

# מודל של תוכניות Java שקול למודל $T$ .

לפני שמראים שקילות, צריך להגדיר מה זה אישור ודחיה בתוכניות Java. מכיוון שפרוצדורה יכולה להחזיר ערך, נגדיר ערך 0 להיות שקר וערך 1 להיות אמת.

## הוכ<sup>1</sup>

כיוון ראשון - תוכניות מחשב לא חלשות יותר ממכונות טיורינג ניתן לבצע סימולציה של מכונות טיורינג ע"י תוכנית:

סרט מערך

ראש מצביע למערך

מצב משתנה

וסימולציה צעד אחר צעד - פשוט.

כיוון שני - מ"ט לא חלשה יותר מתוכנית מחשב

לא נבצע סימולציה לשפה עילית כמו Java, שכן יותר פשוט לסמלץ שפת Assembly, וידוע שיש קומפילירים שממירים שפות עיליות לשפות מכונה

## שפת תכנות בסיסית

אלמנטים (מרכיבים) הבסיסיים

1. זיכרון

(א) משתנים

i. בודדים

ii. מערכים

2. פעולות

(א) השמה

i. קבוע

ii. בין משתנים  $x \leftarrow y$

(ב) אריתמטיות

i.  $+, -, +1$

3. זרימה

---

<sup>1</sup>לא הוכחה מלאה

*goto* (א)

(ב) תנאים - על הזרימה

*if*  $x = 0$  *goto* .i  
*exit* [x] .ii

## איך נייצג?

### זיכרון

לתוכנית מחשב יש מספר סופי של משתנים - ניתן לכל אחד מהם סרט. משתנים בודדים ייוצגו בצורה אונרית - מספר התאים המלאים הוא ערך המשתנה. מערכים ייוצגו באמצעות הסרט, כאשר כל תא בסרט ייצג תא במערך (המערך יהיה סופי). כך יהיה ניתן לגשת לתא בזיכרון לפי הערך של משתנה אונרי.

### פעולות

העתקה זה לא בעיה, ופעולות אריתמטיות - במשתנים בודדים זה לא בעיה כי הם אונרים, ובמערכים יש לנו מצבים סופיים, אז ניתן לכתוב טבלת כפל/חיבור/כל פעולה אחרת.

### זרימה

נשים לב שבכל צעד במכונת טיורינג יש גם *if* (שכן יש שורה נפרדת לכל מצב  $\times$  כל אות בסרט) וגם *goto* (המצב הבא אליו המכונה תלך). לכן אין בעיה לבצע בקרת זרימה.

## התזה של Church-Turing

אין מודל חישובי אוטומטי חזק יותר ממכונת טיורינג במודל  $T$ . הכוונה שברגע שמגדירים את הכללים של המודל, כל צעד וצעד, אז אפשר לסמצלץ אותו במכונת טיורינג.

### מכונת טיורינג אוניברסלית

כמו שניתן לתכנת מחשבים, ניתן לחשוב על מכונת טיורינג אחת שיכולה לעשות כל מה שמכונות טיורינג יכולות לעשות. למכונה הזו קוראים "המכונה האוניברסלית". מסמנים בד"כ  $U$ , מקבלת כקלט זוג:

$M$  תיאור של  $M$ .

$w$  מחרוזת

ומחזירה (מסמלצת) את  $M$  על  $w$  ומחזירה את מה ש  $M$  על  $w$  הייתה מחזירה (עד כדי קידוד)

בעולם המחשבים, המכונה האוניברסלית היא בעצם מערכת ההפעלה - היא מקבלת תוכנית וקלט ומריצה את התוכנית על הקלט.

## סימון

תהי  $M$  מכונת טיורינג. נסמן ב- $L(M)$  את השפה ש- $M$  מזהה.

$$L(M) = \{w \in \Sigma^* \mid M \text{ accepts } w\}$$

מגדירים לפי זיהוי ולא לפי הכרעה שכן לא מובטח לנו ש- $M$  מכריעה שפה. לכל מכונה  $L(M)$  מוגדרת, אבל לא לכל מכונה מוגדרת השפה שהיא מכריעה.

## הבחון יהיה עד כאן

# האם ישנן בעיות שלא ניתנות לפתרון בעזרת מחשב?

האם ישנן שפות שלא ניתנות לזיהוי/הכרעה ע"י מ"ט?

## תשובה - וודאי

משיקולי ספירה/שיקולי עצמה.  
נראה שיש "יותר" שפות ממכונות.

מכונות נסמן ב- $A$  את אוסף כל המכונות מעל  $\Sigma$ .  $|A| \leq ?$   
כל מ"ט/תוכנית מתוארת ע"י מחרוזת מעל א"ב סופי  $\Pi$  (א"ב של התוכניות)  
תוכניות ב- $\Pi^*$ .  $|\Pi^*| = \aleph_0$ .  $|A| = \aleph_0$

שפות נסמן ב- $B$  את אוסף כל השפות מעל  $\Sigma$ . שפה  $L$  הינה קבוצה חלקית  $\Sigma^*$ .  
 $B \subseteq 2^{\Sigma^*}$ . אוסף כל תתי הקבוצות של  $\Sigma^*$  הוא  $2^{\Sigma^*}$ .  
 $|B| = 2^{\aleph_0} = |\Sigma^*| < \aleph_0$

$$|A| = \aleph_0 < 2^{\aleph_0} = |B|$$

ישנן יותר שפות ממכונות ישנן שפות לא כריעות וישנן שפות לא ניתנות לזיהוי.

## ההוכחה הזאת נכונה, אבל לא מספקת

- היא לא קונסטרוקטיבית - היא אינה מראה אילו שפות אינן ניתנות להכרעה.
- יכול להיות שהשפות האלו בכלל לא מעניינות אותנו.
- למעשה ההוכחה הזאת היא קצת לא מעניינת, שכן לרוב השפות אין תיאור סופי, וקיבלנו שלרוב השפות אין תיאור סופי ולכן אי אפשר להכריע אותן באמצעים סופיים.

נרצה שפה שהיא מעניינת, בעלת תיאור סופי, ואי אפשר לזהות/הכריע אותה באמצעות מכונת טיורינג.

<sup>2</sup>מבחינת עצמה

## דוגמה לשפה מעניינת שאי אפשר לזהות/להכריע באמצעות מ"ט

היינו רוצים דרך לבדוק אוטומטית שתוכנה עושה את מה שהיא אמורה לעשות. נרצה להיות מסוגלים לתת אותה לתוכנית אחרת שתבדוק את זה. נרצה תוכנה  $Y$  שבהינתן תוכנה  $P$  וספסיפקציות  $S, Y(P, S)$  תאשר תוכנות טובות ותדחה תוכנות לא טובות. כדי לפשט את זה, הספסיפקציות יהיו קלטים  $G_1, G_2, G_3, G_4$  כשלכל קלט אנו יודעים אם התוכנה אמורה לאשר או לדחות אותו. לכאורה הפתרון פשוט - נעשה סימולציה לתוכנית. אבל זה בעיה - שכן לא מובטח לנו שהתוכנית תעצור.

### הבעיה

בהינתן תוכנית  $P$  ומחרוזת  $w$ , האם  $P(w) = 1$ , כלומר האם  $\perp$  היינו מריצים את  $P$  על  $w$  היינו מקבלים 1. כלומר נגדיר שפה

$$A_{TM} := \{(P, w) | P(w) = 1\}$$

( $A$  זה acceptance,  $TM$  זה Turing Machine)  
לדוגמה:

$$("Q(x) \{while\ 1; 3\}", "aba") \notin A_{TM}$$

$$("F(y) \{if\ y = 'a' \ return\ 1; \ return\ 0\}", "a") \in A_{TM}$$

### משפט

$A_{TM}$  לא כריעה

### הוכחה

בשליחה: נניח  $A_{TM}$  כריעה. תהי  $D_{A_{TM}}$  תוכנית שמכריעה את  $A_{TM}$ . בעזרת  $D_{A_{TM}}$  נבנה תוכנית אחרת  $stupid$ :

```
stupid(x)
{
    a ← DATM(x, x);
    return !a;
}
```

$stupid$  תמיד עוצרת (שכן  $D_{A_{TM}}$  תוכנית להכרעה ולכן תמיד עוצרת). אבל  $stupid$  היא גם מתרוזת, אז מה יקרה אם נפעיל אותה על עצמה:מה  $stupid(stupid)$  תחזיר:

שתי אפשרויות:

$$\begin{aligned} 1. \quad & \text{stupid}(\text{stupid}) = 1 \\ & \Leftarrow a = 0 \text{ (בניה)} \\ & D_{ATM}(\text{st}, \text{st}) = 0 \text{ (בניה)} \\ & \Leftarrow (D_{ATM} \text{ הגדרת } (st, st) \notin A_{TM}) \\ & \Leftarrow (A_{TM} \text{ הגדרת } st(st) \neq 1) \\ & \text{והגענו לסתירה.} \end{aligned}$$

$$\begin{aligned} 2. \quad & \text{stupid}(\text{stupid}) = 0 \\ & \Leftarrow a = 1 \text{ (בניה)} \\ & D_{ATM}(\text{st}, \text{st}) = 1 \text{ (בניה)} \\ & \Leftarrow (D_{ATM} \text{ הגדרת } (st, st) \in A_{TM}) \\ & \Leftarrow (A_{TM} \text{ הגדרת } st(st) = 1) \\ & \text{שוב סתירה} \end{aligned}$$

מסקנה -  $\text{stupid}$  לא קיימת. ההנחה היחידה בבניית  $st$  הייתה ש  $D_{ATM}$  קיימת  $\Leftarrow D_{ATM}$  לא קיימת  $\Leftarrow A_{TM}$  לא כריעה ■

## טענה

$A_{TM}$  ניתנת לזיהוי.

## הוכחה

המכונה  $U$  מזהה את  $A_{TM}$  -  $U(P, w)$ .  $U$  מריצה את  $P$  על  $w$ .

# בעיית העצירה - Halting Problem

$$H = \{(P, w) | P \downarrow w\}$$

רוצים לדעת אם השפה עוצרת.

## משפט

$H$  לא כריעה

## הוכחה

בשליה. נניח  $H$  כריעה. תהי  $D_H$  תוכנית שמכריעה את  $H$ . נבנה  $D_{ATM}$  תוכנית שמכריעה את  $A_{TM}$ .

הערה - אם  $A \subset B$  ו  $B$  כריעה זה לא אומר ש  $A$  כריעה

נבנה את  $D_{ATM}$ :

```

D_{ATM}(P, w)
{
  1. a ← D_H(P, w)
  2. if a == 0 return 0;
  3. else
    return(U(P, w));
}

```

מתקיים:

אם  $P(w) = 1$  אזי  $P \downarrow w$  ולכן  $a == 1$  ולכן נגיע לשורה 3 ונחזיר 1.  
אם  $P(w) = 0$  אזי  $P \downarrow w$  ונגיע לשורה 3 ונחזיר 0.  
אם  $P \uparrow w$  אזי  $D_H$  יחזיר 0 ונחזיר 0 בשורה 2.  
← נחזיר 1 אם  $P(w) = 1$  ונחזיר 0 אם  $P(w) \neq 1$  כנדרש.

← מסקנה

הכרענו את  $A_{TM}$  בסתירה;  
← משהו לא בסדר.  
←  $D_H$  לא קיימת ■