה אחרת אם אפשר להגיע מאחת לשנייה בצעד אחד.

קונפיגורציות מיוחדות: התחלה – $q_S\ w$, קבלה – $u\ q_Y\ v$, דחייה – $u\ q_N\ v$.

האם *DTM* תמיד עוצרת?

קל לבנות *DTM* שלא עוצרת – פשוט נגדיר שתמיד זזים ימינה בלי תנאי עצירה. אבל האם קיימת בעיה כך שאי אפשר לבנות עבורה *DTM* שעוצרת?

בעיית ההתאמה של פוסט – *Emil Post Correspondence Problem*. נתון סט של אבני דומינו, לדוגמה:

$$\begin{bmatrix} bc \\ ca \end{bmatrix}, \begin{bmatrix} a \\ ab \end{bmatrix}, \begin{bmatrix} ca \\ a \end{bmatrix}, \begin{bmatrix} abc \\ c \end{bmatrix}$$

המטרה היא לייצר סדרה של האבנים (עם חזרות) כך שהמחרוזות למעלה ולמטה זהות.

מכונת טיורינג נאיבית שעושה את זה, מונה לפי הסדר את כל הסדרות האפשריות, לפי האורך בסדר עולה, עד שהיא תמצא סדרה תקינה.

עבור הסדרה $\begin{bmatrix} acc \\ ba \end{bmatrix}, \begin{bmatrix} ca \\ a \end{bmatrix}, \begin{bmatrix} abc \\ ab \end{bmatrix}$ אין פיתרון, אז המכונה לא תעצור.

עבור מכונה M וקלט x, נרשום:

לבעיות הכרעה: $M(x) = 1$ אם M מקבלת את x, $M(x) = 0$ אם M לא מקבלת את x, $M(x) = \infty$ אם המכונה לא עוצרת בהינתן x.

לבעיות חיפוש: $M(x) = y$ אם בהינתן x, M עוצרת עם $y \coloneqq y_1 y_2 \dots y_m$ כתוב על הסרט. ו- $M(x) = \infty$ אם M לא עוצרת בהינתן x.

שפות כריעות (וכריעות למחצה)

נאמר ש-*DTM* כלשהו M **מכריע** את השפה L אם לכל קלט x, M עוצר ומתקיים:

$$M(x) = 1 \Leftrightarrow x \in L, \qquad M(x) = 0 \Leftrightarrow x \notin L$$

אם קיים *DTM* שמכריע את השפה L, נאמר ש-L ניתנת להכרעה (או **כריעה**).

שפות כריעות נקראות גם רקורסיביות. נגדיר את המחלקה: $R \coloneqq \{L \mid L\ is\ recursive\}$.

נאמר ש-*DTM* כלשהו M **מקבל** את השפה L אם לכל קלט x, מתקיים

$$x \in L \Leftrightarrow M(x) = 1, \qquad x \notin L \Leftrightarrow M(x) \neq 1$$

כאשר ב- $M(x) \neq 1$ הכוונה ש-M דוחה את x או לא עוצרת.

אם קיים *DTM* שמקבל את L, נאמר ש-L ניתנת להכרעה למחצה (או **כריעה למחצה**).

שפות כריעות למחצה נקראות גם *Recursively Enumerable* – שפות הניתנות למניה רקורסיבית. כלומר, אפשר למנות את כל המחרוזות האפשריות בשפה, אחת אחרי השנייה. נגדיר את המחלקה: $RE \coloneqq \{L \mid L\ is\ recursively\ enumerable\}$.

מתקיים $R \subseteq RE$, כי הדרישה שה-*DTM* תמיד יעצור היא דרישה חזקה יותר. אם יש *DTM* שתמיד עוצר, בפרט הוא עוצר על מילים ששייכות לשפה.

בהמשך נוכיח ש- $R \neq RE$.