11.5.09 (1)

סיבוכיות - שיעור 9

סיבוכיות מקום
- SPACE, L, NL
- בעיות שלמות NL
- coNL=NL
- בעיות שלמות ב- PSPACE

אלגוריתמי קירוב
- VC
- LP
- set cover

בעיות שלמות ב- PSPACE

למרות שזה נראה הזוי, לא יודעים להוכיח ש- $P \neq PSPACE$ (זה היה ייתכן אם $P=NP=PH$).

<u>תרגיל</u>: הוכיחו שהשפה הבאה היא PSPACE שלמה:

$SPACETM = \{ \langle \alpha, w, 1^n \rangle :$ האם מ"ט M_α על הקלט w מגיעה ל-1 תוך שימוש ב-n מקום $\}$

<u>נוסחה מכומתת לחלוטין</u> (Totaly quntified boolean formula): היא נוסחה מהצורה:

$$\exists/\forall x_1 \; \exists/\forall x_2 \dots \exists/\forall x_3 \,.\, \underbrace{u(x_1, x_2, \dots, x_n)}_{\text{נוסחה ללא כמתים}}$$

<u>דוגמאות</u>:

$$\exists x_1 \forall x_2 \,.\, \underbrace{(x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2)}_{x_1 = x_2} \leftarrow \text{משפט שקר}$$

$$\forall x_1 . \exists x_2 . (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2) \leftarrow \text{משפט אמת}$$

נגדיר את <u>TQBF</u> להיות שפת כל הנוסחאות המכומתות לחלוטין שהן אמת.

<u>הערה</u>: SAT היא המקרה הפרטי שבו כל הכמתים הם $\exists$.

<u>טענה</u>: TQBF שייכת ל- PSPACE.

(א נשים לב שברקורסיה כמות הזכרון מושפעת מעומק הרקורסיה ולא ממספר הצעדים א/)

<u>הוכחה</u>: נראה פרוצדורה לפתרון TQBF. בהינתן קלט $\exists/\forall x_1 \exists/\forall x_2 \dots \exists/\forall x_n . u(x_1, \dots, x_n)$ כאשר u היא נוסחה בגודל m, הפרוצדורה קוראת לעצמה ברקורסיה על נוסחת TQBF שמתקבלת ע"י הצבת T ב- x_1 ועל הנוסחה שמתקבלת ע"י הצבת F ב- x_1. אם כמת x_1 היה $\exists$ נקבל אמ"מ לפחות אחת מהקריאות החזירה T בכמת $\forall$ נקבל אמ"מ שתי הקריאות החזירו T. קל לראות שהאלגוריתם נכון וכמות הזכרון שהוא משתמש היא $O(m \cdot n)$ זכרון (כי עומק הרקורסיה n והפונקציה צריכה $O(m)$ זכרון על מנת להציב ערכים ב- x_1). ∎

<u>תרגיל</u>: שפרו ל- $O(m+n)$ זכרון. (אפשר לזכור רק את ההצבות שהצבנו)

<u>מסקנה</u>: $PH \subseteq PSPACE$

<u>הערה</u>: TQBF לא מוכיחה שנמצאת ב- PH, למרות שאולי זה נראה כך ממבט ראשון.
הסיבה: מספר החלפות הכמתים ב- TQBF יכול להיות n ולא קבוע כמו ב- PH.

סיבוכיות- שיעור 9

<u>משפט</u> TQBF היא PSPACE שלמה.

<u>הוכחה</u>: נותר להראות שהיא PSPACE-קשה.

תהי L שפה כלשהי ב- <u>PSPACE</u> ונראה ש- $L \le_p TQBF$.
תהי M מכונה המחשבת <u>את L</u> <u>במקום</u> $O(s(n))$ עבור איזשהו פולינום $s(n)$.

נראה כיצד ניתן בהינתן $x \in \{0,1\}^n$ <u>לבנות</u> נוסחה מכומתת שלמה ψ בגודל $O(s(n)^2)$ כך ש- M מקבלת את x <u>אמ"מ</u> ψ היא נוסחת אמת.

יהא $G_{M,x}$ גרף הקונפיגורציות של M על קלט x. ונסמן ב- $m = O(s(n))$ את מספר הביטים הנדרשים לייצוג קודקוד בגרף (כלומר מספר הקודקודים $\ge 2^m$).

נבנה ψ שתתקיים אם יש מסלול באורך $\ge 2^m$ מ- C_{start} קודקוד מצב התחלתי ל- C_{accept}.

<u>נסיון 1</u> נשתמש ב- 2^m משתנים $C_1, C_2, \ldots, C_{2^m}$ כשכל משתנה הוא בעצם m משתנים בוליאניים ומייצג קודקוד בגרף הקונפיגורציות.

$$\psi = \exists C_1, \ldots \exists C_{2^m}. (C_1 = C_{start}) \wedge (C_{2^m} = C_{accept}) \wedge \underbrace{\varphi(C_1, C_2)}_{\text{קיימת קשת מ-}C_1\text{ אל }C_2} \wedge \varphi(C_2, C_3) \ldots \wedge \varphi(C_{2^m-1}, C_{2^m})$$

כאשר $\varphi(C,D)$ היא נוסחה בגודל פולי' הבודקת שיש בגרף הקונפיגורציות קשת מ- C ל- D. ניתן לבנות את φ כמו שעשינו במשפט Cook-Levin (תרגיל).

* נשים לב שלא השתמשנו בשני הכמתים וזו למעשה נוסחת SAT *

אבל הנוסחה שבנינו היא בגודל <u>מעריכי</u>!

<u>נסיון 2</u> נשתמש ברעיון של SAVITCH. נבנה את ψ ברקורסיה כאשר $\psi_i(C,D)$ אומרת האם יש מסלול באורך 2^i מ- C ל- D. את ψ ניקח להיות $\psi_m(C_{start}, C_{accept})$ את ψ_0 נגדיר להיות φ כמו קודם (היא בגודל פולינומי!).

נגדיר את ψ_i כך:
$$\psi_i(C_1, C_2) = \exists D. \psi_{i-1}(C_1, D) \wedge \psi_{i-1}(D, C_2)$$

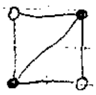
* שים לב השתמשנו ב- $\forall$ אלא רק ב- $\exists$. הבעיה של הנוסחה הזו שהיא עדיין בגודל מעריכי ולכן זו נוסחה סופית. *

<u>נסיון 3 (ומוצלח)</u> נשתמש בכמת $\forall$:

$$\psi_i(C_1,C_2) = \exists D. \forall (D_1, D_2) \in \{(C_1, D), (D, C_2)\}. \psi_{i-1}(D_1, D_2) = \exists D. \forall D_1 D_2. (D_1 = C_1 \wedge D_2 = D) \vee (D_1 = D \wedge D_2 = C_2) \rightarrow \psi_{i-1}(D_1, D_2)$$

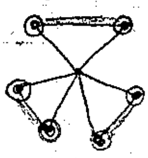
<u>תרגיל</u> להשתכנע שהכל עובד.

<u>אלגוריתמי קירוב</u> (Approximation Alg.)

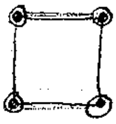


נזכר בבעיית ה- VC: <u>קלט</u>: $G=(V,E)$ גרף לא מכוון ומספר k.
<u>פלט</u>: "כן" אמ"מ קיימת $C \subseteq V$ בגודל $\le k$ כך שלכל קשת $(i,j) \in E$, מתקיים שלפחות אחד מ- i ו- j ב- C.

11.5.09 3



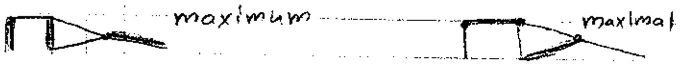
דוגמא: האלגוריתם בדוגמא הבאה ניתן כיסוי בגודל 6, אופטימום הוא 4. יחס של 1.5.



דוגמא: אלגוריתם: 4
אופטימום: 2
יחס של 2.

אלגוריתם: 1. בנה זיווג מקסימלי $M \subseteq E$ (maximal matching). כלומר, להוסיף קשתות שלא נוגעות בקשתות הקיימות (זיווג) עד שאי אפשר. (מקסימלי)

למשל:



2. נקח את כל הקודקודים ש-M נוגעת בהם $C=\{v \mid \exists e\in M .\ v\in e\}$

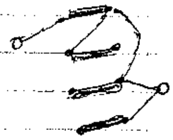
ניתוח האלגוריתם: א. זמן ריצה פולינומי - ברור.

ב. הפלט הוא כיסוי צמתים חוקי כי כל קשת ב-E חייבת לגעת ב-M כיוון ש-M מקסימלית.

ג. יחס הקירוב הוא ≤ 2:

טענה: לכל קלט G, גודל הקבוצה $|ALG(G)| \le 2\cdot|OPT(G)|$

הוכחה: גודל הפלט הוא $|ALG(G)|=|C|=2|M|$ כש-M הוא הזיווג המקסימלי שהאלג' מצא.



כיוון ש-M הוא זיווג, הגודל של האופטימום הוא לפחות $|M|$ כי האופטימום חייב להכיל לפחות קודקוד אחד מכל קשת בזיווג.
$|M|\le|OPT| \Rightarrow |C|\le 2|OPT|$ ☒

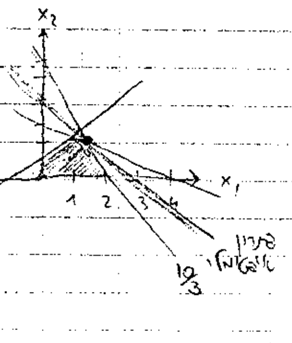
תרגיל (קשה): הראו שהאלג' החמדן Greedy שלוקח כל פעם הקודקוד בדרגה המקסימלית, נותן יחס קירוב של $\ge \log n$ ~, (כלומר יש קלטים שעליהם יתנהג בצורה לא טובה).

הערה: ידוע שזה NP-קשה לקרב ביחס 1.36 [DINUR SAFRA 2002]. ידוע שזה קשה (אך לא NP-קשה) לקרב עד כדי $2-\varepsilon$ [KHOTR 03].

תכנות לינארי (Linear Programing)

דוגמא לבעיית תכנות לינארי:

maximize x_1+x_2
subject to $x_1\ge 0;\ x_2\ge 0;\ x_1+2x_2\le 4;\ 4x_1+2x_2\le 12;\ x_1-x_2\ge -1$



משפט [KHACHITAN '79]: קיים אלגוריתם פולינומיאלי לפתרון LP. (בשם Ellipsoid algorithm) (במציאות משתמשים באלגוריתם simplex שבפועל מהיר).

מאוד חשוב!

קירוב ל-VC בעזרת LP

בהינתן קלט ל-VC, ניתן לרשום תכנית אופטימיזציה שמייצגת פתרון מדויק לבעיה:

המשך...

$$\text{minimize} \quad x_1+x_2+\dots+x_n$$

$$\text{Subject to} \quad \forall \{i,j\}\in E .\ x_i+x_j\ge 1$$

$$\forall i .\ x_i\in\{0,1\}$$

← האילוץ הזה הופך את הבעיה ל- Integer Programing

<u>טענה:</u> הפתרון האופטימלי של התוכנית הזו, הוא בדיוק גודל הכיסוי בצמתים הקטן ביותר.

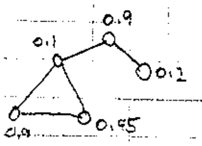
<u>הערה:</u> הראנו בעבר רדוקציה מ- VC ל- IP ולכן IP היא NP-קשה (ובעצם NP-שלמה). האילוץ $x_i\in\{0,1\}$ אינו אילוץ לינארי. נחליף אותו באילוץ לינארי כדי לקבל תוכנית לינארית:

$$\text{minimize} \quad x_1+x_2+\dots+x_n$$

$$\text{Subjet to} \quad \forall \{i,j\}\in E .\ x_i+x_j\ge 1$$

$$\forall i .\ 0\le x_i\le 1$$

נסמן את האופטימום של התוכנית הזו ב- OPT^*. ברור ש $OPT^*\le OPT$, בנוסף, אפשר לחשב את OPT^* בזמן פולינומיאלי.



<u>אלגוריתם:</u> 1. מצא פתרון אופטימלי $x_1,\dots,x_n$ ל- LP שערכו אופטימלי, כלומר $\sum x_i = OPT^*$.

2. תוציא את $C=\{i \mid x_i\ge \frac12\}$ ככיסוי

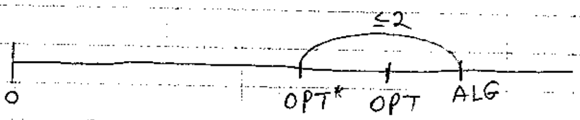
<u>ניתוח:</u> א. זמן ריצה פולינומיאלי בגלל אלג' האליפסואיד.

ב. הפלט הוא כיסוי חוקי בצמתים כי לכל קשת $\{i,j\}\in E$ מתקיים $x_i+x_j\ge 1$ ולכן חייב להיות ש- $x_i\ge\frac12$ או $x_j\ge\frac12$ ולכן הקשת $\{i,j\}$ מכוסה.

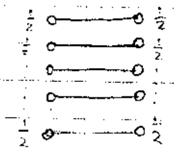
ג. יחס הקירוב הוא 2:

$$|C|\le 2\cdot\sum x_i = 2\cdot OPT^*\le 2\cdot OPT \quad ☒$$

ויזואלית:

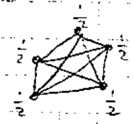


<u>דוגמא</u>

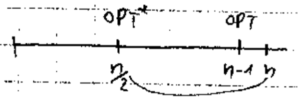


$OPT=5$
$OPT^*=5$
$ALG=10$

<u>דוגמא</u>



$OPT=4$ $n-1$
$OPT^*=\frac52$ $\frac n2$
$ALG=5$ n



מה חשיבה!

* הדוגמא מראה שכדי לקבל פקטור טוב יותר מ-2, צריך להשתמש בתוכנית אחרת *