סיבוכיות – רשימות מהשיעורים

שיעור 1 – 23.02.2010

<u>מכונות טיורינג</u>

הוכחת המשפט $P \neq EXPTIME$:

נגדיר: $L=\left\{x \middle| x=\langle M\rangle \#1^c \#1^k, \text{M does not accept } x \text{ within } c|x|^k \text{ time}\right\}$. נוכיח כי $L \in EXPTIME \setminus P$.

$L \in EXPTIME$:

נבנה מ"ט שתכריע את L בזמן אקספוננציאלי:

1. קלט: $x=\langle M\rangle \#\underbrace{1^c}_{c}\#\underbrace{1^e}_{e}\#$.
2. נריץ את $U\left(x, M, c\cdot|x|^e\right)$ (מ"ט אוניברסלית שמקבלת את האלגוריתם של M , מריצה עליו את הקלט x לכל היותר $c\cdot|x|^e$ צעדים).
3. נקבל תשובה מ-U :
   1. אם לא עצרנו במסגרת הזמן הנתון – נקבל.
   2. אם עצרנו וקיבלנו – נדחה.
   3. אם עצרנו ודחינו – נקבל.

זמן הריצה של המ"ט שבנינו:

שלב 1 הוא פולינומי.

שלב 2 הוא מספר הצעדים של מ"ט אוניברסלית = (מס' צעדים מסומלצים)^2. לכן, הזמן הוא n^n , כי $|x|=n$ ו-$c, e \leq n$.

$L \notin P$:

נניח בשלילה ש-M פותרת את L , $M \in Time\left(c\cdot n^k\right)$.

נבנה $x=\langle M\rangle \#1^c \#1^k \#$.

כעת, לכל x : בגלל ש-M רצה בגבולות הזמן המותר, האלגוריתם שלנו יסמלץ אותה עד הסוף, ולכן יענה ההפך ממנה. לכן, $x \in L \Rightarrow M(x)=0$ ו- $x \notin L \Rightarrow M(x)=1$. סתירה!

מש"ל.

הגדרה

$f:\mathbb{N}\to\mathbb{N}$ היא ***time-constructable*** אם קיימת מ"ט M כך ש:

1. $M \in \text{Time}\left(f(n)\right)$.
2. $M(x)=\left\lfloor f(|x|)\right\rfloor$.

<u>סיבוכיות זיכרון</u>

דוגמה לסיבוכיות זיכרון: כפל מטריצות.

קלט: $A, B \in \{0,1\}^{m\times m}$.

פלט: $A\cdot B$.

אלגוריתם בפסאודו קוד לפתרון הבעיה:

```
for i = 1,...,m
  for j = 1,...,m
    result = 0
    for k = 1,...,m
      result = result + A_ik + B_kj
     end
    output result
    end
  end
```

זיכרון: $4\log m$, כי יש 4 משתנים, כל אחד מהם מכיל ערך בין 1 ל-m .

לכן, בעיה זו היא ב-L .

שיעור 2 – 02.03.2010

משפט:
אפשר לתרגם מ"ט עם k סרטים שרצה בזמן $T(n)$ למ"ט עם סרט אחד שרצה בזמן $O(k\cdot T^2(n))$.
<u>החסם הדוק</u>: $L=\{x0^n x \mid x\in\{0,1\}^n\}$. יש לה מ"ט עם 2 סרטים וזמן לינארי.

טענה:
כל מ"ט עם סרט אחד דורשת זמן $\Omega(n^2)$.

אלגוריתמים ידועים
- אלגוריתם זרימה – משנות ה-50.
- אלגוריתם *Edmonds* – משנת 65.
- *Minimum Spanning Tree* – סיבוכיות זמן של האלגוריתם הכי טוב שידוע היום – כמעט לינארי ב-RAM.
- סיבוכיות כפל שתי מטריצות עם האלגוריתם הטריביאלי:
  - $O(m^3)$ במודל RAM – גישה ישירה לקלט.
  - $O(m^5)$ על מ"ט: m^3 צעדי גישה לזיכרון, וכל גישה עלולה לקחת $O(m^2)$ צעדים.

<u>גרף הקונפיגורציות</u>
נתונה מכונה דטרמיניסטית M, קלט $x\in\{0,1\}^n$. אזי $G_x=(V,E)$, כאשר הקודקודים הם כל הקונפיגורציות האפשריות, והקשתות מכוונות לפי המעברים האפשריים בין הקונפיגורציות.
<u>מסקנה</u>: מכיוון ש-M עוצרת על x, היא לא נכנסת למעגל, ולכן עוצרת תוך $|V|$ צעדים. לכן, $\text{SPACE}(S(n))\subseteq\text{Time}\underbrace{\left(2^{O(s(n))}\cdot s(n)\cdot n\right)}_{\text{חסם על }|V|}$.

PSPACE סגורה תחת רדוקציות פולינומיות:
נניח $L\in\text{PSPACE}$, ו-$L'\leq_p L$. נפתור את $L'\in\text{PSPACE}$:
בהינתן קלט x:
1. נריץ רדוקציה $\varphi(x)$.
2. נבדוק $\varphi(x)\stackrel{?}{\in}L$.

מש"ל.

משפט:
LOG סגורה תחת רדוקציות LSPACE.
<u>טענה</u>: אם $L\in\text{LOG}$ נפתרת על ידי מ"ט $M\in\text{LOG}$ ו-$L'\leq_{\text{LOG}}L$ רדוקציה $f\in\text{LOG}$, אזי $L'\in\text{LOG}$.
<u>אלגוריתם</u>: קלט $x\in\{0,1\}^n$.
1. נחשב את $f(x)$.
2. נריץ את M על $f(x)$.

האלגוריתם טוב: $x\in L' \Leftrightarrow f(x)\in L \Leftrightarrow M(f(x))$ מקבלת.
<u>הבעיה</u>: זה אלגוריתם יקר, כי אין לנו איפה לרשום את $f(x)$.
<u>הפתרון</u>: נשנה את האלגוריתם.
<u>האלגוריתם החדש</u>:
1. נריץ את M כאילו יש לנו את הקלט $f(x)$.
2. כל פעם ש-M מבקשת קלט מ-$f(x)$, נניח את הביט ה-i, נריץ מחדש את f על x, עם ביטים שיוצאים לסרט פלט, ונספור עד שנגיע לביט ה-i, ניקח אותו ונחזיר להרצה של M.

<u>סיבוכיות הזיכרון</u>:
1. סיבוכיות של M: $O\left(\log\underbrace{|f(x)|}_{\text{Poly}(n)}\right)=O(\log n)$.
2. סיבוכיות הזכרון של f: $O(\log n)$.
3. אדמיניסטרציה: $O(\log n)$.

<u>סיבוכיות זמן</u>: ממילא $2^{O(s)}=\text{Poly}(n)$.

הגדרה:
שפה L היא **שלמה** במחלקה C אם $L\in C$ וכל $L'\in C$ מקיימת $L'\le_p L$.

משפט *Cook/Levin*
SAT היא NP-שלמה.
צ"ל: לכל שפה $L\in$NP שנפתרת על ידי $M(x,u)$ יש רדוקציה $f_M:\{0,1\}^*\to\{0,1\}^*$ כך ש-$f\in P$ ו-$x\in L\Leftrightarrow f(x)\in$SAT.
כלומר, צ"ל שיש רדוקציה $f_M:x\to\varphi_{M,x}$ כך ש-$x\in L$ $\Leftrightarrow$ $\varphi_{M,x}$ ספיק.
בהינתן קלט x, סט המשתנים של הפסוק: $u_1,\ldots,u_{p(n)}$, ו-$\forall i,j,s.x_{i,j,s}$. נבנה את הפסוק של טבלת החישוב.
אם $x\in L$ אזי קיים u כך ש-$M(x,u)$=1, ואז קיימת השמה למשתנים $x_{i,j,s}$ שתספק את הפסוק.
אם קיימת השמה ל-u ו-$x_{i,j,s}$ שמספרת את הפסוק, אזי $M(x,u)$=1, ולכן $x\in L$.

שיעור 3 – 09.03.2010

<u>משפט</u>: SAT היא NP-שלמה.

צריך להראות:

1. $SAT \in NP$.
2. SAT היא NP-קשה.

צריך להראות: לכל שפה $L \in NP$ מתקיים $L \leq_p SAT$.

נקבע $L \in NP$. יש מ"ט $M \in P$ (M מ"ט דטרמיניסטית שרצה בזמן פולינימי), ו- $x \in L \Leftrightarrow \exists y . M(x,y)=1, |y|=poly(|x|)$.

אנחנו צריכים להראות פונקציה $f:\{0,1\}^* \to \{0,1\}^*$ כך ש:

1. $x \in L \Leftrightarrow f(x) \in SAT$.
2. $f \in P$.

הרדוקציה:

f ידעת את מ"ט M , עובדת על קלט $x \in \{0,1\}^*$ ותבנה פסוק SAT , שנקרא לו $\varphi_{M,x}$.

המשתנים של $\varphi_{M,x}$: $x_{i,j,s}$ לכל $1 \leq i \leq s(n)$ ו- $1 \leq j \leq T(n)$ (אנחנו נפרש: $x_{i,j,s}$ בתור בזמן j במקום i יש תוכן s). לכל i,j מוגדר s יחיד שרק עבורו $x_{i,j,s}$=true . s כולל גם אם הראש הקורא נמצא על התו i , אם כן, מה המצב של המכונה, ובכל מקרה, מה כתוב בסרט העבודה בתא. בסך הכל, $O(1)$ אפשרויות.

הפסוק: $\left(\bigvee_{s \in S} x_{i,j,s}\right) \wedge \left(\bigwedge_{\substack{s_1,s_2 \\ s_1 \neq s_2}} \neg\left(x_{i,j,s_1} \wedge x_{i,j,s_2}\right)\right)$ - $O(s(n) \cdot T(n))$ פסוקים (כי כותבים פסוק כזה לכל i,j) – פולינומי.

מצב תחילי נכון: $x_{1,0,x_1} \vee x_{2,0,x_2} \vee \ldots$.

מצב בסוף מקבל: $\bigvee_{i=1,\ldots,s(n)} x_{i,T(n),s_{acc}}$.

מצב רגיל: $\bigvee_{(s_1,s_2,s_3,s_4) \in Valid} x_{i,j,s_1} \wedge x_{i-1,j-1,s_2} \wedge x_{i,j-1,s_3} \wedge x_{i+1,j-1,s_4}$ לכל i,j .

$\wedge$ של כל זה, זה הפסוק $\varphi_{M,x}$.

<u>טענה א'</u>: אם $x \in L$ (כלומר $\exists w . M(x,w)=1$) אזי $\varphi_{M,x}$ ספיק.

$x_{i,0,s}$ - השמה שמשקפת ראש קורא על התא השמאלי וקלט $x_1,\ldots,x_n$ (עבור $1 \leq i \leq n$).

$x_{\substack{n+1 \\ \vdots \\ n+m},0,s}$ - השמה שמשקפת את העד.

עבור $x_{i,j,s}$ ($j>0$), נבחר את ההשמה שמתאימה לערך של התא ה-i בזמן t בטבלת החישוב של M על הקלט ב- x_1 .

$$x_{i,j,s} = \begin{cases} \text{true} & \begin{array}{c} s \text{ משקף את הערך} \\ \text{של התא ה} - i \\ \text{בזמן } t \text{ בטבלת} \\ \text{החישוב} \end{array} \\ \text{false} & \text{אחרת} \end{cases} .$$

בגלל ש-M מקבלת את x,w , אזי הפסוק שאומר שבזמן $t(n)$ עוצרים במצב מקבל – יסתפק, ולכן $\varphi_{M,x}$ ספיק (מצאנו לו השמה מספקת).

<u>טענה ב'</u>: אם $\varphi_{M,x}$ ספיק, אזי $x \in L$.

נבחר את w להיות מה שיוצא מהמשתנים $x_{n+1,0},\ldots,x_{n+m,0}$.

<u>טענה</u>: (באינדוקציה) M מקבלת את (x,w) , כלומר $x_{i,j,s}$ מתאים למצב של המכונה M על הקלט ב- x_1 , בזמן j ובתא i .

לכן, M מקבלת את x,w , ולכן $x \in L$.

לכן: $x \in L \Leftrightarrow \exists w . M(x,w)=1 \Leftrightarrow \varphi_{M,x}$ ספיק $\Leftrightarrow \varphi_{M,x} \in SAT$.

סיבוכיות מקום:

נכתוב אלגוריתם Log שבהינתן M,x מוציא את הפסוק $\varphi_{M,x}$:

לכל i,j , פשוט פולטים את הפסוקים למדפסת.

בזיכרון יש רק את i,j שהם בגודל לוגריתמי.

מסקנה: לכל שפה $L\in NP$, $L\le_{Log} SAT$.

פסוקית: $\ell_1\vee\ldots\vee\ell_k$.

פסוק CNF : $\bigwedge \underbrace{c_i}_{\text{פסוקית}}$.

פסוק הוא 3SAT אם כל פסוקית v היא על לכל היותר 3 ליטרלים.

השפה 3SAT

קלט: פסוק 3SAT φ .

פלט: האם הפסוק ספיק?

ברור ש- $3SAT\in NP$.

אפשרות א': להראות שכל $L\in NP$ מקיימת $L\le_{Log} 3SAT$.

אפשרות ב': נראה $SAT\le_{Log} 3SAT$.

למה זה מספיק? לכל $A\in NP$: $A\le_{Log} SAT\le_{Log} 3SAT$.

מסקנה: אם $A\in NP$ ו- B היא NP -קשה, ו- $A\le B$, אזי גם A היא NP -קשה.

רדוקציה $SAT\le_{Log} 3SAT$ בבית.

Clique

קלט: גרף $G=(V,E)$, ומספר k ($1\le k\le|V|$).

פלט: האם יש ב- G קליקה בגודל k ?

נראה $3SAT\le Clique$:

נבנה רדוקציה. $f:\{0,1\}^*\to\{0,1\}^*$. צריך: $x\in 3SAT \Leftrightarrow f(x)\in Clique$.

בהינתן קלט פסוק 3SAT $c_1\wedge c_2\wedge\ldots\wedge c_m$ כאשר $c_i=\ell_{i,1}\vee\ell_{i,2}\vee\ell_{i,3}$. נשים לב שלכל c_i יש רק השמה אחת לא מספקת.

הקודקודים:

לכל פסוקית c_i על b משתנים ($b\in\{1,2,3\}$) נשים 2^b-1 קודקודים בגרף, ונשייך אותם ל- 2^b-1 ההשמות המספקות של c_i .

הצלעות:

נחבר 2 קודקודים בקשת, אם שתי תתי-ההשמות עקביות, כלומר לא סותרות.

$k=m$ – מספר הפסוקיות.

נכונות:

טענה א': אם $x\in 3SAT$ אזי ב- $G=(V,E)$ יש קליקה בגודל m .

$x\in 3SAT$. לכן, יש לו השמה מספקת $a_1,\ldots,a_n$. מבין הקודקודים ששייכים ל- c_i נבחר את הקודקוד שעקבי עם ההשמה a . קיבלנו m קודקודים (אחד לכל פסוקית), והם כולם עקביים, ולכן קיבלנו קליקה בגודל m .

טענה ב': אם יש ב- $f(x)$ קליקה בגודל m , אזי x הוא ספיק.

נבחר השמה שמרחיבה (איחוד) את m תתי ההשמות של הקליקה בגודל m . נקבל השמה שעקבית עם כל m תתי ההשמות, ולכן מספקת את כל m הפסוקיות, ולכן x ספיק.

אלגוריתם Log למימוש הרדוקציה:

לכל $i,j=1,\ldots,m$:

נכתוב את 7 ההשמות המספקות של i .

נכתוב את 7 ההשמות המספקות של j .

נבדוק מי מהן עקביות, ולפי זה נחליט אילו קשתות להוציא.

יש לנו $O(\log m)$ זיכרון, ופולטים את הגרף למדפסת.

יכול להיות ש- $NP=Log$.

מ"ט לא דטרמיניסטית

$\forall q,\sigma.\ \delta(q,\sigma)\subseteq Q\times\{L,R\}\times\Sigma$

למכונה יש סרט קלט, סרט עבודה וסרט פלט (כמו במקרה הדטרמיניסטי).

המכונה מקבלת קלט מסוים אם ורק אם יש מסלול ממצב תחילי למקבל.

סיבוכיות הזיכרון היא הסיבוכיות על המסלול הכי גרוע ממצב תחילי למסיים.

STCON

קלט: גרף *מכוון* $G=(V,E)$, $s,t\in V$.
פלט: האם יש דרך מ- s ל- t ?
בבירור $\text{STCON}\in\text{NP}$ ו- $\text{STCON}\in\text{P}$.
נוכיח ש- $\text{STCON}\in\text{NL}$:

האלגוריתם:
נתחיל ב- s . כל פעם ננחש שכן לעבור אליו, ונספור את מספר הצעדים שניחשנו.
נעצור:
- אם הגענו ל- t (ואז נקבל).
- אם הגענו ל- n צעדים (ואז נדחה).
- אם הגענו לקודקוד בלי יציאה (ואז נדחה).

נכונות:
אם $(G,s,t)\in\text{STCON}$, אזי יש מסלול בין s ו- t ב- G , ואורכו לכל היותר $n-1$. לכן, אם המכונה תנחש לפי המסלול, היא תקבל.
אם $(G,s,t)\notin\text{STCON}$, אזי כל ניחוש מוביל רק לשכנים שקשורים ל- s , ולכן הוא לא מוביל ל- t , ולכן הוא יסתיים בדחייה (כל ניחוש יסתיים בדחייה).

סיבוכיות זיכרון:
אנחנו רק צריכים לשמור את מספר הצעדים שעשינו, ואת הצומת הנוכחי (שניהם ב- $O(\log n)$).

משפט: STCON היא NL -קשה. כלומר, כל שפה $L\in\text{NL}$ מקיימת $L\leq_{\text{Log}}\text{STCON}$.
רדוקציה:
תהי $L\in\text{NL}$, הנפתרת על ידי מ"ט M , ונקבע קלט x .
נבנה גרף $G_{M,x}=(V,E)$ ונגדיר s,t כך ש- $x\in L$ $\Leftrightarrow$ s קשירה ל- t ב- $G_{M,x}$.
הגרף $G_{M,x}$ יהיה גרף הקונפיגורציות של המכונה M על הקלט x .
s תהיה הקונפיגורציה התחילית של M .
t היא הקונפיגורציה המקבלת של M .

נכונות:
$x\in L$ $\Leftrightarrow$ מההגדרה יש מסלול ממצב תחילי למצב מקבל $\Leftrightarrow$ s יהיה קשיר ל- t ב- $G_{M,x}$ $\Leftrightarrow$ $G_{M,x}\in\text{STCON}$.
סיבוכיות הזיכרון של הרדוקציה:
???

מסקנה 1: $\text{NL}\subseteq\text{P}$, בגלל ש- $\text{STCON}\in\text{P}$.
אם $A\in\text{NL}$, אזי $A\leq_{\text{Log}}\text{STCON}$ (כי STCON שלמה ב- NL), ו- $\text{STCON}\in\text{P}$.
טענה 2: $\text{Nlog}\subseteq\text{DSpace}(\log^2 n)$ (משפט *Savitch* משנת 1970). במילים: כל מה שאפשר לפתור עם מכונה לא דטרמיניסטית בזיכרון לוגריתמי, אפשר לפתור במכונה דטרמיניסטית בזיכרון $O(\log^2 n)$.
נראה כי $\text{STCON}\in\text{Space}(\log^2 n)$.
זה מספיק כי אם $A\in\text{NL}$, אזי $A\leq_{\text{L}}\text{STCON}$, ו- $\text{STCON}\in\text{Dspace}(\log^2 n)$, ולכן $A\in\text{Dspace}(\log^2 n)$.
ננית כי $A\leq_{\text{L}}\text{STCON}$ עם f , כלומר $x\in A\Leftrightarrow f(x)\in\text{STCON}$.
בהינתן קלט $x\in A$, היינו רוצים: לחשב את $f(x)$ (יקר מידי) ולהריץ את M על $f(x)$ (אבל $f(x)$ ארוך מידי). בפועל: מריצים את M כאילו יש לה קלט, וכשהיא מבקשת ביט מ- $f(x)$, מריצים את f , ונותנים ל- M את הביט ש- f פלטה.
ואכן: $\text{Reach}(s,t,n)\Leftrightarrow\exists v.\text{Reach}\left(s,v,\frac{n}{2}\right)\wedge\text{Reach}\left(v,t,\frac{n}{2}\right)$. עוברים על כל v אפשרי לקודקוד אמצע.

אם נסמן את סיבוכיות הזיכרון של $\text{Reach}(s,t,k)$ ב- $S(k)$, אזי $S(k)=\underbrace{\log k}_{\text{בשביל }v}+S\left(\frac{k}{2}\right)$.
$S(1)=O(1)$, $S(n)=O(\log^2 n)$.

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 $\frac{n}{n^2}=\frac{1}{n}$.

שיעור 5 – 23.03.2010

TQBF

השפה שמכילה נוסחאות מכומתות $\exists x_1 . \forall x_2 .... \varphi(x_1,...,x_m)$ שערכן T .

ראינו $\text{TQBF} \in \text{PSPACE}$.

<u>משפט</u>: TQBF היא PSPACE -שלמה.

תהי $A \in \text{PSPACE}$. צריכים להראות $A \leq \text{TQBF}$.

לכל קלט $x \in \{0,1\}^n$, צריכים לבנות פסוק TQBF $\varphi(x)$ כך ש- $x \in A \Leftrightarrow \varphi(x) \in \text{TQBF}$.

$A \in \text{PSPACE}$ נפתרת על ידי מ"ט M . נסתכל על גרף הקונפיגורציות של M, x . $x \in A$ $\Leftrightarrow$ בגרף הקונפיגורציות יש מסלול מהמצב התחילי למצב המקבל. $|V| = 2^{\text{poly}(n)}$, ולכן $m = \log|V| = \text{poly}(n)$.

נסמן: $\text{Reach}(x_1, x_2, t)$ - אפשר להגיע מהקונפיגורציה c_1 לקונפיגורציה c_2 בגרף הקונפיגורציות M, x תוך 2^t צעדים. אזי $x \in A \Leftrightarrow \text{Reach}(c_{init}, c_{accept}, \log|V|)$.

נכתוב פסוק (בלי כמתים): $\text{Reach}(c_1, c_2, 0)$ - שומר שאפשר לעבור מ- c_1 ל- c_2 עבור המכונה M והקלט x . אורך הפסוק: $O(m)$.

נגדיר כעת: $\text{Reach}(c_1, c_2, t) = \exists c . \forall b \in \{0,1\} . \exists A, B . \begin{array}{l} (b=0) \to (A=c_1) \wedge (B=c) \\ \wedge (b=1) \to (A=c) \wedge (B=c_2) \\ \wedge \text{Reach}(A, B, t-1) \end{array}$.

נסמן: $\ell(t)$ - אורך הפסוק עבור t .

אזי: $\begin{cases} \ell(0) = O(m) \\ \ell(t) = O(m) + \ell(t-1) \end{cases}$. נקבל: $\ell(t) = O(t \cdot m)$.

לכן, $\ell(m) = O(m^2) = \text{poly}(n)$.

<u>הערה</u>: אם היינו בוחרים בצורה $\text{Reach}(c_1, c_2, t) = \exists c . \text{Reach}(c_1, c, t-1) \wedge \text{Reach}(c, c_2, t-1)$, אזי היינו מקבלים $\ell(t) = 2\ell(t-1)$, ובסוף $\ell(t) = O(2^m)$.

<u>נכונות</u>:

אם $\text{Reach}(c_1, c_2, t)$, אזי יש קודקוד ביניים c .

עבור $b=0$ נבחר $A=c_1$, $B=c$, ואז $\text{Reach}(c_1, c, t-1)$.

עבור $b=1$ נבחר $A=c$, $B=c_2$, ואז $\text{Reach}(c, c_2, t-1)$.

לכן, הפסוק שאנו יוצרים הוא T .

מה הסיבוכיות של הרדוקציה?

$$\begin{array}{c} \text{Reach}(c_{init}, c_{acc}, m) \\ \text{Reach}(A, B, m-1) \\ \vdots \\ \text{Reach}(A_m, B_m, 0) \end{array}$$

<u>הגדרה</u>: $\mathbf{PSPACE}^{\text{TQBF}}$ - כל השפות L שאפשר לפתור על ידי מ"ט M שרצה ב- PSPACE , ומבצעת קריאות אורקל לשפה TQBF .

<u>טענה</u>: $\text{PSPACE}^{\text{TQBF}} = \text{PSPACE}$.

$\text{PSPACE} \subseteq \text{PSPACE}^{\text{TQBF}}$ כי לא חייבים לעשות קריאות אורקל.

$\text{PSPACE}^{\text{TQBF}} \subseteq \text{PSPACE}$ כי אם M^{TQBF} פותרת שפה A , אזי נבנה M' שתריץ את M , ובכל פעם שיש קריאה ל- TQBF , תפתור אותו בעצמה עם אלגוריתם PSPACE שיש לה.

השאילתה שיוצרה: אורכה $\text{poly}(n)$, ולכן נפתור את בעיית ה- TQBF בזיכרון $\text{poly}(\text{poly}(n))$.

<u>עוד שאלה</u>: $\text{P}^{\text{TQBF}} = ?$

לפחות P .

הוא מכיל את השפה TQBF . לכן, גם את כל PSPACE :

לכל $M \in \text{PSPACE}$, $M \leq \text{TQBF}$, ולכן $\forall x . \varphi(x) \in \text{TQBF} \Leftrightarrow x \in M$.

<u>מסקנה</u>: לא נצליח להפריד בין P ל- PSPACE על ידי טיעוני לכסון בלבד.

היינו מוצאים איזה ייצוג מתוחכם של מ"ט. היינו מוצאים איזו שפה ב- PSPACE כך שאם הייתה ב- P , אז היה מספיק זמן לסימולציה והיינו הופכים את עצמנו.

טיעון כזה היה מראה $\text{PSPACE} \not\subseteq \text{P}$.

באותו מחיר היה מראה שלכל אורקל O , $\mathrm{PSPACE}^O \not\subseteq \mathrm{P}^O$.
מצד שני, אנחנו יודעים שיש אורקל O ($O=\mathrm{TQBF}$) שעבורו $\mathrm{P}^O=\mathrm{PSPACE}^O$:
$\forall O.\ \mathrm{Time}^O\left(n^2\right)\not\subseteq \mathrm{Time}^O\left(n\right)$.
מכיוון ש- TQBF היא NPSPACE -שלמה, ו- $\mathrm{TQBF}\in\mathrm{DPSPACE}$, אזי $\mathrm{NPSPACE}=\mathrm{PSPACE}$.
<u>משפט</u>: $\mathrm{NLSPACE}=\mathrm{coNSPACE}$.
<u>משפט</u>: $\overline{\mathrm{stCon}}\in\mathrm{NL}$.
<u>מסקנה</u>: $\mathrm{stCon}\in\mathrm{NL}$.
<u>הגדרה</u>: נגיד שמ"ט לא דטרמיניסטית פותרת שפה A עם **אפס שגיאה** אם:
(מ"ט לא דטרמיניסטית מבצעת התפצלויות ויש לה מצבים מסיימים ומצב מסיים – יכול לקבל, לדחות או לומר "לא יודע")
אנחנו דורשים:
1. לכל קלט x , כל המצבים שאפשר להגיע אליהם מהמצב התחילי הם עקביים (או שכולם "כן" או "לא יודע", או שכולם "לא" או "לא יודע"). אסור שיהיה מצב מסיים שיגיד "כן" ומצב מסיים אחר שיגיד "לא".
2. לכל קלט x , יש מסלול ממצב תחילי למצב מסיים שעונה "כן" או "לא".
3. לכל $x\in L$, כל מי שלא עונה "לא יודע" עונה "כן", ולכן $x\notin L$, כל מי שלא עונה "לא יודע" עונה "לא".

<u>שאלה</u>
נתון גרף מכוון $G=(V,E)$ ו- $s,t\in V$. נסמן: $V_i=\left\{v\middle|\begin{matrix}\text{יש מסלול מ-}s\\ \text{ל-}v\text{ באורך }\leqslant i\end{matrix}\right\}$, $c_i=|V_i|$.
נניח משהו מגלה לנו את i,c_i . האם אנחנו יכולים לייצר מ"ט לא דטרמיניסטית עם אפס שגיאה שתפלוט על סרט הפלט את V_i ?
ננחש v_i .
ננחש מסלול מ- s ל- v_1 באורך לכל היותר i (אם לא הצלחנו – נעצור ונגיד "לא יודע").
נפלוט את v_1 לסרט הפלט.
לכל $j\in\{2,\ldots,c_i\}$:
ננחש קודקוד $v_j>v_{j-1}$ (אם לא הצלחנו – נחזיר "לא יודע").
ננחש מסלול באורך לכל היותר i מ- s ל- v_j (אם לא הצלחנו – נחזיר "לא יודע").
נפלוט את v_j לסרט הפלט.
הגענו למצב מסיים שאומר "יודע".
נכונות:
נניח $V_i=\{a_1,\ldots,a_c\}$, ובייצוג שלנו, $a_1<a_2<\cdots<a_c$.
המכונה שתנחש $v_1=a_1$, ותנחש מסלול באורך לכל היותר i מ- s ל- v_1 (יש כזה כי $a_1\in V_i$), תנחש גם $v_j=a_j$, ותנחש מסלול באורך לכל היותר i מ- s ל- v_j (יש כזה כי $a_j\in V_j$).
סדרת הניחושים הזאת מובילה לענף מקבל שאומר "יודע", ועל סרט הפלט מופיע V_i .
כל מסלול שאומר "יודע" חייב להוציא <u>בדיוק</u> c_i קודקודים <u>שונים</u> לסרט הפלט (הם שונים כי $v_1<v_2<\cdots<v_{c_i}$), וכל קודקוד כזה שייך ל- V_i (כי קודקוד נפלט לסרט הפלט אם ורק אם מצאנו מסלול באורך לכל היותר i מ- s אליו). לכן, כל מסלול שאומר "יודע" מוצא בדיוק את הפלט V_i לסרט הפלט.
מצד שני: אם $v_j\notin V_i$, אזי המסלולים שלא פורשים את V_i יסומנו כ"לא יודע". המסלול שפורש את V_i ויגיע לסוף, אבל לא נראה את v , ולכן נענה "לא".
לכן, אפשר לענות גם על השאלה הבאה באפס שגיאה: בהינתן i,c_i וקודקוד v , האם $v\in V_{i+1}$?
$c_0=1$. נניח שאנחנו יודעים את c_i . האם אפשר לחשב את c_{i+1} באפס שגיאה?
נאתחל $c_{i+1}=0$.
לכל $v\in V$:
נריץ פרוצדורה A שבודקת אם $v\in V_{i+1}$. אם כן, נוסיף 1 ל- c_{i+1} .
<u>מסקנה</u>: יש מ"ט לא דטרמיניסטית ב- NL -אפס-שגיאה שפותרת את stCon .
<u>מסקנה</u>: $\overline{\mathrm{stCon}}\in\mathrm{NL}$.
נריץ את מכונת האפס שגיאה.
עלה שאומר $x\in\overline{\mathrm{stCon}}$ נהפוך למצב מקבל.

עלה שאומר $x \notin \overline{\text{stCon}}$ נהפוך למצב דוחה.
עלה שאומר לא יודע נהפוך למצב דוחה.

<u>מסקנה</u>: $\text{stCon} \in \text{coNL}$.

<u>מסקנה</u>: $\text{NL} \subseteq \text{coNL}$.

תהי $A \in \text{NL}$. אזי $A \leq_L \text{stCon}$ רדוקציה φ .

הרכבה של מכונות Space : $A \leq_L B$ ו- $B \in \text{coNL}$ גורר $A \in \text{coNL}$.

<u>טענה</u>: אם $A \subseteq B$ אזי $\text{co}A \subseteq \text{co}B$.

אם $L \in \text{co}A$ אזי $\bar{L} \in A$, ולכן $\bar{L} \in B$, ולכן $L \in \text{co}B$.

<u>מסקנה</u>: $\text{NL} \subseteq \text{coNL}$, ולכן $\text{NL} = \text{coNL} \subseteq \text{co-coNL}$. לכן, $\text{NL} = \text{coNL}$.

<u>סיכום</u>

מה זה חשוב? התמקדנו במ"ט.
יש דברים שאי אפשר בכלל לחשב במ"ט (למשל בעיית העצירה).
מ"ט היא מודל חישוב אוניפורמי, כלומר מכונה אחת שיש לה ייצוג בגודל קבוע סותרת את הבעיה לכל אורך קלט.
ניסינו לסווג שפות (ברות הכרעה) לפי סיבוכיות הזמן והמקום שלהן.
הגדרנו גם חישוב לא דטרמיניסטי:

אם $x \in L$ אזי יש מסלול מקבל.

אם $x \notin L$ אזי אין מסלול מקבל, כלומר כל מסלול דוחה.

<u>מסקנה</u>: אם אנחנו רואים מסלול מקבל, אז אנחנו יודעים בוודאות שהתשובה היא כן.
חוסר דטרמיניזם הוא תכונה מתמטית של השפה.
יש פרוצדורה יעילה שיכולה לבדוק עדויות שהקלט בשפה.

שיעור 6 – 13.04.2010

חישוב יעיל: כל שפה שאפשר לפתור על ידי מ"ט **דטרמיניסטית** שרצה בזמן פולינומי.

הגדרה: **מ"ט הסתברותית**:

- Q - סט מצבים.
- q_{init} - מצב התחלתי.
- q_{acc} - מצב מקבל.
- q_{rej} - מצב דוחה.
- סרטים:
  - סרט קלט (כרגיל).
  - סרט עבודה (כרגיל).
  - **סרט מטבעות**: סרט אינסופי, מאותחל במחרוזות אקראיות.

סימון: אם M מ"ט הסתברותית, נסמן ב- $M(x,r)$ את התשובה ש-M נותנת על הקלט x, כשנותנים לה את המטבעות r. בה"כ, האורך של r הוא זמן הריצה של המכונה על הקלט x: $r\in\{0,1\}^{T(n)}$.

הגדרה: נאמר ששפה L שייכת למחלקה $\text{BPP}_{\alpha,\beta}$, $0\le\alpha<\beta\le1$ אם קיימת מ"ט הסתברותית $M(x,r)$ שרצה בזמן פולינומי, ולכל $x\in L$, $\Pr_r[M(x,r)=1]\ge\beta$, ולכל $x\notin L$, $\Pr_r[M(x,r)=1]<\alpha$.

הגדרה:

- $\text{BPP}=\text{BPP}_{\frac{1}{3},\frac{2}{3}}$.
- $\text{RP}=\text{BPP}_{0,\frac{1}{2}}$.

מה הסתברות הקבלה?

	$x\in L$	$x\notin L$
P חישוב דטרמיניסטי	1	0
NP חישוב לא דטרמיניסטי	>0	0
BPP	$\ge\frac{2}{3}$	$\le\frac{1}{3}$
RP	$\ge\frac{1}{2}$	0

דוגמה לאלגוריתם הסתברותי: PM (*Perfect matching*)

הבעיה: האם קיים זיווג מושלם בגרף דו-צדדי?

הקלט: גרף דו-צדדי $G=(V_1,V_2,E)$, $E\subseteq V_1\times V_2$, $|V_1|=|V_2|=n$.

הקלט בשפה $\Leftrightarrow$ יש זיווג מושלם בגרף.

אפשר לפתור בעיה זו עם אלגוריתם זרימה, ולכן היא ב- P.

נראה שהיא ב- RP (למרות שזה ברור כי $\text{P}\subseteq\text{RP}$):

האלגוריתם:

1. בונים מטריצה $B_{n\times n}$, כך ש- $B_{ij}=\begin{cases}0 & (i,j)\notin E\\ x_{ij} & (i,j)\in E\end{cases}$, כאשר x_{ij} משתנה.
2. לכל i,j בוחרים $a_{ij}\in_R\{1,2,\ldots,2n\}$ (באופן אקראי), ואנחנו מחליפים כל x_{ij} ב- a_{ij}. מקבלים מטריצה $A_{n\times n}$.
3. מחשבים את $\det A$. אם היא 0, אזי אין זיווג מושלם. אחרת, יש זיווג מושלם.

נכונות:

$\det B=\sum_{\pi\in S_n}\text{sign}(\pi)\cdot x_{1,\pi(1)}\cdot\ldots\cdot x_{n,\pi(n)}$. כלומר, $\det B$ הוא פולינום במשתנים $x_1,\ldots,x_n$, מולטי-לינארי מדרגה טוטלית לכל היותר n.

משפט (שוורץ-ציפל): אם $p(x_1,\ldots,x_m)$ פולינום ב-m שאינו זהותית 0 משתנים מדרגה טוטלית לכל היותר D מעל שדה F, ואם $S\subseteq F$, אז $\Pr_{a_1,\ldots,a_m\subseteq S}[p(a_1,\ldots,a_m)=0]\le\frac{D}{|S|}$.

נוכיח את המשפט בהמשך.

מצב 1: אין זיווג מושלם בגרף.
לכל π יש i כך ש-$(i,\pi(i))\notin E$, ואז $b_{i,\pi(i)}=0$, ואז המונום נהרג.
לכן, $(\det B)(x_{11},\ldots,x_{nn})\equiv 0$.

מצב 2:
קיימת פרמוטציה $\pi:[n]\to[n]$, כך ש-$(i,\pi(i))\in E$ לכל $i\in[n]$.
לכן, כל פרמוטציה π שמייצגת זיווג מושלם בגרף G, תורמת את מונום המתאים לה: $\operatorname{sign}(\pi)\cdot x_{1,\pi(1)}\cdot\cdots\cdot x_{n,\pi(n)}$.
המונומים לא יכולים לבטל אחד את השני, בגלל שלכל פרמוטציה כזו יהיו משתנים שונים, ולכן הפולינום לא יהיה זהותית 0.

בהינתן $S=\{1,\ldots,2n\}$, לפי שוורץ-ציפל, $\Pr_{a_{11},\ldots,a_{nn}\subseteq S}\left[(\det B)(a_{11},\ldots,a_{nn})\right]\le\frac{n}{|2n|}=\frac{1}{2}$.

לכן: אם אין זיווג מושלם בגרף, $\Pr_r[M(x,r)]=0$. אם יש זיווג מושלם בגרף, $\Pr_r[M(x,r)]\ge\frac{1}{2}$.
לכן, $\mathrm{PM}\in\mathrm{RP}$.
מעבר לזה: יודעים לחשב דטרמיננטה במקביל מהר, ולכן זה נותן אלגוריתם הסתברותי מקבילי ל-PM. לא ידוע אלגוריתם דטרמיניסטי מקבילי מהיר ל-PM.

טענה: $\mathrm{P}\subseteq\mathrm{RP}\subseteq\mathrm{NP}$.
הוכחה:
1. $\mathrm{P}\subseteq\mathrm{RP}$ - כל אלגוריתם דטרמיניסטי הוא גם הסתברותי.
2. $\mathrm{RP}\subseteq\mathrm{NP}$: $L\in\mathrm{RP}$ נפתרת על ידי מ"ט $M(x,r)$. M היא פתרון NP ל-L: אם $x\in L$ אזי $\Pr_r[M(x,r)=1]\ge\frac{1}{2}>0$, כלומר $\exists w.M(x,w)=1$. ואם $x\notin L$ אזי $\Pr_r[M(x,r)=1]=0$, ולכן $\forall w.M(x,w)=0$.

אמפליפיקציה של שגיאה חד-צדדית

טענה: אם $L\in\mathrm{BPP}_{0,\frac{1}{n}}$, אזי $L\in\mathrm{BPP}_{0,1-e^{-n^2}}$.

הוכחה:

תהי $L\in\mathrm{BPP}_{0,\frac{1}{n}}$. אזי יש מ"ט M, $M(x,r)$ פולינומית כך ש-$\Pr_r[M(x,r)=1]\ge\frac{1}{n}$ ל-$x\in L$ ו-$\Pr_r[M(x,r)=1]=0$ ל-$x\notin L$.
נבנה אלגוריתם חכם M' על קלט x: יבחר $\bar r=r_1,\ldots,r_T$ באופן ב"ת, יריץ את $M(x,r_i)$ ל-$i=1,\ldots,T$, ויקבל אם ורק אם אחד מהם קיבל.
אזי אם $x\notin L$, $\Pr_r[M'(x,\bar r)=1]=0$, כי לכל i, $M(x,r_i)=0$. אם $x\in L$, $\Pr_{\bar r}[M'(x,\bar r)=0]=\Pr_{r_1,\ldots,r_T}[\forall i.M(x,r_i)=0]\overset{\text{בלתי תלויים}}{=}\prod_{i=1}^{T}\Pr_{r_i}[M(x,r_i)=0]\le\left(1-\frac{1}{n}\right)^T\approx e^{-\frac{T}{n}}$.
ניקח $T=n^3$, ונקבל $\Pr_{\bar r}[M'(x,\bar r)=0]\le e^{-n^2}$.
מש"ל.

אמפליפיקציה לשגיאה דו-צדדית

טענה: אם $L\in\mathrm{BPP}_{\alpha,\alpha+\frac{1}{n}}$, אזי $L\in\mathrm{BPP}_{2^{-n^2},1-2^{-n^2}}$.

הוכחה:

ניקח $L\in\mathrm{BPP}_{\alpha,\alpha+\frac{1}{n}}$, הנפתרת על ידי מ"ט $M(x,r)$, כך שמקבלת או דוחה בהסתברויות המתאימות.
נבנה מ"ט חדשה M': מטילה $\bar r=r_1,\ldots,r_T$ באופן ב"ת, ומחשבת את $M(x,r_i)$. היא רואה את האחוז β של ההרצות שקיבלו. אם $\beta>\alpha+\frac{1}{2}n$ - תקבל. אחרת, תדחה.

<u>משפט</u> (צ'רנוף): אם $X_1,\ldots,X_n$ מ"מ בינריים ב"ת, ו-$\forall i.E(X_i)=p$ (באופן שקול: $\Pr[X_i=1]=p$ ו-$\Pr[X_i=0]=1-p$), אזי $\Pr\left[\left|\sum_{i=1}^{n}X_i-pn\right|>\delta\cdot(pn)\right]\le e^{\frac{-\delta^2 pn}{4}}$. כלומר, ההסתברות שכשעושים n ניסיונות ב"ת, נסטה מהממוצע יורדת אקספוננציאלית ב-$\delta^2 p$.

נחזור לניתוח של המכונה M'.

נניח $x\in L$. נסמן מ"מ: $X_i=\begin{cases}1 & M(x,r_i)=1\\ 0 & \text{אחרת}\end{cases}$. נקבל מ"מ $X_1,\ldots,X_T$ בינריים, ב"ת, כך ש-$E(X_i)=\Pr[X_i=1]=p\ge\alpha+\frac{1}{n}\ge\alpha+\frac{1}{2n}$. M' טועה עם עונה לא, וזה קורה אם $\sum X_i\le\left(\alpha+\frac{1}{2n}\right)\cdot T=\Pr\left[\sum X_i\le\left(\alpha+\frac{1}{2n}\right)T\right]\le\Pr\left[\left|\sum X_i-p\cdot T\right|>\frac{1}{2n}\cdot T\right]\underset{\text{צ'רנוף}}{\le}2^{-\Omega\left(\frac{T}{n^2}\right)}$.

$pT\ge\left(\alpha+\frac{1}{n}\right)\cdot T$, ולכן $pT-\sum X_i\ge\frac{1}{2n}T$.

ובפרט, $\text{BPP}_{\frac{1}{3},\frac{2}{3}}$, $\text{BPP}_{2^{-n},1-2^{-n}}$ ו-$\text{BPP}_{\frac{1}{10},\frac{1}{10}+\frac{1}{n}}$ כולם מבטאים את אותה מחלקה.

כל עוד הפער לא זניח ($\frac{1}{\text{Poly}(n)}$), אפשר לעשות אמפליפיקציה ולהוריד את השגיאה להיות אקספוננציאלית קטנה.

<u>טענה</u>: $\text{RP}\subseteq\text{BPP}$.

$L\in\text{RP}\Leftarrow L\in\text{BPP}_{0,\frac{1}{2}}\Leftarrow L\in\text{BPP}_{2^{-n},1-2^{-n}}\Leftarrow L\in\text{BPP}_{\frac{1}{3},\frac{2}{3}}$. מש"ל.

נוכיח את משפט צ'רנוף:

אי שוויון מרקוב: אם X מ"מ, $X\ge0$, אזי $\Pr[X\ge A]\le\frac{E(X)}{A}$.

$E(X)=\sum_i i\cdot\Pr[X=i]\ge\sum_{i\ge A}i\cdot\Pr[X=i]\ge A\cdot\sum_{i\ge A}\Pr[X=i]=A\cdot\Pr[X\ge A]$.

משפט: אם $X_1,\ldots,X_n$ מ"מ בינריים ב"ת, $E(X_i)=p$, ואם $\mu=p\cdot n$, $\delta>0$, ו-$X=\sum_{i=1}^{n}X_i$, אזי $\Pr[X\le(1-\delta)\mu]\le e^{-\frac{\delta^2\mu}{2}}$.

$$\Pr[X\le(1-\delta)\mu]=\Pr[-X\ge-(1-\delta)\mu]\overset{\forall t}{=}\Pr[-tX\ge-(1-\delta)t\mu]=$$
$$=\Pr\left[e^{-tX}\ge\underbrace{e^{-(1-\delta)t\mu}}_{A}\right]\overset{\text{מרקוב}}{\le}\frac{E\left(e^{-tX}\right)}{e^{-(1-\delta)+\mu}}$$

באופן כללי: $e^{-tX_i}=e^{-t(X_1+\cdots+X_n)}=\prod e^{-tX_i}$, וכל e^{-tX_i} ב"ת. לכן:

$$\Pr[X(1-\delta)\mu]\le\ldots\le\frac{E\left(e^{-tX}\right)}{e^{-(1-\delta)+\mu}}=\frac{\prod_{i=1}^{n}E\left(e^{-tX_i}\right)}{e^{-(1-\delta)t\mu}}$$

נבחר $t>0$ באופן הבא: $t=\ln\frac{1}{1-\delta}\Leftrightarrow e^{-t}=1-\delta\Leftrightarrow e^{-t}-1=-\delta$.

נתונים לנו $X_1,\ldots,X_n$ מ"מ בינריים, ויכול להיות שהם תלויים. $E(X_i)=p$. שואלים: מה ההסתברות ש-$X=\sum_{i=1}^{n}X_i$ שונה מהותית מהתוחלת $p\cdot n$? אם אין נתונים נוספים על $X_1,\ldots,X_n$, אפשר להשתמש רק במרקוב: $\Pr[X\ge A]\le\frac{E(X)}{A}$. אם אנחנו יודעים שהם לחלוטין ב"ת, אז אפשר להשתמש בצ'רנוף.

הגדרה: $X_1,\ldots,X_n$ **בלתי תלויים בזוגות** אם לכל $i\neq j$, X_i,X_j ב"ת.

דוגמה:

X_3	X_2	X_1	
0	0	0	$\frac{1}{4}$
1	1	0	$\frac{1}{4}$
1	0	1	$\frac{1}{4}$
0	1	1	$\frac{1}{4}$

אזי $\forall i.E(X_i)=\frac{1}{2}$. כל זוג מתפלג אחיד על $\{0,1\}^2$. השלישיה לא אחידה, מחמיצה 4 מחרוזות מ- 8 אפשרויות.

הגדרה: מ"מ $X_1,\ldots,X_n$ הם **ב"ת ב-k -יות** אם לכל $1\leq i_1<\cdots<i_k\leq n$, המ"מ $X_{i_1},\ldots,X_{i_k}$ הם ב"ת.

צ'בישב: אם $X_1,\ldots,X_n$ ב"ת בזוגות, δ קבוע, אזי $\Pr\left[\left|\sum X_i-\mu\right|\geq\delta\cdot\mu\right]\leq\Omega\left(\frac{1}{\mu}\right)$.

צ'רנוף דורש שהם יהיו ב"ת לחלוטין.

שיעור 7 – 27.04.2010

הבעיה GISO

הגדרה:

שני גרפים $G_1=(V_1,E_1)$, $G_2=(V_2,E_2)$ הם **איזומורפיים** אם $|V_1|=|V_2|=n$, ויש פרמוטציה $\pi\in S_n$ כך ש- $\pi(E_1)=\{(\pi(i),\pi(j))|(i,j)\in E_1\}=E_2$.

איפה הבעיה הזאת נמצאת?

EXP - כי אפשר לעבור על כל הפרמוטציות $\pi\in S_n$ (יש $n!\approx 2^{n\log n}$ כאלו), ולבדוק כל אחת.

PSPACE - כי אפשר גם ב- NP .

NP - אפשר לפתור באמצעות עדות של פרמוטציה: $M(G_1,G_2;\pi)=\begin{cases}1 & \pi(G_1)=G_2\\ 0 & \text{אחרת}\end{cases}$.

אם $G_1\sim G_2$, אז מההגדרה יש π כך ש- $\pi(G_1)=G_2$, ולכן יש עד שיגרום לקבל.

אם $G_1\not\sim G_2$, אז מההגדרה, לכל π , $\pi(G_1)\neq G_2$, ולכן π נדחה.

הבעיה $\overline{\text{GISO}}$=GNISO

הגדרת הבעיה:

קלט: שני גרפים $G_1=(V_1,E_1)$, $G_2=(V_2,E_2)$.

פלט: הקלט בשפה $\Leftrightarrow$ $G_1\not\sim G_2$.

לא ידוע האם GNISO$\in$NP .

עד עכשיו, היה לנו מוכיח כל יכול שחישב עדות, שלח אותה לבודק יעיל, והבודק וידא נכונות.

עכשיו: תתנהל שיחה (אינטואיטיבית) בין המוכיח לבודק. המוכיח הוא כל יכול (כמו קודם). הבודק הוא פולינומי הסתברותי. השיחה קצרה: מספר פולינומי של שאלות ותשובות (פולינומי באורך הקלט). המסרים באורך פולינומי. העבודה של הבודק פולינומית. בסוף, הבודק מחליט אם לקבל את לדחות.

רוצים:

אם $x\in L$ אזי $\exists p\text{ מוכיח }\Pr_{v\text{ בודק}}[\text{יקבל}]\geq 2/3$.

אם $x\notin L$ אזי $\forall p.\Pr_v[\text{יקבל}]\leq 1/3$.

דוגמה:

הקלט: G_1,G_2 , $|V|=n$.

פלט רצוי: הקלט בשפה $\Leftrightarrow$ $G_1\not\sim G_2$.

הבודק:

בוחר באקראי $\pi\in S_n$, $b\in\{1,2\}$, ושומר את הבחירות שלו בסוד.

שולח למוכיח את $H=\pi(G_b)$.

המוכיח:

צריך לשלוח $c\in\{1,2\}$.

הבודק:

מקבל $\Leftrightarrow$ $c=b$.

ניתוח:

אם $(G_1,G_2)\in$GNISO , כלומר $G_1\not\sim G_2$, אזי:

המוכיח ההוגן: מקבל H , יודע את G_1,G_2 , ויודע ש- $H\sim G_1$ או $H\sim G_2$, ויודע ש- H איזומורפי לבדיוק אחד מהם. אחרת, $H\sim G_1$, $H\sim G_2$ אזי $G_1\sim G_2$.

המוכיח הכל יכול: מוצא $c\in\{1,2\}$ כך ש- $H\sim G_c$ ולא איזומורפי לגרף השני. שולח את c .

טענה: $b=c$.

$G_b\sim H$ ו- $G_c\sim H$. יודעים שרק אחד מ- G_1,G_2 איזומורפי ל- H , ולכן $b=c$.

לכן, אם $(G_1,G_2)\in$GNISO , יש מוכיח הוגן שעבורו הבודק תמיד יקבל.

אם $(G_1,G_2)\notin$GNISO , כלומר $G_1\sim G_2$, אזי:

לכל מוכיח p כלשהו, $H\sim G_1\sim G_2\sim H$. H איזומורפי לאחד הגרפים, ולכן H איזומורפי לשניהם. יש הסתברות שווה מבחינת המוכיח שהבודק בחר $b=1$ או $b=2$.

הגדרה: (פרוטוקול אינטראקטיבי)
שיחה בין שני שחקנים: בודק v ומוכיח p.
נתון קלט x.
v משחק ראשון. מטיל מטבעות r_1. מחשב $m_1=V(x,r_1)$.
p משחק שני. $m_2=P(x,m_1)$.
v מחשב: $m_i=V\left(x,m_1,\ldots,m_{i-1},r_1,\ldots,r_{\frac{i-1}{2}}\right)$.
p מחשב: $m_{i+1}=P(x,m_1,\ldots,m_i)$
בסוף, אחרי k סיבובים, v מחליט אם לדחות או לקבל: $V(x,m_1,\ldots,m_k,r_1,\ldots,r_k)\in\{0,1\}$.

הגדרה:
מספר סיבובים – k – מספר מסרים שעברו.

הגדרה:
הפרוטוקול הוא ***Public-con*** (בלי סודות) אם כל פעם ש-v שולח m_i, $m_i=r_{\frac{i-1}{2}}$.

הגדרה:
הפרוטוקול הוא **פולינומי** אם מספר הסיבובים פולינומי. כלומר: $\forall i.|r_i|,|m_i|=\text{Poly}(n)$, כאשר n הוא אורך הקלט.

הגדרה:
פרוטוקול אינטראקטיבי מקבל שפה L עם שגיאה (α,β) אם $x\in L\to\exists p.\Pr_{\text{מטבעות } v}[v\text{ מקבל}]\geq\beta$ ו- $x\notin L\to\forall p.\Pr_{\text{מטבעות } v}[v\text{ מקבל}]\leq\alpha$.

הגדרה:
המחלקה $\mathbf{IP}_k[\alpha,\beta]$ היא אוסף כל השפות L כך שיש פרוטוקול אינטראקטיבי פולינומי עם k סיבובים שמקבל את L עם שגיאה (α,β).

הגדרה:
המחלקה $\mathbf{AM}_k[\alpha,\beta]$ היא כמו IP, חוץ מזה שהפרוטוקול צריך להיות בנוסף public-con.

ראינו: $\text{GNISO}\in\text{IP}_2\left[\frac{1}{2},1\right]$.
נראה:
טענה: $\text{GNISO}\in\underbrace{\text{AM}_2\left[2^{-n},1-2^{-n}\right]}_{\text{AM}}$.

הבעיה:
קלט x, מה שהבודק מקבל - y, תשובה שמוכיח נותן - z.
פרדיקט בדיקה $V(x,y,z)$ כאשר $V\in P$, $|x|,|y|,|z|=\text{Poly}(n)$. כעת:
$x\in L\to\Pr_y[\exists z.V(x,y,z)=1]\geq\beta$, $x\notin L\to\Pr_y[\exists z.V(x,y,z)=1]\leq\alpha$.

רעיון ההוכחה:
נסתכל על הקבוצה $T=\{H\mid H\sim G_1\vee H\sim G_2\}$.
אם $G_1\not\sim G_2$ אז $\forall H\in T.H\sim G_1\ xor\ H\sim G_2$.
אם $G_1\sim G_2$ אז $\forall H\in T.H\sim G_1\wedge H\sim G_2$.
מצפים שבמקרה $G_1\not\sim G_2$, הקבוצה T הרבה יותר גדולה מהמקרה בו $G_1\sim G_2$.
נחפש מערכת הוכחה AM לכך שהקבוצה גדולה.
הגדרה: $\mathbf{Aut}(G)=\{\pi\in S_n\mid\pi(G)=G\}$.
נסתכל על הקבוצה $S=\{(H,\tau)\mid(H\sim G_1\vee H\sim G_2)\wedge\tau\in\text{Aut}(H)\}$.
טענה:
אם $G_1\sim G_2$ אזי $|S|=n!$. אם $G_1\not\sim G_2$ אזי $|S|=2n!$.
טענה:
אם ל-G_1 ו-G_2 אין אוטומורפיזמים לא טריביאליים: אם $G_1\sim G_2$ אזי $|T|=n!$. אם $G_1\not\sim G_2$ אזי $|T|=2n!$.
התכונה של S שחשובה: ניתן להוכיח ישירות ב-S במחלקה NP.

בהינתן קלט (H,τ):

1. בעצמנו יכולים לבדוק ש- $\tau\in \mathrm{Aut}(H)$.
2. יכולים לבקש הוכחה ש- $H\sim G_1$ או $H\sim G_2$.

הוכחות אינטראקטיביות

- שיחה בין בודק למוכיח.
- הוכחה אינטראקטיבית:
  - עם מטבעות סודיים של הבודק - IP .
  - בלי סודות (כלומר מטבעות פומביים) – AM או $\mathrm{AM}(k)$.

פרוטוקול מטבעות סודיים ל- GNISO :

קלט: $G_1=G_2$.

$\underline{P}$: טוען ש- $G_1 \nsim G_2$.

$\underline{V}$: בוחר $\pi\in S_n$, $b\in\{1,2\}$ ושולח את $\pi(G_b)$ ל- P ($challenge$).

$\underline{P}$: שולח c ($response$).

$\underline{V}$: בודק ש- $b=c$.

נסמן: $S=\{(H,\tau)\mid (H\sim G_1\vee H\sim G_2)\wedge\tau\in\mathrm{Aut}(H)\}$, כאשר $\mathrm{Aut}(H)=\{\tau\mid\tau(H)=H\}$.

טענה: אם $G_1\sim G_2$ אזי $|S|=n!$ ($|V(G_1)|=|V(G_2)|=n$). אם $G_1\nsim G_2$ אזי $|S|=2n!$.

הוכחה:

נשים לב: $S=\{(H,\tau)\mid H\sim G_1\wedge\tau\in\mathrm{Aut}(H)\}\cup\{(H,\tau)\mid H\sim G_2\wedge\tau\in\mathrm{Aut}(H)\}$.

אם $G_1\nsim G_2$, אז שתי הקבוצות האלה שונות, וכל אחת בגודל $n!$.

לכן, $(G_1,G_2)\in\mathrm{GNISO}$, $|S|=2n!$ $\Leftarrow$ $(G_1,G_2)\notin\mathrm{GNISO}$, $|S|=n!$.

הערה: נשים לב, שבדיקת שייכות לקבוצה S נמצאת במחלקה NP , כי בהינתן קלט (H,τ) שאנחנו רוצים לדעת אם הוא ב- S , צריך לבדוק:

1. $H\sim G_1$ או $H\sim G_2$: נבקש פרמוטציה π ו- b כך ש- $H=\pi(G_b)$. עושים את זה ב- NP .
2. $\tau\in\mathrm{Aut}(H)$: $\tau(H)=H$.

איך בודקים שקבוצה S היא גדולה? $S\subseteq\{0,1\}^N$

במקרה שלנו, $N=\underbrace{n\log n}_{\text{תאור פרמוטציה}}+\underbrace{n^2}_{\substack{\text{תיאור גרף על } n \text{ קודקודים}\\ \text{לכל קשת, בגרף או לא}}}$.

ניסיון 1:

רוצים לדעת $p=\frac{|S|}{2^N}$.

נדגום $x_1,\ldots,x_T$ באופן אקראי מ- $\{0,1\}^N$. נראה כמה מהם שייכים ל- S (בעזרת המוכיח).

נגיד ש- t מתוך T ב- S . נענה $p\sim\frac{t}{T}$.

אבל p נורא קטן: $\frac{2^{n\log n}}{2^{n^2}}\le 2^{\frac{-n^2}{2}}$. לכן, נצטרך מספר אקספוננציאלי של דגימות כדי לראות בכלל איבר מ- S .

ניסיון 2:

נדחס את העולם הגדול לעולם קטן יותר שהקבוצה S לא תפגע, ונדגום בעולם קטן יותר.

הבעיה: כשממפים עולם בגודל 2^N לעולם בגודל 2^m , חייב להיות לפחות איבר אחד בעולם הקטן כך ש- 2^{N-m} איברים נופלים אליו.

הפתרון: Hashing הסתברותי. נבחר משפחה $H=\{h:\{0,1\}^N\to\{0,1\}^m\}$ עם התכונות:

1. לכל $a\in\{0,1\}^N$, $b\in\{0,1\}^m$, $\Pr_{h\in H}[h(a)=b]=2^{-m}$.
1. לכל $a_1\neq a_2\in\{0,1\}^N$, $b_1,b_2\in\{0,1\}^m$, $\Pr_{h\in H}[h(a_1)=b_1\wedge h(a_2)=b_2]=2^{-2m}$.

הפרוטוקול:

קלט: G_1, G_2 כך ש-$|V(G_1)|=|V(G_2)|=n$, $m\in\mathbb{Z}$, $10\,n!\le 2^m\le 20\,n!$.
הבודק בוחר $h\in H$, $b\in\{0,1\}^m$ ושולח h,b למוכיח (מטבעות פומביים).
המוכיח צריך להראות $a\in S$ כך ש-$h(a)=b$. המוכיח שולח את a והוכחה ש-$a\in S$.
הבודק בודק את ההוכחה ש-$a\in S$ ושאכן $h(a)=b$, ואם הוא מרוצה, מקבל. אחרת, דוחה.

המשפחה H :
ניתן להסתכל עליה בתור מטריצה, כך שכל פונקציה היא שורה, והעמודות הן הערכים שמקבל כל אחד מהאיברים הממופים.
דרישה 1: בכל עמודה האיברים מ-$\{0,1\}^N$ מופיעים לפי ההתפלגות האחידה.
דרישה 2: כל זוג עמודות אחיד.

נגדיר מ"מ X_a לכל $a\in\{0,1\}^N$ באופן הבא: נבחר $h\in H$, ואז $X_a=h(a)$. נקבל 2^N מ"מ תלויים.
נראה:
1. כל X_a מתפלג אחיד.
2. ב"ת בזוגות.

הוכחת נכונות של הפרוטוקול:
אם $G_1\sim G_2$ אזי $|S|=n!$. לכל בחירה $h\in H$ ש-V יבחר,
$$\Pr_{b\in\{0,1\}^n}\left[\exists a\in S.\,h(a)=b\right]=\frac{|\mathrm{Im}(S)|}{2^m}\le\frac{|S|}{2^m}=\frac{n!}{2^m}=q$$.
אם $G_1\not\sim G_2$ אזי $|S|=2n!$. לכל $b\in\{0,1\}^m$,
$$\Pr_{h\in H}\left[\exists a\in S.\,h(a)=b\right]\ge\underbrace{\sum_{a\in S}\Pr_{h\in H}\left[h(a)=b\right]}_{\sum\Pr[A_i]}-\underbrace{\sum_{a_1\ne a_2\in S}\Pr\left[h(a_1)=b\wedge h(a_2)=b\right]}_{\sum_{i\ne j}\Pr[A_i\wedge A_j]}=$$
$$=|S|\cdot 2^{-m}-\frac{1}{2}|S|^2\cdot 2^{-2m}=\frac{|S|}{2^m}-\frac{1}{2}\cdot\left(\frac{|S|}{2^m}\right)^2=\frac{2n!}{2^m}-2\cdot\left(\frac{n!}{2^m}\right)^2=$$
$$=2q-2q^2=2q(1-q)\ge 2\cdot\frac{9}{10}q\ge 1.5\,q$$
ונזכור כי $\Pr\left[\bigcup_{i=1}^{k}A_i\right]\ge\sum\Pr[A_i]-\frac{1}{2}\sum_{i\ne j}\Pr[A_i\wedge A_j]$.

בסך הכל: אם $G_1\sim G_2$ אזי לכל מוכיח, $\Pr_{\substack{\text{מטבעות } V\\ h,b}}\left[\begin{matrix}\text{המוכיח ישכנע}\\ \text{את } V\end{matrix}\right]\le q$, ואם $G_1\not\sim G_2$ אזי קיים מוכיח הוגן (כל פעם שיש $a\in S$ כך ש-$h(a)=b$ מראה אותו) שעבורו $\Pr_{\text{מטבעות } V}\left[\begin{matrix}\text{המוכיח ישכנע}\\ \text{את } V\end{matrix}\right]\ge 1.5\,q$. q קבוע בין $\frac{1}{10}$ ל-$\frac{1}{20}$.

לכן, $\mathrm{GNISO}\in\mathrm{AM}_{q,1.5q}$.

אם רוצים פער גדול יותר, $\mathrm{GNISO}\in\mathrm{AM}_{\varepsilon,1-\varepsilon}$, הפרוטוקול:
- קלט G_1, G_2 , $10\,n!\le 2^m\le 20\,n!$.
- V בוחר $T=O\left(\log\frac{1}{\varepsilon}\right)$ זוגות (h_i,b_i) ($h_i\in H$, $b_i\in\{0,1\}^m$), וישלח אותם ל-p .
- p ישלח (אם הוא יכול) a_i כך ש-$h(a_i)=b_i$ והוכחות ש-$a_i\in S$.
- V יקבל אם אחוז ה-a_i הטובים הוא לפחות $1.25\,q$.

כל זה מאוד תאורטי. המוכיחים האלה לא באמת קיימים, ולכן אלו לא אלגוריתמים. למה זה טוב?
השאלה נכונה באותה מידה ל-NP .
לפחות דבר אחד – אם אנחנו חושבים ש-$\mathrm{P}\ne\mathrm{NP}$, ברגע שנראה שבעיה היא NP-קשה (ולכן מייצגת את המחלקה NP), גם נדע שאין לה אלגוריתם ב-P .
אלגוריתם מראה קלות. אנחנו רוצים להראות קושי.

או ש- $NP=P$, ואז כדי להראות שאלגוריתם יעיל, מספיק להראות בודק יעיל שמדבר עם מוכיח כל יכול).
או ש- $NP\neq P$, ואז יש לנו דרך להוכיח קושי.

$P=AM(0)$ - בודק פולינומי.
NP - מוכיח כל יכול ששולח מסר אחד לבודק פולינומי דטרמיניסטי.
MA - מוכיח כל יכול ששולח מסר אחד לבודק הסתברותי פולינומי.

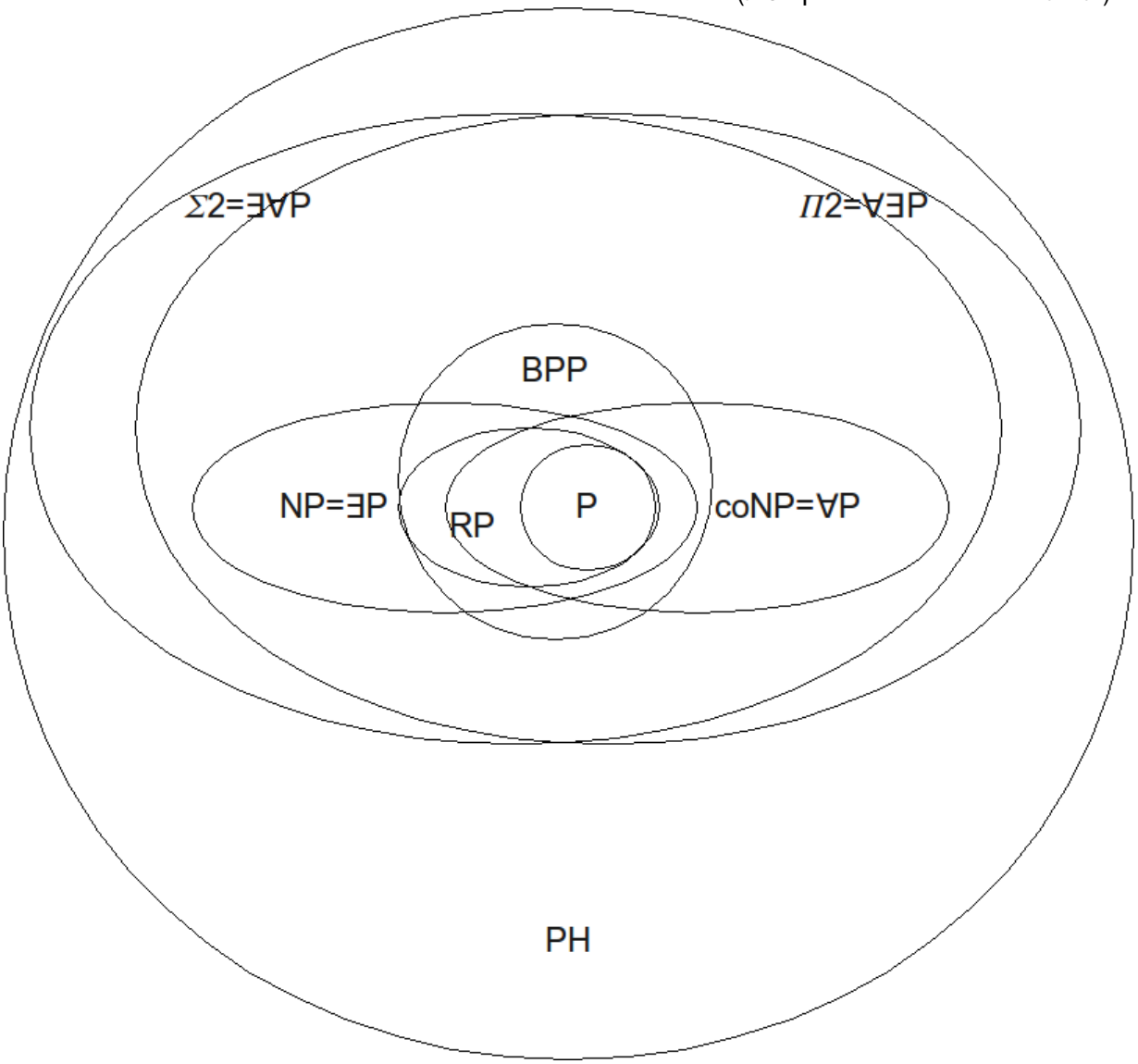
מתקיים: $P\subseteq AM(2)\subseteq AM(3)\subseteq\cdots\subseteq\bigcup_k AM(k)\subseteq\cdots\subseteq AM(Poly(n))$. אבל יש כאן קריסה: לכל k קבוע מתקיים $AM(k)=AM(2)$. כמו כן: $IP=PSPACE=AM(Poly(n))$. בשיעורי הבית נראה $AM\subseteq PSPACE$.

<u>הגדרה</u>: $\Sigma_2=\exists\forall P$ - אוסף כל השפות L כך שיש מ"ט דטרמיניסטית $M(x,y,z)$ פולינומית, $|y|,|z|=Poly(|x|)$, כך שאם $x\in L$ אזי $\exists y.\forall z.M(x,y,z)=1$, ואם $x\notin L$ אזי $\neg\exists y.\forall z.M(x,y,z)=1$ או $\forall y.\exists z.M(x,y,z)=0$.

ראינו כי $P\subseteq RP\subseteq BPP\subseteq PSPACE$, $P\subseteq RP\subseteq NP\subseteq\Sigma_2\subseteq\Sigma_3\subseteq\cdots\subseteq PH\subseteq PSPACE$.

<u>משפט</u>: $BPP\subseteq\Sigma_2$.

ציור (תחת ההנחה ש- PH לא קורסת):



<u>הערה</u>: ממילא $BPP=coBPP\subseteq co\Sigma_2=\Pi_2$.

<u>הוכחה</u>:

תהי $L\in BPP$. בפרט, יש מ"ט הסתברותית $M(x,y)$, $|y|=m=Poly(|x|)$, ומתקיים:

1. $x\in L \Leftarrow \Pr_y[M(x,y)=1]\geq\frac{1}{2}$.
2. $x\notin L \Leftarrow \Pr_y[M(x,y)=1]\leq\frac{1}{4m}$.

$\exists y_1,\ldots,y_{2m}\in\{0,1\}^m.\forall z\in\{0,1\}^m.(M(x,y_1\oplus z)=1)\vee\cdots\vee(M(x,y_{2m}\oplus z)=1)$ שקול לכך ש- $\exists i\in[1,2m].M(x,y_i\oplus z)=1$. הפסוק הראשון הוא פסוק Σ_2 .

נסמן: $G=\{y\in\{0,1\}^m \mid M(x,y)=1\}$.

איזה z יקיימו $M(x,y_1\oplus z)=1$? כל ה- z -ים כך ש- $y_1\oplus z\in G$, כלומר $z\in y_1\oplus G$, כאשר $y_1\oplus G=\{y_1\oplus g \mid g\in G\}$.

נשים לב ש- $\forall y.|y\oplus G|=|G|$.

אם $x \notin L$:

אז לכל בחירה $y_1,\ldots,y_{2m} \in \{0,1\}^m$,

z מקיים $\left(M(x,y_1 \oplus z)=1\right) \vee \cdots \vee \left(M(x,y_{2m} \oplus z)=1\right) \Leftrightarrow z \in (y_1 \oplus G) \cup \cdots \cup (y_{2m} \oplus G)$.

ולכן, מספר ה-z-ים שמספקים את הפסוק הוא לכל היותר $|y_1 \oplus G| + \cdots + |y_{2m} \oplus G|$.

$|y_1 \oplus G| + \cdots + |y_{2m} \oplus G| = 2m \cdot |G| \overset{|G| \le \frac{2^m}{4m}}{\le} 2m \cdot \frac{2^m}{4m} \le \frac{2^m}{2}$, ולכן בפרט יש z שעבורו הפסוק לא נכון, ולכן כל הפסוק False.

זה נכון כי גודל מרחב ה-z-ים הוא 2^m, וכמות ה-z-ים ה"טובים" היא לכל היותר חצי מהמרחב. לכן, יש z-ים בחצי השני.

אם $x \in L$:

לכל $z \in \{0,1\}^m$,

$$\Pr_{\underbrace{y_1,\ldots,y_{2m}}_{\text{בחירות בלתי תלויות}}} \underbrace{\left[z \in (y_1 \oplus G) \vee \cdots \vee z \in (y_{2m} \oplus G)\right]}_{\text{מאורעות בלתי תלויים}} =$$

$$= 1 - \Pr_{y_1,\ldots,y_{2m}}\left[z \notin (y_1 \oplus G) \wedge \cdots \wedge z \notin (y_{2m} \oplus G)\right] = 1 - \prod_{i=1}^{2m} \Pr_{y_i}\left[z \notin y_i \oplus G\right] \ge 1 - 2^{-2m}$$

אם $x \in L$, אזי $|G| \ge \frac{2^m}{2}$, ולכן $\Pr_y[z \notin y \oplus G] = \Pr_y\left[\underbrace{z \oplus y}_{\substack{\text{איבר אקראי} \\ \text{ב-} \{0,1\}^m}} \notin G\right] = \frac{2^m - |G|}{2^m} \le \frac{1}{2}$.

ראינו שלכל $z \in \{0,1\}^m$, $\Pr_{y_1,\ldots,y_{2m}}[z \text{ לא מכוסה}] \le 2^{-2m}$.

$$\Pr_{y_1,\ldots,y_{2m}}[\text{יש איזשהו } z \text{ לא מכוסה}] = \Pr\left[\bigvee_z z \text{ לא מכוסה}\right] \le$$

$$\le \sum_z \Pr[z \text{ לא מכוסה}] \le 2^m \cdot 2^{-2m} = 2^{-m} < 1$$

בפרט, יש $y_1,\ldots,y_{2m} \in \{0,1\}^m$ שמכסים את כל ה-z-ים, והפסוק True.

<u>אלגוריתמי קירוב</u>

בעיית MAX-CUT

<u>קלט</u>: גרף $G=(V,E)$.

<u>פלט</u>: הגודל של החתך הכי גדול בגרף, ומהו.

בעיה זו היא NP-קשה. לכן, נרצה לבצע לה קירוב.

נמצא חתך שגודלו "כמעט" כמו החתך הכי גדול בגרף.

בתרגיל: תמיד יש חתך שגודלו $\frac{|E|}{2}$, וגם ראינו איך למצוא אותו.

<u>הערה</u>: אם פותרים בעיה NP-קשה אחת עם פקטור קירוב 2, אז (כמו עכשיו) זה לא אומר כלום על קירוב של בעיות קשות אחרות.

נחזור לבעיית MAX-CUT בהמשך.

<u>דוגמה 1</u>: VC - *Vertex Cover*

<u>קלט</u>: $G=(V,E)$.

<u>פלט</u>: קבוצה $A \subseteq V$ שמכסה את כל הקשתות וגודלה מינימלי.

בעיה זו היא NP-קשה, ולכן נרצה לקרב אותה.

אלגוריתם קירוב ל- VC

<u>הצעה</u>: כל פעם ניקח את הקודקוד עם הדרגה הכי גבוהה, ונזרוק את כל הקשתות שנוגעות בו (זה אלגוריתם חמדני). כל פעם עושים את הצעד <u>הלוקלי</u> הכי טוב.

<u>האלגוריתם</u>:

1. נמצא זיווג מקסימלי בגרף, $e_1,\ldots,e_t$.

<u>הערה</u>: זיווג מקסימלי (*maximal*) – זיווג שלא ניתן להרחבה. לעומת זיווג אופטימלי (*maximum matching*) שהוא הזיווג הגדול ביותר הקיים בין כל הזיווגים.

2. נוציא כפלט את כל הקודקודים $v_1,\ldots,v_{2t}$ שמופיעים בקשתות $e_1,\ldots,e_t$.

נכונות:

זהו באמת כיסוי קודקודי, כי לכל $e\in E$, אם e לא נוגעת ב-$v_1,\ldots,v_{2t}$, אז אפשר להוסיף אותה לזיווג, בסתירה למקסימליות. לכן, זהו כיסוי קודקודי.

הכיסוי "כמעט" מינימלי: כל כיסוי מכסה את כל הצלעות, ולכן בפרט הוא מכסה את $e_1,\ldots,e_t$. ולכן, חייב לקחת קודקוד אחד (לפחות) מכל קשת, והקודקודים האלה זרים, ולכן צריך לפחות t קודקודים. קיבלנו כיסוי בגודל $2t$, כשכל כיסוי גודלו לפחות t , דהיינו, קירוב בפקטור 2 .

בעיית אופטימיזציה מוגדרת על ידי: לכל קלט x , מוגדר אוסף פתרונות אפשריים $W(x)$ (למשל VC , קליקה בגודל לפחות k , וכו'), ולכל פתרון מוגדרת עלות $C(x,s)$, $x\in\{0,1\}^n$, $s\in W(x)$.

דוגמאות:

1. x - גרף, $W(x)$ - אוסף החתכים, $C(x,s)$ - מספר הקשתות בחתך. מחפשים C מקסימלי.
2. x - גרף, $W(x)$ - אוסף כיסויים קודקודיים, $C(x,s)$ - מספר הקודקודים בכיסוי. מחפשים C מינימלי.

לבעיית מקסימום, יש פתרון אופטימלי בעלות $\mathrm{OPT}(x)$: נגיד שיש לנו אלגוריתם c -קירוב לבעיה. אם לכל קלט x , נמצא פתרון $s(x)$ כך ש- $\frac{\mathrm{OPT}(x)}{c}\le \mathrm{Cost}(x,s(x))\le \mathrm{OPT}(x)$.

לביית מינימום, $\mathrm{OPT}(x)\le \mathrm{Cost}(x,s(x))\le c\cdot\mathrm{OPT}(x)$.

ראינו: אלגוריתם פולינומי דטרמיניסטי שפותר את בעיית 2 -קירוב ל- VC .

דוגמה 2: SC - *Set Cover*

קלט: עולם $W=\{w_1,\ldots,w_m\}$ ואוסף קבוצות $S_1,\ldots,S_n\subseteq W$.

פלט: $i_1,\ldots,i_k\in[n]$ כך ש- $W=S_{i_1}\cup\cdots\cup S_{i_k}$.

הערה:

בעיית ה- SC היא יותר כללית מבעיית VC , כי בבעיית VC רוצים לכסות את כל הקשתות עם קודקודים, כמופע SC כאשר $W=E=\{e_1,\ldots,e_n\}$, ולכל קודקוד v , $S_v=\left\{\begin{array}{l}\text{כל הקשתות}\\ \text{שנוגעות ב-}v\end{array}\right\}$.

אלגוריתם (חמדני):

בכל שלב, נבחר את הקבוצה S_i שמכסה הכי הרבה איברים חדשים.

הערה:

במקרה הפרטי של VC , כל פעם נבחר את הקודקוד עם הדרגה הכי גבוהה, ונמחק את כל הקשתות שקשורות אליו.

נכונות:

האלגוריתם מחזיר פתרון: בסוף הריצה, כל העולם אכן מכוסה.

כמה רע זה יכול להיות? נניח שנתון לנו קלט $S_1,\ldots,S_n$, שאפשר לפתור בעלות OPT . באיזה עליות האלגוריתם החמדן יפתור אותו?

בהתחלה, יש m איברים לכסות. אפשר לכסות אותם עם $S_{i_1},\ldots,S_{i_{\mathrm{OPT}}}$, ולכן יש קבוצה $|S_{ij}|\ge\frac{m}{\mathrm{OPT}}$ (לפי שובך היונים). לכן, בפרט, הקבוצה הכי גדולה מבין $S_1,\ldots,S_n$ גודלה לפחות $\frac{m}{\mathrm{OPT}}$, ולכן יישאר לנו לכסות לכל היותר $\left(1-\frac{1}{\mathrm{OPT}}\right)\cdot m$ איברים.

נסמן: r_k - כמה איברים נותרו לאלגוריתם החמדן אחרי k צעדים. מתחילים עם $r_0=m$.

באופן כללי, אם אחרי k צעדים, נותרו לנו r_k איברים, ניתן לכסות אותם על ידי OPT קבוצות, ולכן הקבוצה שנבחר (הכי גדולה שאפשר) תכיל לפחות $\frac{r_k}{\text{OPT}}$ איברים חדשים.

לכן, $\begin{cases} \underbrace{r_{k+1}}_{\substack{\text{כמה נשאר אחרי}\\ \text{הצעד הנוסף}}} \le \left(1-\frac{1}{\text{OPT}}\right)\cdot r_k \\ r_0=m \end{cases}$.

<u>מטרה</u>: למצוא t כך ש- $r_t<1$.

$$r_t \le \left(1-\frac{1}{\text{OPT}}\right)^t \cdot r_0 = \left(1-\frac{1}{\text{OPT}}\right)^t \cdot m \overset{1-x<e^{-x}}{<} e^{-\frac{t}{\text{OPT}}}\cdot m$$

לכן, אם ניקח $t=\text{OPT}\cdot\ln m$, אז $r_t<e^{-\ln m}\cdot m=1$.

האלגוריתם החמדן המשתמש בלכל היותר $\text{OPT}\cdot\ln m$ קבוצות, משיג יחס קרוב $\ln m$.

<u>שאלות</u>:

1. האם האלגוריתם החמדן משיג אולי יחס קירוב יותר טוב?
2. האם אולי יש אלגוריתם קירוב פולינומי אחר שמשיג יחס קירוב יותר טוב?
3. מה יחס הקירוב של האלגוריתם החמדן עבור VC ?

נחזור לבעיית MAX-CUT :

<u>קלט</u>: גרף $G=(V,E)$.

<u>פלט</u>: **גודל** חתך מקסימלי בגרף.

נרצה למצוא חתך בגרף שגודלו "כמעט" מקסימלי.

<u>אלגוריתם קירוב</u>:

הבעיה שקולה לתוכנית $\max \frac{1}{2}\cdot\sum_{(i,j)\in E}(1-y_i y_j)$, כאשר $\forall i\in[n].\ y_i\in\{\pm 1\}$.

<u>הסבר</u>: אם y_1, y_2 באותו צד (כלומר שניהם 1 או שניהם -1), אז מכפלתם היא 1, ולכן נקבל תרומה של $1-1=0$ לסכום. אם הם בצדדים שונים, נקבל תרומה של $\frac{1}{2}\cdot(1+1)=1$ לסכום.

Linear Programming – בעיה ב- P .

כשהבעיה היא לא לינארית – יש מקרים שידוע שהם ב- P , ויש מקרים שידוע שהם קשים.

Integer Programming – NP -קשה.

<u>נכונות</u>:

אם יש חתך $A\subseteq V$ שגודלו k, אזי נבחר $y_i=\begin{cases}1 & i\in A\\ -1 & i\notin A\end{cases}$, ואז $e\in E$ תורמת לסכום 1 אם היא בחתך, או 0 אחרת. התוכנית תחזיר לפחות k.

אם יש פתרון שגודלו k, נסמן $A=\{i\,|\,y_i=1\}$, $\bar{A}=\{i\,|\,y_i=-1\}$. אזי A חתך בגודל k.

במקום לפתור את הבעיה בשלמים, נפתור את הבעיה בעולם אחר.

$\max \frac{1}{2}\cdot\sum_{(i,j)\in E}\left(1-\underbrace{v_i\cdot v_j}_{\text{מכפלה פנימית}}\right)$, כאשר $v_i\in\mathbb{R}^n$, $\|v_i\|=1$ לכל i.

<u>נכונות</u>:

הערך שהתוכנית מחזירה הוא לפחות OPT , כי ראינו שיש פתרון ב- $1,-1$ שפותר את הבעיה. אפשר להרחיב אותו באמצעות $v_i=(y_i,0,\ldots,0)$.

את התוכנית הזות אפשר לפתור ב- P בזמן פולינומי דטרמיניסטי.

איך נחלץ מהפתרון $v_1,\ldots,v_n$ חתך?

נבחר על מישור אקראי $w\in\mathbb{R}^n$, $\|w\|=1$.

כל הקודקודים ש- $v_i\cdot w>0$ יהיו בצד אחד.

כל הקודקודים ש- $v_i\cdot w<0$ יהיו בצד השני.

בעיית MAX-CUT

<u>קלט</u>: גרף לא מכוון $G=(V,E)$.

<u>פלט</u>: חתך $A\subseteq V$ כך ש- $|E(A,\bar{A})|$ מקסימלי.

הבעיה היא NP -קשה.

<u>ראינו</u>: תמיד יש חתך שגודלו לפחות $\frac{|E|}{2}$. אפשר למצוא אותו ב- P .

נראה אלגוריתם הסתברותי שמוצא חתך שמשקלו $\alpha\cdot\mathrm{OPT}$, $\alpha\approx 0.86\ldots$.

תיאור האלגוריתם:

1. פותרים (בקירוב) את התוכנית $\max\frac{1}{2}\sum_{(i,j)\in E}1-v_i\cdot v_j$, כאשר $v_i\in\mathbb{R}^n$ ולכל $i\in\{1,\ldots,n\}$, $\|v_i\|=1$.
2. נבחר $\bar{r}\in_R\mathbb{R}^n$, $\|\bar{r}\|=1$ ונגדיר $A=\{i\,|\,v_i\cdot\bar{r}\geq 0\}$.

<u>הערה</u>:

אם $n=1$, אזי $v_i\in\{1,-1\}$, והתוכנית מחזירה ערך OPT ו- $A=\{i\,|\,v_i=1\}$ חתך אופטימלי.

<u>הערה</u>:

כל פתרון $v_i\in\{1,-1\}$ ניתן לתרגום לפתרון של וקטורים n -ממדיים מהצורה $\{\pm 1,0,\ldots,0\}$, ולכן בפרט, אם OPT פתרון אופטימלי של MAX-CUT , $\widetilde{\mathrm{OPT}}$ פתרון אופטימלי של התוכנית, אזי $\widetilde{\mathrm{OPT}}\geq\mathrm{OPT}$.

<u>נראה</u>: $\alpha\cdot\widetilde{\mathrm{OPT}}\leq E_{\bar{r}}\left[\begin{matrix}\text{גודל חתך}\\ \text{שמקבלים}\end{matrix}\right]$, $\mathrm{OPT}\geq E[\text{גודל חתך}]$.

צ"ל: $\alpha\cdot\mathrm{OPT}\leq\alpha\cdot\widetilde{\mathrm{OPT}}\underset{\text{נראה}}{\leq}E_{\bar{r}}\left[\begin{matrix}\text{גודל החתך}\\ \text{שמקבלים}\end{matrix}\right]\leq\mathrm{OPT}$.

בכך נראה כי $\alpha\cdot\mathrm{OPT}\leq\alpha\cdot\widetilde{\mathrm{OPT}}\leq\mathrm{OPT}$.

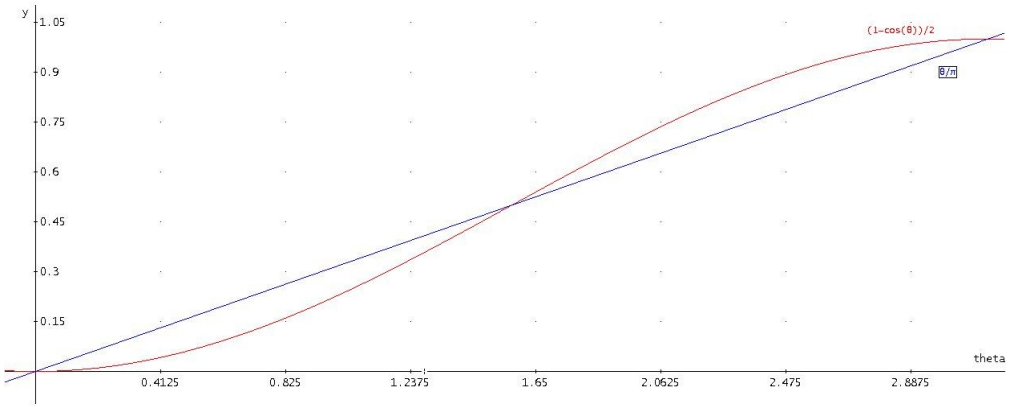
נסמן ב- W_e - מ"מ שאומר "הקשת e בתוך החתך שנוצר". $W=\sum_{e\in E}W_e$. אזי:

$$E[W]=\sum_{e\in E}E[W_e]=\sum_{e\in E}\frac{\theta_e}{\pi}\geq\sum_{e\in E}\alpha\cdot\frac{1-\cos\theta_e}{2}=$$

$$=\alpha\sum_{(i,j)=e\in E}\frac{1-v_i\cdot v_j}{2}\geq\alpha\cdot\widetilde{\mathrm{OPT}}\geq\alpha\cdot\mathrm{OPT}$$

כאשר θ_e היא הזווית בין v_i ל- v_j ($e=(i,j)$).

לכל $0\leq\theta\leq\pi$, $\frac{\theta}{\pi}\geq\alpha\cdot\frac{1-\cos\theta}{2}$.



$v_1,\ldots,v_n$ הם הפתרונות של התוכנית, ולכן הם מקרבים את OPT .

<u>המטרה הבאה</u>: להראות שיש בעיות אופטימיזציה ש- NP -קשה לקרב אותן עד כדי איזה פקטור.
לצורך זה, נחזור ונסתכל על השלב בין מוכיחים כל יכולים ובודקים הסתברותיים.

<u>הגדרה</u>:

המחלקה $\mathbf{PCP}\left(r(n),q(n)\right)$ היא מחלקת כל השפות L עבורן יש מוכיח ובודק על קלט באורך n, הבודק מטיל $r(n)$ מטבעות, מחשב $q(n)$ ביטים שהוא רוצה לקרוא מההוכחה, וקורא אותם, ואז מחשב פרדיקט $V\left(\underbrace{x}_{\text{קלט}},\underbrace{y}_{\text{מטבעות}},\underbrace{w_{i_1},\ldots,w_{i_{q(n)}}}_{\substack{\text{הביטים מההוכחה}\\ \text{שהוא קורא}}}\right)$ כך ש:

$$x\in L \;\Rightarrow\; \exists \underbrace{w}_{\text{הוכחה}}.\ \forall \underbrace{y}_{\substack{\text{מטבעות}\\ \text{של } V}}.\ V(x,y,w)=1$$

$$x\notin L \;\Rightarrow\; \forall \underbrace{w}_{\text{מוכיח}}.\ \Pr_{\underset{\text{מטבעות } V}{y}}\left[V(x,y,w)=1\right]\le\frac{1}{2}.$$

איך אלגוריתמים כאלו עובדים?

<u>קלט</u>: $x\in\{0,1\}^n$ ידוע לכולם.

1. המוכיח כותב הוכחה (ענקית) על השולחן $w=(w_1,\ldots,w_m)$.
   אפשר לדמיין שאחרי שהמוכיח כתב את ההוכחה, הוא כיסה אותה (והיא לא ניתנת לשינוי).
2. הבודק ההסתברותי מטיל $r(n)$ מטבעות, מחשב איזה $q(n)$ ביטים מההוכחה הוא רוצה לקרוא, $i_1,\ldots,i_{q(n)}$, ומחליט אם לקבל או לדחות.

מה אורך ההוכחה המקסימלי ל- $\mathrm{PCP}\left(r(n),q(n)\right)$?

לכל הטלת מטבע $y\in\{0,1\}^{r(n)}$, הבודק קורא $q(n)$ מקומות. יש $2^{r(n)}$ הטלות אפשריות.
לכן, יש לכל היותר $2^{r(n)}\cdot q(n)$ מקומות שהבודק יכול לקרוא.
לכן, נניח בה"כ שאורך ההוכחה הוא לכל היותר $2^{r(n)}\cdot q(n)$.

<u>טענה</u>: $\mathrm{NP}\subseteq\mathrm{PCP}\left(O(\log n),\mathrm{Poly}(n)\right)$.

נניח כי $L\in\mathrm{PCP}\left(O(\log n),\mathrm{Poly}(n)\right)$. נבנה אלגוריתם NP בשבילה.

תיאור האלגוריתם (בהינתן $x\in\{0,1\}^n$):

המוכיח ייתן את אותה הוכחה של מערכת ה- PCP. האורך שלה הוא לכל היותר $\mathrm{Poly}(n)=2^{O(\log n)}\cdot\mathrm{Poly}(n)=2^{r(n)}\cdot q(n)$.

הבודק יעבור על כל האפשרויות של $y\in\{0,1\}^{r(n)}$. לכל אחת מהן, יקרא את הביטים המתאימים מההוכחה, יראה מה הבודק של PCP עושה.

- אם עבור איזשהו y, הבודק דוחה, גם הוא ידחה.
- אם כולם מקבלים – יקבל.

נכונות:

אם $x\in L$, אזי המוכיח ההוגן של PCP יש לו את התכונה שלכל y הבודק מקבל, ולכן נקבל.

אם $x\notin L$, לכל מוכיח, לחצי מהאפשרויות של y, הבודק ידחה. בפרט, יש y עבורו הבודק ידחה, ולכן נדחה.

זמן ריצה:

הבודק רץ ב- P כי $2^{r(n)}=\mathrm{Poly}(n)$.

<u>דוגמה</u>: $\mathrm{GNISO}\in\mathrm{PCP}\left(\mathrm{Poly}(n),1\right)$.

בפרט, אורך ההוכחה אקספוננציאלי', ונקרא רק ביט אחד מתוכה.

<u>קלט</u>: גרפים G_1,G_2 על n קודקודים.

<u>הוכחה</u>: לכל גרף H על n קודקודים, כך ש- $H\sim G_1$ או $H\sim G_2$. הביט אומר למי הוא איזומורפי.

משפט ה- PCP : $\mathrm{NP}\subseteq\mathrm{PCP}\left(O\left(\log n\right),O\left(1\right)\right)$.

אורך ההוכחה $2^{O\left(\log n\right)}\cdot O\left(1\right)=\mathrm{Poly}\left(n\right)$.

בודק הסתברותי מספיק לו לקרוא מספר קבוע של ביטים מההוכחה.

נראה (לא עד הסוף) כי $\mathrm{NP}\subseteq\mathrm{PCP}\left(\mathrm{Poly}\left(n\right),O\left(1\right)\right)$.

<u>טענה</u>: אם A בעיה NP -קשה, ו- $A\in\mathrm{PCP}\left(\mathrm{Poly}\left(n\right),O\left(1\right)\right)$, אזי גם $\mathrm{NP}\subseteq\mathrm{PCP}\left(\mathrm{Poly}\left(n\right),O\left(1\right)\right)$.

תהי $L\in\mathrm{NP}$. תהי רדוקציה φ $L\leq_{\mathrm{p}}A$. בהינתן x , במקום לשאול האם $x\in L$, נשאל האם $\varphi\left(x\right)\in A$.

נבקש מהמוכיח הוכחת PCP של A עבור $\varphi\left(x\right)$, ונבדוק אותה.

הבעיה A תהיה QuadEq :

<u>קלט</u>: אוסף משוואות ריבועיות מעל $\mathbb{Z}_2$: $A_{\underbrace{m}_{\text{משוואות}}\times\underbrace{n^2}_{\substack{\text{נעלמים}\\(\text{נעלם לכל } i,j)}}}$ $\cdot$ m משוואות $\begin{cases}x_1x_2+x_3x_5+x_1x_7=1\\x_1x_2+x_1x_{10}=0\\\vdots\end{cases}$

<u>פלט</u>: האם יש $x_1,\ldots,x_n\in\mathbb{Z}_2$ כך שכל המשוואות מסתפקות?

ניתן לפשט:

<u>קלט</u>: $A_{m\times n^2}$, $\vec{b}\in\left\{0,1\right\}^m$.

<u>פלט</u>: האם יש $u\in\left\{0,1\right\}^n$ כך ש- $A\underbrace{\begin{pmatrix}u_1u_1\\u_1u_2\\\vdots\\u_iu_j\\\vdots\\u_nu_n\end{pmatrix}}_{u\otimes u}=\vec{b}$?

נוכיח כי $\mathrm{QuadEq}\in\mathrm{NP}$:

המוכיח ייתן לנו $u\in\left\{0,1\right\}^n$. נחשב את $u\otimes u$ ונבדוק האם $A\left(u\otimes u\right)=\vec{b}$.

הבעיה שלנו: אנחנו הולכים לקרוא חלק קטן מההוכחה. יכול להיות שהשגיאה נמצאת במקום מצומצם (למשל כל המשוואות מסופקות למעט אחת). איך נמצא את זה?

רעיון I:

המוכיח במקום לתת $u\in\left\{0,1\right\}^n$, ייתן קידוד של u באורך 2^n , או שהוא ייתן קידוד של $u\otimes u$ באורך $2^{\left(n^2\right)}$.

אם $a\in\left\{0,1\right\}^n$, אז הקידוד שלה $c\left(a\right)$ הוא מחרוזת באורך 2^n באופן הבא: על כל מיקום ב- $c\left(a\right)$, ניתן להתאים $y\in\left\{0,1\right\}^n$, והערך במיקום הזה יהיה $a\cdot y=\sum_{i=1}^{n}a_iy_i$.

<u>תרגיל</u>: אם $a_1,a_2\in\left\{0,1\right\}^n$, $a_1\neq a_2$, אזי $\underbrace{\Delta\left(c\left(a_1\right),c\left(a_2\right)\right)}_{\substack{\text{מספר המקומות בהם}\\a_1,a_2\text{ שונים}}}=\frac{2^n}{2}$. כלומר, a_1 ו- a_2 יכולים להיות שונים רק בביט אחד, אבל אחרי הקידוד, השוני נמצא בכל מקום.

בתור בודקים, קיבלנו וקטור $w_1,\ldots,w_{2^n}$, והמוכיח טוען שזו מילת קוד. אנחנו נסתכל רק על מספר קבוע של ביטים מ- $c\left(a\right)$.

רוצים פרוצדורת בדיקה:

- אם זו מילת קוד – תמיד נקבל.
- אם זה רחוק ממילת קוד – נדחה בהסתברות טובה.

פרוצדורת הבדיקה:

<u>קלט</u>: $w_1,\ldots,w_{2^n}$ שאמור להיות מילת קוד.

תיאור האלגוריתם:

נבחר $\bar{r}_1,\bar{r}_2\in_{\mathrm{R}}\left\{0,1\right\}^n$ באופן אחיד וב"ת.

נקרא $w_{\bar{r}_1},w_{\bar{r}_2},w_{\bar{r}_1\oplus\bar{r}_2}$, ונבדוק האם $w_{\bar{r}_1\oplus\bar{r}_2}=w_{\bar{r}_1}\oplus w_{\bar{r}_2}$.

אם יש שוויון – נקבל. אחרת – נדחה.

<u>טענה</u>: אם $w=c(u)$ מילת קוד, אז הבדיקה תמיד תקבל.

$$(c(u))_{\bar{r}_1+\bar{r}_2}=u\cdot(\bar{r}_1+\bar{r}_2)=u\cdot\bar{r}_1+u\cdot\bar{r}_2=(c(u))_{\bar{r}_1}+(c(u))_{\bar{r}_2}$$

<u>טענה</u> (ללא הוכחה): אם w δ-רחוקה מכל מילת קוד (כלומר לכל $a\in\{0,1\}^n$, $\Delta(w,c(a))\geq\delta\cdot 2^n$), אזי $\Pr[\text{נדחה}]\geq\delta$.

המוכיח נותן $w\in\{0,1\}^{2^n}$.

1. אם w רחוק מכל מילת קוד, נתפוס אותו.
2. אם w לא מילת קוד, אבל $\frac{1}{100}$ קרוב למילת קוד.

אנחנו נקפיד תמיד לשאול קואורדינטות אקראיות מתוך w. בהסתברות לכל היותר δ נקבל ערך לא לפי $c(a)$. מבחינה אפקטיבית, אנחנו באמת עובדים עם $c(a)$.

נניח שאנחנו מעוניינים ב-w_y. נבחר $r\in\{0,1\}^n$. נשאל: $\underbrace{w_r}_{\text{מקום אקראי}}$, $\underbrace{w_{r+y}}_{\substack{\text{בפני}\\ r+y\\ \text{עצמו אקראי}}}$. אזי $\Pr_r[w_r\neq(c(a))_r]\leq\delta$, $\Pr_r[w_{r+y}\neq(c(a))_{r+y}]\leq\delta$. לכן, חוץ מהסתברות לכל היותר 2δ, נקבל $c(a)_r=a\cdot r$, $c(a)_{r+y}=a\cdot(r+y)$, ומזה נחלץ את $c(a)_y=a\cdot y$. כמו כן:

$$c(a)_r+c(a)_{r+y}=a\cdot r+a\cdot(r+y)=\underbrace{a\cdot r+a\cdot r}_{0}+a\cdot y=a\cdot y=c(a)_y$$

הפרוטוקול ל- QuadEq :

<u>קלט</u>: $A_{m\times n^2}$, $\vec{b}\in\{0,1\}^m$.

<u>פלט</u>: האם יש $u\in\{0,1\}^n$ כך ש-$A(u\otimes u)=\vec{b}$?

ההוכחה:

מוכיח הוגן מספק $T_1=c(u)$, $T_2=c(u\otimes u)$.

אורך ההוכחה: $2^n+2^{(n^2)}$.

הבודק:

1. בודק ש-T_1 הוא δ-קירוב למילת קוד, עם בטחון $1-\varepsilon$ (דורש לקרוא $\frac{3\cdot\ln\frac{1}{\varepsilon}}{\delta}$ ביטים).
2. בודק ש-T_2 הוא δ-קירוב למילת קוד, עם בטחון $1-\varepsilon$.
3. בוחר $r_1,r_2\in\{0,1\}^n$, ובודק ש-$(T_2)_{\bar{r}_1\otimes\bar{r}_2}=(T_1)_{r_1}\cdot(T_1)_{r_2}$, וחוזר על השלב הזה $n\ln\frac{1}{\varepsilon}$ פעמים.

   (נראה: אם $T_1=c(u)$, $T_2=c(v)$, אזי $v=u\otimes u$).
4. בוחר $\alpha_1,\ldots,\alpha_m\in_R\{0,1\}$, ובודק $(T_2)_A=\sum_k\alpha_k b_k$.

נכונות:

מה קורה אם יש u כך ש-$A(u\otimes u)=\vec{b}$, והמוכיח הוגן, כלומר $T_1=c(u)$, $T_2=c(u\otimes u)$?

בדיקה 1: ראינו שכשמקבלים מילת קוד בדיקה 1 תמיד עוברת.

בדיקה 2:

$$(T_2)_{r_1\otimes r_2}=(c(u\otimes u))_{r_1\oplus r_2}=(u\otimes u)\cdot(r_1\otimes r_2)=$$

$$=\underbrace{(u\ \text{מכפלה פנימית}\ r_1)}_{\text{כפל}}\cdot\underbrace{(u\ \text{מכפלה פנימית}\ r_2)}=(c(u))_{r_1}\cdot(c(u))_{r_2}=T_{r_1}\cdot T_{r_2}$$

ולכן הבדיקה תעבור.

בדיקה 3: מכיוון $A(u\otimes u)=\vec{b}$, נובע שלכל $k=1,\ldots,m$ מתקיים $A_k\cdot(u\otimes u)=b_k$. בפרט, לכל $\alpha_1,\ldots,\alpha_k$,

$$(T_2)_{\sum_k\alpha_k A_k}=\left(\sum_k\alpha_k A_k\right)\cdot(u\otimes u)=\sum_k\alpha_k b_k.$$

<u>הגדרה</u>: המחלקה $\mathbf{PCP}\left(r(n),q(n)\right)$ היא מחלקת כל השפות L עבורן יש פרוטוקול אינטראקטיבי בין מוכיח כל יכול לבודק הסתברותי.

עבור קלט $x\in\{0,1\}^n$, המוכיח כותב הוכחה $w\in\{0,1\}^{2^{r(n)}\cdot q(n)}$ ומכסה אותה, הבודק מטיל $r(n)$ מטבעות $y=y_1\cdots y_{r(n)}$, בוחר $q(n)$ ביטים שרוצה לקרוא $i_1,\ldots,i_{q(n)}$ וקורא אותם מההוכחה.

הוא מפעיל פרדיקט $V\left(\underbrace{x}_{\text{קלט}},\underbrace{y}_{\text{מטבעות}},w_{i_1},\ldots,w_{i_{q(n)}}\right)$ שמחליט אם לקבל או לדחות, כך ש:

$$x\in L \quad\Leftarrow\quad \underbrace{\exists w}_{\text{יש מוכיח}}\ .\ \underbrace{\forall y}_{\substack{\text{לכל בדיקה}\\ \text{של הבודק}}}\ .\ V(x,y,w)=1 .$$

$$x\notin L \quad\Leftarrow\quad \forall w.\Pr_y\left[V(x,y,w)=0\right]\geq 1/2$$ (כלומר לכל מוכיח, נתפוס אותו מרמה בהסתברות לפחות $1/2$).

משפט ה- PCP

$\mathrm{NP}\subseteq\mathrm{PCP}\left(O(\log n),O(1)\right)$ (ראינו בשיעור שעבר את הכיוון ההפוך).

נוכיח: $\mathrm{NP}\subseteq\mathrm{PCP}\left(\mathrm{Poly}(n),O(1)\right)$. בתוצאה הזאת, אורך ההוכחה $2^{\mathrm{Poly}(n)}$ אקספוננציאלי באורך הקלט.

1. מספיק להוכיח עבור שפה ספציפית שהיא NP-שלמה, שהיא גם שייכת למחלקה $\mathrm{PCP}\left(\mathrm{Poly}(n),O(1)\right)$.
2. נעבוד עם הבעיה QuadEq:

<u>הקלט</u>:

1. מטריצה בולאנית $A_{m\times n^2}$, נסמן ב-A_i את השורה ה-i של A, ונסמן $A_i=\left(a_{i1},\ldots,a_{in^2}\right)$.
2. וקטור $\vec{b}\in\{0,1\}^m$.

<u>הפלט</u>: הקלט $A,\vec{b}$ בשפה אם יש $u\in\{0,1\}^n$ כך ש-$A(u\otimes u)=\vec{b}$.

<u>קידוד</u>: $c:\{0,1\}^k\to\{0,1\}^{2^k}$.

התכונה של הקידוד תהיה: אם $u_1\neq u_2$, $u_1,u_2\in\{0,1\}^k$, אזי $c(u_1),c(u_2)$ שונים בלפחות חצי מהקואורדינטות.

הקוד: אם $u\in\{0,1\}^k$, $c(u)\in\{0,1\}^{2^k}$, ואז $\left(c(u)\right)_y=u\cdot y$ (כמו בשיעור הקודם).

<u>תכונה</u>: $\left(c(u)\right)_A+\left(c(u)\right)_B=\left(c(u)\right)_{A\oplus B}$.

למה? $\left(c(u)\right)_A+\left(c(u)\right)_B=u\times A+u\times B=\sum u_iA_i+\sum u_iB_i=$
$=\sum u_i(A_i+B_i)=u\cdot(A\oplus B)=\left(c(u)\right)_{A\oplus B}$.

לכן, אפשר להסיק כי $\left(c(u)\right)_A=\left(c(u)\right)_B+\left(c(u)\right)_{A\oplus B}$.

כזכור, יש 2^{2^k} מילים, ורק 2^k מילות קוד.

<u>טענה</u>: מילה אקראית, בהסתברות גבוהה, $1/4$-רחוקה מכל מילת קוד.

<u>משפט</u>: $\mathrm{QuadEq}\in\mathrm{PCP}\left(\mathrm{Poly}(n),O(1)\right)$.

נראה פרוטוקול $\mathrm{PCP}\left(\mathrm{Poly}(n),O(1)\right)$ לבעיה:

<u>קלט</u>: $A_{m\times n^2}$, $\vec{b}$.

מוכיח הוגן צריך לעשות:

אם $\langle A,\vec{b}\rangle\in\mathrm{QuadEq}$, אז יש $u\in\{0,1\}^n$ כך ש-$A(u\otimes u)=\vec{b}$.

לכן, המוכיח נותן $T_1=c(u)$, $T_2=c(u\otimes u)$.

הבודק:

בודק ש-T_1 היא 99/100-קרובה למילת קוד.

בודק ש-T_2 היא 99/100-קרובה למילת קוד.

בוחר $r_1,r_2\in\{0,1\}^n$, ובודק ש-$\left(T_2\right)_{r_1\otimes r_2}=\left(T_1\right)_{r_1}\cdot\left(T_1\right)_{r_2}$, $O\left(\log\frac{1}{\varepsilon}\right)$ פעמים.

בוחר $\alpha_1,\ldots,\alpha_m\in\{0,1\}$ באקראי, מחשב $C=\sum\alpha_i A_i$, בוחר $B\in_R\{0,1\}^{n^2}$, ובודק ש- $(T_2)_B+(T_2)_{B\oplus C}=\sum\alpha_i\vec{b}_i$. על השלב הזה חוזרים $O\left(\log\frac{1}{\varepsilon}\right)$ פעמים.

הבדיקה ש- $T\in\{0,1\}^{2^t}$ היא ε-קרובה למילת קוד:

נחזור על התהליך הבא $O\left(\log\frac{1}{\varepsilon}\right)$ פעמים:

נבחר $A,B\in\{0,1\}^t$, ונוודא ש- $T_A+T_B=T_{A\oplus B}$.

אם T באמת מילת קוד, אז יש u כך ש- $T=c(u)$, והבדיקה תמיד עוברת.

<u>משפט</u>: אם T δ-רחוקה ממילת קוד, אזי המבחן ייכשל בהסתברות לפחות δ.

<u>תכונה 1</u>: זהו באמת פרוטוקול $\mathrm{PCP}(\mathrm{Poly}(n),O(1))$.

הבודק מטיל $O(n^2)$ מטבעות, קורא $O\left(\log\frac{1}{\varepsilon}\right)=O(1)$ ביטים מההוכחה (את ε אנחנו קובעים, ונגיד ש- $\varepsilon=1/100$). אורך ההוכחה $2^{n^2}+2^n$.

שלמות:

נניח $(A,\vec{b})\in\mathrm{QuadEq}$, ונניח שהמוכיח הוגן. אזי יש $u\in\{0,1\}^n$ כך ש- $A(u\otimes u)=\vec{b}$.

המוכיח ייתן $T_1=c(u)$, $T_2=c(u\otimes u)$.

הבדיקה הראשונה תצליח.

הבדיקה השנייה: $(T_2)_{r_1\otimes r_2}\overset{?}{=}(T_1)_{r_1}\cdot(T_2)_{r_2}$.

$$(T_2)_{r_1\otimes r_2}=\left(c(u\otimes u)\right)_{r_1\otimes r_2}=(u\otimes u)\cdot(r_1\otimes r_2)=$$
$$=\sum_{1\le i,j\le n}(u\otimes u)_{i,j}\cdot(r_1\otimes r_2)_{i,j}=\sum_{i,j}u_i\cdot u_j\cdot(r_1)_i\cdot(r_2)_j=$$
$$=\left(\sum_i u_i\cdot(r_1)_i\right)\cdot\left(\sum_j u_j\cdot(r_2)_j\right)=(u\cdot r_1)\cdot(u\cdot r_2)=(c(u))_{r_1}\cdot(c(u))_{r_2}=(T_1)_{r_1}\cdot(T_1)_{r_2}$$

הבדיקה השלישית:

u מקיים $A(u\otimes u)=\vec{b}$. לכן, לכל $1\le i\le m$, $A_i\cdot(u\otimes u)=\vec{b}_i$.

לכן גם $\underbrace{\left(\sum\alpha_i A_i\right)}_{C}\cdot(u\otimes u)=\sum\alpha_i(A_i\cdot(u\otimes u))=\sum\alpha_i\vec{b}_i$.

מכיוון שהמוכיח הוגן, $(T_2)_C=\sum\alpha_i\vec{b}_i$.

לכן: $(T_2)_B+(T_2)_{B\oplus C}=(T_2)_C=\sum\alpha_i\vec{b}_i$.

לכן, המבחן תמיד יעבור. נקבל:

<u>תכונה</u>: אם הקלט בשפה והמוכיח הוגן, אז כל הבדיקות תעבורנה.

תקפות:

נניח כי $(A,\vec{b})\notin\mathrm{QuadEq}$, ונקבע מוכיח מרושע כלשהו, שנותן לנו $T_1\in\{0,1\}^{2^n}$, $T_2\in\{0,1\}^{2^{n^2}}$.

אם T_1 היא 1/100-רחוקה מכל מילת קוד, נתפוס אותו בבדיקה הראשונה בהסתברות $1-\varepsilon$ וסיימנו. בה"כ ניתן להניח כי T_1 היא 99/100-קרובה לאיזו מילת קוד $c(u)$.

אם T_2 היא 1/100-רחוקה מכל מילת קוד, נתפוס אותו בבדיקה הראשונה בהסתברות $1-\varepsilon$ וסיימנו. בה"כ ניתן להניח כי T_2 היא 99/100-קרובה לאיזו מילת קוד $c(v)$.

בא המבחן השני. המטרה שלנו היא לבדוק ש- $v=u\otimes u$. אם המוכיח בחר לתת לנו $v\neq u\otimes u$, נגלה את זה.

מה למעשה עושה הבדיקה השנייה? $(T_2)_{r_1\otimes r_2}\overset{?}{=}(T_1)_{r_1}\cdot(T_2)_{r_2}$. קוראים 3 ביטים: ביט אחד מ- T_2 ו-2 ביטים מ- T_1. כל מקום שאנחנו קוראים בפני עצמו מיקום אקראי.

יש סיכוי של לכל היותר 0.03 שאחד מהביטים שקראנו שונה מ- $c(u)$, $c(v)$ בהתאמה. חוץ מזה, מה שאנחנו רואים:

$$(T_2)_{r_1\otimes r_2}\overset{?}{=}(T_1)_{r_1}\cdot(T_1)_{r_2}$$
$$(c(v))_{r_1\otimes r_2}\overset{?}{=}(c(u))_{r_1}\cdot(c(u))_{r_2}$$
$$v\cdot(r_1\otimes r_2)\overset{?}{=}(u\cdot r_1)\cdot(u\cdot r_2)=(u\otimes u)\cdot(r_1\otimes r_2)$$

מכיוון ש- $v\neq u\otimes u$, לא נעבור את המבחן.
אנחנו משווים 2 מילות קוד שונות במיקום אקראי. מכיוון שכל 2 מילות קוד שונות בחצי מהמיקומים, בהסתברות 0.5 נתפוס את השגיאה.
עכשיו, בה"כ T_1 קרובה ל- $c(u)$, T_2 קרובה ל- $c(v)$ והקלט לא בשפה. לכן, בהכרח $A(u\otimes u)\neq\vec{b}$, ולכן יש לפחות $1\le i\le m$ אחד כך ש- $A_i\cdot(u\otimes u)\neq\vec{b}_i$.
בבדיקה השלישית, בוחרים $\alpha_1,\ldots,\alpha_m$ באקראי. נסמן: $C=\sum\alpha_i A_i$. בודקים ש- $(T_2)_B+(T_2)_{B\oplus C}\overset{?}{=}\sum\alpha_i\vec{b}_i$. קוראים 2 ביטים מ- T_2 .
בהסתברות לכל היותר 0.02 יש ביט שקראנו שלא משקף את $c(u\otimes u)$. חוץ מזה, אנחנו רואים $\underbrace{(c(u\otimes u))_B+(c(u\otimes u))_{B\oplus(\sum\alpha_i A_i)}}_{(c(u\otimes u))_{\sum\alpha_i A}=(u\otimes u)\cdot\sum\alpha_i A_i}\overset{?}{=}\sum\alpha_i\vec{b}_i$.

בודקים: $\sum\alpha_i(A\cdot(u\otimes u))=\sum\alpha_i\vec{v}_i$. כלומר: $\sum_{i=1}^{m}\alpha_i\cdot(A_i\cdot(u\otimes u)-b_i)\overset{?}{=}0$.

$\underbrace{\vec{\alpha}}_{\text{וקטור באורך } n}\cdot\underbrace{(A(u\otimes u)-\vec{b})}_{\text{וקטור באורך } n}\overset{?}{=}0$. $(c(A(u\otimes u)-\vec{b}))_{\vec{\alpha}}\overset{?}{=}0$, ולכן נתפוס את המוכיח מרמה בהסתברות 0.5 .
בסך הכל: $\Pr\left[\begin{matrix}\text{נטעה}\\ \text{ונגיד בסדר}\end{matrix}\right]\le\frac{2}{100}+\frac{1}{2}=0.6$.

משפט: $\mathrm{NP}=\mathrm{PCP}(O(\log n),O(1))$.

<u>הגדרה</u>: $Y,N\subseteq\{0,1\}^*$ הוא **בעיית הבטחה** (*promise problem*) אם $Y\cap N=\emptyset$.
<u>הגדרה</u>: $L\subseteq\{0,1\}^*$ הוא **שפה** אם בעיית ההבטחה המתאימה: $\langle L,\{0,1\}^*\setminus L\rangle$.
<u>הגדרה</u>: אלגוריתם A פותר בעיית הבטחה $\langle Y,N\rangle$ אם לכל $x\in Y$, $A(x)=1$ ולכל $x\in N$, $A(x)=0$.
מה קורה עם הקלטים $x\notin Y\cup N$? לא אכפת לנו.
<u>דוגמה</u>: $\mathrm{Gap}_{[\alpha,\beta]}\mathrm{Max3SAT}$, $\alpha<\beta$.
היא בעיית הבטחה שמקבלת פסוק 3SAT .
Y - כל פסוק 3SAT $\varphi=\bigwedge_{i=1}^{m}c_i$ כך שיש השמה שמספקת לפחות βm פסוקיות.
N - כל פסוק 3SAT $\varphi=\bigwedge_{i=1}^{m}c_i$ כך שאין השמה שמספקת αm פסוקיות.
<u>מטרה</u>: לטעון ישי $\beta<1$ כך ש- $\mathrm{Gap}_{[\beta,1]}\mathrm{Max3SAT}$ היא NP -קשה.
<u>הגדרה</u>: יהיו $\langle Y_1,N_1\rangle$, $\langle Y_2,N_2\rangle$ בעיות הבטחה. נגיד $\langle Y_2,N_2\rangle\le\langle Y_1,N_1\rangle$ אם יש רדוקציה $\varphi:\{0,1\}^*\to\{0,1\}^*$ שמקיימת $x\in Y_2\Leftrightarrow\varphi(x)\in Y_1$, $x\in N_2\Leftrightarrow\varphi(x)\in N_1$.
כמו במקרה הכללי, לרדוקציה יש סיבוכיות. ניתן להגדיר רדוקציות במחלקות, למשל $\le_{\mathrm{L}}$, $\le_{\mathrm{P}}$.
<u>הגדרה</u>: בעיית הבטחה $\langle Y,N\rangle$ היא NP -קשה אם לכל $A\in\mathrm{NP}$ מתקיים $\langle A,A^c\rangle\le_{\mathrm{P}}\langle Y,N\rangle$.
<u>משפט</u>: יש קבוע $\beta<1$ כך ש- $\mathrm{Gap}_{[\beta,1]}\mathrm{Max3SAT}$ היא NP -קשה. נניח שאנחנו יודעים $\mathrm{NP}=\mathrm{PCP}(O(\log n),O(1))$, ונוכיח את המשפט.
תהי $A\in\mathrm{NP}$. לכן, $A\in\mathrm{PCP}(O(\log n),O(1))$, ולכן יש לה פרוטוקול PCP כנ"ל.
לכל $x\in\{0,1\}^n$ קלט של A , נבנה פסוק 3SAT φ_x כך ש- $x\in A\Rightarrow\varphi_x$ ספיק (כלומר יש השמה המספקת את כל הפסוקיות), ו- $x\notin A\Rightarrow$ כל השמה תספק לכל היותר β מהפסוקיות.

הרדוקציה:

נתון $x\in\{0,1\}^n$.

בפרוטוקול ה- PCP , המוכיח נותן לנו הוכחה $w=w_1\dots w_k$, $k=\text{Poly}(n)$.

הבודק מטיל $y\in\{0,1\}^{r(n)}$ מטבעות ($r(n)=O(\log n)$). לכל $y_1,\dots,y_{r(n)}$ אפשרי, מסתכל על $q(n)=O(1)$ ביטים $w_{i_1},\dots,w_{i_{q(n)}}$ ומחשב $V(x,y,w_{i_1},\dots,w_{i_{q(n)}})$ שמחליט אם לקבל או לדחות.

לכל $y_1,\dots,y_{r(n)}$ אפשרי, נבנה פסוק 3SAT שהמשתנים שלו הם $w_{i_1},\dots,w_{i_{q(n)}}$ והערך שלו $V(x,y,w_{i_1},\dots,w_{i_{q(n)}})$.

הפסוק $\varphi_{x,y_1,\dots,y_{r(n)}}$ הוא פסוק 3SAT באורך קבוע $2^{O(q(n))}$.

הפלט של הרדוקציה: $\bigwedge_{y_1,\dots,y_{r(n)}}\varphi_{x,\bar{y}}$.

הרדוקציה רצה ב- P , כי יש $2^{O(\log n)}=\text{Poly}(n)$ אפשרויות ל- y.

שלמות:

אם $x\in A$, יש מוכיח שנותן $w=w_1\cdots w_k$ כך שלכל $y\in\{0,1\}^n$ מתקיים $V(x,y,w_{i_1},\dots,w_{i_{q(n)}})=1$, ולכן עבור ה- w הזה, לכל y , $\varphi_{x,y}$ ספיק, ולכן w מספק את φ_x.

תקפות:

אם $x\notin A$, אז לכל $w=w_1\cdots w_k$ (נחשוב עליו כהוכחה שמוכיח רשע מספר למערכת ה- PCP), $\Pr_y\left[V(x,y,w_{i_1},\dots,w_{i_{q(n)}})=0\right]\geq 1-\beta$. לכן, $\Pr_y\left[\begin{matrix}v\text{ לא מספק}\\ \varphi_{x,y}\text{ את}\end{matrix}\right]\geq 1-\beta$.

ב- $\varphi_{x,y}$ יש מספר קבוע, B , של פסוקיות.

$\varphi_x=\bigwedge_{y_1,\dots,y_{r(n)}}\underbrace{\varphi_{x,y}}_{\substack{\text{פסוק 3SAT}\\ \text{באורך קבוע}}}$, ולכן בסך הכל ב- φ_x יש $B\cdot 2^{r(n)}$ פסוקיות, ולכן ב- w שנקבל, לפחות $(1-\beta)\cdot 2^{r(n)}$ פסוקים $\varphi_{x,y}$ לא מסתפקים עם w , ולכן לפחות $(1-\beta)\cdot 2^{r(n)}$ פסוקיות של φ_x לא מסתפקות, ולכן אחז הפסוקיות שלא מסתפקות הוא לפחות $\gamma=\frac{1-\beta}{B}=\frac{(1-\beta)\cdot 2^{r(n)}}{B\cdot 2^{r(n)}}$, ו- $\gamma>0$ קבוע. זה נכון עם $\beta=\frac{1}{2}$.

מש"ל.

<u>הגדרה</u>: $\text{Gap}_{[\alpha,\beta]}\text{maxClique}$:

- Y - גרפים על n קודקודים בהם יש קליקה בגודל לפחות β.
- N - גרפים על n קודקודים בהם הקליקה המקסימלית היא בגודל לכל היותר α.

<u>טענה</u>: נניח כי $A\in\text{Gap}_{[a,b]}$ היא NP -קשה. אם יש אלגוריתם P שמקרב את A עד כדי פקטור $\frac{b}{a}$, אזי $\text{P}=\text{NP}$.

נניח $\mathrm{NP}=\mathrm{PCP}\left(O\left(\log n\right),O\left(1\right)\right)$.

<u>משפט</u>: יש $0<\alpha<\beta\le 1$ כך ש-$\mathrm{Gap}_{\left[\alpha,\beta\right]}\mathrm{max3SAT}$ היא NP-שלמה.

תהי $A\in\mathrm{NP}$. אזי יש לה מערכת PCP, כך שעבור קלט $x\in\left\{0,1\right\}^n$ יש מוכיח שמספק הוכחה $w\in\left\{0,1\right\}^{m=\mathrm{Poly}\left(n\right)}$.

נבנה פסוק 3SAT φ_x שהמשתנים שלו הם $w_1,\ldots,w_m$:

לכל $y\in\left\{0,1\right\}^{O\left(\log n\right)}$ הטלת מטבעות אפשרית לבודק של מערכת ה-PCP, הרדוקציה תעשה את הדבר הבא:

- תחשב איזה שאילתות בודק ה-PCP ישאל $i_1,\ldots,i_q$.
- הרדוקציה תבנה טבלת אמת: לכל השמה אפשרית $w_{i_1},\ldots,w_{i_q}$, הרדוקציה מחשבת האם $V\left(x,y,w_{i_1},\ldots,w_{i_q}\right)$ מקבל או לא.
- הרדוקציה ממירה את טבלת האמת לפסוק 3SAT $\varphi_{x,y}$.

הפלט של רדוקציה: $\varphi_x=\bigwedge_{y\in\left\{0,1\right\}^{O\left(\log n\right)}}\varphi_{x,y}$. φ_x פסוקית באורך k שתלוי רק ב-q.

נכונות הרדוקציה:

אם $x\notin A$, לכל w הוא מבטא איזשהו מוכיח כלשהו של מערכת PCP, ולכן $\Pr_y\left[V\left(x,y,w_{i_1},\ldots,w_{i_q}\right)=1\right]\le\alpha$, ולכן $\Pr_y\left[\begin{matrix}\text{מספק }\varphi_{x,y}\\ \text{עם }w\end{matrix}\right]\le\alpha$. לכן, עבור חלק $1-\alpha$ יש לפחות פסוקית אחת של $\varphi_{x,y}$ שלא מסתפקת, ולכן בסך הכל חלק $\beta=\frac{1-\alpha}{k}$ מהפסוקיות לא מסתפקות.

אם $x\in A$, אזי יש מוכיח הוגן שנותן $w\in\left\{0,1\right\}^m$ כך שלכל $y\in\left\{0,1\right\}^{O\left(\log n\right)}$, $V\left(x,y,w_{i_1},\ldots,w_{i_q}\right)=1$, ולכן ההשמה w מספקת כל $\varphi_{x,y}$, ולכן φ_x ספיק לגמרי. אפשר לקחת $\beta=1$.

בעיה NP-קשה: אפשר לתת עדות לקלטים בשפה.	שפה: האם פסוק 3SAT ספיק לגמרי?
$\mathrm{PCP}\left(O\left(\log n\right),O\left(1\right)\right)$: אפשר לתת עדות לשייכות בשפה שאפשר לבדוק רק עם מעט שאילתות.	בעיית ההבטחה: <u>כן</u> – יש השמה שמספקת הרבה פסוקיות. <u>לא</u> – כל השמה מספקת מעט פסוקיות.

<u>טענה</u>: אם $\mathrm{Gap}_{\left[\alpha,\beta\right]}\mathrm{max3SAT}$ היא NP-קשה עבור $0<\alpha<\beta\le 1$, אזי $\mathrm{NP}\subseteq\mathrm{PCP}\left(O\left(\log n\right),O\left(1\right)\right)$.

תהי $L\in\mathrm{NP}$. יש לה רדוקציה φ ל-$\mathrm{Gap}_{\left[\alpha,\beta\right]}\mathrm{max3SAT}$.

המטרה שלנו: לבנות מערכת PCP עבור A.

עבור קלט $x\in\left\{0,1\right\}^n$ יש פסוק 3SAT $\varphi_x\left(z_1,\ldots,z_n\right)=\bigwedge_{i=1}^{m}c_i$ כך שאם $x\in L$ אז יש השמה w שמספקת המון פסוקיות, ואם $x\notin L$, כל השמה w מספקת לכל היותר $\alpha\cdot m$ פסוקיות.

נבנה את מערכת ה-PCP:

המוכיח מספק השמה למשתני φ_x.

הבודק מטיל $i\in\left\{1,\ldots,n\right\}$. שואל מה הערכים של המשתנים שמופיעים בפסוקיות c_i. מקבל ערכים של משתנים. אם הפסוקית מסתפקת – מקבל. אחרת – דוחה.

נכונות:

אם $x\in L$, יש השמה w למשתני φ_x שמספקת המון פסוקיות, ולכן $\Pr_i\left[\begin{matrix}\text{מספק }c_i\\ \text{עם }w\end{matrix}\right]\ge\beta$ והבודק יקבל בהסתברות לפחות β.

אם $x\notin L$, לכל השמה w, ההשמה מספקת לכל היותר $\alpha\cdot m$ פסוקיות. לכן, כל מוכיח למערכת ה-PCP מגדיר השמה w, ולכן $\Pr\left[V\text{ יקבל}\right]\le\alpha$.

מספר השאילתות - 3. המערכת פולינומית. מספר המטבעות – $\log m$, כאשר m הוא מספר הפסוקיות ב-φ_x. φ_x הוא הפלט של הרדוקציה מ-L ל-Gap-max3SAT. φ_x פולינומית, ובפרט $m=\mathrm{Poly}\left(\left|x\right|\right)$, $\log m=O\left(\log\left|x\right|\right)$.

לכן, $L\in\mathrm{PCP}_{\left[\alpha,\beta\right]}\left(\log m,3\right)$.

רוצים להתחיל עם בעיית $\text{Gap}_{[\alpha,\beta]}\text{max3SAT}$ NP-קשה, ולהסיק ממנה שבעיות Gap אחרות הן NP-קשות.
לא רוצים לעשות את כל העבודה מחדש. רוצים להפעיל רדוקציות.

<u>הגדרה</u>: רדוקציה שמעבירה Gap ל- Gap נקראת **רדוקציה משמרת פער**.

ראינו: $3SAT \le Clique \le IS \le VC$.
<u>תזכורת</u>: $3SAT \le Clique$.
האם הרדוקציה משמרת פער?
אם התחלנו עם $\varphi \in Yes$, אזי יש השמה שמספקת $\beta \cdot m$ פסוקיות, ויש קליקה בגודל $\beta \cdot m$ לפחות .
אם התחלנו עם $\varphi \in NO$, אזי כל השמה מספקת לכל היותר $\alpha \cdot m$ פסוקיות, ולכן הקליקה המתקבלת גודלה לכל היותר $\alpha \cdot m$.
<u>תזכורת</u>: $Clique \le IS$.
<u>הרדוקציה</u>:
קלט: $G=(V,E)$.
פלט: $\bar{G}=(V,\bar{E})$.
<u>טענה</u>: יש ב-G קליקה בגודל k $\Leftrightarrow$ יש ב-$\bar{G}$ קבוצה ב"ת בגודל k .
האם הרדוקציה משמרת פער? כן!

נניח שהיינו יודעים כי $\text{Gap}_{[\sqrt{n},10\sqrt{n}]}\text{maxIS}$ (פער פי 10) היא NP-קשה.
הרדוקציה תיתן ש- $\text{Gap}_{[n-\sqrt{n},n-10\sqrt{n}]}\text{minVC}$ NP-קשה.
אבל $n-10\sqrt{n}$ מאוד קרוב ל-$n-\sqrt{n}$. נעלם לנו הפער הכפלי.

בפועל ראינו: $\text{Gap}_{[\alpha\cdot m,\beta\cdot m]}\text{maxIS}$ היא NP-שלמה. לכן, $\text{Gap}_{[(1-\alpha)\cdot m,(1-\beta)\cdot m]}\text{minVC}$ היא NP-שלמה.
נשאר לנו פער כפלי, אולי קטן יותר.
ל- VC באמת יש אלגוריתם קירוב עם פקטור 2 .

הבעיה 3SAT3 :
<u>קלט</u>: פסוק 3SAT $\varphi(x_1,\ldots,x_n)=\bigwedge_{i=1}^{m} c_i$, שכל משתנה x_i מופיע לכל היותר 3 פעמים.
<u>פלט</u>: ספיק או לא.

<u>טענה</u>: $3SAT \le 3SAT3$.
הרדוקציה (על קלט פסוק 3SAT $\varphi(x_1,\ldots,x_n)=\bigwedge_{i=1}^{m} c_i$):
לכל משתנה x_i שמופיע k_i פעמים, נכניס k_i משתנים שנקרא להם x_{ij} .
מחליף את המופע ה-j -י של x_i במשתנה x_{ij} , ונוסיף פסוקית $x_{ij_1}\to\cdots\to x_{ij_{k_i}}\to x_{ij_1}$, תוך שימוש בשקילות $a\to b\equiv\neg a\vee b$.
הפסוק החדש הוא פסוק 3SAT3 φ' .
כמה פסוקיות יש בו? $m+\sum_{i=1}^{m} k_i = m+3m=4m$.

נכונות:
אם φ ספיק, יש השמה ל-$x_1,\ldots,x_n$ שמספקת את φ . נבחר ל-x_{ij} את הערך של x_i , והשמה זו תספק את כל הפסוקיות.
אם φ' ספיק, אז בהכרח כל x_{ij} מקבלים את אותו ערך w_i , ובהכרח w מספקת את φ ולכן φ ספיק.

מה הרדוקציה מראה לגבי פער?

	φ	φ'
Yes	$\geq \beta m$	$\geq (3+\beta)m$
No	$\leq \alpha m$	יש $(3+\alpha)m$. יכול להיות שיש השמה שתספק $4m-1$.

למשל, ייתכן ש- x_1 או $\neg x_1$ מופיע בכל פסוקית, ואז ל- x_{ij} אפשר לבחור ערך 0 או 1 בצורה שכל הפסוקיות המקוריות יסתפקו.

לא למבחן

<u>עובדה</u>: יש סדרה של גרפים לא מכוונים G_n , כך ש- G_n מדרגה קבועה D ($D=10$) על n קודקודים, ומתקיים $G_n=(V,E)$, $|V|=n$, ולכל $A\subseteq V$, $|E(A,\bar{A})|\geq \min(|A|,|\bar{A}|)$.

בחזרה לרדוקציה:

מקבלת $\varphi(x_1,\ldots,x_n)=\bigwedge_{i=1}^{m} c_i$.

- לכל i , שמופיע n_i פעמים, ניצור n_i משתנים חדשים x_{ij} . נחליף את המופע ה- j -י של x_i ב- x_{ij} .
- נוסיף פסוקיות עוקבות: לכל $i=1,\ldots,n_i$, ניקח גרף על n_i קודקודים כמו בעובדה. הגרף דרגתו קבועה D , ויש לו את התכונה שלכל $A\subseteq V$, $|E(A,\bar{A})|\geq \min(|A|,|\bar{A}|)$, ולכל קשת בגרף x_{ij_2} , x_{ij_1}, x_{ij_2} , נוסיף $x_{ij_2}\rightarrow x_{ij_1}$, $x_{ij_1}\rightarrow x_{ij_2}$.
- כל משתנה מופיע $2D+1$ פעמים. קיבלנו רדוקציה ל- $\text{Gap}_{\frac{1}{2},1}\text{max3SAT}(2D+1)$.
- מספר הפסוקיות ב- φ' : $m+\sum_{i=1}^{n} k_i\cdot 2D = m+2D\cdot 3m=(6D+1)\cdot m$. $\underbrace{\quad}_{\text{ישנות}}$

נכונות:

	φ	φ'
Yes	$\geq \beta m$	$\geq (6D+\beta)m$ (כל הפסוקיות העקביות, βm מהמקוריות)
No	$\leq \alpha m$	$\leq (6D+\alpha)m$

המשך החומר למבחן

נזכר איפה התחלנו:

A - בעיית אופטימיזציה:

על קלט x , יש קבוצה של פתרונות מותרים S_x , לכל אחד יש $c(s)$.

בעיית האופטימיזציה מחפשת פתרון מותר עם ערך אופטימלי.

c -קירוב של בעיית מקסימום ($c\geq 1$):

על קלט x , מוצא ערך O כך ש- $\frac{\text{OPT}_x}{c}\leq O\leq \text{OPT}_x$, כאשר OPT_x הוא הערך האופטימלי.

בהתחלה ראינו אלגוריתמי קירוב (למשל 2 -קירוב ל- VC).

<u>הגדרה</u>: נאמר שבעיית קירוב היא **NP -קשה** אם קיום אלגוריתם פולינומי שפותר את בעיית הקירוב גורר $P=NP$.

<u>הערה</u>: משפט *Cook*: $c=1$ -קירוב הוא NP -קשה. עכשיו נוכיח: $c>1$ -קירוב הוא NP -קשה. ראינו: $c=2$ -קירוב ב- P .

טענה: יש $c>1$ כך ש-c-קירוב של VC היא בעיה NP-קשה.

בסדרת הרדוקציות, הגענו לבעיה $\text{Gap}_{|a,b|}\text{minVC}$, כאשר:

Yes - יש בגרף כיסוי קודקודי בגודל הקטן מ-b.

No - כל כיסוי קודקודי בגרף הוא בגודל הגדול מ-a.

ראינו: יש $0<\alpha<\beta<1$ כך ש-$\text{Gap}_{|\beta n,\alpha n|}\text{VC}$ היא NP-קשה ($n=|V|$).

נניח שיש לנו אלגוריתם c-קירוב ל-VC, שרץ בזמן פולינומי.

נראה איך לפתור את $\text{Gap}_{|\beta n,\alpha n|}\text{VC}$ ב-P, ומשם ינבע P=NP.

בהינתן מופע של $\text{Gap}_{|\beta n,\alpha n|}\text{VC}$, המופע הוא גרף $G=(V,E)$, ואנחנו רוצים לברר אם $G\in\text{Yes}$, ואז יש כיסוי קודקודי שגודלו לכל היותר αn, או $G\in\text{No}$, ואז כל כיסוי קודקודי גודלו לפחות βn.

נפעיל את אלגוריתם הקירוב שיחזיר מספר O.

מקרה 1: $G\in\text{Yes}$.

$\text{OPT}\le O\le c\cdot\text{OPT}$ ו-$\text{OPT}\le\alpha n$. לכן, $O\le c\,\alpha n$.

מקרה 2: $G\in\text{No}$.

$\beta n\le\text{OPT}\le O$. לכן, $O\ge\beta n$.

תיאור האלגוריתם על קלט G, שיודעים $G\in\text{Yes}$ או $G\in\text{No}$:

נפעיל את אלגוריתם הקירוב על G, ונקבל תשובה O.

אם $O<c\,\alpha n$, נענה כן.

אם $O>\beta n$, נענה לא.

האלגוריתם מוגדר היטב כל עוד $c\,\alpha n<\beta n$ $\Leftrightarrow$ $c<\frac{\beta}{\alpha}$.

ראינו: $\text{Gap}_{|a,b|}\min A$ NP-קשה $\Leftarrow$ קשה לקרב את A עד כדי פקטור $\frac{a}{b}$.

באותו אופן, $\text{Gap}_{|a,b|}\text{Max}\,A$ NP-קשה $\Leftarrow$ קשה לקרב את A עד כדי פקטור $\frac{b}{a}$.

בעיית Gap NP-קשה $\Leftarrow$ קושי של קירוב.

ראינו כי בעיית Gap של VC היא NP-קשה $\Leftarrow$ קושי לקרב את VC.

מה לגבי הכיוון ההפוך?

האם קושי של קירוב גורר בעיית Gap NP-קשה?

ראינו כי $\text{Gap}_{\left[\underbrace{0.61}_{\alpha}n,\underbrace{0.62}_{\beta}n\right]}\text{max2SAT}$ היא NP-קשה. לכן, קשה לקרב את max2SAT עד איזה קבוע $c=\frac{\beta}{\alpha}<1$.

מצד שני, $\text{Gap}_{|\alpha,1|}\text{max2SAT}$ לכל $\alpha<1$, ובפרט $\alpha=\frac{1}{c}$, נפתר על ידי 2SAT:

$\varphi\in\text{Yes}$ $\Leftarrow$ ספיק לגמרי.

$\varphi\in\text{No}$ $\Leftarrow$ לא ספיק.

וגם $2\text{SAT}\in\text{P}$, כי 2SAT היא NL-שלמה.

לכן, להגיד שקשה לקרב בעיה עד פקטור של c, לא נותן את האינפורמציה איפה הפער הזה נמצא.

לעומת זאת, לומר שבעיית $\text{Gap}_{|a,b|}\max A$ היא NP-קשה אומר שקשה לקרב עד פקטור c, ובנוסף אומר איפה הקושי הזה נמצא.

בעיית 2SAT היא דוגמה לזה ש-$\text{Gap}_{|0.61,\,c\cdot0.61|}$ היא NP-קשה, ו-$\text{Gap}_{\left[\frac{1}{c},1\right]}$ ב-P.

נזכר כי $\text{IS}\le\text{VC}$. הרדוקציה הייתה נכונה רק בגלל שידענו איפה ה-Gap של IS היה. הרדוקציה הייתה מועילה רק כי ידענו שיש קושי קירוב לפקטור כפלי, וידענו איפה הוא נמצא.

אם רק היינו יודעים שיש קירוב, זה היה יכול להיות $\left[\sqrt{n},10\sqrt{n}\right]$, זה לא היה מועיל לנו כי ב-$\left[n-\sqrt{n},n-10\sqrt{n}\right]$ אין פקטור כפלי.