שיעור 4 – 16.03.2010

משפט *Savitch*: $\text{STCON}\in\text{Space}\left(\log^2 n\right)$.

מסקנה: $\text{NL}\subseteq\text{DSpace}\left(\log^2 n\right)$.

קלט: גרף מכוון $G=(V,E)$ ושני קודקודים s,t .

פלט: האם s קשור ל-t ב-G או לא?

האלגוריתם:

$\text{Reach}\left(v_1,v_2;t\right)$. הערך הוא True $\Leftrightarrow$ אפשר להגיע מ-v_1 ל-v_2 בלכל היותר t צעדים.

$\text{Reach}\left(V_1,v_2;t\right)$

if $t=1$ then return $\left(v_1,v_2\right)\overset{?}{\in}E$

for $v_3\in V$

if $\text{Reach}\left(v_1,v_3;\frac{t}{2}\right)\wedge\text{Reach}\left(v_3,v_2;\frac{t}{2}\right)$ then return True

return False

נכונות:

אם יש מסלול מ-v_1 ל-v_2 בלכל היותר t צעדים, ניקח את הקודקוד v_3 באמצע, יש מסלול מ-v_1 ל-v_3 ומ-v_3 ל-v_2 בלכל היותר $\frac{t}{2}$ צעדים, ולכן נענה כן.

הכיוון ההפוך קל באותה מידה.

כמה זיכרון האלגוריתם לוקח?

נסמן ב-$s(t)$ את שטח העבודה ש-$\text{Reach}\left(v_1,v_2;t\right)$ צורכת.

$s(1)=O(\log n)$, $s(t)=s\left(\frac{t}{2}\right)+O(\log n)$ (כי יש קריאה רקורסיבית אחת בכל פעם על הזיכרון).

אזי ממערכת המשוואות הרקורסיביות: $s(t)=\log t\cdot O(\log n)=O(\log t\cdot\log n)$.

ובפרט, אנחנו מריצים $\text{Reach}(s,t;n)$, ובפרט אנחנו נצרוך $O\left(\log^2 n\right)$.

מש"ל.

כמה זמן האלגוריתם לוקח?

$n^{O(\log n)}$. למה?

נסמן ב-$T(t)$ את זמן הריצה של $\text{Reach}\left(v_1,v_2;t\right)$. אזי נקבל את המערכת הבאה:

$\begin{cases}T(1)=O(1)\\T(t)=n\cdot 2\,T\left(\frac{t}{2}\right)\end{cases}$. לכן, $T(t)\geq n^{\log t}$, ובפרט $T(n)\geq n^{\log n}=2^{\log^2 n}$.

נזכר, שכבר הוכחנו שכל אלגוריתם הרץ בזיכרון $s(n)\geq\log n$, רץ בזמן $\geqslant$ $2^{O(s(n))}$.

שפה TQBF

קלט: $\varphi\left(x_1,\ldots,x_n\right)$.

הקלט בשפה $\Leftrightarrow$ $\exists x_1.\forall x_2.\exists x_3.\ldots.$ ערך הפסוק φ הוא True .

אלגוריתם: $\text{TQBF}\in\text{EXP}$

נעשה טבלה: לכל $x_1,\ldots,x_n$ נכתוב את $\varphi\left(x_1,\ldots,x_n\right)$.

וננסה $x_1=\text{False}$, $b_0=\text{Accept}\left(\varphi_{x_1=\text{False}};x_2,\ldots,x_n\right)$.

וננסה $x_1=\text{True}$, $b_1=\text{Accept}\left(\varphi_{x_1=\text{True}};x_2,\ldots,x_n\right)$.

אם אנחנו בכמת קיים, נעשה $b_0\vee b_1$, אם אנחנו בכמת לכל, נחזיר $b_0\wedge b_1$.

טענה: $\text{TQBF}\in\text{PSpace}$.

נתאר אלגוריתם:

וננסה $x_1=\text{False}$, $b_0=\text{Accept}\left(\varphi_{x_1=\text{False}};x_2,\ldots,x_n\right)$.

וננסה $x_1=\text{True}$, $b_1=\text{Accept}\left(\varphi_{x_1=\text{True}};x_2,\ldots,x_n\right)$.

אם הגענו למקרה בסיס $n=0$, נחשב את הפרדיקט.

נחשב את סיבוכיות הזיכרון:
$s(0)=O(n)$.
צריכים לחשב השמה נתונה בתוך פסוק בגודל לכל היותר n . לכן, $s(n)=s(n-1)+O(n)$.
לכן, $s(n)=O(n^2)$.

<u>משפט</u>: TQBF שלמה ב- PSpace .
צ"ל: לכל $A\in$PSpace , $A\leq_{\text{Log}}$ TEBF . נוכיח:
נתון אלגוריתם $A\in$Pspace הנפתר על ידי מ"ט M . נתון קלט $x\in\{0,1\}^n$.
נסתכל על גרף הקונפיגורציות של M על הקלט x . אזי $x\in A$ $\Leftrightarrow$ $\text{Reach}(s,t,|V|)$, כאשר s הוא המצב התחילי, t הוא המצב המקבל, ו- $|V|$ הוא גודל גרף הקונפיגורציות - $2^{O(s(n))}$, ונתון כי $s(n)=\text{Poly}(n)$.

באופן כללי, $\text{Reach}(c_1,c_2,t)$ $\Leftrightarrow$ $\exists c_3.\text{Reach}\left(c_1,c_3,\frac{t}{2}\right)\wedge\text{Reach}\left(c_3,c_2,\frac{t}{2}\right)$.
אם נפתח את הפסוק רקורסיבית, נקבל פסוק עם כמתי $\exists$ שאורכו t . אנחנו מכוונים ל- $t=|V|=2^{s(n)}$.

$$\exists c_3\in V.\forall b\in\{0,1\}.\exists A,B\in V.\begin{array}{l}(b=0)\rightarrow(A=c_1\wedge B=c_3)\\ \wedge(b=1)\rightarrow(A=c_3\wedge B=c_3)\\ \wedge\text{Reach}\left(A,B,\frac{t}{2}\right)\end{array}\quad\Leftrightarrow\quad\text{Reach}(c_1,c_2,t)$$

נפתח את $\text{Reach}(s,t,|V|)$ לפי הנוסחה הרקורסיבית.
אורך הנוסחה הוא $\ell(m)$.
אזי $\ell(m)=O(\log|V|)+\ell(m-1)=O(m\cdot\log|V|)=O(m\cdot S(n))\underset{\substack{m\leq n\\ s(n)=\text{Ploy}(n)}}{=}\text{Poly}(n)$.
מש"ל.

<u>אקראיות</u>

אלגוריתם לבדיקת $AB=C$ $(Freivalds)$

נתבונן בבעיה הבאה: נתונות מטריצות A,B,C בגודל $n\times n$, ושואלים האם $AB=C$ (עם כניסות מהשדה $\mathbb{Q}$ או $\mathbb{Z}_2$).
<u>פתרון נאיבי</u>: להכפיל את A ב- B ולבדוק אם קיבלנו את C .
<u>זמן הריצה</u>: בצורה נאיבית, $O(n^3)$. אבל אפשר יותר טוב: $O(n^{2.81})$ (בעזרת אלגוריתם של *Strassen*). הכי טוב שידוע: $O(n^{2.37})$ (*Coppersmith-Winograd*).
<u>אלגוריתם של *Freivalds*</u>:
נגריל וקטור x בצורה אחידה מהקבוצה $\{0,1\}^n$. נחשב את Cx ואת $A(Bx)$, ונקבל אם ורק אם הם שווים.
<u>זמן הריצה</u>: $O(n^2)$ פעולות (כי קודם נחשב את Bx ואז את $A(Bx)$).
<u>ניתוח</u>:
אם $AB=C$, אזי האלגוריתם מקבל בסיכוי 1 .
נניח עכשיו ש- $AB\neq C$. אזי האלגוריתם מקבל (כלומר טועה) אם ורק אם $ABx=Cx$, או בצורה שקולה $\underbrace{(AB-C)}_{D}x=0$ (כאשר $D\neq 0$).
<u>טענה</u>: לכל מטריצה $D\neq 0$, אם $x\in\{0,1\}^n$ נבחר בצורה אחידה, אזי $\Pr(Dx=0)\leq 0.5$.
מכיוון ש- $D\neq 0$, קיימת עמודה שאינה 0 . נניח בה"כ שזו העמודה הימנית ביותר.
נשים לב שלכל $x_1,\ldots,x_{n-1}\in\{0,1\}$, מתקיים שלכל היותר אחד מהווקטורים הבאים הוא 0 : $D\begin{pmatrix}x_1\\ \vdots\\ x_{n-1}\\ 0\end{pmatrix}$, $D\begin{pmatrix}x_1\\ \vdots\\ x_{n-1}\\ 1\end{pmatrix}$. לכן, לכל היותר 2^{n-1} מחרוזות x יקיימו $Dx=0$, מה שמוכיח את הטענה.

האלגוריתם יכול להגיד שיש שוויון למרות ש-$AB\neq C$. ניתן לשפר את זה על ידי אמפליפיקציה, כלומר חזרה k פעמים על האלגוריתם. אם בלפחות אחת מהפעמים האלגוריתם אמר "שונים", אז נוציא "שונים".

אם $AB=C$, אזי האלגוריתם המוגבר תמיד מקבל, כמו קודם.

אם $AB\neq C$, אזי הסיכוי שהאלגוריתם טועה הוא לכל היותר 2^{-k}. ניקח למשל $k=1000$, וזה בעצם לא יקרה אף פעם.

<u>זמן הריצה</u>: $O\left(n^2\right)$.

בעיית השוויון EQ בסיבוכיות תקשורת

לאליס יש $x\in\{0,1\}^n$, ולבוב $y\in\{0,1\}^n$, והם רוצים להחליט האם $x=y$ במינימום תקשורת.

<u>פתרון נאיבי</u>: אליס שולחת לבוב את x. אבל זה דורש n ביטים של תקשורת.

<u>עוד פתרון נאיבי</u>:

אליס בוחרת i באקראי בצורה אחידה מתוך $\{1,\ldots,n\}$, ושולחת לבוב את i ואת x_i. בוב מקבל אם ורק אם $x_i=y_i$.

דורש בערך $O(\log n)$ תקשורת.

אבל ייתכן ש-x ו-y נבדלים רק בביט אחד, והסיכוי שנתפוס את זה הוא רק $\frac{1}{n}$.

אם נחזור n פעמים על התהליך, התקשורת תהיה $O(n\log n)$, ובעצם אין בזה טעם, כי לא שיפרנו כלום.

דרך לפתור את זה, היא על ידי שימוש בקוד מתקן שגיאות. זה בעצם פונקציה $E:\{0,1\}^n\to\{0,1\}^{2n}$, עם התכונה שלכל $x\neq y$, שניהם באורך n, מתקיים ש-$E(x)$ שונה מ-$E(y)$ בלפחות 10% מהביטים.

עכשיו אפשר לשלוח מספר קבוע של ביטים אקראיים מ-$E(x)$, ולבדוק אם הם שווים לאלו של $E(y)$. זה מוביל לפרוטוקול עם סיכוי שגיאה קטן, ותקשורת של $O(\log n)$.

<u>פתרון אחר</u>:

יהיו $p_1=2,p_2=3,p_3=5,\ldots,p_{n^2}$ n^2 המספרים הראשוניים הראשונים. נחשוב על x ועל y כעל מספרים בקבוצה $\{0,\ldots,2^n-1\}$.

<u>הפרוטוקול</u>:

אליס בוחרת $i\in\{1,\ldots,n^2\}$ בצורה אחידה, ושולחת לבוב את $x \bmod p_i$ ואת i. בוב מקבל אם ורק אם $x \bmod p_i=y \bmod p_i$.

<u>התקשורת</u>: $\underbrace{2\log n}_{\text{עבור } i}+\underbrace{\log p_{n^2}}_{\approx 2\log n}=O(\log n)$.

<u>נכונות</u>:

אם $x=y$, אזי האלגוריתם תמיד מקבל.

נניח ש-$x\neq y$. הפרוטוקול טועה אם ורק אם $x \bmod p_i=y \bmod p_i$, או בצורה שקולה, $p_i|x-y$, כלומר p_i מחלק את $x-y$. מכיוון ש-$|x-y|\leq 2^n$, מספר המחלקים הראשוניים שלו הוא לכל היותר n (כי מכפלה של n מספרים ראשוניים שונים היא גדולה מ-2^n). כלומר, לכל היותר n מהאינדקסים i הם "בעיתיים". לכן, הסיכוי לשגיאה הוא לכל היותר $\frac{n}{n^2}=\frac{1}{n}$.