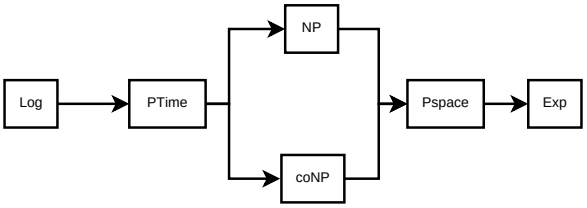
סיבוכיות – הרצאה 4

כל העולם הוא חישוב אחד גדול

15.3.11

תחת החלוקה הגסה לפי עלות זכרון וזמן ורק עבור מכונות דטרמיניסטיות ולא דטרמיניסטיות קיבלנו את תמונת המצב הבאה:



ראינו הרכבה של זמן, אם $T(n) \gg t(n)$, (כלומר $T(n)$ הוא מספיק כדי לסמלץ מכונה שרצה ב- $t(n)$) אז

$$Time(T(n)) \supsetneq Time(t(n))$$

באותו אופן נראה גם הרכבה של מקום: אם $S(n) \gg s(n)$ אז $Space(S(n)) \supsetneq Space(s(n))$.
מחבינתו P זה קל. למרות שייתכנו פולינומים עם חזקות ממש גדולות אפשר איכשהו להתמודד עם זה לעומת נגיד Exp שדורש בכלל זמן אקספוננציאלי. Log הוא מאוד קל. אפשר למקבל. מצד שני יש לנו בעיות מעניינות ב- NP אבל אנחנו לא יודעים האם זה קל או קשה.
בחלק השני של הקורס נתחיל לשחק עם ההגבלות שראינו. אולי עם מעגלים אפשר לפתור את SAT מהר? אולי חישוב הסתברותי יכול לעזור? אולי SAT קשה לפתרון מדוייק אבל יש אלגוריתמי קירוב לבעיות אופימיזציה? אולי SAT קשה אבל יש אלגוריתם פולינומי שטועה על SAT אבל קשה מאוד למצוא קלטים עליהם הוא טועה? אולי הכל לא עובד ו-SAT נורא קשה אבל אם כך, האם אפשר לנצל את הקושי ולעשות משהו מועיל?
כל אלה הם בסדר מבחינתנו בעולם האמיתי.
כדי לסיים עם המיון הראשוני לקל לקשה נדבר על $Logspace$.

סיבוכיות מקום

תזכורת. למכונה מוגבל זכרון יש סרט קלט (שאסור לכתוב עליו) סרט עבודה רגיל וסרט פלט (שרק זורקים עליו פלטים) ועלות הזכרון היא של סרט העבודה בלבד.

עובדה: אם $S(n), s(n)$ מקיימים:

$$\lim \frac{s(n)}{S(n)} = 0$$

אז יש מ"ט שרצה בזכרון $S(n)$ שבהינתן מ"ט M שמשתמשת ב- $s(n)$ זכרון ובהינתן קלט x מסמלצת את M על x.
אז

$$Space(S(n)) \supsetneq Space(s(n))$$

ואפילו

$$\forall O \quad Space^O(S(n)) \supsetneq Space^O(s(n))$$

תזכורת. בהינתן מ"ט M וקלט x שמים בקונפיגורציה כל מה שמשפיע על צעד החישוב הבא: סרט עבודה, מצב מ"ט, ראש קורא קלט, ראש קורא עבודה.
במקרה זה סרט קלט הוא קבוע אז אין צורך להתחשב בו. ראש קורא פלט וסרט הפלט לא משפיעים אז כל אלה לא בקונפיגורציה.
מצב מ"ט זה $O(1)$, סרט עבודה $2^{O(s(n))}$ ראש קורא קלט n, סרט עבודה $s(n)$. נעבוד עם $s(n) \geq \log n$ ולכן מספר הקונפיגורציות הוא $2^{O(s(n))}$.

תזכורת. גרף הקונפיגורציות של M דטרמיניסטית על x: קודקודיו הם הקונפיגורציות האפשריות ויש קשת בין c_1 ל- c_2 אם המכונה M על קלט x תעבור ממצב c_1 למצב c_2.
דרגת היציאה בגרף כזה היא 1, יש מצב תחילי ויש שני סוגים של מצבים מסיימים: מצב מסיים מקבל ומצב מסיים דוחה.
מ"ט עם זכרון s שעוצרת עsל קלט x עוצרת תוך $2^{O(s)}$ צעדים)תחת ההנחנה $s(n) \geq \log n$).

מסקנה. אס $s(n) \geq \log n$ אז

$$Space(s(n)) \subseteq DTime(2^{O(s(n))})$$

ולכן

$$Log \subseteq P$$

$$Pspace \subseteq Exps$$

המושג של מוגבל זכרון הוא די עדין כי גם הקלט וגם הפלט יכולים להיות עצומים כל עוד לא עושים יותר מדי חישובי ביניים.

טענה. אס $A \leq_L B$ וכן $B \leq_L C$ אז $A \leq_L C$.

הוכחה. $A \leq_L B$ לכן יש $\varphi \in L$, $\varphi : \{0,1\}^* \to \{0,1\}^*$ כך ש-

$$\varphi(x) \in B \Leftrightarrow x \in A$$

$B \leq_L C$ לכן יש $\psi \in L$, $\psi : \{0,1\}^* \to \{0,1\}^*$ כך ש-

$$\psi(x) \in C \Leftrightarrow x \in B$$

המועמד הטבעי ברדוקציה (בדיוק כמו רדוקצית זמן) הוא $\psi \circ \varphi(x) = \psi(\varphi(x))$.
כמובן שמתקיים:

$$x \in A \Leftrightarrow \varphi(x) \in B \Leftrightarrow \psi(\varphi(x)) \in C$$

האם $\psi \circ \varphi \in L$?
אלגוריתם נאיבי: בהינתן קלט x נחשב את $\varphi(x)$ ועליו נחשב $\psi(\varphi(x))$.
זה לא יעבוד כי לאו דווקא יש לנו מקום לשמור את $\varphi(x)$. זאת בעיה טיפוסית עבור רדוקציות מקום ועל כן, יש לה פתרון סטנדרטי:
אין לנו מקום לשמור את $\varphi(x)$ אז לא נשמור אותו. במקום זה כל פעם שנצטרך ביט של $\varphi(x)$ פשוט נחשב מחדש. באופן פרומלי נריץ האלגוריתם של ψ כאילו יש לנו $\varphi(x)$ וזה לוקח $Logspace$. כל פעם שנצטרך ביט של $\varphi(x)$ נריץ את האלגוריתם של φ ונספור את ביטי הפלט עד שנגיע לביט שצריכים ואז נמשיך בהרצת ψ.
כן זה בזבזני. אבל באופן תיאורטי זה עובד כי לא משנה את זמן הריצה אסימפטוטית. □

הגדרה. מכונת טיורינג לא דטרמיניסטית מוגבלת זכרון. מ"ט עם סרט קלט $(R, \leftrightarrow)$, סרט עבודה $(RW, \leftrightarrow)$ וסרט פלט $(W, \rightarrow)$.
יחס ה δ של מ"ט הוא יחס.ד

$$\delta : \Sigma \times S \to P(\Sigma \times S \times \{\rightarrow, \leftarrow, \downarrow\})$$

נגיד שמכונה מקבלת קלט x אם יש מסלול חוקי לפי δ ממצב תחילי למקבל.
סיבוכיות הזכרון תהיה המקום על סרט העבודה.

נסמן $NL = NSpace(O(\log n))$.
היינו רוצים למצוא

$$NSpace(s(n)) \subseteq DSpace(?)$$

$$NSpace(s(n)) ? coNSpace(s(n))$$

כמובן היינו רוצים גם למצאות בעיות שלמות ל- $NL, NPSpace$.

STCON

קלט: גרף מכוון $G = (V, E)$ וכן $s, t \in V$.
פלט: כן $\Leftrightarrow$ יש מסלול מ-s ל- t בגרף.
פתרון נאיבי הוא BFS, DFS אבל רצים בזכרון לינארי בגודל הקלט. נוכל להשתמש באי-דטרמיניזם:
נתחיל ב- s. אם בשלב ה- i אנחנו בקודקוד v_i אז נתפצל על כל שכני v_i בשלב הבא. אם הגענו ל- t נעצור ונקבל. אם $i > |V|$ נעצור ונדחה.
זה אלגוריתם לא דטרמיניסטי במהותו. כמה זכרון האלגוריתם לוקח? צריכים לשמור את הקודקוד הנוכחי $|V| = n$ ומונה על מספר הצעדים בין 1 ל- n. סה"כ $O(\log n)$ זכרון. נשים לב שאין צורך לשמור על אילו קודקודים עברנו כי החישוב לא דטרמיניסטי.
נכונות: אם s קשור ל- t אז יש מסלול מ-s ל- t באורך $n \geq$ ולכן יש מסלול מקבל. אם לא כל המסלולים לא יגיעו ל- t ולא יהיה מסלול מקבל.

מסקנה.

$$STCON \in NL$$

בהינתן מ"ט ל"ד M וקלט x נגדיר קונפיגורציה כמו קודם ואת גרף הקודקודים כך שקיימת צלע $\Leftrightarrow$ אפשר לעבור למצב. כלומר, עכשיו דרגת היציאה יכולה להיות יותר מ-1. לבדוק האם יש מסלול ממצב תחילי למצב מקבל זה מאוד דומה ל- $STCON$. למעשה, זה בדיוק אותו דבר!

משפט. $STCON$ היא NL קשה ולכן גם NL שלמה.

הוכחה. בהינתן מכונת טיורינג M וקלט x צריך למצוא רדוקציה φ שתחזיר לנו $G=(V,E)$ כך ש- M מקבלת את $x \Leftrightarrow$ s קשור ל- t ב- G וגם $\varphi \in L$.
הרדוקציה $\varphi(x)$ תבנה גרף שקודקודיו כל הקונפיגורציות, הקשתות לפי גרף הקונפיגורציות.
s הקודקוד של מצב תחילי ו- t הקודקוד של מצב מסיים מקבל.
נכונות: מיידית. M מקבלת את $x \Leftrightarrow$ בגרף הקונפיגורציות יש מסלול ממצב תחילי למקבל $\Leftrightarrow (G,s,t) \in STCON$.
סיבוכיות מקום:
$Forc_1$ קונפיגורציה
$Forc_2$ קונפיגורציה
נדפיס למדפסת 1 אם יש מעבר חוקי מ- c_1 ל- c_2 ו-0 אחרת.
אפשר פשוט למספר את כל הקונפיגורציות האפשריות ויש $2^{O(s(n))}$ כאלה לכן c_1, c_2 יקחו $s(n)$ זכרון. $\square$

הערה. גודל הגרף הוא $2^{O(s(n))}$ ולכן אם יש מסלול אז אורכו $\geq 2^{O(s(n))}$ ואז בה"כ המכונה יכולה לספור $2^{O(s(n))}$ צעדים ואז לעצור.

כעת נרצה לחשוב על $NSpace(s) \subseteq DSpace(?)$.

טענה.

$$STCON \in DSpace(\log^2 n)$$

וממלא $NL \subseteq DSpace(\log^2 n)$ (בדומה למה שעשינו בתחילת השיעור). באופן כללי ינבע:

משפט. *Savich*
לכל $s(n) \geq \log n$ קיים:

$$NSpace((s(n)) \subseteq DSpace(s^2(n))$$

ובפרט

$$NPSpace \subseteq PSpace$$

הוכחה. **(של הטענה)** נתחיל מהוכחה אינטואיטיבית. הקלט (G,s,t) נסמן ב- A את מטריצת השכנויות של G.
$A[i,j]=1 \Leftrightarrow$ יש קשת מ- i ל- j ב- G.
$A[i,j]=0 \Leftrightarrow$ אין קשת מ- i ל- j ב- G.
כעת, $A^2[i,j] \geq 0 \Leftrightarrow$ יש מסלול באורך 2 מ- i ל- j ב- G.
ואז $(A+I)^k[i,j] \geq 0 \Leftrightarrow$ יש מסלול באורך $k \geq$ מ- i ל- j ב- G.
אז s קשור ל- t ב-$G \Leftrightarrow$ יש מסלול מ-s ל- t באורך $n \geq 0 < (I+A)^n[s,t]$.
כעת די לעלות את מטריצת $I+A$ בחזקת n (שזה $Logspace$) והתשובה כתובה ב- $[s,t]$.
סה"כ $\log^2 n$ זכרון כי $\log$ רמות רקורסיה, כל רמה ב- $\log$ זכרון. וזה כואב לנו.
נוכיח פורמלית.

נגדיר פרדיקט $Reach(v_1, v_2, k)$ שמקבל 1 $\Leftrightarrow$ יש מסלול מ- v_1 ל- v_2 באורך $\geq 2^k$.
עבור $k = 0$ התשובה כתובה בקלט.
כיצד נחשב את

Algorithm 1 $Reach(v_1, v_2, k$

```
for w ∈ V do
  if Reach(v1, w, k − 1 ∧ Reach(w, v2, k − 1) then
    return true
  end if
end for
return false
```

נכונות: אם יש מסלול מ- v_1 ל- v_2 באורך $2^k \geq$ אז ניקח קודקוד w באמצע עבורו $Reach(v_1, w, k-1) = T \wedge Reach(w, v_2, k-1) = T$ ונענה כן. אם אין כזה מסלול אז אין כזה w ונדחה.
זכרון: נסמן $T(k)$ זכרון הנדרש לרמה k.

$$T(k) = T(k-1) + O(\log n)$$

מדוע לא $2T(k-1)$? הרי קוראים לרקורסיה פעמיים. אבל אחרי הקריאה הראשונה אין צורך כבר בזכרון שהתמשנו לכן אפשר לדרוס אותו.
לכן, $T(k) = O(n \log n)$. החישוב הוא $Reach(s, t, \log n)$ ולכן הזכרון הוא $O(\log^2 n)$.
□

משפט. אמרמן- סלפציני

$$NL = coNL$$

הוכחה. מספיק להראות $\overline{STCON} \in NL$. יודעים ש- $STCON$ שלמה ב- NL לכן $\overline{STCON}$ שלמה ב- $coNL$ ולכן אם $\overline{STCON} \in NL$ אז $coNL \subseteq NL$ ולכן גם $cocoNL \subseteq NL$ כי $A \subseteq B \Rightarrow coA \subseteq coB$ כי $L \in coA \Rightarrow \overline{L} \in A \Rightarrow \overline{B} \in B \Rightarrow L \in coB$.
יהי (G, s, t) קלט. הקלט בשפה $\Leftrightarrow$ אין מסלול מ- s ל- t ב- G.
צריכים למצוא מכונת NL שבודקת עד לכך שאין מסלול מ-s ל- t.
נגיד שמכונה לא דטרמיניסטית פותרת שפה L עם אפס שגיאה אם עבור $x \in L$ יש מסלול מקבל וכל מסלול שאינו מקבל עונה $quit$. עבור $x \notin L$ יש מסלול דוחה וכל מסלול שאחנו דוחה עונה $quit$.
הקלט הוא (G, s, t). נסמן c_i - מספר הקודקודים ב- G שאפשר להגיע מ- s אליהם תוך $i \geq$ צעדים. לדוגמא $c_0 = 1$.
נגדיר פרדיקט $Reach(v, i, c_{i-1})$ שאמור לענות האם אפשר להגיע מ-s ל- v תוך $i \geq$ צעדים בהינתן מספר c_{i-1}.

אם יש מסלול מ- s ל- v באורך $i \geq$. אז יש w כך ש- $(w, v) \in E$, יש מסלול באורך $i - 1 \geq$ מ-s ל- t.
המסלול בחישוב: החישוב שעבור ה- w הזה ינחש מסלול באורך $i - 1 \geq$ מ- s ל- w יהיה מסלול מקבל. אין אף מסלול דוחה כי כדי לדחות צריך ש- $counter = c_{i-1}$ ולכן בהכרח

Algorithm 2 Reach(v, i, c_{i-1})

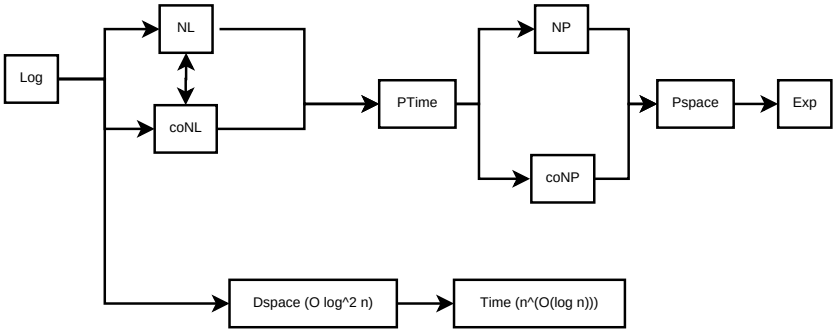
```
count = 0
for w = 1 to n do
  guess a path from s to w of length ≤ i − 1
  if valid path then
    count++
  end if
  if (w, v) ∈ E then
    return true
  end if
end for
if count == c_{i−1} then
  return false
end if
return QUIT
```

ביקרנו בכל הקודקודים שאפשר להגיע אליהם מ- s ל- $i-1$ צעדים ולכן בהכרח ביקרנו בקודם של v ולכן נקבל. אם אין מסלול מ- s ל- v באורך $i \geq$.
המסלול שיינחש מסלולים נכון לכל הקודקודים של המרחק $i-1$ יקבל $counter = c_{i-1}$ ואז אף אחד מהם אין לו קשת ל- v (כי אין מסלול באורך i ל- v) ולכן יחזיר $false$. ברור ששום התפצלות לא יכולה להחזיר $true$ כי אין מסלול באורך $i \geq$.
אז החישוב הוא באפס שגיאה.
האלגוריתם הסופי:

```
c_current = 1
for i = 1 to n do
  for v ∈ V do
    c_i = 0
    if Reach(v, i, c_current) then
      c_i++
    end if
  end for
  c_current = c_i
end for
```

התשובה הסופית תהיה נחשב את c_{n-1} ונענה $Reach(t, n, c_{n-1})$ □

הערה. באותה מידה אפשר להגדיר מכונה לא דטרמיניסטית מוגבלת זמן שכל מסלול בה עונה $accept, reject, quit$ ופותרת את השפה באפס שגיאה.
מחלקת כל השפות שניתנות לפתרון בזמן פולינומי לא דטרמיניסטי באפס שגיאה זה בדיוק $NP \cap coNP$.


Log
NL
coNL
PTime
NP
coNP
Pspace
Exp
Dspace (O log^2 n)
Time (n^(O(log n)))