# אלגוריתמים (67504) - סיכום הרצאות ותרגולים

משה קול (moshe.kol@mail.huji.ac.il)

17 בינואר 2019

**תקציר**

סיכום הרצאות ותרגולים בקורס "אלגוריתמים" (67504) אשר נלמד באוניברסיטה העברית, 2018-2019.
ההרצאות הועברו ע"י פרופ' אלכס סמורודניצקי, התרגולים הועברו ע"י גב' מורן מזרחי.

## תוכן העניינים

# חלק I
# הרצאות

## 1 שבוע 1 - הרצאה - 14.10.18

### 1.1 הקדמה

נפתח בדוגמת חימום.

**דוגמה 1.1.** נתונים $n$ מספרים שלמים $a_1, a_2, \ldots, a_n$. האם קיימים אינדקסים $i, j$ המקיימים $a_i + a_j = 2018$?

**דרך ראשונה** נעבור על כל הזוגות $1 \le i < j \le n$ ונבדוק האם $a_i + a_j = 2018$. זמן הריצה: $\Theta\left(n^2\right)$.

**דרך שניה** נמיין את המספרים בסדר עולה ומעתה נניח כי $a_1 \le a_2 \le \cdots \le a_n$. לכל $1 \le i \le n$ נבדוק האם $2018 - a_i$ נמצא ברשימה ע״י שימוש בחיפוש בינארי. זמן הריצה: $\Theta\left(n \log n\right)$.

### 1.2 אלגוריתמים חמדניים: מבוא

באופן בלתי פורמלי, נאמר כי **אלגוריתם חמדני** (Greedy Algorithm) הוא אלגוריתם איטרטיבי שבכל שלב ממקסם את הרווח המקומי.
בקורס ניתן מספר דוגמאות לבעיות אלגוריתמיות שאפשר לפתור בצורה זו, וגם נפתח תורה יחסית כללית המאפשרת לזהות מקרים מסוימים בהם יש לבעיה פתרון חמדני.

### 1.3 אלגוריתמים חמדניים: בעיית התרמיל השברית (Fractional Knapsack Problem)

בעיית התרמיל השברית

**בעיה 1.2** (בעיית התרמיל השברית).
סיפור המסגרת: גנב נכנס לחנות, מצוייד בתרמיל המוגבל במשקל המירבי שהתרמיל יכול לשאת. המטרה של הגנב היא למלא את התרמיל בפריטים בעלי ערך כולל מירבי. הפריטים ניתנים לחלוקה (למשל, אפשר לקחת $3/4$ מהפריט הראשון).
קלט: $n$ - מספר הפריטים; $W$ - המשקל המירבי שיכול לשאת התרמיל; $n$ זוגות של מספרים $\left(v_1, w_1\right), \ldots, \left(v_n, w_n\right)$ כאשר $v_i$ - הערך של הפריט ה-$i$, $w_i$ - המשקל של הפריט ה-$i$.
פלט: רשימה של מספרים $\left(x_1, x_2, \ldots, x_n\right)$ כאשר $x_i$ מתאר את חלקיות הפריט ה-$i$, ומתקיים:

- $x_i \in [0,1]$ לכל $i$;
- אילוץ המשקל: $\sum_{i=1}^{n} x_i w_i \le W$;
- השווי הכולל $\sum_{i=1}^{n} x_i v_i$ הוא מירבי.

**דוגמה.** $n = 3$, $W = 50$, $\left(v_1 = 120, w_1 = 30\right), \left(v_2 = 100, w_2 = 20\right), \left(v_3 = 60, w_3 = 10\right)$.

**ניסיון ראשון** ניקח $x_1 = x_2 = 1$ ו-$x_3 = 0$. המשקל הכולל הנו: $1 \cdot 30 + 1 \cdot 20 + 0 \cdot 10 = 50$ (עומד באילוץ המשקל), הערך הכולל הנו: $1 \cdot 120 + 1 \cdot 100 + 0 \cdot 60 = 220$.

**ניסיון שני (הפתרון האופטימלי)** לכל פריט נחשב את "הערך הסגולי" שלו: $r_i = \frac{v_i}{w_i}$, ערך זה מייצג את השווי של הפריט ליחידת משקל.

$$r_1 = \frac{120}{30} = 4 \quad r_2 = \tfrac{100}{20} = 5 \quad r_3 = \frac{60}{10} = 6$$

נעדיף לאסוף פריטים עם ערך סגולי גדול יותר על פני אחרים. מכאן נקבל:

| | | | |
|---|---|---|---|
| חלקיות | $x_3 = {}^2\!/_3$ | $x_2 = 1$ | $x_3 = 1$ |
| משקל צבור | 50 | 30 | 10 |

**פתרון.** נחשב את הערכים הסגוליים של הפריטים $r_i = \frac{v_i}{w_i}$, $i = 1, \ldots, n$. נמיין את הפריטים בסדר יורד לפי הערכים הסגוליים שלהם.
מעתה נניח כי $r_1 > r_2 > \cdots > r_n > 0$, כלומר: 1. הערכים הסגוליים חיוביים ממש (אם יש פריט ששוויו הוא 0 נתעלם ממנו); 2. נטפל במקרה בו הערכים הסגוליים שונים זה מזה (אם יש שני פריטים עם אותו ערך סגולי, נתייחס אליהם כאל פריט אחד); 3. האינדקסים $1, 2, \ldots, n$ כעת מציינים את הסדר הממויין של הפריטים (לאו דווקא הסדר שבו קיבלנו את רשימת הפריטים).

יהי $0 \leq t \leq n-1$ אינדקס המקיים $\sum_{i=1}^{t} w_i \leq W$ וגם $\sum_{i=1}^{t+1} w_i > W$.
כלומר, הפריט ה-$t+1$ הוא הפריט הראשון שלא יכול להיכנס בשלמותו לתרמיל, בהינתן שהכנסו את הפריטים הקודמים לתרמיל בשלמותם.
נסכים כי אם $t = 0$ אז הפריט הראשון לא נכנס בשלמותו, ואם $\sum_{i=1}^{n} w_i \leq W$ (כל הפריטים נכנסים בשלמותם) נגדיר $t = n$.
נגדיר את הפתרון החמדן באופן הבא:

1. $x_1 = x_2 = \cdots = x_t = 1$;
2. $x_{t+1} = \frac{W - \sum_{i=1}^{t} w_i}{w_{t+1}}$;
3. $x_{t+2} = x_{t+3} = \cdots = x_n = 0$.

**טענה 1.3.** הפתרון שהגדרנו הוא חוקי ואופטימלי (ז"א הטוב ביותר מבין כל הפתרונות החוקיים).

הוכחה.
(1) אם $t = n$, כלומר, כל הפריטים נכנסים לתוך התרמיל בשלומתם, אז הפתרון החמדן נותן $x_1 = x_2 = \cdots = x_n = 1$, שזה בבירור פתרון חוקי ואופטימלי.
(2) נניח כי $t < n$. נוכיח חוקיות ואופטימליות:
(חוקיות הקואורדינטות) מהגדרת הפתרון החמדן $x_1 = \cdots = x_t = 1$ (בין 0 ל-1), $x_{t+2} = \cdots = x_n = 0$ (בין 0 ל-1), וגם:

$$0 = \frac{\sum_{i=1}^{t} w_i - \sum_{i=1}^{t} w_i}{w_{t+1}} \leq x_{t+1} = \frac{W - \sum_{i=1}^{t} w_i}{w_{t+1}} < \frac{\sum_{i=1}^{t+1} w_i - \sum_{i=1}^{t} w_i}{w_{t+1}} = 1$$

כאשר נעזרנו בהגדרת האינדקס $t$ (ראו פירוט הפתרון מעלה).
(אילוץ המשקל)

$$\begin{aligned}
\sum_{i=1}^{n} x_i w_i &= \sum_{i=1}^{t} x_i w_i + x_{t+1} w_{t+1} + \sum_{i=t+2}^{n} x_i w_i \\
&= \sum_{i=1}^{t} 1 \cdot w_i + \frac{W - \sum_{i=1}^{t} w_i}{w_{t+1}} \cdot w_{t+1} + \sum_{i=t+2}^{n} 0 \cdot w_i \\
&= \sum_{i=1}^{t} w_i + \left( W - \sum_{i=1}^{t} w_i \right) \\
&= W
\end{aligned}$$

(אופטימליות) נראה כי לכל פתרון חוקי אחר $(y_1,\dots,y_n)$ מתקיים $\sum_{i=1}^{n} y_i v_i \geq \sum_{i=1}^{n} x_i v_i$:

$$\begin{aligned}
\sum_{i=1}^{n} x_i v_i - \sum_{i=1}^{n} y_i v_i &= \sum_{i=1}^{n} (x_i - y_i) v_i \\
\left[r_i = \frac{v_i}{w_i}\right] &= \sum_{i=1}^{n} (x_i - y_i) w_i r_i \\
[\text{פיצול הסכום}] &= \sum_{i=1}^{t} (x_i - y_i) w_i r_i + (x_{t+1} - y_{t+1}) w_{t+1} r_{t+1} + \sum_{i=t+2}^{n} (x_i - y_i) w_i r_i \\
[\text{החלפנו } r_i \text{ ב- } r_{t+1}] &\geq \sum_{i=1}^{t} \underbrace{(x_i - y_i)}_{\substack{\geq 0 \\ x_i=1, y_i \leq 1}} w_i r_{t+1} + (x_{t+1} - y_{t+1}) w_{t+1} r_{t+1} + \sum_{i=t+2}^{n} \underbrace{(x_i - y_i)}_{\substack{\leq 0 \\ x_i=0, y_i \geq 0}} w_i r_{t+1} \\
&= r_{t+1} \left( \sum_{i=1}^{t} (x_i - y_i) w_i + (x_{t+1} - y_{t+1}) w_{t+1} + \sum_{i=t+2}^{n} (x_i - y_i) w_i \right) \\
&= r_{t+1} \sum_{i=1}^{n} (x_i - y_i) w_i \\
&= r_{t+1} \left( \underbrace{\sum_{i=1}^{n} x_i w_i}_{=W} - \underbrace{\sum_{i=1}^{n} y_i w_i}_{\leq W} \right) \\
&\geq 0
\end{aligned}$$

והוכחנו כי $\sum_{i=1}^{n} y_i v_i \geq \sum_{i=1}^{n} x_i v_i$, כנדרש.

□

---

**אלגוריתם 1** אלגוריתם חמדן לפתרון בעיית התרמיל השברית

1. עיבוד מוקדם (Pre-Processing): חישוב הערכים הסגוליים $r_i = \frac{v_i}{w_i}$, ומיון הפריטים בסדר יורד לפי הערכים הסגוליים שלהם.
2. אתחול: $K \leftarrow \emptyset$.
3. איטרציה: נעבור על הפריטים מהראשון עד האחרון, ובכל שלב נכניס ל-$K$ כמה שיותר מהפריט הנוכחי (לפי אילוצי משקל).
4. סיום: נעצור כאשר התרמיל מתמלא או כאשר הכנסנו את כל הפריטים.

---

**זמן ריצה** בשלב העיבוד המוקדם אנו מבצעים מיון של הפריטים אשר לוקח $O(n \log n)$ (כאשר $n$ מספר הפריטים). שלב האיטרציה דורש $O(n)$. סה״כ זמן ריצה של האלגוריתם נשלט ע״י שלב המיון ולוקח $O(n \log n)$.

# 2 שבוע 1 - הרצאה - 17.10.18

## 2.1 אלגוריתמים חמדניים: בעיית התרמיל השברית - למת החלפה

אין זה מחייב שבכל בעיה חמדנית שאיתה נתמודד נוכל לתאר את הפתרון החמדן במפורש, כפי שעשינו בהרצאה קודמת עם בעיית התרמיל השברית.
לעתים נאלץ להוכיח את נכונות האלגוריתם החמדן באמצעות שיקולים אחרים - לרוב טענה שתגרוס כי הפתרון החמדן הוא לא פחות טוב מכל פתרון חוקי אחר (ומכאן תנבע האופטימליות שלו).
ננסח ונוכיח את "למת ההחלפה" עבור בעיית התרמיל השברית, אשר באמצעותה ניתן הוכחה נוספת לאופטימליות הפתרון שמוציא אלגוריתם 1.

**למה 2.1** (למת החלפה עבור בעיית התרמיל השברית)**.** יהי $y = (y_1, y_2, \ldots, y_n)$ פתרון חוקי לבעיית התרמיל השברית. נניח כי קיים אינדקס $1 \le j \le n$ כך שמתקיים:

1. $\sum_{i=1}^{j} y_i w_i < W$ (עד הפריט ה-$j$ לא מילאנו את התרמיל)
2. $y_j < 1$ (לא הכנסנו לתרמיל את כל הפריט ה-$j$)

אזי קיים פתרון חוקי $z = (z_1, z_2, \ldots, z_n)$ כך ש-$z_j > y_j$ (כלומר, $z$ מכניס יותר מהפריט ה-$j$) וגם $z$ רווחי יותר מ-$y$.

הוכחה. נבדיל בין שני מקרים:

**מקרה ראשון:** $\sum_{i=1}^{n} y_i w_i < W$

נגדיר:

- $d = W - \sum_{i=1}^{n} y_i w_i$ - אז $d > 0$ לפי הנחת המקרה, ו-$d$ מייצג את עודף המשקל שעדיין אנחנו יכולים להכניס לתרמיל.
- $d_j = (1 - y_j) w_j > 0$ - המשקל מהפריט ה-$j$ שעדיין נותר בחנות.
- $D = \min(d, d_j) > 0$

נגדיר פתרון חדש $z$ באופן הבא:

- לכל $j \ne i$ נגדיר $z_i := y_i$;
- $z_j$ יוגדר באופן הבא $z_j := y_j + \frac{D}{w_j}$.

מתקיים $z_j > y_j$ כי הוספנו ל-$z_j$ גודל חיובי, וגם בבירור $z$ רווחי יותר מ-$y$ (הוספנו עוד משקל מפריט אחד).
נשאר לבדוק את החוקיות של $z$. לכל $i \ne j$ מתקיים $0 \le z_i = y_i \le 1$, בנוסף מתקיים $0 \le y_j < z_j$ וגם:

$$\begin{aligned} z_j &= y_j + \frac{D}{w_j} \\ \text{[by definition of } D\text{]} &\le y_j + \frac{(1 - y_j) w_j}{w_j} \\ &= 1 \end{aligned}$$

נבדוק את אילוץ המשקל:

$$\begin{aligned} \sum_{i=1}^{n} z_i w_i &= \sum_{i \ne j} y_i w_i + z_j w_j \\ &= \sum_{i \ne j} y_i w_i + y_j w_j + D \\ &= \sum_{i=1}^{n} y_i w_i + D \\ \text{[by definition of } D\text{]} &\le \sum_{i=1}^{n} y_i w_i + \left( W - \sum_{i=1}^{n} y_i w_i \right) \\ &= W \end{aligned}$$

**מקרה שני:** $\sum\limits_{i=1}^{n} y_i w_i = W$

לפי הנתון בלמה מתקיים $\sum\limits_{i=1}^{j} y_i w_i < W$ ולכן קיים אינדקס $j < k$ שעבורו $y_k > 0$, ובמילים, קיים פריט זול יותר, $k$, שהכנסנו ממנו גודל חיובי.
נגדיר:

- $d_j = (1 - y_j)\, w_j$ - המשקל שאפשר להוסיף לפריט ה-$j$,
- $d_k = y_k w_k$ - המשקל שהכנסנו מהפריט ה-$k$.
- $D = \min(d_j, d_k)$

נגדיר פתרון חדש $z$ באופן הבא:

- לכל $i \neq j$ וגם $i \neq k$ נגדיר $z_i := y_i$;
- $z_k := y_k - \frac{D}{w_k}$
- $z_j := y_j + \frac{D}{w_j}$

לפי הגדרת $z$ מתקיים $z_j > y_j$. נותר לבדוק כי $z$ הוא פתרון חוקי, אשר רווחי יותר מ-$y$.
(המשך הוכחת המקרה השני הושארה כתרגיל - והיא מאוד דומה להוכחת המקרה הראשון). $\square$

# 3 שבוע 2 - הרצאה - 21.10.18

## 3.1 אלגוריתמים חמדניים: בעיית התרמיל השברית - המשך

**משפט 3.1.** הפתרון החמדן הוא הפתרון האופטימלי היחיד לבעיית התרמיל השברית (בעיה 1.2).

הוכחה. יהי $G = (g_1, \dots, g_n)$ הפתרון החמדן ויהי $y$ פתרון אופטימלי. נוכיח כי $y = G$ ובכך נסיים את הוכחת המשפט. נניח בשלילה כי $y \neq G$. יהי $j$ האינדקס הראשון בו $y_j \neq g_j$, נראה כי בהכרח $y_j < g_j$:

- אם $1 \leq j \leq t$ [1] אזי $y_j \leq 1 = g_j$ כיוון ש-$y$ פתרון חוקי. מאחר ש-$g_j \neq y_j$ אנו מקבלים אי-שוויון חזק $y_j < g_j$.
- אם $j = t+1$ אז נניח בשלילה ש-$g_j > y_j$. במקרה זה מתקיים

$$\sum_{i=1}^{t+1} y_i w_i > \sum_{i=1}^{t+1} g_i w_i = \sum_{i=1}^{t} w_i + g_{t+1} w_{t+1} = W$$

וזוהי סתירה לחוקיות של $y$. לכן מתקיים $y_j < g_j$ (שהרי $y_j \neq g_j$).

- אם $j > t+1$ אז מכיוון ש-$g_j \neq y_j$ ו-$g_j = 0$ לפי הגדרת הפתרון החמדן, מתקיים $y_j > 0 = g_j$ (מחוקיות של $y$). בנוסף

$$\sum_{i=1}^{j} y_i w_i > \sum_{i=1}^{j} g_i w_i = W$$

וזוהי סתירה לחוקיות של $y$.

הראינו כי $y_j < g_j$ וגם כי $j \leq t+1$. נשים לב שתנאי למת ההחלפה 2.1 מתקיימים עבור $y$:

- $y_j < g_j \leq 1$,
- $\sum_{i=1}^{j} y_i w_i < \sum_{i=1}^{j} g_i w_i \leq W$

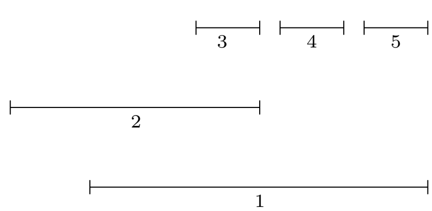
מלמת ההחלפה נובע שקיים פתרון רווחי יותר מ-$y$, בסתירה לבחירתו כפתרון אופטימלי. □

## 3.2 אלגוריתמים חמדניים: בעיית שיבוץ משימות

**בעיית שיבוץ משימות**

**בעיה 3.2** (שיבוץ משימות)**.** נתונות $n$ משימות כך שלא ניתן לבצע שתי משימות בו-זמנית. המטרה היא לבחור תת-קבוצה בגודל מקסימלי של משימות שכן ניתן לבצע.
קלט: $n$ - מספר המשימות; $n$ קטעים על ציר הזמן, הנתונים ע״י נקודות ההתחלה והסיום שלהם $[s_1, f_1], [s_2, f_2], \dots, [s_n, f_n]$.
פלט: תת-קבוצה $S \subseteq [n]$ כך שכל הקטעים עם אינדקסים ב-$S$ זרים זה לזה וגם $|S|$ מקסימלי בתנאי זה.

**דוגמה 3.3.** $n = 5$ והמשימות מתוארות בתור קטעים, באופן הבא:




פתרונות אופטימליים אפשריים $\{2,4,5\}$ או $\{3,4,5\}$.

[1] להגדרת $t$ ראו 1.3

ננסח את הבעיה באופן מעט יותר פורמלי: נסמן ב-$\mathcal{S}$ את מרחב הפתרונות החוקיים לבעיה. כלומר,

$$\mathcal{S} = \{S \subseteq [n] : \text{הקטעים עם אינדקסים ב-}S\text{ זרים זה לזה}\}$$

נגדיר פונקציית ערך $q : \mathcal{S} \to \mathbb{R}^+$ ע״י $q(S) = |S|$.
אנו מחפשים $S^* \in \mathcal{S}$ כך ש-$S^*$ היא נקודת מקסימום של $q$ על $\mathcal{S}$.

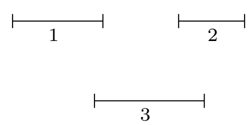
**פתרון נאיבי** נעבור על כל תת-הקבוצות של $[n]$, נבדוק האם תת-קבוצה מגדירה פתרון חוקי ונעקוב אחרי קבוצה מקסימלית כזו. העלות: $\Omega(2^n)$.

**ניסיונות למתן עקרון חמדני** נחפש אלגוריתם חמדני איטרטיבי שבכל שלב מכניס קטע נוסף לפתרון, ע״פ קריטריון חמדני כלשהו. מספר הצעות לעקרונות חמדניים:

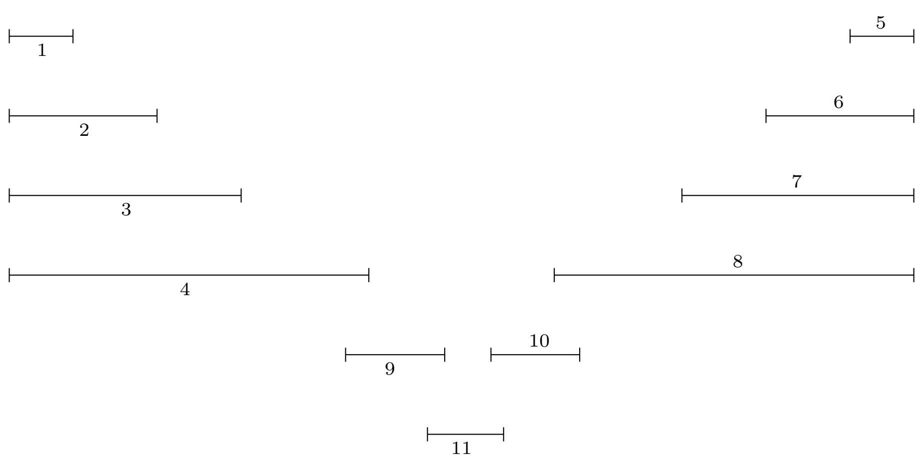
1. בכל שלב נכניס את הקטע הקצר ביותר שלא מתנגש עם הקטעים הקודמים שהכנסנו.
2. בכל שלב נכניס את הקטע שמתנגש עם כמה שפחות מהקטעים שנשארו.
3. בכל שלב נכניס קטע שזמן ההתחלה שלו הוא קרוב כמה שיותר לזמן הסיום של הקטע (המשימה) שכרגע סיימנו. נתחיל מהקטע שזמן הסיום שלו מינימלי.
4. הצעת שיפור: אם יש קטע שממש מוכל בקטע שכרגע רוצים להוסיף, נוסיף אותו במקום.

דוגמאות נגדיות:

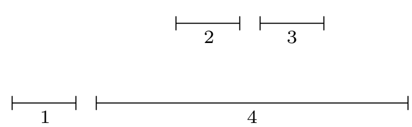
1. (הקטע הקצר ביותר) החמדן יחזיר $\{3\}$ אולם הפתרון האופטימלי הוא $\{1,2\}$




2. (התנגשות עם כמה שפחות קטעים) החמדן יחזיר $\{1,5,11\}$ והאופטימלי $\{1,5,9,10\}$




3. (זמן התחלה קרוב ביותר) החמדן יחזיר $\{1,4\}$ האופטימלי $\{1,2,3\}$



**אלגוריתם חמדן** נמיין את הקטעים בסדר עולה לפי זמני הסיום ובכל שלב נבחר את הקטע שמסתיים ראשון.

---

**אלגוריתם 2** אלגוריתם חמדן לפתרון בעיית שיבוץ משימות

1. עיבוד מוקדם : נמיין את הקטעים בסדר עולה לפי זמני הסיום שלהם. מעתה נניח כי $f_1 \leq f_2 \leq \cdots \leq f_n$.
2. אתחול : $A = [n]$ - מכילה את האינדקסים של הקטעים שעדיין ניתנים להוספה ; $S = \emptyset$ - תכיל את הפתרון החמדן.
3. איטרציה : בכל שלב נוסיף ל-$S$ את האינדקס $t$ הקטן ביותר מ-$A$. נמחק מ-$A$ את כל הקטעים שחותכים את קטע מספר $t$.
4. סיום : נעצור כאשר $A = \emptyset$ ונחזיר את $S$.

---

**זמן ריצה** שלב העיבוד המוקדם דורש $O(n \log n)$. מספר האיטרציות חסום ע״י $n$ (כי יש $n$ קטעים), ובכל איטרציה משווים את הקטע שמוסיפים מול קטעים שכבר הכנסנו. סה״כ $O(n^2)$.

**הערה 3.4.** ניתן לשפר את מימוש ההשוואה בכל איטרציה כדי להשיג יעילות של $O(n \log n)$.

# 4 שבוע 2 - הרצאה - 24.10.18

## 4.1 אלגוריתמים חמדניים: בעיית שיבוץ משימות - המשך

נראה כי האלגוריתם החמדן שהגדרנו מחזיר פתרון חוקי ואופטימלי.

**חוקיות** החוקיות ברורה מדרך פעולתו של האלגוריתם: ברגע שהאלגוריתם מוסיף קטע לפתרון, הוא מוחק מאוסף הקטעים המותרים להוספה ($A$) את כל הקטעים שמתנגשים עם הקטע שכרגע הוספנו.

**אופטימליות** ננסח למת החלפה.

**למה 4.1** (למת החלפה עבור בעיית שיבוץ משימות)**.** יהי $S = \{i_1, i_2, \ldots, i_k\}$ פתרון חוקי כלשהו לבעיה, כאשר $i_1 < i_2 < \cdots < i_k$ (כלומר, האינדקסים ממויינים לפי זמני הסיום של הקטעים המתאימים).
יהי $0 \le j \le k-1$ ויהי $t$ האינדקס המינימלי של הקטע שלא חותך את הקטעים עם אינדקסים $i_1, i_2, \ldots, i_j$.
אזי $S_1 = \{i_1, i_2, \ldots, i_j, t, i_{j+2}, \ldots, i_k\}$ הוא גם פתרון חוקי לבעיה.

הוכחה. צריך להוכיח כי $S_1$ הוא פתרון חוקי לבעיה, כלומר, לא קיים זוג של קטעים נחתכים. מחוקיות הפתרון $S$, מספיק לוודא שקטע מספר $t$ לא חותך אף קטע אחר בפתרון.
לפי בחירת $t$, קטע מספר $t$ בבירור לא חותך את הקטעים $i_1, i_2, \ldots, i_j$.
נוודא שלכל $j+2 \le m \le k$, קטע מספר $t$ לא חותך את קטע $i_m$.
אבחנה: אם נתונים שני קטעים זרים $[s,f]$, $[s',f']$ וגם $f' \ge f$ אז חייב להתקיים $s' > f$, נמחיש:  .
$S$ הוא פתרון חוקי, ולכן הקטע $i_m$ לא חותך את הקטע $i_{j+1}$. מכיוון ש-$f_{i_m} > f_{i_{j+1}}$, מתקיים גם $s_{i_m} > f_{i_{j+1}}$ לפי האבחנה דלעיל.
זמן הסיום של קטע מספר $t$ הוא הקטן ביותר מבין כל הקטעים שלא חותכים את $i_1, \ldots, i_j$.
הקטע $i_{j+1}$ לא חותך את הקטעים $i_1, \ldots, i_j$, ומכאן אנו מקבלים:

$$\underset{\substack{\uparrow \\ \text{מינימליות } t}}{f_t \le} f_{i_{j+1}} < s_{i_m} < f_{i_m}$$

נובע מכך שקטע מספר $t$ לא חותך את קטע $i_m$, כפי שרצינו להוכיח. $\square$

# 5 שבוע 3 - הרצאה - 28.10.18

## 5.1 אלגוריתמים חמדניים: בעיית שיבוץ משימות - המשך

**משפט 5.1.** הפתרון החמדן (ראו אלגוריתם 2) הוא פתרון אופטימלי לבעיה 3.2.

הוכחה. עבור פתרון חוקי $S : i_1 < i_2 < \cdots < i_m$ נגדיר את גודל הרישא המשותפת של $S$, $\ell$, עם הפתרון החמדן $G : g_1 < g_2 < \cdots < g_r$ באופן הבא:

- אם $i_1 = g_1, \ldots, i_r = g_r$ נאמר כי $\ell = r$.
- אחרת, $\ell$ מקיים: $i_1 = g_1, i_2 = g_2, \ldots, i_\ell = g_\ell$ וגם $i_{\ell+1} \neq g_{\ell+1}$. במקרה זה $\ell < r$.

יהי $S^*$ פתרון אופטימלי שהרישא המשותפת שלו עם הפתרון החמדן הינה מקסימלית מבין כל הפתרונות האופטימליים. נרצה להוכיח כי $S^* = G$ ובכך נוכיח אופטימליות של הפתרון החמדן.
נניח בשלילה כי $S^* \neq G$ ונבחין בין שתי אפשרויות:

1. $\ell = r$, כלומר $S^*$ מכיל את כל הקטעים $g_1, \ldots, g_r$ של הפתרון החמדן.

   מכיוון ש-$G \neq S^*$, קיים קטע נוסף $i_{r+1} \in S^*$. מחוקיות $S^*$, הקטע $i_{r+1}$ זר לקטעים $g_1, \ldots, g_r$.

   אולם, זוהי סתירה, כיוון שהאלגוריתם החמדן עוצר אחרי שהוא בוחר את הקטעים $g_1, \ldots, g_r$ (אם קיים קטע נוסף שזר לקטעים $g_1, \ldots, g_r$ אז $A \neq \emptyset$ והאלגוריתם היה ממשיך).

2. $\ell < r$, כלומר מתקיים $i_1 = g_1, \ldots, i_\ell = g_\ell$ וגם $i_{\ell+1} \neq g_{\ell+1}$.

   נפעיל את למת ההחלפה 4.1 על הפתרון $S^*$ עם $j = \ell$.

   לפי למת ההחלפה, הפתרון $S_1 = \{g_1, \ldots, g_\ell, \boldsymbol{g_{\ell+1}}, i_{\ell+2}, \ldots, i_k\}$ הוא פתרון חוקי (הבחירה החמדנית בשלב ה-$\ell+1$ היא $g_{\ell+1}$).

   קיבלנו אפוא פתרון חוקי ואופטימלי (כי הוא שווה בגודלו ל-$S^*$), שהרישא המשותפת שלו עם $G$ היא לפחות $\ell + 1$, בסתירה לבחירתו של $S^*$ כפתרון אופטימלי שהרישא שלו עם הפתרון החמדן היא מקסימלית.

□

## 5.2 אלגוריתמים חמדניים: קבוצת וקטורים בת"ל בעלת משקל מקסימלי

**בעיית מציאת קבוצת וקטורים בת"ל בעלת משקל מקסימלי**

**בעיה 5.2** (קבוצת וקטורים בת"ל בעלת משקל מקסימלי).
קלט:

1. קבוצה סופית של וקטורים $F = \{v_1, v_2, \ldots, v_n\}$ במרחב וקטורי $V$.
2. פונקציית משקל $\mu : F \to \mathbb{R}^+$.

פלט: תת-קבוצה $S \subseteq [n]$ כך שהוקטורים עם אינדקסים ב-$S$ בת"ל, וכך שהמשקל של $S$ מקסימלי בתנאי זה.
משקל של קבוצה $S$ מוגדר באופן הבא: $\mu(S) := \sum_{i \in S} \mu(v_i)$.

**דוגמה 5.3.** נתבונן ב-$V = \mathbb{R}^4$, $n = 5$. נסדר את הוקטורים ב-$F$ במטריצה:

$$F : \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{pmatrix} 1 & 2 & 5 & -1 & \pi \\ 0 & 2 & 4 & -17 & e \\ -1 & 2 & 3 & 3 & 0 \\ 0 & 2 & 4 & 5 & 1/2 \end{pmatrix} \end{array}$$

כאשר המשקלים הם: $\mu(v_1) = 2$, $\mu(v_2) = 4$, $\mu(v_3) = 8$, $\mu(v_4) = 2$, $\mu(v_5) = 1$.
נבחין כי קיימת תלות לינארית בין הוקטורים $\{v_1, v_2, v_3\}$ כי $v_3 = v_1 + 2 \cdot v_2$.
הפתרון האופטימלי: $\{3, 2, 4, 5\}$ (כלומר, לקחת את הוקטורים $v_3, v_2, v_4, v_5$).

**פתרון.** נציג אלגוריתם חמדן לפתרון הבעיה:

**אלגוריתם 3** אלגוריתם חמדן לפתרון בעיית מציאת קבוצת וקטורים בת״ל בעלת משקל מקסימלי

1. <u>עיבוד מוקדם</u>: נמיין את הוקטורים לפי משקלם בסדר יורד. מעתה נניח $\mu(v_1) \geq \mu(v_2) \geq \cdots \geq \mu(v_n)$. (וקטורים עם משקל $0$ או שלילי לא מסייעים להשגת משקל מקסימלי ואנו נתעלם מהם.)
2. <u>אתחול</u>: $A = [n]$, $S = \emptyset$.
3. <u>איטרציה</u>: בכל שלב נכניס ל-$S$ את האינדקס המינימלי מ-$A$. נמחק מ-$A$ את כל הוקטורים התלויים לינארית ב-$S$.
4. <u>סיום</u>: נעצור כאשר $A = \emptyset$ ונחזיר את $S$.

<u>זמן ריצה</u>: יש $n$ איטרציות, ובכל איטרציה אנו מבצעים פתרון של מערכת משוואות לינאריות כדי לקבוע מי הם הוקטורים ב-$A$ שתלויים לינארית ב-$S$ - פעולה זו לוקחת $O(n^3)$. בסה״כ זמן הריצה הוא $O(n^4)$.

<u>חוקיות הפתרון החמדן</u>: לפי דרך פעולתו של האלגוריתם, לאחר כל איטרציה, הקבוצה $S$ הינה בת״ל.
<u>אופטימליות הפתרון החמדן</u>:
ניתן להוכיח את אופטימליות הפתרון החמדן בעזרת למת החלפה בסגנון שהראינו בדוגמאות הקודמות. נציג ניסוח אפשרי:

**למה.** יהי $S = \{i_1, \ldots, i_k\}$ פתרון חוקי ($i_1 < i_2 < \cdots < i_k$). אזי בכל שלב $0 \leq j \leq k-1$ ניתן להחליף את הבחירה $i_j$ של הפתרון $S$ בבחירה חמדנית של הפתרון החמדן.

אולם, בעיה זו טומנת בחובה מבנה קומבינטורי מעניין שנקרא מטרואיד. ננסח למת החלפה מטרואידית.

**למה 5.4** (למת החלפה מטרואידית עבור בעיה 5.2)**.** יהיו $X, Y \subseteq V$ שתי תת-קבוצות סופיות של וקטורים בת״ל במרחב וקטורי $V$, כך שמתקיים $|X| < |Y|$. <u>אזי</u> קיים $y \in Y \setminus X$ כך ש-$X \cup \{y\}$ בת״ל.

הוכחה. נגדיר את $U$ להיות תת-מרחב וקטורי של $V$ הנפרש ע״י איברי $X$. מכיוון ש-$|X| < |Y|$ ו-$Y$ בת״ל, לא ייתכן כי $Y \subseteq U$ (אחרת, $Y$ הייתה קבוצה בת״ל ב-$U$ שגדולה מגודל מימדו).
לכן, קיים $y \in Y$ כך ש-$y \notin U$. כלומר, $y$ אינו תלוי לינארית באיברי $X$, ובפרט $y \notin X \implies X \cup \{y\}$ בת״ל. □

**משפט 5.5.** הפתרון החמדן (ראו אלגוריתם 3) הוא פתרון אופטימלי לבעיה 5.2.

הוכחה. יהי $S^*$ פתרון אופטימלי כלשהו, ויהי $G$ הפתרון החמדן.
נוכיח כי $|G| = |S^*|$. אמנם, אם $|G| < |S^*|$, לפי למת ההחלפה יש $g \in G \setminus S^*$ כך ש-$S_1 = S^* \cup \{g\}$ קבוצה בת״ל ולכן פתרון חוקי.
במקרה זה, $\mu(S_1) = \mu(S^* \cup \{g\}) = \mu(S^*) + \mu(g) > \mu(S^*)$, בסתירה לאופטימליות של $S^*$.
אם $|G| > |S^*|$, אז לפי למת ההחלפה, קיים $s \in S^* \setminus G$ כך ש-$G \cup \{s\}$ בת״ל. כלומר, הוקטור $s$ לא תלוי לינארית ב-$G$, בסתירה לכך שהאלגוריתם החמדן עצר אחרי בניית $G$ (הקבוצה $A$ תהיה לא ריקה שכן תכיל את $s$).
עתה נסמן, $k := |S^*| = |G|$. נרשום את שני הפתרונות בסדר יורד של משקלי הוקטורים.
נוכיח כי $\mu(G) = \mu(S^*)$ ומכאן תנבע אופטימליות $G$.
נניח בשלילה כי $\mu(G) \neq \mu(S^*)$, אז מכיוון ש-$S^*$ אופטימלי חייב להתקיים $\mu(G) < \mu(S^*)$:

$$\mu(G) = \sum_{i=1}^{k} \mu(g_i) < \sum_{i=1}^{k} \mu(s_i) = \mu(S^*)$$

זה גורר קיום אינדקס $1 \leq j \leq k$ כך ש-$\mu(g_j) < \mu(s_j)$.
נגדיר $X := \{g_1, g_2, \ldots, g_{j-1}\}$ ו-$Y := \{s_1, s_2, \ldots, s_j\}$. אז $X$ ו-$Y$ קבוצות בת״ל וגם $|X| < |Y|$.
מתקיימים תנאי למת ההחלפה, לכן קיים $1 \leq i \leq j$ כך ש-$X \cup \{s_i\}$ בת״ל.
מאחר ש-$s_i$ לא תלוי באיברי $X$, הוא מותר לבחירה באיטרציה ה-$j$ של האלגוריתם החמדן.
מצד שני, מתקיים $\mu(s_i) \geq \mu(s_j) > \mu(g_j)$, ולכן האלגוריתם החמדן טעה כאשר העדיף את $g_j$ על-פני $s_i$, סתירה. □

מיון

# 6 שבוע 3 - הרצאה - 31.10.18

## 6.1 אלגוריתמים חמדניים: מטרואידים - מבוא

נתבונן בבעיית האופטימיזציה הבאה:

**בעיית אופטימיזציה גנרית**

**בעיה 6.1** (בעיית אופטימיזציה גנרית).
קלט:

(1) קבוצה סופית $B = \{x_1, \ldots, x_n\}$;

(2) פונקציית משקל $\mu : B \to \mathbb{R}^+$;

(3) אוסף $\mathcal{F}$ של תת-קבוצות של $B$, המגדירות את מרחב הפתרונות החוקיים לבעיה.

פלט: $S \in \mathcal{F}$ תת-קבוצה של $B$ (חוקיות), בעלת משקל מקסימלי (אופטימליות). כאשר המשקל $S$ מוגדר כך:

$$\mu(S) := \sum_{s \in S} \mu(s)$$

נביא כעת דוגמאות לבעיות אופטימיזציה מסוג 6.1. בכל דוגמה נפרט את הקלט לבעיה, כאשר נבצע את ההתאמות הנדרשות:

1. בעיית התרמיל השלם.

   הקלט:

   (א) $B = \{x_1, \ldots, x_n\}$ - קבוצת הפריטים;

   (ב) $(v_1, w_1), \ldots, (v_n, w_n)$ כאשר $v_i$ ערך הפריט ה-$i$ ו-$w_i$ משקל הפריט ה-$i$;

   (ג) $W$ המשקל המקסימלי של התרמיל;

   (ד) פונקציית משקל $\mu : B \to \mathbb{R}^+$ כאשר $\mu(x_i) = v_i$ לכל $1 \leq i \leq n$;

   (ה) מרחב הפתרונות החוקיים $\mathcal{F} = \left\{ S \subseteq [n] : \sum_{i \in S} w_i \leq W \right\}$.

2. בעיית שיבוץ משימות.

   הקלט:

   (א) $B = [n]$;

   (ב) $[s_1, f_1], \ldots, [s_n, f_n]$;

   (ג) $\mathcal{F} = \{S \subseteq [n] :$ הקטעים עם אינדקסים ב-$S$ זרים זה לזה$\}$;

   (ד) $\mu : B \to \mathbb{R}^+$ אשר מוגדרת $\mu(i) = 1$ (בבעיה זו, אנו מחפשים $S \in \mathcal{F}$ כך ש-$|S|$ מקסימלי).

3. קבוצת וקטורים בת"ל בעלת משקל מקסימלי.

   הקלט:

   (א) $F = \{v_1, \ldots, v_n\}$, $n$ וקטורים;

   (ב) $B = [n]$;

   (ג) $\mathcal{F} = \{S \subseteq [n] :$ קבוצת הוקטורים עם אינדקסים ב-$S$ הנה בלתי תלויה לינארית$\}$;

   (ד) פונקציית המשקל $\mu : F \to \mathbb{R}^+$ היא חלק מהקלט לבעיה, נגדיר $\mu' : B \to \mathbb{R}^+$ ע"י $\mu'(i) = \mu(v_i)$.

**אבחנה:** בשלושת הדוגמאות האחרונות מתקיים $\mathcal{F} \neq \emptyset$ (קבוצת הפתרונות החוקיים אינה ריקה), $\mathcal{F}$ תורשתית (סגורה להכלה) - ז"א אם $T \in \mathcal{F}$ ו-$S \subseteq T$ אז $S \in \mathcal{F}$.

**אלגוריתם 4** אלגוריתם חמדן "גנרי" לפתרון בעיות מסוג בעיה 6.1

1. עיבוד מוקדם: נמיין את הפריטים בסדר משקלים יורד $\mu(x_1) \geq \mu(x_2) \geq \cdots \geq \mu(x_n) > 0$.
2. אתחול: $A = [n]$ (האינדקסים שניתן להוסיף), $S = \emptyset$ (הפתרון החמדן).
3. איטרציה: נעבור על $A$ ונמחק אינדקס של כל פריט $x \in A$ כך ש-$S \cup \{x\} \notin \mathcal{F}$. נעביר מ-$A$ ל-$S$ את האינדקס המינימלי.
4. סיום: נעצור כאשר $A = \emptyset$ ונחזיר את $S$.

## 6.2 אלגוריתמים חמדניים: מטרואידים - הגדרות

**הגדרה 6.2** (מטרואיד). הזוג $(B, \mathcal{F})$, כאשר $B$ קבוצה סופית ו-$\mathcal{F}$ היא אוסף תת-קבוצות לא ריק של $B$, ייקרא **מטרואיד**, אס"ם $\mathcal{F}$ מקיימת שתי תכונות:

1. $\mathcal{F}$ תורשתי: אם $T \in \mathcal{F}$ ו-$S \subseteq T$ אז $S \in \mathcal{F}$.
2. למת החלפה: אם $S, T \in \mathcal{F}$ ו-$|S| < |T|$ אז קיים איבר $t \in T \setminus S$ כך ש-$S \cup \{t\} \in \mathcal{F}$.

**משפט 6.3.** תהי נתונה בעיה אלגוריתמית מסוג בעיה 6.1. אם הזוג $(B, \mathcal{F})$ מהווה מטרואיד, אז האלגוריתם החמדן הגנרי מחזיר פתרון אופטימלי לכל פונקציית משקל $\mu$.

הוכחת המשפט דומה מאוד להוכחת משפט 5.5, והשלמת הפרטים הושארה כתרגיל.

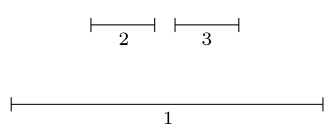
**משפט 6.4.** תהי נתונה בעיה אלגוריתמית מסוג בעיה 6.1. נניח כי משפחת הפתרונות החוקיים $\mathcal{F}$ לא ריקה ותורשתית, אך אינה מקיימת את למת ההחלפה. אזי קיימת פונקציית משקל $\mu$ עבורה האלגוריתם החמדן הגנרי לא מחזיר פתרון אופטימלי.

דוגמאות לבעיות שבהן למת ההחלפה לא מתקיימת:

- בעיית התרמיל השלם. $W = 50$, $(v_1, 25), (v_2, 25), (v_3, 50)$.

  נגדיר $S = \{3\}$ ו-$T = \{1,2\}$. $S, T$ מהוות פתרונות חוקיים, וגם $|S| < |T|$ אבל לא ניתן להעביר אף פריט מ-$T$ ל-$S$ ולכן למת ההחלפה לא מתקיימת.

- בעיית שיבוץ משימות. נתבונן ב-3 המשימות הבאות:




  נגדיר $S = \{1\}$ ו-$T = \{2,3\}$. $S, T$ מהוות פתרונות חוקיים, וגם $|S| < |T|$ אבל לא ניתן להוסיף קטע מ-$T$ ל-$S$ מבלי שתיווצר התנגשות.

הוכחת משפט 6.4 תושלם בשבוע הבא.

# 7 שבוע 4 - הרצאה - 4.11.18

## 7.1 אלגוריתמים חמדניים: מטרואידים - המשך

הוכחה (של משפט 6.4). מכיוון שתכונת ההחלפה לא מתקיימת עבור $\mathcal{F}$, קיימות $S,T \in \mathcal{F}$ כך ש-$|S| > |T|$ אך לכל $t \in T \setminus S$ מתקיים $S \cup \{t\} \notin \mathcal{F}$.

נגדיר פונקציית משקל $\mu : B \to \mathbb{R}^+$ באופן הבא:

$$\mu(x) = \begin{cases} 1 & x \in S \\ 1-\varepsilon & x \in T \setminus S \\ \varepsilon & \text{Otherwise} \end{cases}$$

כאשר $\varepsilon := \frac{|T|-|S|}{2|B|}$ (נשים לב ש-$0 < \varepsilon$ כי $|T| > |S|$).

נוכיח שהאלגוריתם החמדן יחזיר פתרון שאינו אופטימלי עבור פונקציית משקל זו.

על-פי דרך פעולתו, ב-$|S|$ האיטרציות הראשונות, האלגוריתם יבחר בכל איברי $S$.

באיטרציה הבאה, האלגוריתם יתחיל למחוק מ-$A$ את כל איברי $T \setminus S$ (כי למת ההחלפה לא מתקיימת עבור $T, S$), לכן המשקל הכולל של הפתרון החמדן, $G$, מקיים:

$$\begin{aligned} \mu(G) &\le \mu(S) + \mu((S \cup T)^c) \\ [\text{לפי הגדרת } \mu] &= 1 \cdot |S| + \varepsilon \cdot |(S \cup T)^c| \\ [\text{אי שוויון חזק כי } 1 \le |T|] &< |S| + \varepsilon \cdot |B| \\ [\text{הגדרת } \varepsilon] &= |S| + \frac{|T|-|S|}{2|B|} \cdot |B| \\ &= \frac{|T|+|S|}{2} \end{aligned}$$

לעומת זאת, $T \in \mathcal{F}$ ומתקיים

$$\begin{aligned} \mu(T) &\ge (1-\varepsilon) \cdot |T| \\ &= |T| - \varepsilon \cdot |T| \\ [|T| \le |B|] &\ge |T| - \varepsilon \cdot |B| \\ &= |T| - \frac{|T|-|S|}{2|B|} \cdot |B| \\ &= \frac{|T|+|S|}{2} \end{aligned}$$

מכאן קיבלנו $\mu(G) < \frac{|T|+|S|}{2} \le \mu(T)$, ולכן הפתרון החמדן איננו אופטימלי. $\square$

## 7.2 אלגוריתמים דינאמיים: חישוב של פונקציות המוגדרות רקורסיבית

אלגוריתם דינאמי פועל לפי רעיון של "הפרד ומשול": מחלקים את הבעיה הנתונה למספר תת-בעיות, פותרים את תתי-הבעיות, ומאחדים את פתרונן לפתרון הבעיה כולה.

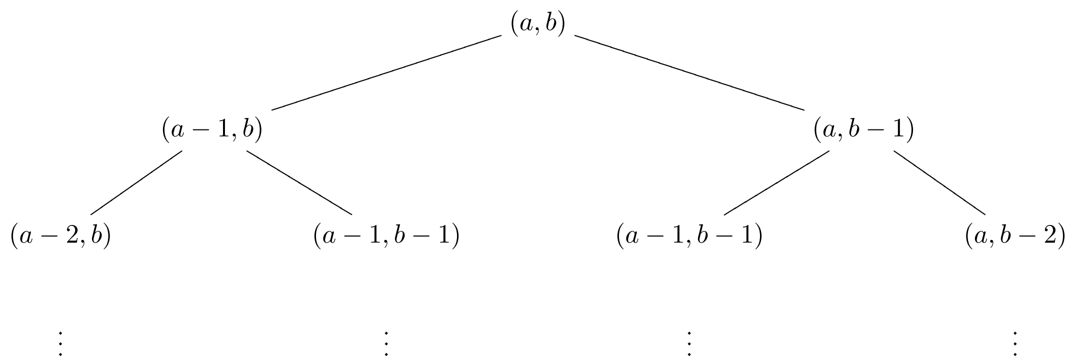
**בעיה 7.1** (חישוב של פונקציות המוגדרות רקורסיבית). תהי $T(a,b)$ פונקציה של שני משתנים $a,b \in \mathbb{N} \cup \{0\}$ המוגדרת באופן רקורסיבי:

$$T(a,b) = \begin{cases} 1 & a = 0 \vee b = 0 \\ T(a-1,b) + T(a,b-1) & \text{Otherwise} \end{cases}$$

נרצה לחשב את $T(2018, 2018)$.

הדרך הנאיבית לפתור את השאלה היא באמצעות אלגוריתם רקורסיבי שמחקה את כלל הנסיגה שעלינו לחשב.

ניתן לתאר את האלגוריתם באמצעות עץ הקריאות הרקורסיביות שלו :




נשים לב שעבור $T\left(2018,2018\right)$ אנו מקבלים עץ בינארי מלא שהוא לפחות בגובה $2018$ (שהרי המסלול הקצר ביותר בעץ הוא השמאלי ביותר או הימני ביותר) ולכן עץ הקריאות מכיל לפחות $2^{2018+1}-1$ קודקודים, שמתפרשים כקריאות רקורסיביות - מספר אקספוננציאלי.
בשיטה זו אנו מחשבים את הערכים של הקודקודים מספר פעמים, למשל $T\left(a-1,b-1\right)$ יחושב פעמיים. כדי להימנע מכך, נרצה לחשב פעם אחת את הערך $T\left(a-1,b-1\right)$ "ולזכור" אותו לפעם הבאה.
נשים לב לכך שלצורך החישוב של $T\left(2018,2018\right)$ מספיק לפתור פעם אחת כל תת-הבעיות מהצורה $T\left(a,b\right)$ עבור $0\le a,b\le 2018$, סה"כ $2019^2$ תתי-בעיות.
נשתמש בתובנה זו כדי לבנות אלגוריתם יעיל לפתרון הבעיה.
הרעיון הטכני שנשתמש בו הוא ארגון המידע (ערכי הבעיות החלקיות) בטבלה.
עבור הבעיה הנוכחית, נגדיר טבלה ריבועית, $R$, עם $2019^2$ תאים, כאשר בתא $R\left(a,b\right)$ נכתוב את הערך $T\left(a,b\right)$.
נדאג למלא את הטבלה באופן שבו כאשר ניגש לתא $(a,b)$ שני התאים $(a-1,b)$ ו-$(a,b-1)$ יהיו מלאים.

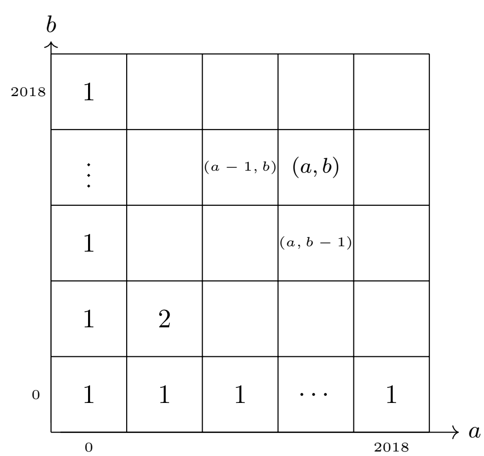
---

**אלגוריתם 5** אלגוריתם דינאמי לפתרון בעיה 7.1

---

1. נגדיר טבלה ריבועית $R\left(a,b\right)$ עבור $0\le a,b,\le 2018$.
2. נאתחל את מקרי הבסיס : את הערכים $R\left(a,0\right)$ ($0\le a\le 2018$) ו-$R\left(0,b\right)$ ($0\le b\le 2018$).
3. נמלא באיטרציה ה-$i$ ($0\le i\le 2n$) את האלכסון $a+b=i$.
4. נחזיר את הערך $R\left(2018,2018\right)$.

---

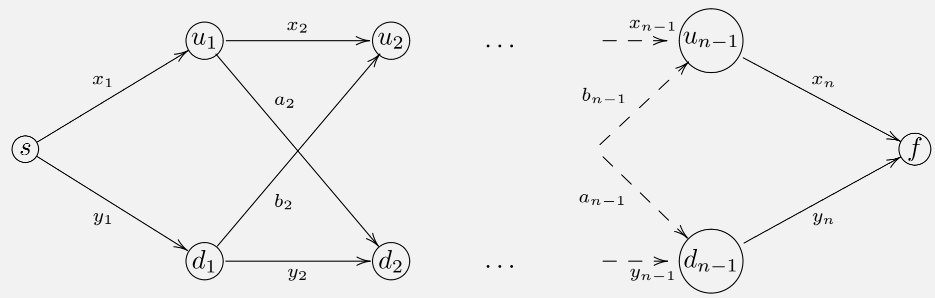



איור 7.1 : המחשה לטבלה $R$ באלגוריתם 5

## 7.3 אלגוריתמים דינאמיים: בעיית ניתוב משימות

**בעיית ניתוב משימות**

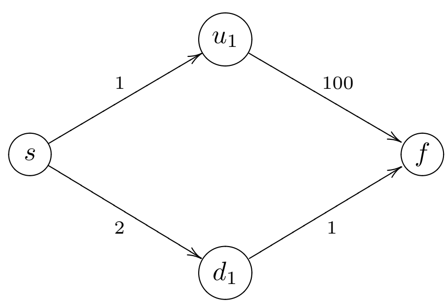
**בעיה 7.2** (בעיית ניתוב משימות).
סיפור המסגרת: במפעל יש 2 פסי ייצור זהים ובכל אחד מהם יש $n$ תחנות עבודה. למעברים בין תחנות יש מחירים שונים. פריט עובר תהליך ייצור במפעל, ונרצה לבחור מסלול עבור הפריט שמחירו מינימלי.



הקלט: גרף מכוון $G=(V,E)$ כנ״ל, והמחירים $x_1,\ldots,x_n,a_2,\ldots,a_{n-1},y_1,\ldots,y_n,b_2,\ldots,b_{n-1}$.
הפלט: מסלול בעל מחיר מינימלי מ-$s$ ל-$f$.

דרכי פתרון אפשריים:

- מעבר על כל האפשרויות: נעבור על כל המסלולים האפשריים מ-$s$ ל-$f$. עבור כל מסלול נחשב את מחירו, ונחזיר את המסלול בעל המחיר המינימלי. מס' המסלולים הכולל הוא $2^{n-1}$ - לא יעיל.
- בעיה מוכרת: בעיה 7.2 היא מקרה פרטי של בעיית מציאת מסלול מינימלי בגרף בין $s$ ל-$f$. בעיה זו ניתנת לפתרון ע״י האלגוריתם של Dijkstra בזמן ריצה $O\left(|V|\log|V|+|E|\right)$ שמתורגם ל-$O\left(n\log n\right)$ בסימוני הבעיה שלנו. נראה שבמקרה שלנו קיימת תכונה שתאפשר לנו לפתור את הבעיה בזמן יעיל יותר.
- אלגוריתם חמדני: רעיון מיידי הוא בכל שלב לבחור ללכת בכיוון הזול מבין השניים. רעיון זה לא צולח, הנה דוגמה נגדית:



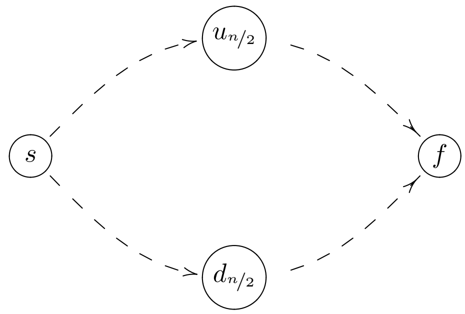
כאן האלגוריתם החמדני יבחר במסלול $s\rightsquigarrow u_1\rightsquigarrow f$ במחיר של 101, בעוד שהמסלול האופטימלי הוא $s\rightsquigarrow d_1\rightsquigarrow f$ במחיר של 3.

נבחין בעיקרון מרכזי שקרוי על שמו של החוקר ריצ'רד בלמן אשר הציג לראשונה את התכנון הדינמי (Dynamic Programming).

**הצעה 7.3** (עקרון האופטימליות של Bellman). כל תת-פתרון של פתרון אופטימלי הוא בעצמו אופטימלי.

עקרון זה מלמד שאם אנו רוצים למצוא את המסלול הזול ביותר מ-$s$ ל-$f$, נוכל למצוא את המסלול הזול ביותר מ-$s$ לקודקוד אמצע מסוים $w$, ולאחר מכן למצוא את המסלול הזול ביותר מ-$w$ ל-$f$.

נדגים עקרון זה באמצעות הדיאגרמה הבאה :




על-מנת למצוא מסלול אופטימלי מ-$s$ ל-$f$ נוכל למצוא מסלול אופטימלי בין $s$ ל-$u_{n/2}$ (למשל, ע״י מעבר על כל המסלולים האפשריים ובחירת הזול ביותר), ומשם מסלול אופטימלי בין $u_{n/2}$ ל-$f$.
באותו אופן, נמצא מסלול אופטימלי בין $s$ ל-$d_{n/2}$ ומ-$d_{n/2}$ ל-$f$. כאן אנו מחשבים 4 מסלולים אופטימליים בעלות של לכל היותר $4 \cdot 2^{n/2}$ לעומת $2^{n-1}$ מסלולים בסה״כ.
הרעיון המרכזי הוא שמסלול אופטימלי בין $s$ ל-$f$ יעבור דרך $u_{n/2}$ או $d_{n/2}$, וברגע שהגענו לקודקוד אמצע, כבר לא משנה לנו כיצד הגענו לשם, וכל כוחנו מוקדש למציאת מסלול זול ביותר בין קודקוד האמצע הזה לקודקוד הסיום.

# 8 שבוע 4 - הרצאה - 7.11.18

## 8.1 אלגוריתמים דינאמיים: בעיית ניתוב משימות - המשך

נשוב לפתרון בעיה 7.2 באמצעות תכנון דינאמי.

**כלל פיצול** עלינו להגדיר "כלל פיצולי" של הבעיה הגדולה לתת-בעיות.
נתבונן בהחלטה הראשונה - יש שתי אפשרויות: ללכת מ-$s$ ל-$u_1$ או מ-$s$ ל-$d_1$.
החלטה זו מפצלת את הבעיה לשתי תת-בעיות: אם החלטנו ללכת מ-$s$ ל-$u_1$ אז עלינו למצוא מסלול אופטימלי בין $u_1$ ל-$f$. מדובר בבעיה קטנה יותר.
אם נמשיך לפצל את תת-הבעיות הקטנות יותר באותו אופן, נגיע למשפחה של תת-בעיות שנראית כך: למצוא מסלול אופטימלי מאחת התחנות $s, d_1, \ldots, d_{n-1}, u_1, \ldots, u_{n-1}$ ל-$f$.
בסה"כ יש $2n-1$ תת-בעיות שעלינו לפתור.
נרשום את **תתי-הבעיות** בצורה מסודרת: מציאת המחיר האופטימלי עבור מסלול מ-$u_k$ ל-$f$ לכל $1 \leq k \leq n-1$, מציאת המחיר האופטימלי עבור מסלול מ-$d_k$ ל-$f$ לכל $1 \leq k \leq n-1$, ומציאת המחיר האופטימלי עבור מסלול מ-$s$ ל-$f$ (הבעיה המקורית שנפתור בעזרת תתי-הבעיות האחרות שהגדרנו).

**בניית כלל נסיגה** נסמן ב-$p^*$ את מחיר המסלול האופטימלי מ-$s$ ל-$f$. נסמן ב-$p_u[k]$ את המחיר של המסלול האופטימלי מ-$u_k$ ל-$f$.
נסמן ב-$p_d[k]$ את המחיר של המסלול האופטימלי מ-$d_k$ ל-$f$.
מתקיים $p^* = \min\left(x_1 + p_u[1], y_1 + p_d[1]\right)$. לכל $1 \leq k \leq n-1$ מתקיים

$$p_u[k] = \begin{cases} x_n & k = n-1 \\ \min\left(x_{k+1} + p_u[k+1], a_{k+1} + p_d[k+1]\right) & k < n-1 \end{cases}$$

כאשר הביטוי $x_{k+1} + p_u[k+1]$ מתאר את ההחלטה ללכת מ-$u_k$ ל-$u_{k+1}$ והביטוי $a_{k+1} + p_d[k+1]$ מתאר את ההחלטה מ-$u_k$ ל-$d_{k+1}$.
נוסחה דומה תקפה עבור $p_d$.

---

**אלגוריתם 6** אלגוריתם לפתרון בעיית ניתוב משימות (בעיה 7.2)

1. נבנה טבלה בגודל $2n-1$.
2. נמלא את העמודה ה-$n-1$ (תנאי הבסיס).
3. נמלא את הטבלה מימין לשמאל, כאשר באיטרציה ה-$t$ ($1 \leq t \leq n-1$) נמלא את העמודה $k = n-t$ לפי כלל הנסיגה דלעיל.
4. באיטרציה ה-$n$ נחשב את $p^*$, ונחזיר את ערכו.

| | 1 | $\ldots$ | $n-2$ | $n-1$ | |
|---|---|---|---|---|---|
| $p^*$ | $p_u[1]$ | $\ldots$ | $p_u[n-2]$ | $x_n$ | $u$ |
| | $p_d[1]$ | $\ldots$ | $p_d[n-2]$ | $y_n$ | $d$ |

---

**הערה 8.1.** נשים לב שבאלגוריתם 6 אנו מחזירים את הערך של המסלול האופטימלי מ-$s$ ל-$f$ ולא את המסלול עצמו.
כדי להחזיר את המסלול האופטימלי, נזכור בעת מילוי כל תא את הבחירה שנעשתה בזמן חישוב המינימום.

**זמן ריצה:** מחיר מילוי כל תא בטבלה הוא $O(1)$ בהינתן העמודה מימינו. עלינו למלא $O(n)$ תאים, ולכן זמן הריצה הוא $O(n)$.

# 9 שבוע 5 - הרצאה - 11.11.18

## 9.1 אלגוריתמים דינאמיים: בעיית כפל מטריצות

**שאלה:** תהי $A$ מטריצה מסדר $n \times t$ ($n$ - מספר השורות, $t$ - מספר העמודות), תהי $B$ מטריצה מסדר $t \times m$.
כמה פעולות כפל נדרשות בכדי לחשב את המטריצה $C = AB$?

**תשובה:** לכל $1 \leq i \leq n$ ולכל $1 \leq j \leq m$ נחשב את $C_{ij}$ ע״י הנוסחה

$$C_{ij} = \sum_{k=1}^{t} A_{ik} B_{kj}$$

יש $n \cdot m$ קואורדינטות במטריצת המכפלה $C$, ולכל קואורדינטה אנו מבצעים $t$ פעולות כפל.
סה״כ נדרשים $n \cdot t \cdot m$ פעולות כפל.

**סימון:** אם $A$ מטריצה מסדר $n \times t$ ו-$B$ מטריצה מסדר $t \times m$, נרשום

$${}_nC_m = {}_nA_t B_m$$

**דוגמה:** יהיו $A, B, C$ מטריצות מלבניות ונרצה לחשב את המטריצה $D = ABC$. לפי אסוציאטיביות כפל מטריצות, יש לנו שתי דרכים לחשב את המכפלה:

1. $D = (AB)\, C$
2. $D = A\,(BC)$

מסתבר שהבחירה בין שתי האפשרויות האלה יכולה לשנות את מספר פעולות הכפל הנדרשות.
לצורך הדגמה נרשום

$${}_{10}D_{100} = {}_{10}A_{50} B_{20} C_{100}$$

נחשב את מספר פעולות הכפל הדרושות בשני המקרים:

1. $D = {}_{10}\,(AB)_{20}\, C_{100}$. כדי לכפול את $AB$ אנו מבצעים $10 \cdot 50 \cdot 20 = 10^4$ פעולות כפל, ולאחר מכן אנו מבצעים עוד $10 \cdot 20 \cdot 100 = 2 \cdot 10^4$ פעולות כפל, ובסה״כ $3 \cdot 10^4$ פעולות כפל.
2. $D = {}_{10}A_{50}\,(BC)_{100}$. כדי לכפול את $BC$ אנו מבצעים $50 \cdot 20 \cdot 100 = 10^5$ פעולות כפל, ולאחר מכן עוד $10 \cdot 50 \cdot 100$ פעולות כפל, בבירור סידור זה ״יקר״ יותר מהאפשרות הקודמת.

דוגמה זו ממחישה כי בחירת מיקום הסוגריים משפיעה על מספר פעולות הכפל הכוללני. נתעניין בתכנון אלגוריתם שמחזיר את הסידור האופטימלי של סוגריים כדי לקבל מספר פעולות כפל מינימלי. ננסח זאת במדויק בבעיה הבאה.

**בעיית כפל מטריצות**

**בעיה 9.1** (בעיית כפל מטריצות). קלט: $n+1$ מספרים טבעיים $p_0, \ldots, p_n$ המציינים מימדים של $n$ מטריצות מלבניות $A_1, \ldots, A_n$, כאשר המטריצה ה-$i$ היא מסדר $p_{i-1} \times p_i$ (כלומר, $p_{i-1}$ שורות ו-$p_i$ עמודות).
פלט: חלוקת הסוגריים (ז״א קביעת סדר פעולות הכפל) לצורך החישוב היעיל ביותר מבחינת מספר הכפלים הנדרש כדי לחשב את המכפלה

$$B = A_1 A_2 \cdots A_n$$

אנו נתעניין באלגוריתם תכנון דינאמי כדי לפתור את בעיה 9.1.
נבחן שתי דרכים אפשריות לפצל את הבעיה לתתי-בעיות:

1. מפצלים לפי פעולת הכפל האחרונה

$$B = (A_1 A_2 \cdots A_k)\,(A_{k+1} \cdots A_n)$$

במקרה זה, תת-הבעיות הם חלוקת הסוגריים במכפלה $A_i \cdots A_j$ ($i \leq j$) לצורך חישוב המכפלה עם מספר כפלים מינימלי.
מספר תת-הבעיות הוא $O\left(n^2\right)$ (ביתר דיוק, $\binom{n}{2} + n = \Theta\left(n^2\right)$).

2. מפצלים לפי פעולת הכפל הראשונה

$$B = A_1 \cdots A_{k-1}\left(A_k A_{k+1}\right) \cdots A_n$$

במקרה זה, קשה לתאר את תת-הבעיות, ומספר תת-הבעיות הוא אקספוננציאלי.

אנו נתכנן אלגוריתם תכנון דינאמי עם פיצול הבעיה לפי פעולת הכפל האחרונה.

**תתי-בעיות:** מציאת מספר הכפלים המינימלי הנדרש לחישוב המכפלה $A_i A_{i+1} \ldots A_j$, לכל $1 \le i \le j \le n$.

**נוסחת הרקורסיה:** נסמן ב-$P[i,j]$ את המחיר (מס' הכפלים) האופטימלי לחישוב המכפלה $A_i A_{i+1} \cdots A_j$.
הפלט המבוקש לבעיה כולה הוא $P[1,n]$, ונוסחת הרקורסיה הכללית נתונה ע"י

$$P[i,j] = \begin{cases} 0 & i=j \\ \min_{i \le k \le j-1}\left\{P[i,k] + P[k+1,j] + p_{i-1}p_k p_j\right\} & i<j \end{cases}$$

במקרה $i=j$ אין מטריצות לכפול ולכן מספר הכפלים שווה 0.
אחרת, אנו עוברים על כל האינדקסים $k$ המציינים את "מיקום הפיצול" עבור פעולת הכפל האחרונה. אם פיצלנו בנקודה $k$ כלשהי, אז מספר פעולות הכפל יהיה סכום תתי-הבעיות $P[i,k]$ (המכפלה של $A_i \cdots A_k$) ו-$P[k+1,j]$ (המכפלה של $A_{k+1} \cdots A_j$) ולא נשכח להוסיף את העלות של המכפלה $\left(A_i \cdots A_k\right)\left(A_{k+1} \cdots A_j\right)$.
סדר המטריצה $A_i \cdots A_k$ הוא $p_{i-1} \times p_k$, סדר המטריצה $A_{k+1} \cdots A_j$ הוא $p_k \times p_j$, ולכן מספר הכפלים הנדרשים לחישוב המכפלה $\left(A_i \cdots A_k\right)\left(A_{k+1} \cdots A_j\right)$ הוא $p_{i-1} \cdot p_k \cdot p_j$.

**האלגוריתם:** נבנה טבלה ריבועית $T[i,j]$, $1 \le i,j \le n$, כאשר בתא ה-$[i,j]$, $i \le j$, נרשום את $P[i,j]$ ועבור $i > j$ נרשום $T[i,j] = 0$.
הפתרון לבעיה כולה יימצא בתא $T[1,n]$.

**דוגמה למילוי טבלה** עבור $n=3$, מטריצות $A_1$ מסדר $10 \times 50$, $A_2$ מסדר $50 \times 20$ ו-$A_3$ מסדר $20 \times 100$ ומכפלה

$$B = A_1 A_2 A_3$$

אנו מקבלים את הטבלה הבאה:

| $j$ | | | | |
|---|---|---|---|---|
| 3 | $3 \cdot 10^4$ | $10^5$ | 0 | |
| 2 | $10^4$ | 0 | 0 | |
| 1 | 0 | 0 | 0 | |
| | 1 | 2 | 3 | $i$ |

**אופן מילוי הטבלה:** נמלא את הטבלה ב-$n$-איטרציות כאשר באיטרציה ה-$d$ נמלא את כל התאים $T[i,j]$ הנמצאים על האלכסון $j-i=d$ ($d = 0,1,\ldots,n-1$).
עבור $d=0$ נמלא 0 בכל תא $[i,j]$. למילוי האלכסונים נשתמש בנוסחת הרקורסיה שפיתחנו קודם לכן, נרשום אותה שוב במונחים של $T$, לשם נוחות:

$$T[i,j] = \begin{cases} 0 & i>j \\ 0 & i=j \\ \min_{i \le k \le j-1}\left\{T[i,k] + T[k+1,j] + p_{i-1}p_k p_j\right\} & i<j \end{cases}$$

נצדיק מדוע התאים $[i,k]$ ו-$[k+1,j]$, $1 - i \le k \le j$, בחישוב ה-min אכן מלאים בזמן מילוי האלכסון $j-i=d$. נשים לב כי

$$\underset{\underset{k<j}{\uparrow}}{k - i < j} - i = d$$

וגם

$$j - (k+1) \le j - 1 - i = d - 1 < d$$
$$\underset{k \ge i}{\uparrow}$$

לכן התאים $T[i,k]$ ו-$T[k+1,j]$, כאשר $i \le k \le j-1$, כבר מולאו בעת מילוי האלכסונים הקודמים.

**חילוץ הפתרון:** כדי להחזיר את חלוקת הסוגריים האופטימלית נזכור בעת מילויו של כל תא $[i,j]$ את הבחירה של $i \leq k \leq j-1$ המשיגה את המינימום בנוסחת הרקורסיה.
בחירת $k$ הזו קובעת את חלוקת הסוגריים האופטימלית לפתרון תת-הבעיה המתאימה.

**זמן ריצה:** חישוב כל תא $T\left[i,j\right]$ ניתן לביצוע ב-$O\left(n\right)$ פעולות. לכן כל הטבלה ניתנת למילוי ב-$O\left(n^3\right)$ פעולות.

**הוכחת נכונות:** נוכיח את הטענה הבאה:

**טענה 9.2.** לכל $1 \leq i \leq j \leq n$ מתקיים $T\left[i,j\right] = P\left[i,j\right]$.

הוכחה. נוכיח באינדוקציה על אופן מילוי הטבלה, כלומר על $d = j - i$.
בסיס: $d = 0$. מתקיים $T\left[i,j\right] = 0 = P\left[i,j\right]$.
צעד: נניח נכונות הטענה עבור לכל היותר $d-1$ ונראה את נכונותה עבור $d$. מתקיים, לפי נוסחת הרקורסיה

$$T\left[i,j\right] = \min_{i \leq k \leq j-1} \left\{T\left[i,k\right] + T\left[k+1,j\right] + p_{i-1}p_kp_j\right\}$$

כעת, לכל $i \leq k \leq j-1$, מאחר ש-$k - i < j - i = d$ וגם $j-(k+1) \leq j-1-i < d$ אנו מקבלים שלפי הנחת האינדוקציה $T\left[i,k\right] = P\left[i,k\right]$ ו-$T\left[k+1,j\right] = P\left[k+1,j\right]$.

$$T\left[i,j\right] = \min_{i \leq k \leq j-1} \left\{P\left[i,k\right] + P\left[k+1,j\right] + p_{i-1}p_kp_j\right\} = P\left[i,j\right]$$

ובכך סיימנו את צעד האינדוקציה. $\square$

# 10 שבוע 5 - הרצאה - 14.11.18

## 10.1 אלגוריתמים דינאמיים: בעיית התרמיל השלם

בעיית התרמיל השלם

**בעיה 10.1 (בעיית התרמיל השלם).**
קלט:

(1) $W$ - המשקל המירבי של התרמיל;

(2) $n$ זוגות של מספרים $(v_1, w_1), \dots, (v_n, w_n)$, כאשר $v_i$ - ערך הפריט ה-$i$, $w_i$ - משקל הפריט ה-$i$.

הפריטים אינם ניתנים לחלוקה (בשונה מבעיית התרמיל השברית - בעיה 1.2).
פלט:
תת-קבוצה $S \subseteq [n]$ של אינדקסים, כך שמתקיים:

(1) $\sum_{i\in S} w_i \le W$ - אילוץ המשקל;

(2) $\sum_{i\in S} v_i$ מירבי בתנאי זה.

ניסיונות לפתרון:

- מעבר על כל האפשרויות: מעבר על כל תת-קבוצה של אינדקסים מ-$[n]$ ובחירת הקבוצה עם המחיר המשתלם ביותר. לוקח $\Omega(2^n)$ פעולות - לא יעיל.
- אלגוריתם חמדני:
  - "תמיד נכניס את הפריט היקר ביותר", לא עובד.
    למשל, $n=3$, $W=50$ והפריטים $(v_1=150, w_1=50), (100,25), (100,25)$.
    הפתרון האופטימלי הוא $S^*=\{2,3\}$, אולם ע"פ הכלל החמדני נקבל את הפתרון $G=\{1\}$ שאיננו אופטימלי.
  - "תמיד נכניס את הפריט בעל הערך הסגולי $r_i=\frac{v_i}{w_i}$ הגבוה ביותר", לא עובד.
    למשל, $n=3$, $W=50$, $(v_1=150, w_1=30), (100,25), (100,25)$.
    גם במקרה הזה, ע"פ הכלל החמדני נקבל $G=\{1\}$, פתרון שאינו אופטימלי לעומת $S^*=\{2,3\}$.

נפנה לאלגוריתם תכנון דינאמי. נציע דרכים לפצל את הבעיה לתת-בעיות:

1. נקבע סדר מעבר על הפריטים, למשל, $1,2,\dots,n$, ועבור פריט בשלב הנוכחי נחליט האם להכניס אותו או לא. זה יפצל את הבעיה ל-2.

2. נפצל את הבעיה ל-$n$ תתי-בעיות לפי מספר הפריטים שייכנסו לתרמיל בפתרון האופטימלי.

בנוגע להצעה השניה, ייתכן שקשה לתאר נוסחת רקורסיה. נמשיך עם ההצעה הראשונה.

**תתי-בעיות:** מציאת השווי המקסימלי של תרמיל כאשר קבוצת הפריטים שניתן להכניס אליו היא תת-קבוצה של הפריטים $i,\dots,n$, ומגבלת המשקל היא $u$, לכל $1\le i\le n$ ולכל $0\le u\le W$.

**נוסחת רקורסיה:** נסמן ב-$K[i,u]$ את הערך האופטימלי לפתרון הבעיה הבאה: מציאת תת-קבוצה של פריטים מתוך הפריטים $i,i+1,\dots,n$ כאשר המשקל המירבי שיכול לשאת התרמיל בשלב זה הוא $u$, ושווי הפריטים יהיה מקסימלי.
פתרון הבעיה כולה יינתן ע"י $K[1,W]$.
נרשום את נוסחת הרקורסיה עבור הפיצול הראשון:

$$K[1,W]=\max\left\{\underbrace{K[2,W]}_{\text{הפריט הראשון לא נכנס}},\ \overbrace{K[2,W-w_1]+v_1}^{\text{הפריט הראשון כן נכנס}}\right\}$$

נוסחת הרקורסיה במקרה הכללי הינה:

$$K[i,u]=\begin{cases} i=n, w_n>u: & 0 \\ i=n, w_n\le u: & v_n \\ i<n, w_i>u: & K[i+1,u] \\ i<n, w_i\le u: & \max\{K[i+1,u], K[i+1,u-w_i]+v_i\} \end{cases}$$

מכאן, כל תת הבעיות הן מהצורה $K[i,u]$ עבור $1\le i\le n$ ו-$u$ משקל כלשהו.

**הגדרת טבלה:** נגדיר טבלה $T$, כאשר בתא ה-$[i,u]$ נציב את הערך $K[i,u]$.

**מהו מספר השורות בטבלה $T$? כלומר, מהו מס' הערכים השונים האפשריים עבור $u$?**
עבור $i$ מסוים, המשקל שנשאר בתרמיל תלוי בהחלטות הקודמות שביצענו, ושייך לקבוצת הערכים

$$\left\{W-\sum_{\substack{t\in T\\ T\subseteq[i-1]}} w_t\right\}$$

אפשר לבחור משקלים $w_1,\dots,w_{i-1}$ כך שכל הסכומים בביטוי דלעיל יהיו שונים $\Leftarrow$ בסה"כ $2^{i-1}$ ערכים אפשריים שונים עבור $u$.
זה אומר שגודל הטבלה הוא אקספוננציאלי.
**האם זה מפתיע שלא הצלחנו לתכנן אלגוריתם יעיל לפתרון הבעיה?**
לא, בעיה זו נחשבת NP-קשה. ובכל זאת, נוכל להשתמש באלגוריתם שתיכננו אם נבצע שתי הנחות מקלות:

- כל המשקלים $\{w_i\}_{i=1}^n$ של הפריטים הם מספרים שלמים, וכך גם $W$.
- $W$ פולינומיאלי ב-$n$.

מההנחה המקלה הראשונה עולה כי הערכים האפשריים של המשקל $u$ הם בתחום $0\le u\le W$ כאשר $u$ מספר שלם.
מכאן נובע, שגודל הטבלה הוא $n\cdot(W+1)$, כיוון שיש $n$ אפשרויות עבור $i$ ו-$W+1$ אפשרויות עבור $u$.
נוכל למלא כל תא בטבלה ב-$O(1)$ ולכן זמן הריצה יהיה $O(n\cdot W)$.
**האם מדובר בפתרון יעיל?**
התשובה לשאלה זו תלויה בגודל של $W$. אם $W=2^n$ אז זמן הריצה הוא $n\cdot 2^n$.
אולם, מההנחה המקלה השניה, $W$ פולינומיאלי ב-$n$, ולכן הפתרון יעיל.

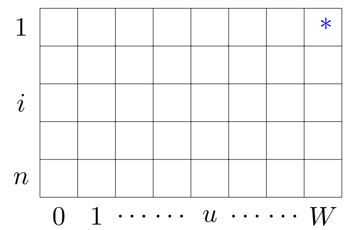
בשיעור הבא נשלים לתאר את מילוי הטבלה.

# 11 שבוע 6 - הרצאה - 18.11.18

## 11.1 אלגוריתמים דינאמיים: בעיית התרמיל השלם - המשך

נשלים את חובנו מהשבוע הקודם, ונגדיר במדויק את הטבלה ואופן מילויה עבור בעיה 10.1.

**הגדרת טבלה:** מכיוון שכל משקלי הפריטים הם מספרים שלמים, וגם $W$ מספר שלם, הערך של $u$ הוא מס' שלם בין 0 ל-$W$.
נבנה טבלה $T$ עם $n$ שורות ו-$W+1$ עמודות. ערך הפתרון האופטימלי יימצא בתא $T\left[1,W\right]$.



<u>דוגמה:</u>
$n=3$, $W=50$ והפריטים $\left(v_1=150, w_1=30\right), \left(100,25\right), \left(100,25\right)$.
הפתרון האופטימלי לבעיה הוא $S^*=\{2,3\}$ והערך האופטימלי (הרווח) הוא 200.
נדגים את מילוי הטבלה לפי נוסחת הרקורסיה שפותחה:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | $\cdots$ | 0 | 100 | $\cdots$ | 100 | 150 | $\cdots$ | 150 | 200 |
| 2 | 0 | 0 | $\cdots$ | 0 | 100 | $\cdots$ | 100 | 100 | $\cdots$ | 100 | 200 |
| 3 | 0 | 0 | $\cdots$ | 0 | 100 | $\cdots$ | 100 | 100 | $\cdots$ | 100 | 100 |
| | 0 | 1 | $\cdots$ | 24 | 25 | $\cdots$ | 30 | 31 | $\cdots$ | 49 | 50 |

**אופן מילוי:** נמלא את הטבלה ב-$n$ איטרציות, כאשר באיטרציה הראשונה נמלא את השורה ה-$n$ המתאימה לתנאי השפה של הרקורסיה, ובאיטרציה ה-$i$ נמלא את השורה ה-$n-i+1$ ($1\le i\le n$).

**חילוץ הפתרון:** התא $T\left[1,W\right]$ מכיל את הפתרון של הבעיה כולה. כדי להחליט אילו פריטים להכניס לתרמיל, נזכור בעת מילוי כל תא האם הפריט נבחר או לא.

**זמן ריצה:** מילוי של תא בשורה ה-$i$, בהינתן שהשורה ה-$i+1$ כבר מולאה, עולה $O\left(1\right)$.
גודל הטבלה הוא $O\left(n\cdot W\right)$ ובסה"כ קיבלנו שזמן הריצה של האלגוריתם הוא $O\left(n\cdot W\right)$.

## 11.2 אלגוריתמי קירוב: בעיית חלוקת משימות בין מכונות (Load Balancing)

ישנן בעיות שלא ידוע להן פתרון בזמן יעיל, לפיכך, נתעניין במציאת אלגוריתמים שמחזירים פתרון מקורב לבעיה. מחלקה זו של אלגוריתמים נקראת "אלגוריתמי קירוב" (approximation algorithms).
נראה כעת בעיה NP-קשה שעולה בהקשרים של מערכות הפעלה, ונציע לה אלגוריתם שמחזיר פתרון מקורב.

**בעיית חלוקת משימות בין מכונות**

**בעיה 11.1** (בעיית חלוקת משימות בין מכונות)**.**
<u>קלט:</u>

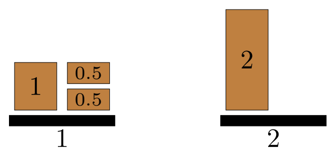
(1) $k$ מכונות זהות;

(2) $n$ מספרים חיוביים, $t_1,\ldots,t_n$, המסמנים זמני ריצה של $n$ משימות.

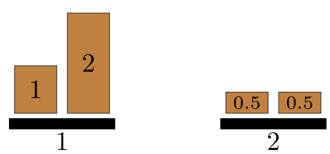
<u>פלט:</u>
חלוקה מאוזנת של $n$ המשימות בין $k$ המכונות, כך שהמכונה העמוסה ביותר תעבוד בזמן המינימלי האפשרי.

נתאר באופן פורמלי את מרחב הפתרונות החוקיים לבעיה:
פלט הבעיה הוא פונקציה $S:[n]\to[k]$, ומרחב הפתרונות החוקיים $\mathcal{S}=\{S:[n]\to[k]\}$.
לכל מכונה $1\leq j\leq k$ נגדיר $T_j\left(S\right):=\sum\limits_{i\in S^{-1}(j)}t_i$, זמן הריצה הכולל של המכונה ה-$j$ עבור הפתרון $S$.
נגדיר את פונקציית הערך $q:\mathcal{S}\to\mathbb{R}^+$ ע״י $q\left(S\right)=\max\limits_{1\leq j\leq k}\left\{T_j\left(S\right)\right\}$ לכל $S\in\mathcal{S}$.
מטרת הבעיה היא למצוא נקודת מינימום של פונקציית הערך $q$: אם $S^*$ הוא פתרון אופטימלי לבעיה אז $q\left(S^*\right)=\min\limits_{S\in\mathcal{S}}\left\{q\left(S\right)\right\}$.

**דוגמה:** יש $k=2$ מכונות, $n=4$ משימות שזמני הריצה שלהן $\left(t_1=1,\frac{1}{2},\frac{1}{2},2\right)$.
הפתרון האופטימלי הוא להקצות את $t_1,t_2,t_3$ למכונה מס׳ $1$, ולהקצות את $t_4$ למכונה מס׳ $2$.
ערך הפתרון האופטימלי הוא $2$.




**כלל החלטה חמדני:** בבעיה זו אלגוריתם חמדן לא ישיג את הפתרון האופטימלי, אך יקרב אותו, כפי שנראה בהמשך.
כלל ההחלטה החמדני קובע כי בכל שלב נשלח את המשימה שמגיעה למכונה הכי פחות עמוסה בשלב הזה.
זמני המשימות אינם ממויינים, והאלגוריתם החמדן עובר על המשימות לפי סדר הגעתן.
למשל, עבור הדוגמה הקודמת, האלגוריתם החמדן יציב את המשימה הראשונה באחת מן המכונות (כי שתי המכונות ריקות בהתחלה), נניח מכונה מס׳ $1$.
לאחר מכן יציב את המשימה השניה והשלישית (שזמן הריצה שלהן הוא $\frac{1}{2}$) במכונה מס׳ $2$. לבסוף, האלגוריתם החמדן יציב את משימה מספר $4$ (שזמן הריצה שלה הוא $2$) במכונה מס׳ $1$.
בסה״כ, קיבלנו שערך הפתרון החמדן הוא $3$, והוא איננו אופטימלי.




**<u>חסמים תחתונים על ערך הפתרון האופטימלי:</u>** יהי $S^*$ פתרון אופטימלי לבעיה.

**חסם תחתון ראשון** נסמן ב-$t_{max}$ את זמן הריצה של המשימה הארוכה ביותר, כלומר $t_{max}:=\max\limits_{1\leq i\leq n}\left\{t_i\right\}$.
אזי מתקיים $q\left(S^*\right)\geq t_{max}$, ובמילים, כל פתרון לבעיה, בפרט הפתרון האופטימלי, מקצה את המשימה הארוכה ביותר למכונה כלשהי, ולכן ערך כל פתרון לבעיה הוא לפחות $t_{max}$, זמן הריצה של המשימה הארוכה ביותר.

**חסם תחתון שני**

**למה 11.2.** $q\left(S^*\right)\geq\frac{1}{k}\sum\limits_{i=1}^{n}t_i$.

הוכחה. לפי הגדרה $q\left(S^*\right)=\max\limits_{1\leq j\leq k}\left\{T_j\left(S^*\right)\right\}$. המקסימום של קבוצת מספרים סופית הוא לכל הפחות ממוצע איבריה

$$q\left(S^*\right)=\max_{1\leq j\leq k}\left\{T_j\left(S^*\right)\right\}\geq\frac{1}{k}\sum_{j=1}^{k}T_j\left(S^*\right)$$

לפי הגדרת $T_j\left(S^*\right)$ נקבל כי

$$\frac{1}{k}\sum_{j=1}^{k}T_j\left(S^*\right)=\frac{1}{k}\sum_{j=1}^{k}\sum_{i\in\left(S^*\right)^{-1}(j)}t_i$$

ביטוי הסכימה הכפול מבטא סכימה של זמני הריצה של כל המשימות (עבור כל מכונה $j$ אנו סוכמים את זמני הריצה של כל המשימות שהוקצו לה), וקיבלנו

$$q\left(S^*\right) \geq \frac{1}{k}\sum_{j=1}^{k} T_j\left(S^*\right) = \frac{1}{k}\sum_{i=1}^{n} t_i$$

כנדרש. □

**משפט 11.3.** יהי $G$ הפתרון של האלגוריתם החמדן, ויהי $S^*$ פתרון אופטימלי. אזי $1 \leq \frac{q(G)}{q(S^*)} \leq 2 - \frac{1}{k}$.

הוכחה. ראשית, לפי הגדרת הפתרון האופטימלי (נקודת מינימום של פונקציית הערך $q$) מתקיים $q\left(S^*\right) \leq q\left(G\right)$.
יהי $1 \leq j^* \leq k$ האינדקס של המכונה העמוסה ביותר בפתרון החמדן. במילים אחרות, $q\left(G\right) = T_{j^*}\left(G\right)$.
יהי $1 \leq \ell \leq n$ האינדקס של המשימה האחרונה שמוקצית למכונה ה-$j^*$.
נתבונן בשיקול של האלגוריתם החמדן לפני שהחליט לשלוח את משימה $\ell$ למכונה $j^*$.
נגדיר עבור $1 \leq j \leq k$ את $F_j\left(G\right)$ בתור זמן הריצה הכולל של מכונה מספר $j$, לאחר חילוק $\ell - 1$ המשימות הראשונות.
האבחנה הקריטית היא שלפי דרך פעולתו של האלגוריתם החמדן, $F_{j^*}\left(G\right) = \min\limits_{1\leq j\leq k}\left\{F_j\left(G\right)\right\}$.
עתה,

$$q\left(G\right) = T_{j^*}\left(G\right) = t_\ell + F_{j^*}\left(G\right) = t_\ell + \min_{1\leq j\leq k}\left\{F_j\left(G\right)\right\}$$

המינימום של קבוצה סופית של מספרים הוא לכל היותר ממוצע איבריה, ולכן

$$\begin{aligned} t_\ell + \min_{1\leq j\leq k}\left\{F_j\left(G\right)\right\} &\leq t_\ell + \frac{1}{k}\sum_{j=1}^{k} F_j\left(G\right) \\ &= t_\ell + \frac{1}{k}\sum_{j=1}^{k}\sum_{\substack{i\in G^{-1}(j)\\ 1\leq i\leq \ell-1}} t_i \\ &= t_\ell + \frac{1}{k}\sum_{i=1}^{\ell-1} t_i \\ &= \left(1-\frac{1}{k}\right)t_\ell + \frac{1}{k}\sum_{i=1}^{\ell} t_i \end{aligned}$$

נשתמש בכך ש-$t_\ell \leq t_{max}$ וגם בכך ש-$t_i \geq 0$ לכל $i \in [n]$:

$$\left(1-\frac{1}{k}\right)t_\ell + \frac{1}{k}\sum_{i=1}^{\ell} t_i \leq \left(1-\frac{1}{k}\right)t_{max} + \frac{1}{k}\sum_{i=1}^{n} t_i$$

כעת נשתמש בשני החסמים התחתונים דלעיל: $t_{max} \leq q\left(S^*\right)$, $\frac{1}{k}\sum\limits_{i=1}^{n} t_i \leq q\left(S^*\right)$ ונקבל

$$\begin{aligned} \left(1-\frac{1}{k}\right)t_{max} + \frac{1}{k}\sum_{i=1}^{n} t_i &\leq \left(1-\frac{1}{k}\right)q\left(S^*\right) + q\left(S^*\right) \\ &= \left(2-\frac{1}{k}\right)q\left(S^*\right) \end{aligned}$$

ולבסוף, קיבלנו כי

$$q\left(G\right) \leq \left(2-\frac{1}{k}\right)q\left(S^*\right)$$

כנדרש. □

**הערה 11.4.**

(1) הניתוח שביצענו הדוק עבור $k=2$. בדוגמה שראינו קודם, מתקיים $\frac{q(G)}{q(S^*)} = \frac{3}{2} \leq \left(2-\frac{1}{2}\right)$.

(2) אם נמיין את זמני הריצה של המשימות בסדר יורד, כלומר $t_1 \geq t_2 \geq \cdots \geq t_n$, נקבל שהאלגוריתם החמדן המשופר מקיים $\frac{q(G)}{q(S^*)} \leq \frac{3}{2}$.

אולם, זה יהיה אלגוריתם offline, שצריך לקבל את כל הקלט מראש. האלגוריתם שהצגנו לעיל הוא אלגוריתם online במובן שאין חשיבות לסדר המעבר על המשימות, ולכן ניתן להקצות משימות מבלי לנבא את העתיד.

(3) לכל $\varepsilon > 0$ קיים אלגוריתם המשיג קירוב $1+\varepsilon$ לפתרון האופטימלי, שזמן הריצה שלו $n^{(1/\varepsilon)^{1.5}}$. כדי לקבל תחושה, עבור $\varepsilon = \frac{1}{100}$ זמן הריצה הוא $n^{1000}$.

## 11.3 אלגוריתמי קירוב: הגדרות

נתבונן בשני סוגי בעיות אופטימיזציה: בעיות מקסימיזציה ובעיות מינימיזציה.

**הגדרה 11.5.** יהי $\mathcal{S}$ מרחב הפתרונות החוקיים לבעיה אלגוריתמית נתונה, ותהי פונקציית ערך $q : \mathcal{S} \to \mathbb{R}^+$.
**בעיית מקסימיזציה** היא הבעיה של מציאת פתרון אופטימלי $S^* \in \mathcal{S}$ כך ש-$q(S^*) = \max_{S \in \mathcal{S}} \{q(S)\}$.
**בעיית מינימיזציה** היא הבעיה של מציאת פתרון אופטימלי $S^* \in \mathcal{S}$ כך ש-$q(S^*) = \min_{S \in \mathcal{S}} \{q(S)\}$.

נגדיר במדויק את האמירה "האלגוריתם מקרב את הבעיה".

**הגדרה 11.6.** יהי $c \geq 1$. נאמר כי אלגוריתם נתון הוא **אלגוריתם $c$-מקרב לבעיית מקסימיזציה** נתונה עם פתרון אופטימלי $S^*$, אם לכל קלט האלגוריתם מחזיר פתרון חוקי $S$ כך שמתקיים $q(S) \geq \frac{q(S^*)}{c}$.
נאמר כי אלגוריתם נתון הוא **אלגוריתם $c$-מקרב לבעיית מינימיזציה** נתונה עם פתרון אופטימלי $S^*$, אם לכל קלט האלגוריתם מחזיר פתרון חוקי $S$ כך שמתקיים $q(S) \leq c \cdot q(S^*)$.

**דוגמה.** האלגוריתם שהצענו עבור בעיה 11.1 הוא אלגוריתם $\left(2 - \frac{1}{k}\right)$-מקרב לבעיה.

## 11.4 העשרה: 3 בעיות על גרפים

הקלט לכל אחת מהבעיות הוא גרף $G = (V, E)$ לא מכוון עם $n$ קודקודים ו-$m$ צלעות.

1. קלט נוסף: זוג קודקודים $v, w \in V$. פלט: מסלול באורך מינימלי מ-$v$ ל-$w$.

   יש פתרון אופטימלי יעיל (למשל, האלגוריתם של Dijkstra).

2. אין קלט נוסף. מחפשים קבוצת קודקודים מינימלית השולטת על כל צלע בגרף. כלומר, תת-קבוצה $C \subseteq V$ כך שלכל $(x, y) \in E$ מתקיים $x \in C$ או $y \in C$.

   לבעיה זו ניתן אלגוריתם יעיל 2-מקרב.

3. אין קלט נוסף. מחפשים תת-קבוצה של קודקודים בגודל מקסימלי כך שכל שני קודקודים מחוברים בצלע (קליקה).

   בעיה זו היא NP-שלמה.

# 12 שבוע 6 - הרצאה - 21.11.18

## 12.1 אלגוריתמי קירוב: בעיית כיסוי קבוצות (Set Cover)

**בעיית כיסוי קבוצות**

**בעיה 12.1** (בעיית כיסוי קבוצות).
קלט:

(1) מספר טבעי $n$;

(2) $r$ תתי-קבוצות $A_1, A_2, \ldots, A_r$ של $[n]$ כך ש-$\bigcup_{i=1}^{r} A_i = [n]$.

פלט:
תת-קבוצה $S \subseteq [r]$ של אינדקסים כך ש-$\bigcup_{i \in S} A_i = [n]$ וגם $|S|$ מינימלי בתנאי זה.

דוגמאות לסיפורי מסגרת:

1. באוניברסיטה יש $n$ תחומי התמחות ו-$r$ אנשים. מעוניינים לבחור ועדה עם מספר מינימלי של אנשים, כך שעבור כל תחום התמחות קיים לפחות אדם אחד בועדה שמתמחה בו.
2. בגדר האוניברסיטה יש חור, ונתונים $r$ קרשים שביחד מכסים את החור בגדר. מעוניינים לבחור מספר מינימלי של קרשים שיכסו את החור בגדר.

בעיה זו היא NP-קשה ואנו נציג אלגוריתם מקרב.

**דוגמה 12.2.** $n = 10$ ו-$r = 6$

$$A_1 = \{1,2,3,4,5\} \qquad A_2 = \{6,7,8,9,10\}$$
$$A_3 = \{1,2,3\} \qquad A_4 = \{6,7,8\}$$
$$A_5 = \{1,2,3,6,7,8\} \qquad A_6 = \{3,4,5\}$$

נתעניין בקבוצה מינימלית של אינדקסים שמכסים את כל המספרים $\{1, \ldots, 10\}$. הפתרון $S^* = \{1,2\}$ הוא אופטימלי.

לטובת האלגוריתם המקרב, ניעזר בכלל חמדני טבעי שניסוחו (באופן לא פורמלי): בכל שלב נבחר קבוצה המכסה "כמה שיותר" ממה שעוד לא כיסינו.

**דוגמה 12.3.** עבור הנתונים שהצגנו בדוגמה 12.2 פתרון חמדן לפי הכלל דלעיל הוא $G = \{5,1,2\}$.
באיטרציה הראשונה, אנו צריכים לכסות את $X_0 = \{1,2,3,\ldots,10\}$. נבחר ב-$A_5$ כיוון שהיא מכילה את המספר הגדול ביותר של איברים.
באיטרציה השניה, אנו צריכים לכסות את $X_1 = \{4,5,9,10\}$ מבין הקבוצות (קו-תחתון מסמן איברים שעדיין לא כיסינו):

$$A_1 = \{1,2,3,\underline{4},\underline{5}\} \qquad A_2 = \{6,7,8,\underline{9},\underline{10}\}$$
$$A_3 = \{1,2,3\} \qquad A_4 = \{6,7,8\}$$
$$A_5 = \{1,2,3,6,7,8\} \qquad A_6 = \{3,\underline{4},\underline{5}\}$$

בשלב זה, שלושת הקבוצות $A_1, A_2, A_6$ ניתנות לבחירה, ונניח שהאלגוריתם החמדן בחר את $A_1$.
באיטרציה השלישית והאחרונה, אנו צריכים לכסות את $X_2 = \{9,10\}$ מבין הקבוצות:

$$A_1 = \{1,2,3,4,5\} \qquad A_2 = \{6,7,8,\underline{9},\underline{10}\}$$
$$A_3 = \{1,2,3\} \qquad A_4 = \{6,7,8\}$$
$$A_5 = \{1,2,3,6,7,8\} \qquad A_6 = \{3,4,5\}$$

נבחר בקבוצה $A_2$, וניוותר עם הקבוצה $X_3 = \emptyset$, ובשלב זה כיסינו את $[10]$ ולכן נעצור.

נגדיר במדויק את האלגוריתם החמדן:

**אלגוריתם 7** אלגוריתם חמדן המקרב את בעיית כיסוי הקבוצות

1. אתחול: נאתחל $S=\emptyset$ (הפתרון שיוחזר), ו-$X=[n]$ (קבוצת האיברים שעדיין נשאר לכסות).
2. איטרציה: יהי $1\le i^*\le r$ כך ש-
$$|X\cap A_{i^*}|=\max_{1\le i\le r}\left\{|X\cap A_i|\right\}$$
נעדכן $S\leftarrow S\cup\{i^*\}$, ונבצע $X\leftarrow X\setminus A_{i^*}$.
3. סיום: נעצור כאשר $X=\emptyset$ ונחזיר את $S$.

**משפט 12.4.** אלגוריתם 7 משיג $\lceil\ln n\rceil$ קירוב לבעיה 12.1.

**הערה 12.5.** מסתבר שאי-אפשר (=קשה מבחינת סיבוכיות) להשיג קירוב טוב יותר מ-$O(\log n)$.

הוכחה. נוכיח כי אם $G$ הוא הפתרון של האלגוריתם החמדן ו-$S^*$ הוא פתרון אופטימלי אז $\frac{|G|}{|S^*|}\le\lceil\ln n\rceil$.
נסמן ב-$k$ את גודל הפתרון האופטימלי $S^*$.
נסמן ב-$t$ את מספר האיטרציות של האלגוריתם החמדן עד עצירתו (נשים לב ש-$t\le r$ כי במקרה הכי גרוע בוחרים בכל הקבוצות, ולכן האלגוריתם החמדן עוצר).
בכל איטרציה של האלגוריתם החמדן מתווסף אינדקס לפתרון, ולכן גודל הפתרון החמדן הוא כמספר האיטרציות, כלומר: $|G|=t$.
בסימונים אלה, נרצה להוכיח כי $t\le k\lceil\ln n\rceil$.
נסמן ב-$X_j$ את קבוצת האיברים שנותר לכסות אחרי האיטרציה ה-$j$, כאשר נגדיר $X_0=[n]$. נבחין כי $X_t=\emptyset$ (נובע מתנאי העצירה של האלגוריתם החמדן).

**למה.** לכל $0\le j\le t-1$ מתקיים $|X_{j+1}|\le\left(1-\frac{1}{k}\right)|X_j|$.

הערה. ברמה האינטואיטיבית, נשים לב שגודל הפתרון האופטימלי הוא $k$, לכן יש $k$ קבוצות שמכסות את $[n]$. נתבונן באיטרציה הראשונה של האלגוריתם החמדן.
מעיקרון שובך היונים, יש לפחות קבוצה אחת (מתוך $k$ הקבוצות ששייכות לפתרון האופטימלי) שגודלה הוא לפחות $\frac{n}{k}$.
מאחר שהאלגוריתם החמדן בוחר את הקבוצה עם מספר האיברים הגדול ביותר, מספר האיברים שיישאר לכסות באיטרציה הבאה הוא לכל היותר $n-\frac{n}{k}=\left(1-\frac{1}{k}\right)n$.
נוכיח את הלמה פורמלית.

הוכחה. נסמן ב-$i^*$ את אינדקס הקבוצה המתווספת לכיסוי באיטרציה ה-$j+1$.
ע״פ דרך פעולתו של האלגוריתם החמדן: $|X_j\cap A_{i^*}|=\max_{1\le i\le r}\left\{|X_j\cap A_i|\right\}$.
כעת, בוודאי מתקיים $\max_{1\le i\le r}\left\{|X_j\cap A_i|\right\}\ \ge\ \max_{i\in S^*}\left\{|X_j\cap A_i|\right\}$ (מקסימום של קבוצה גדול שווה מכל מקסימום של תת-קבוצה שלה).
מקסימום של קבוצה גדול שווה מממוצע איבריה, ולכן

$$|X_j\cap A_{i^*}|\ge\max_{i\in S^*}\left\{|X_j\cap A_i|\right\}\ge\frac{1}{k}\sum_{i\in S^*}|X_j\cap A_i| \tag{12.1}$$

מצד שני, מכיוון ש-$S^*$ הוא פתרון חוקי לבעיה, מתקיים $\bigcup_{i\in S^*}A_i=[n]$, ולכן

$$\bigcup_{i\in S^*}(X_j\cap A_i)=X_j\cap\left(\bigcup_{i\in S^*}A_i\right)=X_j\cap[n]=X_j$$

ע״פ חסם האיחוד

$$|X_j|=\left|\bigcup_{i\in S^*}(X_j\cap A_i)\right|\le\sum_{i\in S^*}|X_j\cap A_i|$$

משילוב עם אי-שוויון (12.1) נקבל

$$|X_j\cap A_{i^*}|\ge\frac{1}{k}|X_j|$$

עתה, מאופן פעולת האלגוריתם החמדן $|X_{j+1}| = |X_j \setminus A_{i^*}|$, ומכאן

$$|X_{j+1}| = |X_j \setminus A_{i^*}| = |X_j| - |X_j \cap A_{i^*}| \leq |X_j| - \frac{|X_j|}{k} = \left(1 - \frac{1}{k}\right)|X_j|$$

כנדרש. $\square$

חזרה להוכחת המשפט.
נסמן $u := k\lceil \ln n \rceil$. נוכיח כי $t \leq u$ ובכך נסיים.
נניח בשלילה כי $t > u$. מכאן, אחרי $u$ איטרציות האלגוריתם החמדן לא עצר, כלומר $|X_u| \geq 1$.
לפי הלמה שהוכחנו מתקיים

$$\begin{aligned}1 \leq |X_u| &\leq \left(1 - \frac{1}{k}\right)|X_{u-1}| \leq \left(1 - \frac{1}{k}\right)^2 |X_{u-2}| \leq \cdots \leq \\ &\leq \left(1 - \frac{1}{k}\right)^u |X_0| = \left(1 - \frac{1}{k}\right)^u n\end{aligned}$$

כעת, ע"פ תכונה של הפונקציה האקספוננציאלית $(1-x)^y < e^{-xy}$ ולכן

$$\left(1 - \frac{1}{k}\right)^u n < e^{-u/k} n = e^{-\lceil \ln n \rceil} n \leq e^{-\ln(n)} \cdot n = 1$$

כלומר, קיבלנו $1 < 1$ וזוהי סתירה.
על כן, $t \leq u$, כלומר $t \leq k\lceil \ln n \rceil$, כנדרש. $\square$

# 13 שבוע 7 - הרצאה - 25.11.18

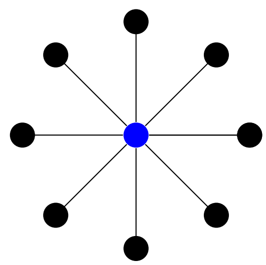
## 13.1 אלגוריתמי קירוב: בעיית כיסוי קודקודים (Vertex Cover)

**בעיית כיסוי קודקודים**

**בעיה 13.1** (בעיית כיסוי קודקודים).
<u>קלט:</u> גרף לא מכוון $G = (V, E)$.
<u>פלט:</u> תת-קבוצה של קודקודים $S \subseteq V$ כך שלכל צלע $e \in E$ $\{x, y\} = e$ מתקיים $x \in S$ או $y \in S$, וגם $|S|$ מינימלי בתנאי זה.

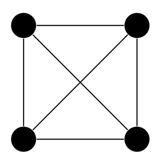
<u>דוגמאות:</u>

1. גרף הכוכב



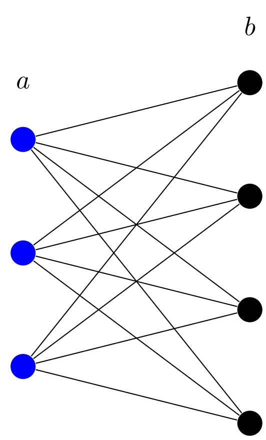
בגרף זה הקודקוד שמסומן בכחול הוא כיסוי קודקודים מינימלי בגרף, שהרי כל צלע בגרף מערבת את הקודקוד האמצעי $\Leftarrow$ $|S^*| = 1$.

2. הגרף השלם על $n$ קודקודים - $K_n$



כל $n - 1$ קודקודים נותנים כיסוי אופטימלי: ראשית, כל $n - 1$ קודקודים מהווים כיסוי קודקודים. שנית, אף קבוצה של $n - 2$ קודקודים אינה כיסוי כי הצלע בין הקודקודים שלא בחרנו אינה מכוסה.

3. גרף דו-צדדי שלם $K_{a,b}$ כאשר $a \leq b$




יש כיסוי בגודל $a$ (כל הקודקודים משמאל) ולכן $|S^*| \leq a$. נראה בהמשך ע"י שיקול קומבינטורי שאכן $|S^*| = a$.

בעיה זו הינה NP-שלמה, ואנחנו מחפשים אלגוריתם מקרב.

**רדוקציה לבעיית כיסוי על ידי קבוצות** נציג רדוקציה של בעיית כיסוי קודקודים לבעיית כיסוי על ידי קבוצות (בעיה 12.1), באופן הבא: לכל קודקוד $x \in V$ נתאים את קבוצת הצלעות שעוברות דרך הקודקוד הזה, ונסמנה $A_x$.
בעיית כיסוי קודקודים הופכת לבעיית כיסוי קבוצות מינימלי של אוסף כל הצלעות בגרף מתוך הקבוצות $\{A_x\}_{x \in V}$.
האלגוריתם החמדן יוסיף בכל שלב לכיסוי את הקודקוד שדרכו עוברות הכי הרבה צלעות שעוד לא כוסו. פתרון זה, כפי שראינו, ייתן $\ln(n)$ קירוב ($n$ מספר הקודקודים).
ואמנם, בבעיית כיסוי קודקודים, ניתן להשיג קירוב טוב יותר.

**אלגוריתם 2-מקרב לבעיית כיסוי קודקודים** בעיית כיסוי קודקודים היא בעיית מינימיזציה, ולכן נחפש חסם תחתון על גודל הפתרון האופטימלי לבעיה (=גודל כיסוי קודקודים מינימלי).

**הגדרה 13.2.** זיווג בגרף הוא אוסף של צלעות ללא קודקודים משותפים.

מהו גודל הזיווג המקסימלי בדוגמאות שראינו?

1. בגרף הכוכב, גודל הזיווג המקסימלי הוא 1, כי כל זיווג מערב את הקודקוד האמצעי.
2. בגרף השלם $K_n$ גודל הזיווג המקסימלי הוא $\left\lfloor \frac{n}{2} \right\rfloor$, פשוט חלוקה של הקודקודים לזוגות.
3. בגרף הדו-צדדי השלם $K_{a,b}$, $a \leq b$, גודל הזיווג המקסימלי הוא $a$.

**למה 13.3.** יהי $G = (V, E)$ גרף לא מכוון. אם $G$ מכיל זיווג עם $t$ צלעות, אזי כל כיסוי קודקודים ב-$G$ מכיל לפחות $t$ קודקודים.

הוכחה. יהי $S$ כיסוי קודקודים ב-$G$. ע״פ הגדרת המושג כיסוי קודקודים, $S$ מכיל קודקוד על כל אחת מ-$t$ הצלעות שבזיווג. לצלעות אלה אין קודקודים משותפים, ולכן כל $t$ הקודקודים ש-$S$ מכיל שונים זה מזה. מכאן נובע, $|S| \geq t$. $\square$

**הערה 13.4.** בתור מסקנה מלמה 13.3, נוכל להוכיח כי גודל כיסוי קודקודים מינימלי בגרף דו-צדדי שלם $K_{a,b}$, $a \leq b$, הינו $a$. ראינו קודם שגודל כיסוי קודקודים בגרף $K_{a,b}$ הוא לכל היותר $a$ (כי יש כיסוי בגודל $a$). מכיוון שיש זיווג בגודל $a$, נקבל מלמה 13.3 שכל כיסוי קודקודים (בפרט המינימלי) חייב להכיל לפחות $a$ קודקודים. מכאן נסיק שגודל כיסוי קודקודים מינימלי ב-$K_{a,b}$ הוא $a$.

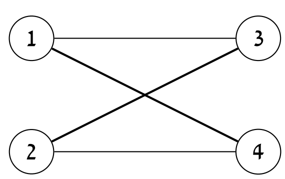
---

**אלגוריתם 8** אלגוריתם 2-מקרב לבעיית כיסוי קודקודים

1. אתחול: $S = \emptyset$ (הכיסוי שנחזיר), $X = E$ (קבוצת הצלעות שנותר לכסות).
2. איטרציה: בכל שלב נבחר צלע כלשהי $X \ni e = \{x, y\}$ ונעדכן $S \leftarrow S \cup \{x\} \cup \{y\}$. נמחק מ-$X$ את כל הצלעות שעוברות דרך $x$ או דרך $y$.
3. סיום: נעצור כאשר $X = \emptyset$, ונחזיר את $S$.

---

**דוגמה 13.5.** נתבונן בגרף $K_{2,2}$, ונריץ את אלגוריתם 8 עליו.




באיטרציה הראשונה של האלגוריתם נוכל לבחור כל צלע, נניח שבחרנו $e_1 = \{1, 4\}$. נוסיף את הקודקודים 1 ו-4 של הצלע $e_1$ ל-$S$ ונקבל $S = \{1, 4\}$.
באיטרציה השניה, אנו מוחקים את הצלעות $\{2, 4\}$ ו-$\{1, 3\}$. נותרה הצלע היחידה $e_2 = \{2, 3\}$. נוסיף את שני הקודקודים שלה לכיסוי $S$, ונקבל $S = \{1, 2, 3, 4\}$.
גודל כיסוי מינימלי, $S^*$, ב-$K_{2,2}$ שווה $|S^*| = 2$, וגודל הכיסוי שמצאנו שווה $|S| = 4$. בזאת קיבלנו פתרון שהוא 2-קירוב, $\frac{|S|}{|S^*|} = 2$.

**טענה 13.6.** אלגוריתם 8 הוא אלגוריתם 2-מקרב לבעיית כיסוי קודקודים (בעיה 13.1).

הוכחה. נרצה להוכיח שלכל גרף לא מכוון $G=(V,E)$, האלגוריתם מחזיר כיסוי $S$ כך ש-$\frac{|S|}{|S^*|}\le 2$, כאשר $S^*$ פתרון אופטימלי לבעיה.
נסמן ב-$t$ את מספר האיטרציות של האלגוריתם עד עצירתו. הפתרון $S$ שהאלגוריתם בונה הוא בגודל $2t$, כי בכל איטרציה מוסיפים 2 קודקודים ל-$S$.
נסמן ב-$e_i$ את הצלע שהאלגוריתם בוחר באיטרציה ה-$i$, עבור $1\le i\le t$.
מדרך פעולתו של האלגוריתם, לצלעות $e_1,e_2,\ldots,e_t$ אין קודקודים משותפים $\Leftarrow$ צלעות אלה מהוות זיווג ב-$G$.
מלמה 13.3 נקבל כי $|S^*|\ge t$, כאשר $S^*$ הוא פתרון אופטימלי לבעיה.
לכן מתקיים:

$$\frac{|S|}{|S^*|}\le\frac{2t}{t}=2$$

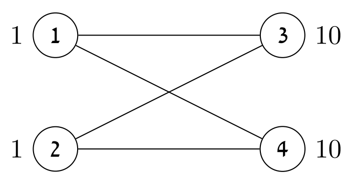
כנדרש. □

## 13.2 אלגוריתמי קירוב: בעיית כיסוי קודקודים ממושקל (Weighted Vertex Cover)

**בעיית כיסוי קודקודים ממושקל**

**בעיה 13.7** (בעיית כיסוי קודקודים ממושקל).
קלט: גרף לא מכוון $G=(V,E)$; פונקציית משקל $w:V\to\mathbb{R}^+$.
פלט: $S\subseteq V$ כיסוי קודקודים בעל משקל מינימלי, כאשר המשקל של $S$ מוגדר ע"י $w(S)=\sum_{x\in S}w(x)$.

**דוגמה 13.8.** נתבונן בגרף $K_{2,2}$, הפעם עם משקלים על הקודקודים.




הפתרון האופטימלי הוא $S^*=\{1,2\}$, עם משקל $w(S^*)=2$ - שהרי בצורה כזו אנו משיגים כיסוי קודקודים בעל משקל מינימלי.
אם נפעיל את אלגוריתם 8 על הגרף הנ"ל, נקבל $S=\{1,2,3,4\}$, ומשקל $w(S)=22$.
אלגוריתם זה השיג 11-קירוב על הדוגמה הזאת, אולם אם נשנה את משקלי הקודקודים של 3 ו-4 נוכל לקבל קירוב "גדול כרצוננו".

בבעיה זו נפנה לתכנון לינארי, ונשתמש באסטרטגיה הכללית הבאה להשגת אלגוריתם קירוב לבעיה 13.7:

1. ננסח את הבעיה כבעיית אופטימיזציה על משתנים המקבלים ערכים שלמים ומקיימים אילוצים לינאריים. (זוהי בעיית ILP)
2. נסיר את דרישת שלמות הערכים מהמשתנים, ונחליף דרישה זו באילוצים לינאריים נוספים. (כעת קיבלנו בעיית LP שניתנת לפתרון ע"י אלגוריתם יעיל)
3. נמצא פתרון אופטימלי (באופן יעיל) לבעיה שבנינו בשלב 2.
4. נבנה פתרון מקורב לבעיה המקורית בהתבסס על הפתרון משלב 3.

הבא ניישם את האסטרטגיה דלעיל עבור בעיה 13.7.

1. נתאים לכל קודקוד $v\in V$ משתנה $x(v)$ אשר יוגדר באופן הבא: $x(v):=\begin{cases}v\in S: & 1\\ v\notin S: & 0\end{cases}$, כאשר $S$ הוא כיסוי הקודקודים שאותו אנו מחפשים.
המשתנים $\{x(v)\}_{v\in V}$ מקיימים את האילוצים הבאים:
(א) לכל $v\in V$ מתקיים $x(v)\in\{0,1\}$;
(ב) מכיוון ש-$S$ הוא כיסוי קודקודים, לכל צלע $E\ni e=\{a,b\}$ מתקיים $a\in S$ או $b\in S$, כלומר, $x(a)=1$ או $x(b)=1$ (או שניהם).
ניתן לבטא דרישה זו ע"י אילוץ לינארי יחיד $x(a)+x(b)\ge 1$.

אנו רוצים למזער את פונקציית המשקל של הכיסוי $w\left(S\right)=\sum_{v\in S}w\left(v\right)=\sum_{v\in V}w\left(v\right)\cdot x\left(v\right)$.
בסה"כ קיבלנו בעיית תכנון לינארי בשלמים, השקולה לבעיה המקורית:

$$\begin{aligned}\min\quad & \sum_{v\in V}w\left(v\right)\cdot x\left(v\right)\\ \text{subject to}\quad & x\left(v\right)\in\left\{0,1\right\}\quad\forall v\in V\\ & x\left(a\right)+x\left(b\right)\geq1\quad\forall\left\{a,b\right\}=e\in E\end{aligned}$$

2. בשלב זה עלינו להסיר את הדרישה על שלמות המשתנים. במקום הדרישה $x\left(v\right)\in\left\{0,1\right\}$ לכל $v\in V$, נדרוש שהמשתנים יקיימו אילוץ "חלש" יותר $0\leq x\left(v\right)\leq1$.

$$\begin{aligned}\min\quad & \sum_{v\in V}w\left(v\right)\cdot x\left(v\right)\\ \text{subject to}\quad & 0\leq x\left(v\right)\leq1\quad\forall v\in V\\ & x\left(a\right)+x\left(b\right)\geq1\quad\forall\left\{a,b\right\}=e\in E\end{aligned}$$

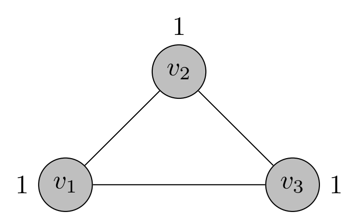
3. נמצא פתרון אופטימלי לבעיית התכנון הלינארי שהגדרנו בשלב 2. נסמן ב-$x^*$ פתרון אופטימלי לבעיה המקורית (בעיית תכנון הלינארי בשלמים), ונסמן ב-$x_{\text{LP}}^*$ את הפתרון האופטימלי לבעיית התכנון הלינארי שבנינו בשלב 2.

4. נבנה פתרון $x$ לבעיה המקורית על-סמך הפתרון האופטימלי $x_{\text{LP}}^*$.

כל פתרון של בעיית התכנון לינארי בשלמים היא גם פתרון של בעיית התכנון הלינארי, ולכן –

$$\sum_{v\in V}w\left(v\right)\cdot x_{\text{LP}}^*\left(v\right)\leq\sum_{v\in V}w\left(v\right)\cdot x^*\left(v\right)$$

(כלומר, הפתרון $x_{\text{LP}}^*$ משיג ערך טוב יותר של הפונקציה הלינארית שאנו מנסים למזער, על פני הפתרון $x^*$).
למשל, נתבונן בגרף $K_3$ הממושקל הבא:




הפתרון האופטימלי לבעיית תכנון הלינארי בשלמים הוא $x^*=\begin{pmatrix}1\\1\\0\end{pmatrix}$ אשר מקודד את הכיסוי עם הקודקודים $\left\{1,2\right\}$ (קודקוד 1 נכנס לכיסוי, קודקוד 2 נכנס לכיסוי ואילו קודקוד 3 לא נכנס לכיסוי).

הפתרון האופטימלי לבעיית תכנון הלינארי (שתיכננו בשלב 2) הוא $x_{\text{LP}}^*=\begin{pmatrix}0.5\\0.5\\0.5\end{pmatrix}$.

מדוגמה זו עולה כי המשקל של $x^*$ הוא $\underbrace{1}_{w(v_1)}\cdot\underbrace{1}_{x^*(v_1)}+1\cdot1+1\cdot0=2$, ואילו המשקל של $x_{\text{LP}}^*$ הוא $1\cdot0.5+1\cdot0.5+1\cdot0.5=1.5$.

נשתמש בטכניקה נפוצה להגדרת פתרון $x$ מתוך פתרון $x_{\text{LP}}^*$ שנקראת rounding. נגדיר את הפתרון $x$, לכל $v\in V$, באופן הבא:

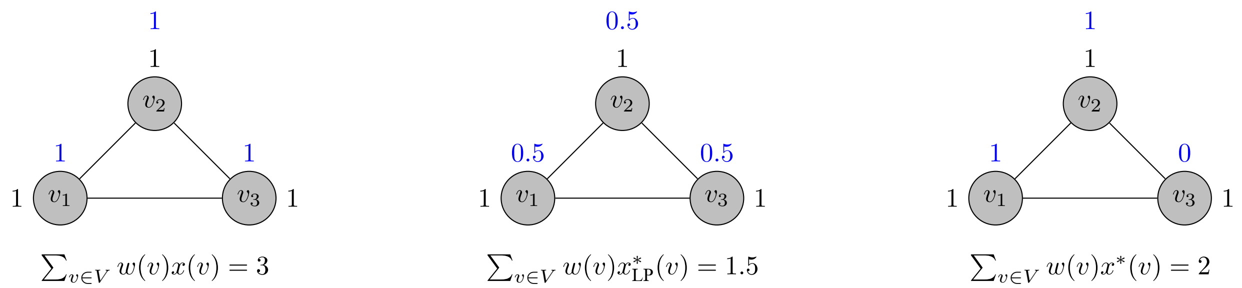
$$x\left(v\right):=\begin{cases}x_{\text{LP}}^*\left(v\right)\geq\frac{1}{2}: & 1\\ \text{Otherwise}: & 0\end{cases}$$

בשיעור הבא נוכיח את נכונות האלגוריתם.

# 14 שבוע 7 - הרצאה - 28.11.18

## 14.1 אלגוריתמי קירוב: בעיית כיסוי קודקודים ממושקל - המשך

נשוב ונתבונן בדוגמה הבאה:




$\sum_{v\in V} w(v)x^*(v) = 2$ | $\sum_{v\in V} w(v)x^*_{\text{LP}}(v) = 1.5$ | $\sum_{v\in V} w(v)x(v) = 3$

(א) $x^*$: פתרון אופטימלי של בעיית תכנון לינארי בשלמים

(ב) $x^*_{\text{LP}}$: פתרון אופטימלי של בעיית תכנון לינארי

(ג) $x$: הפתרון המקורב שהתקבל מעיגול $x^*_{\text{LP}}$

איור 14.1: דוגמה הממחישה את ההבדלים בין שלושת הפתרונות: $x^*$, $x^*_{\text{LP}}$ ו-$x$. בכל אחד מן האיורים מסומנים בכחול המשקלים שנותן כל אחד מן הפתרונות לקודקודים השונים.

**משפט 14.1.**

1. הפתרון $x$ שבנינו הוא פתרון חוקי לבעיית כיסוי קודקודים.
2. מתקיים כי: $\sum_{v\in V} w\left(v\right)x\left(v\right) \leq 2\sum_{v\in V} w\left(v\right)x^*\left(v\right)$.

ובמילים אחרות: נסמן ב-$S$ את הכיסוי שאנו מחזירים, כלומר, $S = \{v \in V : x\left(v\right) = 1\}$. נסמן ב-$S^*$ את הכיסוי האופטימלי. נרצה להראות כי $w\left(S\right) \leq 2w\left(S^*\right)$.

הוכחה.

1. חוקיות: על פי הגדרת הפתרון $x$ (הפתרון המעוגל) מתקיים $x\left(v\right) \in \{0,1\}$ לכל $v \in V$.
   נותר להוכיח שהקבוצה $\{v \in V : x\left(v\right) = 1\}$ אכן מגדירה כיסוי.
   תהי $e = \{a,b\} \in E$. מכיוון ש-$x^*_{\text{LP}}$ הוא פתרון חוקי לבעיית ה-LP מתקיים $x^*_{\text{LP}}\left(a\right) + x^*_{\text{LP}}\left(b\right) \geq 1$.
   לכן מתקיים $x^*_{\text{LP}}\left(a\right) \geq \frac{1}{2}$ או $x^*_{\text{LP}}\left(b\right) \geq \frac{1}{2}$. לכן, ע״פ הגדרת $x$ מתקיים $x\left(a\right) = 1$ או $x\left(b\right) = 1 \Leftarrow x\left(a\right) + x\left(b\right) \geq 1$.
   זה נכון לכל צלע $\{a,b\}$ בגרף, לכן $\{v \in V : x\left(v\right) = 1\}$ מגדירה כיסוי.
2. 2-קירוב: נשים לב כי הפתרון $x^*$ הוא פתרון אופטימלי לבעיית תכנון לינארי בשלמים, ולכן הוא פתרון חוקי של בעיית תכנון הלינארי (השברית). $x^*_{\text{LP}}$ הוא פתרון אופטימלי של בעיית תכנון הלינארי (השברית) ולכן

$$\sum_{v\in V} w\left(v\right)x^*\left(v\right) \geq \sum_{v\in V} w\left(v\right)x^*_{\text{LP}}\left(v\right)$$

יתר על כן, נבחין כי ע״פ הגדרת הפתרון $x$ מתקיים לכל $v \in V$ כי $x\left(v\right) \leq 2x^*_{\text{LP}}\left(v\right)$ (הסבר: אם $x^*_{\text{LP}}\left(v\right) \geq \frac{1}{2}$ אז $x\left(v\right) = 1$ ואי-השוויון מתקיים; אם $x^*_{\text{LP}}\left(v\right) < \frac{1}{2}$ אז $x\left(v\right) = 0$ ואי-השוויון מתקיים).

מכאן נקבל:

$$\sum_{v\in V} w\left(v\right)x\left(v\right) \leq \sum_{v\in V} w\left(v\right)\left(2x^*_{\text{LP}}\left(v\right)\right) = 2\sum_{v\in V} w\left(v\right)x^*_{\text{LP}}\left(v\right) \leq 2\sum_{v\in V} w\left(v\right)x^*\left(v\right)$$

כנדרש.

□

## 14.2 אלגוריתמי קירוב: בעיית פתרון אופטימלי של מערכת משוואות לינאריות מודולו 2 (Max Lin-2)

**בעיית פתרון אופטימלי של מערכת משוואות לינאריות מודולו 2**

**בעיה 14.2** (בעיית פתרון אופטימלי של מערכת משוואות לינאריות מודולו 2).
קלט: מטריצה $A$ מסדר $k \times n$ ($k$ שורות, $n$ עמודות) מעל השדה $\mathbb{F}_2$, ווקטור $b$ באורך $k$ מעל $\mathbb{F}_2$, המייצגים מערכת משוואות לינאריות $Ax = b$ כאשר $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ וקטור משתנים.
פלט: השמה של ערכים למשתנים $x_1, x_2, \ldots, x_n$ המספקת מספר מירבי של משוואות.

**דוגמה 14.3.** $n = 2$, $k = 3$

$$\overbrace{\begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}}^{A} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \overbrace{\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}}^{b}$$

נבחין כי מערכת זו היא ללא פתרון. אנו דורשים שיתקיים

$$\left\langle \begin{pmatrix} 1 \\ 0 \end{pmatrix} \mid \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle = 1, \left\langle \begin{pmatrix} 1 \\ 1 \end{pmatrix} \mid \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle = 1, \left\langle \overbrace{\begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix}}^{=\begin{pmatrix} 0 \\ 1 \end{pmatrix}} \mid \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle = 1$$

אבל זה לא ייתכן לאף השמה של $x_1$ ו-$x_2$ כי

$$\left\langle \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} \mid \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle = \left\langle \begin{pmatrix} 1 \\ 0 \end{pmatrix} \mid \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} 1 \\ 1 \end{pmatrix} \mid \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\rangle = 1 + 1 = 0$$

נחפש השמה שמספקת מספר מקסימלי של משוואות. ואכן $x_1 = x_2 = 1$ מספקת 2 משוואות (מספר מקסימלי).

זוהי בעיה NP-קשה, וניתן פתרון 2-קירוב לבעיה זו. כל קירוב יותר טוב מ-2-קירוב הוא בעצמו NP-קשה.

**הערה 14.4.** בעיה זו עולה בהקשרים של קודים לתיקון שגיאות.

בשיעור הבא ניתן אלגוריתם 2-מקרב.

# 15 שבוע 8 - הרצאה - 2.12.18

## 15.1 אלגוריתמי קירוב: בעיית פתרון אופטימלי של מערכת משוואות לינאריות מודולו $2$ - המשך

נכניס צורת רישום שאיתה נעבוד במהלך ההרצאה.

צורת רישום 15.1. נסמן את השורות של המטריצה $A$ ב-$r_1,\dots,r_k\in\mathbb{F}_2^n$; נסמן את עמודות המטריצה $A$ ב-$c_1,\dots,c_n\in\mathbb{F}_2^k$.

נאמר שמשוואה מספר $i$ מסתפקת ע"י השמה $s\in\mathbb{F}_2^n$ אס"ם $\langle r_i,s\rangle=b_i$ (כאשר $\left\langle\begin{pmatrix}r_1\\\vdots\\r_k\end{pmatrix},\begin{pmatrix}s_1\\\vdots\\s_k\end{pmatrix}\right\rangle:=\sum_{j=1}^k r_js_j$).

נפתח בדוגמה נוספת, מעט שונה מדוגמה 14.3.

**דוגמה 15.2.** $n=k=3$

$$\underbrace{\begin{pmatrix}1&0&0\\0&1&1\\1&1&1\end{pmatrix}}_{A}\underbrace{\begin{pmatrix}x_1\\x_2\\x_3\end{pmatrix}}_{x}=\underbrace{\begin{pmatrix}1\\1\\1\end{pmatrix}}_{b}$$

נטען שלא קיימת השמה $s\in\mathbb{F}_2^n$ אשר מספקת את כל המשוואות.

נשים לב ש-$r_3=r_1+r_2$, ולכן מלינאריות

$$\langle r_3,s\rangle=\langle r_1,s\rangle+\langle r_2,s\rangle$$

ואמנם, אם $s$ מספקת את משוואות $1$ ו-$2$ אז $\langle r_1,s\rangle=\langle r_2,s\rangle=1$ ולכן $\langle r_3,s\rangle=1+1=0$, כלומר, $s$ לא מספקת את משוואה $3$.

ההשמה $s^*=\begin{pmatrix}1\\0\\0\end{pmatrix}$ מספקת $2$ משוואות כי מתקיים

$$As^*=s_1^*c_1+s_2^*c_2+s_3^*c_3=1\cdot c_1+0\cdot c_2+0\cdot c_3=\begin{pmatrix}1\\0\\1\end{pmatrix}$$

וקטור זה שווה לוקטור $b$ בשני מקומות. □

ניתן אלגוריתם 2-מקרב לבעיה זו, כלומר, נחזיר פתרון $s$ המספק לפחות חצי ממספר המשוואות שפתרון אופטימלי $s^*$ מספק.

**מרחב הפתרונות החוקיים** $\mathcal{S}$ לבעיה זו הוא אוסף כל ההשמות של ערכים ל-$n$ משתנים בינאריים $x_1,\dots,x_n$, כלומר $\mathcal{S}=\mathbb{F}_2^n$.

נגדיר פונקציה $Y:\mathcal{S}\to\mathbb{N}\cup\{0\}$, כאשר בהינתן השמה $s\in\mathcal{S}$, $Y(s)$ שווה למספר המשוואות ש-$s$ מספקת. כלומר,

$$\begin{aligned}Y(s)&:=\left|\{i\in[k]:\langle r_i,s\rangle=b_i\}\right|\\&=\left|\{i\in[k]:(As)_i=b_i\}\right|\\&=\left|\left\{i\in[k]:\left(\sum_{j=1}^n s_jc_j\right)_i=b_i\right\}\right|\end{aligned}$$

אנו מחפשים השמה $s^*$ המקיימת $Y(s^*)=\max_{s\in\mathcal{S}}\{Y(s)\}$.

נשים לב שמעבר על כל האפשרויות עולה $2^n$ - יקר מידי.

רעיון: השמה מקרית של ערכים למשתנים תספק "בערך" חצי מהמשוואות; נהפוך רעיון זה לגישה דטרמיניסטית.

נגדיר על $\mathcal{S}$ פונקציית הסתברות ע"י $\mathbb{P}(\{s\})=\frac{1}{2^n}$ לכל $s\in\mathcal{S}$ (לכל השמה נתנו הסתברות שווה).

ביחס למרחב ההסתברות $(\mathcal{S},\mathbb{P})$ הפונקציה $Y$ הופכת למשתנה מקרי. נתעניין ב-$\mathbb{E}[Y]$.

**הגדרה 15.3** (משוואה ריקה טובה, משוואה ריקה רעה)**.** תהי $Ax=b$ מערכת משוואות לינאריות מעל $\mathbb{F}_2$. נאמר שמשוואה $i$ היא **משוואה ריקה טובה** אס"ם $r_i=0$ וגם $b_i=0$; נאמר שמשוואה $i$ היא **משוואה ריקה רעה** אס"ם $r_i=0$ וגם $b_i=1$.

**למה 15.4.** תהי $Ax=b$ מערכת משוואות לינאריות מעל $\mathbb{F}_2$, כאשר $A$ מטריצה מסדר $1\times n$, $x\in\mathbb{F}_2^n$, $b\in\mathbb{F}_2^1$ (כלומר, $A$ מטריצה עם שורה אחת). אזי מתקיים

$$\mathbb{E}[Y]=\begin{cases}r_1=0,b_1=0: & 1\\ r_1=0,b_1=1: & 0\\ r_1\neq 0: & 1/2\end{cases}$$

**הוכחה.** במקרה שבו ל-$A$ יש רק שורה בודדת, מתקיים עבור כל השמה $s \in \mathcal{S}$

$$Y(s) = \begin{cases} 1 & \langle r_1, s \rangle = b_1 \\ 0 & \text{Otherwise} \end{cases}$$

נבחין בין 3 מקרים:

1. $r_1 = 0$, $b_1 = 0$. במקרה זה, המשוואה מספקת לכל השמה $s \in \mathcal{S}$. (קראנו למשוואה מסוג זה "משוואה ריקה טובה", כי היא תמיד מסתפקת.)
2. $r_1 = 0$, $b_1 = 1$. במקרה זה, המשוואה לא מסתפקת לכל השמה $s \in \mathcal{S}$. (קראנו למשוואה מסוג זה "משוואה ריקה רעה", כי היא אף פעם לא מסתפקת.)
3. $r_1 \neq 0$. במקרה זה, נטען כי המשוואה מסתפקת ע"י חצי מההשמות האפשריות.

כדי לראות זאת, נסמן $V = \{s \in \mathcal{S} : \langle r_1, s \rangle = 0\}$, כלומר $V$ הוא מרחב כל ההשמות האפשרויות שעבורם המשוואה מקבלת ערך $0$.

אזי $V$ הוא תת-מרחב וקטורי מעל $\mathbb{F}_2^n$ (כי $V$ הוא למעשה מרחב הפתרונות של מערכת משוואות הומוגנית) ממימד $n-1$ (שהרי $\text{nullity}(A) + \text{rank}(A) = n$, כאשר $\text{nullity}(A)$ מייצג את מימד מרחב הפתרונות של מערכת משוואות הומוגנית מהצורה $Ax = 0$).

מכיוון ש-$V$ מוגדר מעל שדה סופי בן שני איברים, והוא ממימד $n-1$, אנו מקבלים כי $|V| = 2^{n-1}$.

כלומר, חצי מההשמות מקיימות $\langle r_1, s \rangle = 0$ והחצי הנותר מקיים $\langle r_1, s \rangle = 1$.

$\Leftarrow$ עבור השמה מקרית שנבחר מתוך $\mathcal{S}$, המשוואה מסתפקת בסיכוי $\frac{1}{2}$ ולא מסתפקת בסיכוי $\frac{1}{2}$.

עתה, לפי הגדרת התוחלת מתקיים:

$$\mathbb{E}[Y] = \frac{1}{2^n} \sum_{s \in \mathcal{S}} Y(s)$$

במקרה של "משוואה ריקה טובה", המשתנה $Y$ תמיד מקבל ערך $1$, ולכן $\mathbb{E}[Y] = 1$.
במקרה של "משוואה ריקה רעה", המשתנה $Y$ תמיד מקבל ערך $0$, ולכן $\mathbb{E}[Y] = 0$.
במקרה של משוואה לא ריקה, $Y$ מקבל ערך $1$ עבור חצי מההשמות האפשריות, ולכן

$$\mathbb{E}[Y] = \frac{2^{n-1}}{2^n} = \frac{1}{2}$$

בכך הסתיימה ההוכחה. □

**למה 15.5** תהי $Ax = b$ מערכת משוואות לינאריות מודולו $2$, עם $k$ משוואות. יהי $\beta$ מספר המשוואות הריקות הרעות במערכת ויהי $\gamma$ מספר המשוואות הריקות הטובות במערכת.
יהי $Y(s)$ מספר המשוואות שההשמה $s$ מספקת. אזי

$$\mathbb{E}[Y] = \frac{k + \gamma - \beta}{2}$$

**הוכחה.** לכל $1 \leq i \leq k$ נגדיר $Y_i : \mathcal{S} \to \{0,1\}$ להיות המשתנה המקרי האחראי על המשוואה ה-$i$, כלומר:

$$Y_i(s) = \begin{cases} 1 & \langle r_i, s \rangle = b_i \\ 0 & \text{Otherwise} \end{cases}$$

מתקיים $Y = \sum_{i=1}^{k} Y_i$, ולכן, לפי לינאריות התוחלת

$$\mathbb{E}[Y] = \sum_{i=1}^{k} \mathbb{E}[Y_i] = 1 \cdot \gamma + 0 \cdot \beta + \frac{1}{2} \overbrace{(k - \beta - \gamma)}^{\text{מספר משוואות לא ריקות}} = \frac{k + \gamma - \beta}{2}$$

כנדרש. □

**מסקנה 15.6.** לכל מערכת משוואות לינאריות מודולו 2, ניתן לחשב את התוחלת $\mathbb{E}\left[Y\right]$ של מספר המשוואות המסופקות באופן יעיל.

הוכחה. לפי למה 15.5, כדי לחשב את $\mathbb{E}\left[Y\right]$ מספיק לספור את המשוואות הריקות ולסווגן לפי טובות ורעות (ע״י כך נמצא את $\beta$ ואת $\gamma$).
לשם כך, נעבור על המטריצה $A$ ולכל שורה ריקה $r_i$ נבדוק האם $b_i = 0$ או $b_i = 1$, ועל פי כך נחליט אם היא משוואה ריקה טובה או רעה. זמן ריצה: $O\left(k \cdot n\right)$ (עוברים על כל שורה במטריצה, ולכל שורה מבצעים בדיקה שלוקחת $O\left(n\right)$ הקובעת אם השורה ריקה או לא). □

**מסקנה 15.7.** $\mathbb{E}\left[Y\right] \geq \frac{1}{2}Y\left(s^*\right)$.

הוכחה. יהי $\beta$ מספר המשוואות הריקות הרעות במערכת. משוואות אלה אינן מסופקות ע״י שום השמה, וגם לא ע״י השמה אופטימלית. לכן,

$$Y\left(s^*\right) \leq k - \beta$$

לפי למה 15.5 אנו מקבלים:

$$\mathbb{E}\left[Y\right] = \frac{k+\gamma-\beta}{2} \geq \frac{k-\beta}{2} \geq \frac{1}{2}Y\left(s^*\right)$$

כנדרש. □

נרצה להשתמש במסקנה 15.6 ומסקנה 15.7 ע״מ להציג אלגוריתם דטרמיניסטי אשר מחזיר השמה שמספקת לפחות חצי מהמשוואות.
הרעיון המרכזי הוא בשלב הראשון לחלק את מרחב הפתרונות החוקיים ($\mathcal{S} = \mathbb{F}_2^n$) לשני חלקים: ההשמות שבהן $x_1 = 0$ וההשמות שבהן $x_1 = 1$.
בכל אחד מן המקרים, מקבלים מערכת משוואות חדשה, כאשר מספר המשתנים שנותרו להשים קטן ב-1. בכל אחד מן המקרים נוכל לחשב את תוחלת מספר המשוואות המסופקות ע״י השמה מקרית (כתוצאה ממסקנה 15.6), ולבחור באופן חמדני את ההשמה שתוחלת מספר המשוואות המסופקות על-ידה הוא גדול יותר. נמשיך באופן הזה עד שנקבל השמה שתוחלת מספר המשוואות המסופקות על-ידה הוא לפחות מחצית מספר המשוואות המסופקות ע״י השמה אופטימלית.

**דוגמה 15.8.** נשוב ונתבונן בדוגמה 15.2. מרחב הפתרונות החוקיים ניתן לחלוקה לשני חלקים: השמות שבהן $x_1 = 0$ והשמות שבהן $x_1 = 1$.
במקרה שבו $x_1 = 0$ אנו מקבלים את מערכת המשוואות החדשה הבאה:

$$\begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

מערכת זו מכילה משוואה ריקה רעה אחת ($\beta = 1$), ולכן מלמה 15.5 אנו מקבלים $\mathbb{E}\left[Y|x_1=0\right] = \frac{3+0-1}{2} = 1$.
במקרה שבו $x_1 = 1$ אנו מקבלים את מערכת המשוואות החדשה הבאה:

$$\begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$$

(מכיוון ש-$x_1 = 1$ המערכת המקורית הופכת להיות $\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ x_2 \\ x_3 \end{pmatrix}$, וזה שקול למערכת $\begin{pmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$.)
מערכת זו מכילה משוואה ריקה טובה אחת ($\gamma = 1$), ולכן מלמה 15.5 אנו מקבלים $\mathbb{E}\left[Y|x_1=1\right] = \frac{3+1-0}{2} = 2$.
קיבלנו שתוחלת מספר המשוואות המסופקות ע״י השמה מקרית שעבורה $x_1 = 1$ גדול יותר מתוחלת מספר המשוואות המסופקות ע״י השמה מקרית שעבורה $x_1 = 0$, ולכן, באופן חמדני, נעדיף את ההשמות שבהן $x_1 = 1$.
בנוסף, נזכור שמנוסחת התוחלת השלמה נובע: $\mathbb{E}\left[Y\right] = \mathbb{P}\left(x_1=0\right)\mathbb{E}\left[Y|x_1=0\right] + \mathbb{P}\left(x_1=1\right)\mathbb{E}\left[Y|x_1=1\right]$.

# 16 שבוע 8 - הרצאה - 5.12.18

## 16.1 אלגוריתמי קירוב: בעיית פתרון אופטימלי של מערכת משוואות לינאריות מודולו $2$ - המשך

נרשום במדויק את האלגוריתם שתיארנו בצורה מרושלת בהרצאה הקודמת.

**אלגוריתם 9** אלגוריתם חמדן המשיג $2$-קירוב לבעיית פתרון אופטימלי של מערכת משוואות לינאריות מודולו $2$

1. אתחול: $A' \leftarrow A$, $b' \leftarrow b$. ($A'$ ו-$b'$ הם המטריצה והוקטור של המערכת לאורך האיטרציות)
2. איטרציה: באיטרציה ה-$t+1$, עבור $0 \le t \le n-1$, יש בידינו השמה של ערכים $s_1$ ל-$x_1$, ... , $s_t$ ל-$x_t$.
נחשב את $\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=0\right]$ ואת $\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=1\right]$.
נציב ל-$x_{t+1}$ את הערך $s_{t+1}$ שעבורו התוחלת המותנית גדולה מבין השניים.
נמחק מ-$A'$ את העמודה הראשונה שלה, $c$ (למעשה, $c$ היא העמודה ה-$t+1$ של המטריצה המקורית $A$).
נגדיר $b' \leftarrow b' - s_{t+1} \cdot c$.
נעדכן $t \leftarrow t+1$.
3. סיום: נעצור כאשר $t=n$, ונחזיר את ההשמה $s$.

תזכורת - נוסחת התוחלת השלמה: לכל $0 \le t \le n-1$

$$\begin{aligned}\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t\right] &= \mathbb{P}\left(x_{t+1}=0\right)\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=0\right] \\ &\quad + \mathbb{P}\left(x_{t+1}=1\right)\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=1\right] \\ &= \frac{1}{2}\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=0\right] + \frac{1}{2}\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=1\right]\end{aligned}$$

**טענה 16.1.** אלגוריתם 9 משיג $2$-קירוב לבעיה.

הוכחה. תהי $s=\begin{pmatrix}s_1\\ \vdots\\ s_n\end{pmatrix}$ ההשמה המוחזרת ע"י אלגוריתם 9, ותהי $s^*$ השמה אופטימלית. נוכיח כי $\frac{1}{2}Y\left(s^*\right) \le Y\left(s\right)$.
נסמן לכל $0 \le t \le n$ את $\mathbb{E}_t := \mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t\right]$.
בפרט מתקיים: $\mathbb{E}_0=\mathbb{E}\left[Y\right]$, $\mathbb{E}_n=Y\left(s\right)$.
נוכיח כי $\mathbb{E}_0 \le \mathbb{E}_1 \le \cdots \le \mathbb{E}_n$. לשם כך, מספיק להוכיח כי לכל $0 \le t \le n-1$ מתקיים $\mathbb{E}_t \le \mathbb{E}_{t+1}$.

$$\begin{aligned}\mathbb{E}_{t+1} &\overset{\text{לפי}}{\underset{\text{הגדרה}}{=}} \mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=s_{t+1}\right] \\ \left[\text{דרך הפעולה החמדנית}\right] &= \max\left\{\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=0\right],\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=0\right]\right\} \\ &\ge \frac{1}{2}\left(\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=0\right]+\mathbb{E}\left[Y|x_1=s_1,\ldots,x_t=s_t,x_{t+1}=0\right]\right) \\ \left[\text{נוסחת התוחלת השלמה}\right] &= \mathbb{E}_t\end{aligned}$$

לבסוף קיבלנו

$$Y\left(s\right)=\mathbb{E}_n \ge \mathbb{E}_{n-1} \ge \cdots \ge \mathbb{E}_0=\mathbb{E}\left[Y\right] \ge \frac{1}{2}Y\left(s^*\right)$$

כאשר אי-השוויון האחרון נובע מלמה 15.5. $\square$

# 17 שבוע 9 - הרצאה - 12.12.18

## 17.1 רשתות זרימה: מבוא

בעיית הזרימה

**בעיה 17.1** (בעיית הזרימה)**.**

קלט: רשת זרימה $N=(V,E,c,s,t)$.
פלט: זרימה חוקית $f$ ברשת הזרימה הנתונה $N$, בעלת שטף $|f|$ מקסימלי.

**הגדרה 17.2** (רשת זרימה)**. רשת זרימה** היא חמישייה $N=(V,E,c,s,t)$ כאשר

- $G=(V,E)$ גרף מכוון
- $c:E\to\mathbb{R}_{>0}$ פונקציית קיבול
- $s\in V$ קודקוד מקור ("יצרן חומר")
- $t\in V$ קודקוד בור ("סופג חומר")

**הערה 17.3.** אנו נבצע שתי הנחות מקלות על רשת הזרימה:

- אין צלעות שנכנסות לקודקוד המקור או צלעות שיוצאות מקודקוד הבור.[2]
- אין לולאות בגרף - כלומר, אין צלע מקודקוד לעצמו.

**הגדרה 17.4** (זרימה חוקית ברשת זרימה)**. זרימה חוקית ברשת זרימה** היא פונקציה $f:E\to\mathbb{R}_{\geq 0}$ המקיימת שני אילוצים:

1. אילוץ הקיבול: לכל $e\in E$ מתקיים $f(e)\leq c(e)$. במילים: הזרימה בכל צלע אינה גדולה מהקיבול של הצלע.
2. חוק שימור החומר: לכל $x\in V\setminus\{s,t\}$:

$$\sum_{u:(u,x)\in E} f(u,x)=\sum_{v:(x,v)\in E} f(x,v)$$

במילים: לכל קודקוד, הזרימה הנכנסת לקודקוד שווה לזרימה היוצאת מהקודקוד, פרט לקודקוד המקור וקודקוד הבור.

**הגדרה 17.5** (שטף)**. השטף** של זרימה $f$ מוגדר להיות סכום ערכי הזרימה שיוצאים מקודקוד $s$, כלומר

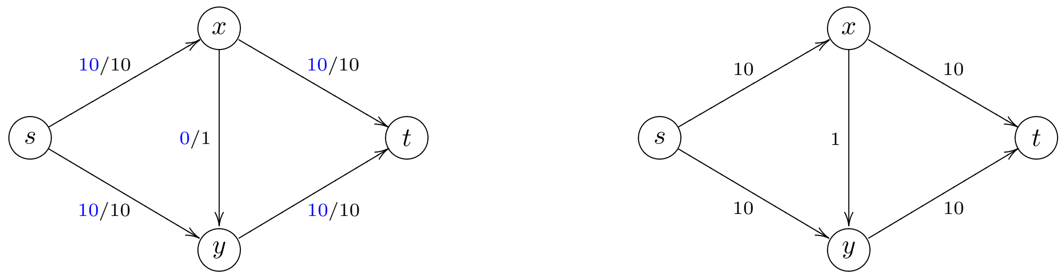
$$|f|\coloneqq\sum_{u:(s,u)\in E} f(s,u)$$

בעיה 17.1 היא בעיית מקסימיזציה חשובה עם הרבה שימושים, הניתנת לפתרון יעיל. הבעיה נפתרה לראשונה בשנת 1956 ע"י Ford&Fulkerson.

**דוגמה 17.6.** נתבונן ברשת הזרימה המתוארת באיור 17.1. רשת זרימה זו תלווה אותנו לאורך הנושא.
הזרימה האופטימלית ברשת היא בעלת שטף 20. נטען כי אין זרימה עם שטף גדול מ-20 ברשת, שהרי לכל זרימה $f$ מתקיים

$$\begin{aligned}|f|&=f(s,x)+f(s,y)\\&\leq c(s,x)+c(s,y)\quad[\text{אילוץ הקיבול}]\\&=10+10=20\end{aligned}$$

[2] הנחה זו לא משמעותית, שכן גם לבעיית הזרימה שמנוסחת ללא הנחה זו יש פתרון יעיל (כמעט באותה דרך).




(ב) (א)

איור 17.1: דוגמה לרשת זרימה על $4$ קודקודים $s, x, y, t$, כאשר: $s$ - קודקוד המקור, $t$ - קודקוד הבור. באיור (ב) מתוארת זרימה מקסימלית, כאשר ערכי הזרימה מסומנים בכחול.

**הערה 17.7.** נשים לב שבעיית הזרימה ניתנת לפתרון ע״י תכנון לינארי. המשתנים הם ערכי הזרימה על הצלעות. אילוץ הקיבול וחוק שימור החומר מגדירים אילוצים לינאריים על המשתנים, ואנחנו רוצים למקסם פונקציה (שטף הזרימה) שהיא פונקציה לינארית של המשתנים.

היתרונות של האלגוריתם של F&F על-פני פתרון הבעיה בעזרת תכנון לינארי הם:

1. האלגוריתם של F&F יעיל יותר;
2. אם ערכי הקיבול הם מספרים שלמים, אז האלגוריתם של F&F ייתן זרימה אופטימלית במספרים שלמים (לעומת תכנון לינארי, שעלול להחזיר פתרון ״שברי״).

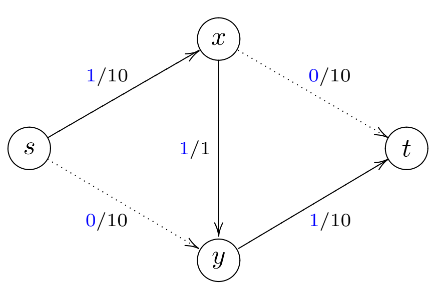
   זה חשוב שכן ביישומים של הבעיה נרצה שזרימה אופטימלית שתוחזר תהיה במספרים שלמים. אם למשל רשת הזרימה מתארת רשת כבישים וערכי הקיבול מתארים את מספר המשאיות שיכולות לנוע בכביש מבלי שיווצר פקק, אז נרצה שערכי הזרימה האופטימלית יהיו במספרים שלמים ולא שבריים.

**מספר אבחנות**

- **מרחב הפתרונות החוקיים** לבעיית הזרימה הוא מרחב כל הזרימות החוקיות ברשת זרימה.
- בכל רשת זרימה קיימת זרימה חוקית, והיא **זרימת האפס**. זרימה זו מקיימת את אילוץ הקיבול ואת חוק שימור החומר. מכאן נסיק שמרחב הפתרונות החוקיים אינו ריק.
- אם $f, g$ זרימות חוקיות אז גם $\frac{f+g}{2}$ היא זרימה חוקית, ובאופן כללי, אוסף הזרימות החוקיות הוא **קבוצה קמורה**: לכל $\lambda \in [0,1]$ הזרימה $\lambda f + (1-\lambda)\, g$ היא זרימה חוקית.
- אם $f, g$ שתי זרימות חוקיות, אז $f+g$ מקיימת את חוק שימור החומר (ולא בהכרח מקיימת את אילוץ הקיבול). אם בנוסף אילוץ הקיבול מתקיים, אז $f+g$ זרימה חוקית.

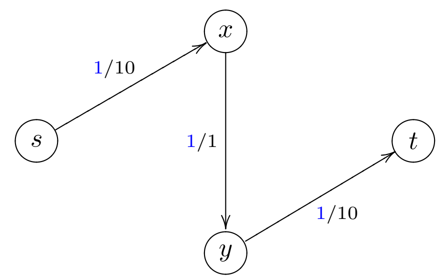
**דוגמה לזרימה לא טריוויאלית ברשת** נמצא מסילה מכוונת פשוטה בין $s$ ל-$t$ ונשלח בה זרימה קבועה השווה לקיבול המינימלי של צלע במסילה.

למשל, ברשת הזרימה של דוגמה 17.6, נמצא מסלול $s \rightsquigarrow x \rightsquigarrow y \rightsquigarrow t$, והקיבול המינימלי במסלול הוא $1$ (הקיבול של הצלע $(x, y)$).

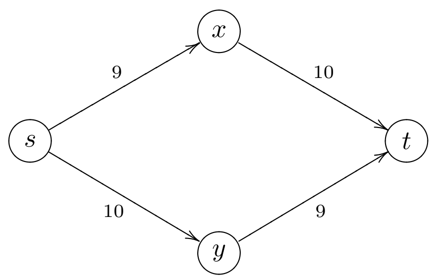



הזרימה שמצאנו היא חוקית, שכן אילוץ הקיבול מתקיים וגם אילוץ שימור החומר מתקיים (בודקים עבור $x$ ו-$y$).

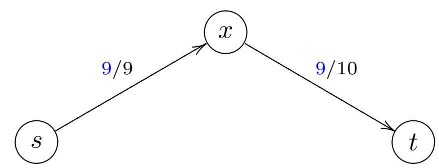
**רעיון ראשוני** הרעיון הראשון שנציע הוא: בכל שלב נמצא מסילה פשוטה בין $s$ ל-$t$ ברשת. נשלח בה זרימה קבועה השווה לקיבול המינימלי של צלע במסילה, ונוסיף את הזרימה החדשה שהגדרנו לזרימה שכבר קיימת ברשת, תוך כדי כך שנדאג לא להפר את אילוץ הקיבול.
נראה שרעיון זה פשוט מידי, אבל מקריסתו נחכים.



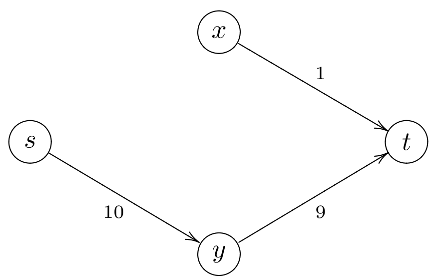

(א) מסלול $s-x-y-t$ וזרימה $f_1$ קבועה שנעביר בו



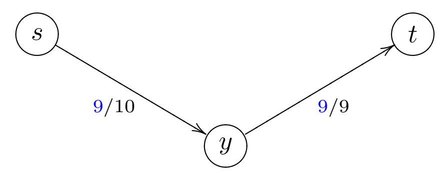

(ב) עדכון הקיבולים על הצלעות (לא נציג צלעות שהקיבול עליהם שווה 0)



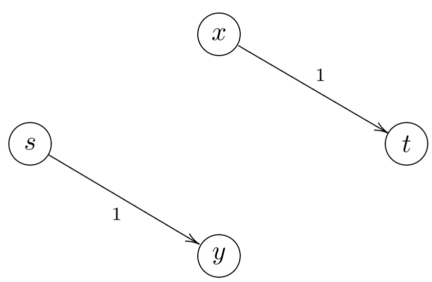

(ג) מסלול $s-x-t$ וזרימה $f_2$ קבועה שנעביר בו




(ד) עדכון הקיבולים על הצלעות אחרי שהעברנו את הזרימה $f_1+f_2$



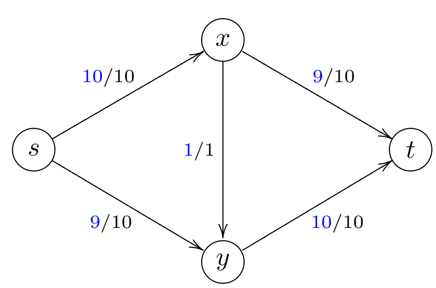

(ה) מסלול $s-y-t$ וזרימה $f_3$ קבועה שנעביר בו




(ו) עדכון הקיבולים על הצלעות אחרי שהעברנו את הזרימה $f_1+f_2+f_3$

איור 17.2: הזרימה שאיתה מתחיל האלגוריתם היא זרימת האפס, שנסמנה $f_0$ ומתקיים $|f_0|=0$. בכל שלב האלגוריתם מוצא מסילה פשוטה בין $s$ ל-$t$ (בגרף המעודכן הכולל צלעות עם קיבול שאינו אפס) ומעביר בה זרימה קבועה. האלגוריתם עוצר כאשר לא קיים מסלול בין $s$ ל-$t$.

שלבי האלגוריתם על רשת הזרימה מדוגמה 17.6 מתוארים באיור 17.2. נתבונן בזרימה $f_1+f_2+f_3$ שמחזיר האלגוריתם:




נשים לב שע"י שימוש ברעיון זה קיבלנו זרימה בעלת שטף שאינו מקסימלי (קיבלנו זרימה $f_1+f_2+f_3$ עם שטף $19$, והשטף המקסימלי הוא $20$, כפי שראינו בדוגמה 17.6).

**למה 17.8.** קיימת זרימה אופטימלית $f$ ברשת עם התכונה הבאה: אין שני קודקודים $y \neq x$, כך ש-$(x,y),(y,x) \in E$ וגם $f(x,y) > 0$ וגם $f(y,x) > 0$.

הוכחה. תהי $g$ זרימה אופטימלית ברשת.
נגדיר קבוצה $B := \{(x,y) \in E : (y,x) \in E \wedge (g(x,y) > 0) \wedge (g(y,x) > 0)\}$. נשים לב שאם $(x,y) \in B$ אז $(y,x) \in B$.
נגדיר פונקציה $f$ על הצלעות של הרשת באופן הבא:

$$f(e) = \begin{cases} e \notin B: & g(e) \\ (x,y) = e \in B: & g(e) - \min\{g(x,y), g(y,x)\} \end{cases}$$

נראה כעת ש-$f$ זרימה חוקית. הפונקציה $f$ היא אי-שלילית, ומקיימת את אילוץ הקיבול כי לכל $e \in E$ מתקיים

$$f(e) \leq g(e) \leq c(e)$$

חוק שימור החומר מתקיים עבור $f$ כי לכל קודקוד $x \in V \setminus \{s,t\}$, הזרימה הנכנסת והזרימה היוצאת מ-$x$ קטנות באותה מידה.
נראה כי $|f| = |g|$ ולכן $f$ היא גם זרימה אופטימלית. אמנם, לפי ההנחה על הרשת, אין צלעות ברשת הנכנסות למקור ולכן כל הצלעות $e = (s,x) \in E$ לא שייכות לקבוצה $B$. לכן, לכל צלע כזו (היוצאת מהמקור) מתקיים $f(e) = g(e)$. לכן, ע״פ הגדרת השטף, מתקיים $|f| = |g|$.
כדי לסיים את הטיעון, נבחין כי מתקיים לכל צלע $(x,y) \in B$ כי $\min\{f(x,y), f(y,x)\} = 0$.
בנוסף, לא ייתכן שיופיעו זוג קודקודים חדש (=שלא היה ב-$B$ קודם לכן) $u,v \in V$ כך ש-$f(u,v) > 0$ וגם $f(v,u) > 0$, כי רק הקטנו את הזרימה במעבר מ-$g$ ל-$f$. $\square$

# 18 שבוע 10 - הרצאה - 16.12.18

## 18.1 רשתות זרימה: הגדרות טכניות

בהרצאה הקודמת הצגנו רעיון כללי לבניית זרימה ברשת. ע״י שימוש ברעיון הזה קיבלנו זרימה טובה, אבל לא אופטימלית. נרצה לייצר את האפשרות ״להחזיר זרימה״.
למשל, הבעיה שעלתה בהפעלת הרעיון הכללי עבור דוגמה 17.6 (באיור 17.2) היא שבחרנו להעביר זרימה של $1$ בין $x$ ל-$y$. נרצה לפתוח קשת חדשה בין $y$ ל-$x$ שדרכה נוכל ״להתחרט״.
נפתח במספר הגדרות טכניות שיעזרו לנו בפרמול הרעיון של ״החזרת זרימה״ – להקטין זרימה בצלעות מסוימות.
מעתה נטפל בזרימות שבהן אין זוג קודקודים עם זרימה דו-כיוונית (הצידוק לכך מצוי בלמה 17.8).

**הגדרה 18.1** (רשת זרימה מורחבת). **רשת זרימה מורחבת** היא רביעייה $N' = (V, c', s, t)$, כאשר $V$ היא קבוצת קודקודים, $s$ - קודקוד מקור, $t$ - קודקוד בור, ו-$c' : V \times V \to \mathbb{R}_{\geq 0}$ היא פונקציית קיבול מורחבת, עם התכונות הבאות:

- לכל $x \in V$, $c'(x, x) = 0$
- לכל $x \in V$, $c'(x, s) = 0$
- לכל $x \in V$, $c'(t, x) = 0$

**הערה 18.2.** נחשוב על רשת זרימה מורחבת כעל רשת זרימה כאשר אנו לוקחים בחשבון כל צלע אפשרית בין קודקוד לקודקוד. נרצה שההגדרה שלנו תכבד את ההנחות שביצענו בהגדרה של רשת זרימה (רגילה), ולכן נדרוש מפונקציית הקיבול כי $c'(x, x) = 0$, כאשר הגדרה זו מגלמת את הדרישה של חוסר לולאות; $c'(x, s) = 0$, כאשר הגדרה זו מגלמת את הדרישה של איסור קיום צלעות הנכנסות למקור; $c'(t, x) = 0$, כאשר הגדרה זו מגלמת את הדרישה של איסור קיום צלעות היוצאות מקודקוד הבור.

**הגדרה 18.3** (זרימה מורחבת ברשת זרימה מורחבת). **זרימה מורחבת ברשת זרימה מורחבת** $N' = (V, c', s, t)$ זו פונקציה $f' : V \times V \to \mathbb{R}$, המקיימת שלוש תכונות:

1. <u>אנטי-סימטריה</u>: לכל $x \neq y \in V$ מתקיים $f'(x, y) = -f'(y, x)$;
2. <u>אילוץ הקיבול</u>: לכל $x, y \in V$ מתקיים $f'(x, y) \leq c'(x, y)$;
3. <u>חוק שימור החומר</u>: לכל $x \in V \setminus \{s, t\}$ מתקיים $\sum_{v \in V} f'(x, v) = 0$.

**הערה 18.4.** שימו לב שהבגדרה של $f'$ הטווח הוא כל $\mathbb{R}$, בניגוד ל-$\mathbb{R}_{\geq 0}$.

**הגדרה 18.5** (הרחבה של רשת זרימה). תהי $N = (V, E, c, s, t)$ רשת זרימה (רגילה). נרחיב אותה לרשת זרימה מורחבת $N' = (V, c', s, t)$, כאשר $c' : V \times V \to \mathbb{R}_{\geq 0}$ היא הרחבה של פונקציית הקיבול $c$, ומוגדרת באופן הבא:

$$c'(x, y) = \begin{cases} (x, y) \in E : & c(x, y) \\ (x, y) \notin E : & 0 \end{cases}$$

**הגדרה 18.6** (הרחבה של זרימה). תהי $f$ זרימה (רגילה) ברשת $N$, ותהי $N'$ רשת זרימה מורחבת שהיא הרחבה של $N$. נגדיר את הפונקציה $f' : V \times V \to \mathbb{R}$ שהיא **הרחבה של זרימה חוקית** $f$, באופן הבא:

$$f'(x, y) = \begin{cases} (x, y) \in E, f(x, y) > 0 : & f(x, y) \\ (y, x) \in E, f(y, x) > 0 : & -f(y, x) \\ \text{Otherwise} : & 0 \end{cases}$$

<u>הנחה</u>: כשאנחנו מדברים על זרימה חוקית אנו מניחים שלא קיימים $x, y \in V$ כך ש-$f(x, y) > 0$ וגם $f(y, x) > 0$.

**הגדרה 18.7** (שטף של זרימה מורחבת). **שטף של זרימה מורחבת** $f'$ מוגדר באופן הבא:

$$|f'| \coloneqq \sum_{v \in V} f'(s, v)$$

בתרגיל תוכיחו את הלמה הבאה.

**למה 18.8.**

(1) תהי $N=(V,E,c,s,t)$ רשת זרימה רגילה. אזי ההרחבה שלה $N'$ היא רשת זרימה מורחבת.

(2) תהי $f$ זרימה (רגילה) ברשת זרימה $N$, אזי ההרחבה $f'$ היא זרימה מורחבת ברשת זרימה מורחבת.

(3) $|f'|=|f|$.

**הגדרה 18.9** (צמצום של רשת מורחבת)**.** תהי $N'=(V,c',s,t)$ רשת זרימה מורחבת. נגדיר רשת זרימה (רגילה) $N=(V,E,c,s,t)$ באופן הבא:

- קבוצת הצלעות $E\subseteq V\times V$ מוגדרת ע״י

$$E\coloneqq\{(x,y)\in V\times V: c'(x,y)>0\}$$

- פונקציית קיבול $c:E\to\mathbb{R}_{>0}$ מוגדרת ע״י

$$\forall e\in E,\quad c(e)\coloneqq c'(e)$$

נאמר על $N$ שהיא **הצמצום** של הרשת המורחבת $N'$.

**הגדרה 18.10** (צמצום של זרימה מורחבת)**.** תהי $f'$ זרימה מורחבת ברשת זרימה מורחבת $N'$. תהי $N=(V,E,c,s,t)$ רשת זרימה (רגילה) שהיא צמצום של $N'$. נגדיר את הפונקציה $f:E\to\mathbb{R}_{\geq 0}$ שהיא **צמצום של הזרימה המורחבת** $f'$, באופן הבא:

$$\forall e\in E,\quad f(e)\coloneqq\max\{f'(e),0\}$$

**הערה 18.11.** התבוננו באיור 18.2 (שלב 5 באלגוריתם פורד-פולקרסון) להמחשה של המושגים "הרחבה של זרימה" ו-"צמצום של זרימה מורחבת".

## 18.2 רשתות זרימה: אלגוריתם פורד-פולקרסון

נפתח ברשימה של הגדרות שיקלו על כתיבת האלגוריתם של פורד ופולקרסון.

**הגדרה 18.12** (קיבול שיורי, אוסף צלעות שיורי, גרף שיורי, רשת שיורית, מסילת הרחבה, קיבול שיורי של מסילה $P$, זרימה שיורית)**.**
תהי $N=(V,c,s,t)$ רשת זרימה מורחבת ותהי $f$ זרימה מורחבת ברשת זו.
**הקיבול השיורי** הוא פונקציה $c_f:V\times V\to\mathbb{R}_{\geq 0}$ שמוגדרת ע״י $c_f(x,y)\coloneqq c(x,y)-f(x,y)$ לכל $x,y\in V$.
**אוסף הצלעות השיורי** הוא הקבוצה $E_f\coloneqq\{(x,y)\in V\times V: c_f(x,y)>0\}$.
**הגרף השיורי** הוא הגרף המכוון $G_f=(V,E_f)$.
**הרשת השיורית** היא רשת הזרימה המורחבת $N_f=(V,c_f,s,t)$.
**מסילת הרחבה** זו מסילה פשוטה $P$ בין $s$ ל-$t$ בגרף השיורי $G_f$.
**הקיבול השיורי של מסילה** $P$ מוגדר ע״י $c_f(P)=\min_{e\in P}\{c_f(e)\}$.
**הזרימה השיורית** היא פונקציה $\Delta_{f,P}:V\times V\to\mathbb{R}$ שמוגדרת ע״י:

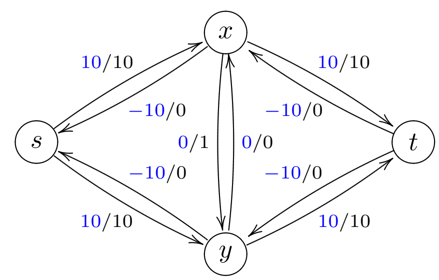
$$\Delta_{f,P}(x,y)=\begin{cases}(x,y)\in P: & c_f(P)\\ (y,x)\in P: & -c_f(P)\\ \text{Otherwise}: & 0\end{cases}$$

**אלגוריתם 10** האלגוריתם של פורד ופולקרסון למציאת זרימה אופטימלית ברשת זרימה

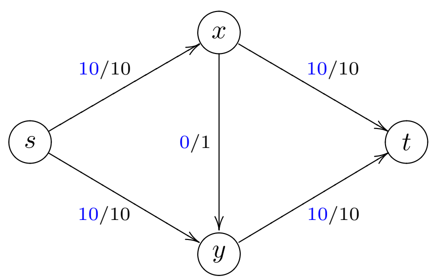
קלט: רשת זרימה רגילה $N=(V,E,c,s,t)$.
פלט: זרימה חוקית $g$ ברשת $N$ בעלת שטף מקסימלי.

1. נרחיב את רשת $N$ לרשת זרימה מורחבת $N'$.
2. אתחול: נגדיר $f\equiv 0$ (זרימת האפס).
3. איטרציה: בכל שלב נמצא מסילת הרחבה $P$ בגרף השיורי $G_f$ (למשל, ע"י BFS), ונעדכן את הזרימה המצטברת $f\leftarrow f+\Delta_{f,P}$.
4. עצירה: נעצור כאשר לא נותרו מסילות הרחבה בגרף השיורי.
5. נצמצם את רשת הזרימה המורחבת $N'$ לרשת זרימה רגילה $N$, ונצמצם את הזרימה $f$ שקיבלנו בשלב הקודם לזרימה רגילה $g$ ברשת $N$.
6. נחזיר את $g$.

איור 18.1 מתאר דוגמה לריצת האלגוריתם על הרשת מאיור 17.1.
הזרימה המצטברת ברשת המורחבת $N'_4$ וצמצומה לזרימה רגילה ברשת המקורית $N$ מתוארות באיור 18.2.

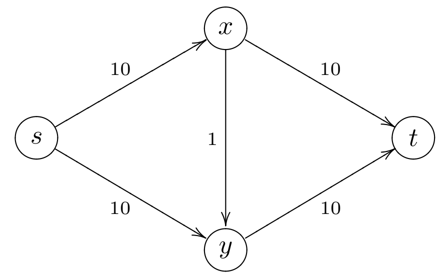



(א) הזרימה המצטברת ברשת המורחבת $N'_4$

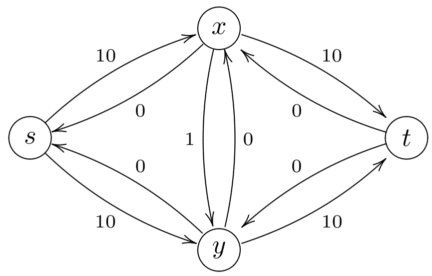



(ב) הצמצום של הזרימה המצטברת לזרימה רגילה

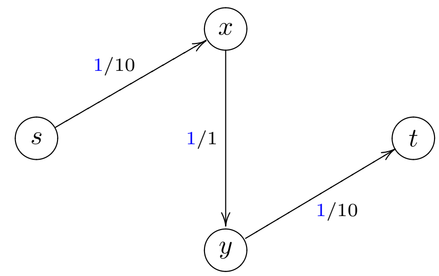
איור 18.2: המחשה לשלב 5 באלגוריתם 10



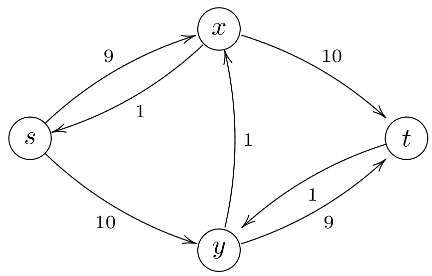

(א) הרשת המקורית $N_0 = N$



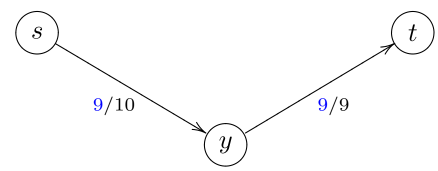

(ב) הרשת השיורית $N_0'$ (שימו לב שצריכה להיות גם צלע בין $s$ ל-$t$ עם קיבולים 0)



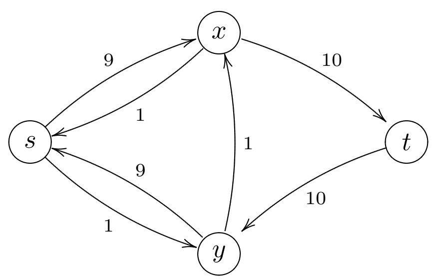

(ג) מסלול $P_1 : s - x - y - t$ וזרימה שמועברת בו $f_1$



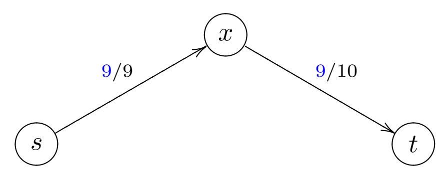

(ד) הרשת השיורית $N_1'$



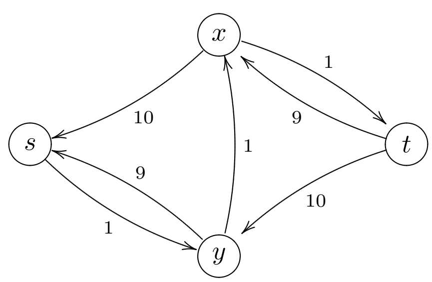

(ה) מסלול $P_2 : s - y - t$ בגרף השיורי וזרימה שמועברת בו $f_2$



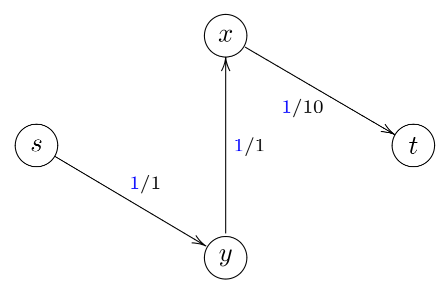

(ו) הרשת השיורית $N_2'$



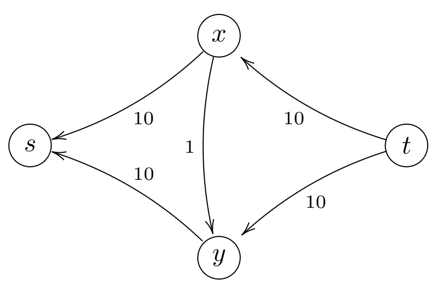

(ז) מסלול $P_3 : s - x - t$ בגרף השיורי וזרימה שמועברת בו $f_3$




(ח) הרשת השיורית $N_3'$




(ט) מסלול $P_4 : s - y - x - t$ בגרף השיורי וזרימה שמועברת בו $f_4$




(י) הרשת השיורית $N_4'$

איור 18.1 : דוגמה לריצת אלגוריתם 10

# 19 שבוע 10 - הרצאה - 19.12.18

## 19.1 רשתות זרימה: ניתוח אלגוריתם פורד-פולקרסון

משמעות הלמה הבאה היא שבאלגוריתם פורד-פולקרסון אנו שומרים על זרימה מורחבת חוקית, באופן כזה שהשטף שלה גדל מאיטרציה לאיטרציה.

**למה 19.1.** תהי $N$ רשת זרימה (מורחבת) ו-$f$ זרימה (מורחבת) ברשת זו. תהי $P$ מסילת הרחבה בגרף השיורי $G_f$, ותהי $g = f + \Delta_{f,P}$. אזי:

(1) הזרימה השיורית $\Delta_{f,P}$ היא זרימה חוקית (מורחבת) ברשת השיורית $N_f$.

(2) הזרימה $g = f + \Delta_{f,P}$ היא זרימה חוקית ברשת $N$ (המורחבת).

(3) $|g| = |f| + c_f\left(P\right)$.

הוכחה.

(1) על פי הגדרת הזרימה השיורית, $\Delta_{f,P} : V \times V \to \mathbb{R}$ וזו פונקציה אנטי-סימטרית. נותר לוודא שאילוץ הקיבול נשמר וחוק שימור החומר.

אילוץ הקיבול: נבדוק כי לכל $(x,y) \in V \times V$ מתקיים $\Delta_{f,P}\left(x,y\right) \le c_f\left(x,y\right)$. נבחין בין שתי אפשרויות:

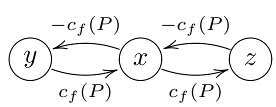
- $\tilde{e} = (x,y) \in P$: במקרה זה

$$\Delta_{f,P}\left(\tilde{e}\right) \overset{\text{על פי}}{\underset{\text{הגדרה}}{=}} c_f\left(P\right) = \min_{e \in P}\left\{c_f\left(e\right)\right\} \le c_f\left(\tilde{e}\right)$$

- $\tilde{e} = (x,y) \notin P$: אזי ע"פ ההגדרה של הזרימה השיורית, $\Delta_{f,P}\left(\tilde{e}\right) \le 0$, ומכך שהקיבול השיורי $c_f\left(\tilde{e}\right)$ הוא אי-שלילי מתקבלת הטענה.

חוק שימור החומר: עלינו לוודא כי לכל $x \in V \setminus \{s,t\}$ מתקיים $\sum_{v \in V} \Delta_{f,P}\left(x,v\right) = 0$. מספיק לוודא זאת רק עבור קודקודים הנמצאים על המסילה $P$ כי לכל קודקוד אחר הזרימה היוצאת היא $0$ באופן מיידי: אם $x$ לא נמצא על המסילה $P$, הזרימה השיורית על כל הצלעות היוצאות מ-$x$ היא $0$ ולכן גם הזרימה היוצאת הכוללת היא $0$.

אם $x$ נמצא ב-$P$: נסמן ב-$y$ את הקודקוד הקודם ל-$x$ וב-$z$ את הקודקוד שבא אחריו (זכרו ש-$x \ne s$ וגם $x \ne t$ לכן יש קודקודים כאלה).




מכאן שהצלעות היחידות היוצאות מ-$x$ עם זרימה לא אפס הן $(x,y)$ ו-$(x,z)$,

$$\begin{aligned}\sum_{v \in V} \Delta_{f,P}\left(x,v\right) &= \Delta_{f,P}\left(x,y\right) + \Delta_{f,P}\left(x,z\right)\\ &= -c_f\left(P\right) + c_f\left(P\right) = 0\end{aligned}$$

ולכן $x$ מקיים את חוק שימור החומר.

(המשך ההוכחה בהרצאה הבאה.)

□

# 20 שבוע 11 - הרצאה - 23.12.18

## 20.1 רשתות זרימה: ניתוח אלגוריתם פורד-פולקרסון - המשך

נמשיך את הוכחת למה 19.1.

הוכחה (של למה 19.1).

(2) עלינו להוכיח כי הזרימה $g=f+\Delta_{f,P}$ היא זרימה חוקית ברשת $N$ (המורחבת).

<u>אנטי-סימטריה</u>: עבור $x,y\in V$ מתקיים:

$$\begin{aligned} g(y,x) &\overset{\text{הגדרת}}{\underset{g}{=}} (f+\Delta_{f,P})(y,x) \\ &= f(y,x)+\Delta_{f,P}(y,x) \\ &= -f(x,y)-\Delta_{f,P}(x,y) && [\text{אנטי סימטריה עבור } f \text{ ו-} \Delta_{f,P}] \\ &= -(f+\Delta_{f,P})(x,y) \\ &= -g(x,y) \end{aligned}$$

<u>חוק שימור החומר</u>: יהי $x\in V\setminus\{s,t\}$, אזי –

$$\begin{aligned} \sum_{v\in V} g(x,v) &= \sum_{v\in V}(f+\Delta_{f,P})(x,v) \\ &= \sum_{v\in V} f(x,v)+\sum_{v\in V}\Delta_{f,P}(x,v) \\ &= 0 && [\text{חוק שימור החומר עבור } f \text{ ו-} \Delta_{f,P}] \end{aligned}$$

<u>אילוץ הקיבול</u>: יהיו $x,y\in V$, אזי –

$$\begin{aligned} g(x,y) &= f(x,y)+\Delta_{f,P}(x,y) \\ &\le f(x,y)+c_f(x,y) && [\text{אילוץ הקיבול עבור } \Delta_{f,P}] \\ &= f(x,y)+(c(x,y)-f(x,y)) && [\text{לפי הגדרת קיבול שיורי}] \\ &= c(x,y) \end{aligned}$$

(3) נחשב את השטף של הזרימה השיורית $\Delta_{f,P}$. נסמן ב-$y$ את הקודקוד הראשון אחרי $s$ במסילה $P$.
הצלע $(s,y)$ היא הצלע היוצאת מ-$s$ היחידה עם זרימה שונה מאפס (לפי הגדרת $\Delta_{f,P}$), לכן –

$$|\Delta_{f,P}|=\sum_{v\in V}\Delta_{f,P}(s,v)=\Delta_{f,P}(s,y)=c_f(P)$$

מכאן נקבל –

$$\begin{aligned} |g| &= \sum_{v\in V} g(s,v) \\ &= \sum_{v\in V} f(s,v)+\sum_{v\in V}\Delta_{f,P}(s,v) \\ &= |f|+c_f(P) \end{aligned}$$

כנדרש.

□

## 20.2 רשתות זרימה: זרימות וחתכים ברשת

**הגדרה 20.1** (חתך ברשת זרימה). **חתך** $(S,T)$ **ברשת זרימה** זו חלוקה של הקודקודים $V = S \sqcup T$ (כאשר $\sqcup$ מציין איחוד זר), כך ש-$s \in S$ ו-$t \in T$.

**הגדרה 20.2** (קיבול חתך ברשת זרימה). **הקיבול של חתך** $(S,T)$ ברשת זרימה מוגדר ע״י

$$c(S,T) := \sum_{x \in S, y \in T} c(x,y)$$

**הגדרה 20.3** (זרימה בחתך ברשת זרימה). תהי $f$ זרימה ברשת (המורחבת) ויהי $(S,T)$ חתך ברשת. נגדיר את **הזרימה בחתך** ע״י

$$f(S,T) := \sum_{x \in S, y \in T} f(x,y)$$

**למה 20.4.** תהי $f$ זרימה ברשת ו-$(S,T)$ חתך ברשת, אזי $f(S,T) = |f|$.

הוכחה.

$$\begin{aligned}
f(S,T) &= \sum_{x \in S, y \in T} f(x,y) \\
&= \sum_{x \in S} \sum_{y \in T} f(x,y) \\
[V = S \sqcup T \text{ כי}] &= \sum_{x \in S} \left( \sum_{v \in V} f(x,v) - \sum_{u \in S} f(x,u) \right) \\
&= \sum_{x \in S} \sum_{v \in V} f(x,v) - \sum_{x \in S} \sum_{u \in S} f(x,u) \qquad (20.1)
\end{aligned}$$

עתה מתקיים:

$$\sum_{x \in S} \sum_{v \in V} f(x,v) = \sum_{v \in V} f(s,v) + \sum_{\substack{x \in S \\ x \neq s}} \sum_{v \in V} f(x,v)$$

נשים לב שאם $x \in S$ ו-$s \neq x$ אז גם $x \neq t$ כי דרשנו בהגדרת החתך ש-$s \in S$ ו-$t \in T$ והקבוצות $S$ ו-$T$ זרות. מכאן נקבל ע״פ חוק שימור החומר:

$$\begin{aligned}
&= \sum_{v \in V} f(s,v) + \sum_{\substack{x \in S \\ x \neq s}} 0 \\
&= |f|
\end{aligned}$$

נחשב

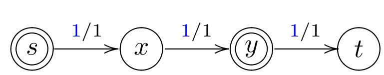
$$\sum_{x \in S} \sum_{u \in S} f(x,u) = \sum_{\{x,u\} \in S} \left( f(x,u) + f(u,x) \right) = \sum_{\{x,u\} \in S} 0 = 0$$

נציב במשוואה (20.1):

$$\begin{aligned}
f(S,T) &= \sum_{x \in S} \sum_{v \in V} f(x,v) - \sum_{x \in S} \sum_{u \in S} f(x,u) \\
&= |f| - 0 \\
&= |f|
\end{aligned}$$

כנדרש. □

**דוגמה 20.5.** נתבונן ברשת הבאה



נגדיר חתך $S=(s,y)$ ו-$T=\{x,t\}$, נסמן ב-$f$ את הזרימה שמצוינת באיור. נחשב את הזרימה שעוברת בחתך $(S,T)$,

$$\begin{aligned} f(S,T) &= f(s,x)+f(s,t)+f(y,x)+f(y,t) \\ &= 1+0+(-1)+1 \\ &= 1=|f| \end{aligned}$$

**למה 20.6.** תהי $g$ זרימה ברשת ויהי $(S,T)$ חתך ברשת, אזי $|g|\le c(S,T)$.

הוכחה. לפי למה 20.4 מתקיים $|g|=g(S,T)$, ולכן –

$$|g|=g(S,T)=\sum_{x\in S,y\in T} g(x,y) \underset{\text{הקיבול}}{\overset{\text{אילוץ}}{\le}} \sum_{x\in S,y\in T} c(x,y)=c(S,T)$$

כנדרש. $\square$

## 20.3 רשתות זרימה: משפט השטף והחתך

**משפט 20.7** (משפט השטף והחתך (Max-Flow Min-Cut Theorem)). תהי $N=(V,c,s,t)$ רשת זרימה (מורחבת) ותהי $f$ זרימה ברשת. שלושת התנאים הבאים שקולים:

**(1)** $f$ זרימה אופטימלית.

**(2)** לא קיימת מסילת הרחבה בגרף השיורי $G_f$.

**(3)** קיים חתך ברשת $(S,T)$ כך ש-$|f|=c(S,T)$.

הוכחה. $(1)\Leftarrow(2)$: נניח בשלילה שקיימת מסילת הרחבה $P$ בגרף השיורי $G_f$. אזי, לפי למה 19.1 הזרימה $g=f+\Delta_{f,P}$ היא זרימה חוקית ברשת ומתקיים –

$$|g|=|f|+c_f(P)>|f|$$

בסתירה לאופטימליות של $f$.
$(3)\Leftarrow(1)$: תהי $g$ זרימה כלשהי ברשת. לפי למה 20.6 מתקיים לכל חתך $(S,T)$ ברשת כי $|g|\le c(S,T)$, בפרט עבור החתך $(S,T)$ שעבורו $|f|=c(S,T)$.
מכאן שמתקיים $|g|\le|f|$, ולכן $f$ אופטימלית.
$(2)\Leftarrow(3)$: נגדיר חתך ברשת באופן הבא:

$$\begin{aligned} S &:= \{s\}\cup\left\{x\in V:\begin{smallmatrix}\text{קיים מסלול בין } s\\ \text{ל-} x \text{ בגרף } G_f\end{smallmatrix}\right\} \\ T &:= V\setminus S \end{aligned}$$

מכיוון שלא קיימת מסילת הרחבה בגרף השיורי $G_f$, אנו מקבלים כי $t\notin S$ ולכן $t\in T$. מהגדרת $S$ מתקיים $s\in S$ ולכן $(S,T)$ מהווה חתך ברשת.
יהי $x\in S$ ו-$y\in T$. נטען כי $c(x,y)=f(x,y)$.
נניח בשלילה כי $c(x,y)\neq f(x,y)$. לפי אילוץ הקיבול בהכרח $f(x,y)<c(x,y)$ ולכן

$$c_f(x,y)=c(x,y)-f(x,y)>0$$

מכאן ש-$(x,y)$ היא צלע בגרף השיורי $E_f$, כלומר $(x,y)\in E_f$.
מכאן נובע שניתן להגיע מקודקוד המקור $s$ ל-$y$: מגיעים מ-$s$ ל-$x$ באמצעות המסילה המובטחת לנו מהגדרת $S$, ולאחר מכן עוברים על הצלע המכוונת $(x,y)$.
הראינו כי קיים מסלול מ-$s$ ל-$y$ בגרף $G_f$, ולכן $y\in S$. זוהי סתירה לכך ש-$y\in T$.
מכאן נסיק שהנחת השלילה שלנו הייתה שגויה, ו-$c(x,y)=f(x,y)$ לכל $x\in S$ ו-$y\in T$.
לבסוף –

$$|f|=f(S,T)=\sum_{x\in S,y\in T} f(x,y)=\sum_{x\in S,y\in T} c(x,y)=c(S,T)$$

כנדרש. $\square$

**מסקנות ממשפט השטף והחתך**

1. כאשר אלגוריתם פורד-פולקרסון עוצר, יש ברשת זרימה $f$ כך שלא קיימת מסילת הרחבה בגרף השיורי $G_f$. לכן, לפי משפט השטף והחתך, $f$ היא זרימה אופטימלית.

2. נניח כי אלגוריתם פורד-פולקרסון עצר עם זרימה $f$, ונגדיר חתך $(S,T)$ ברשת באופן הבא:

$$S := \{s\} \cup \left\{ x \in V : \substack{\text{קיים מסלול בין } s \\ \text{ל-}x \text{ בגרף } G_f} \right\}$$
$$T := V \setminus S$$

אזי החתך $(S,T)$ הוא בעל קיבול מינימלי.

הוכחה. יהי $(U,W)$ חתך ברשת שונה מ-$(S,T)$. אזי –

$$c(S,T) = |f| \leq c(U,W)$$

(השוויון נובע מהוכחת משפט השטף והחתך, האי-שוויון נובע מלמה 20.6.)

מכאן נובע ש-$(S,T)$ הוא חתך ברשת בעל קיבול מינימלי. $\square$

3. ניתן למצוא חתך מינימלי ברשת באופן יעיל: מריצים חיפוש BFS עם קודקוד התחלה $s$ על הגרף השיורי $G_f$ שמתקבל בסוף ריצת אלגוריתם פורד-פולקרסון. מטרת הרצת BFS היא למצוא את קבוצת הקודקודים שנגישים מ-$s$ – בדרך זו אנו מוצאים את הקבוצה $S$. מגדירים את הקבוצה $T$ ע״י $T = V \setminus S$, ומחזירים את החתך $(S,T)$.

4. מינימום הקיבולים של חתכים ברשת שווה למקסימום השטפים על הזרימות ברשת. (בניסוח דומה: הקיבול של חתך מינימלי שווה לשטף של זרימה מקסימלית.)

   מעובדה זו קיבל משפט השטף והחתך את שמו הלועזי: Max-Flow Min-Cut Theorem.

# 21 שבוע 11 - הרצאה - 26.12.18

## 21.1 רשתות זרימה: אלגוריתם פורד-פולקרסון - עצירה וזרימה בשלמים

מתברר שאם הקיבולים ברשת זרימה $N$ הם מספרים ממשיים אז בחירת מסלולים גרועה באלגוריתם פורד-פולקרסון עלולה לגרום למצב שבו האלגוריתם לא עוצר.
אם הקיבולים ברשת זרימה הם שלמים, אז האלגוריתם עוצר, ויתרה מזאת, מחזיר זרימה בשלמים. זהו תוכנה של הטענה הבאה:

**טענה 21.1.** תהי $N=(V,E,c,s,t)$ רשת זרימה כך ש-$c:E\to\mathbb{N}$ (פונקציית הקיבול בשלמים). אזי אלגוריתם פורד-פולקרסון עוצר לאחר לכל היותר $\sum_{v\in V} c\left(s,v\right)$ איטרציות (זה מספר שלם כי כל הקיבולים שלמים), ומחזיר זרימה אופטימלית בשלמים.

הוכחה. נוכיח תחילה כי לאורך ההרצה של האלגוריתם, הזרימות (המורחבות) שנבנות ברשת מקבלות ערכים שלמים.
נוכיח זאת באינדוקציה על מספר האיטרציה.
נסמן ב-$f_i$ את הזרימה ברשת לאחר האיטרציה ה-$i$ של האלגוריתם, ונסמן ב-$P_i$ את מסילת ההרחבה הנבחרת באיטרציה ה-$i$.
<u>בסיס:</u> $i=0$, הזרימה $f_0$ היא זרימת האפס, ולכן מקבלת ערכים שלמים.
<u>צעד:</u> נניח כי $f_i$ מקבלים ערכים שלמים ונוכיח עבור $f_{i+1}$. על-סמך דרך פעולתו של האלגוריתם –

$$f_{i+1}=f_i+\Delta_{f_i,P_i}$$

מספיק להראות כי $\Delta_{f_i,P_i}$ מקבלת רק ערכים שלמים. הזרימה $\Delta_{f_i,P_i}$ מקבלת רק שלושה ערכים שונים $+c_{f_i}\left(P_i\right)$, $-c_{f_i}\left(P_i\right)$ ו-0.
לכן, מספיק לבדוק כי $c_{f_i}\left(P_i\right)$ הוא מספר שלם. לפי ההגדרה –

$$c_{f_i}\left(P_i\right)=\min_{e\in P_i}\left\{c_{f_i}\left(e\right)\right\}=\min_{e\in P_i}\left\{c\left(e\right)-f_i\left(e\right)\right\}$$

נטען כי הקיבול השיורי הוא מינימום של מספר סופי של מספרים שלמים. אכן, על-פי ההנחה $c\left(e\right)$ מספר שלם ו-$f_i\left(e\right)$ מספר שלם לפי הנחת האינדוקציה. מכאן ש-$c_{f_i}\left(P_i\right)$ הוא מספר שלם, כמינימום של מספרים שלמים.
לכן, אם האלגוריתם עוצר, הוא בהכרח יחזיר זרימה אופטימלית בשלמים.
כדי להוכיח עצירה, ניווכח כי לפי למה 19.1 מתקיים –

$$\left|f_{i+1}\right|=\left|f_i\right|+c_{f_i}\left(P_i\right)$$

מכיוון ש-$c_{f_i}\left(P_i\right)$ הוא מינימום של מספרים חיוביים וגם מספר שלם, אנו מקבלים כי $c_{f_i}\left(P_i\right)\geq 1$, לכן –

$$\left|f_{i+1}\right|\geq\left|f_i\right|+1$$

מכיוון ש-$\left|f_0\right|=0$ אנו מקבלים כי $\left|f_i\right|\geq i$ לכל $i$.
מכאן שהאלגוריתם יעצור לאחר לכל היותר $\left|f^*\right|$ איטרציות (כאשר $f^*$ היא זרימה אופטימלית שמוחזרת בסוף האלגוריתם).
מלמה 20.6 אנו יודעים כי לכל זרימה $g$ ברשת ולכל חתך ברשת $(S,T)$ ברשת מתקיים $|g|\leq c\left(S,T\right)$.
בפרט זה נכון עבור $f^*$ וחתך $\left(\left\{s\right\},V\setminus\left\{s\right\}\right)$, לכן –

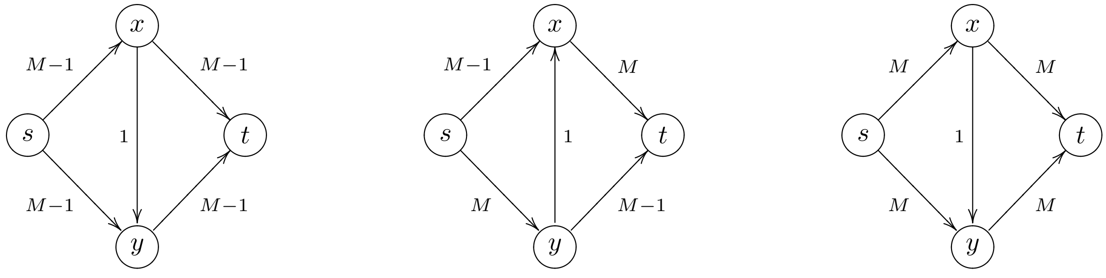
$$\left|f^*\right|\leq c\left(\left\{s\right\},V\setminus\left\{s\right\}\right)=\sum_{v\in V}c\left(s,v\right)$$

ובכך הוכחנו את הטענה. □

**דוגמה 21.2.** יהי $M\in\mathbb{N}$ מספר טבעי "גדול". נרצה להראות שהחסם שקיבלנו עבור מספר האיטרציות באלגוריתם פורד-פולקרסון הדוק.
נתבונן ברשת הזרימה שבאיור 21.1. בכל שלב אנו בוחרים מסילה באורך 3:

1. באיטרציה הראשונה אנו בוחרים מסילה $s-x-y-t$ ומעבירים בה זרימה קבועה השווה ל-1.
2. באיטרציה השניה אנו בוחרים מסילה $s-y-x-t$ ומעבירים בה זרימה קבועה השווה ל-1.

נבחין כי בתום שתי האיטרציות הללו, נשארה רשת דומה לרשת שממנה התחלנו, רק שהקיבולים על הצלעות $\left(s,x\right),\left(x,t\right),\left(s,y\right),\left(y,t\right)$ הם $M-1$.
אם נחזור על בחירת המסילות, נבצע בסה"כ $2M$ איטרציות, שזה בדיוק $c\left(s,x\right)+c\left(s,y\right)$, ולכן החסם שקיבלנו בטענה הקודמת הדוק.




איור 21.1: המחשה לכך שהחסם $\sum_{v\in V} c(s,v)$ על מספר האיטרציות של אלגוריתם פורד-פולקרסון הדוק. באיורים מוצגים הגרפים השיוריים לאחר האיטרציה ה-0, האיטרציה ה-1 והאיטרציה ה-2.

## 21.2 רשתות זרימה: אלגוריתם אדמונדס-קארפ

אלגוריתם אדמונדס-קארפ הוא שיפור של אלגוריתם פורד-פולקרסון, אשר מורה כיצד יש לבחור את המסלולים.
הכלל על-פיו אלגוריתם אדמונדס-קארפ בוחר מסילת הרחבה הוא לפי אורכה, והוא בוחר במסילה הקצרה ביותר.
מבחינת מימוש, יש לבצע חיפוש של מסילת הרחבה בין קודקוד המקור $s$ לקודקוד הבור $t$ באמצעות חיפוש רוחב (BFS), אשר מוצא מסלול קצר ביותר (מבחינת מספר הצלעות) בין $s$ ל-$t$.

**משפט 21.3** (משפט אדמונדס-קארפ)**.** אם בכל שלב באלגוריתם פורד-פולקרסון נבחר את מסילת ההרחבה כמסילה קצרה ביותר בין קודקוד המקור $s$ לקודקוד הבור $t$ בגרף השיורי, אז מספר האיטרציות של האלגוריתם עד העצירה הוא לכל היותר $|V|\cdot|E|$.

**מסקנה 21.4.** זמן הריצה של אלגוריתם אדמונדס-קארפ חסום מלעיל ע״י $O\left(|V||E|\cdot(|V|+|E|)\right)$.

הוכחה. יש $O\left(|V|\cdot|E|\right)$ איטרציות לכל היותר, וכל איטרציה עולה לכל היותר $O\left(|V|+|E|\right)$ מהרצת BFS, עדכון הזרימה והגרף השיורי. $\square$

**תיאור קצר (לא פורמלי) של הוכחת המשפט של אדמונדס-קארפ:** נסמן ב-$f_i$ את הזרימה ברשת אחרי האיטרציה ה-$i$.
נסמן ב-$P_i$ את מסילת ההרחבה הנבחרת באיטרציה ה-$i$.
נסמן ב-$\delta_i\left(x\right)$ את המרחק של הקודקוד $x\in V$ מקודקוד המקור $s$ בגרף השיורי $G_{f_i}$.
למה (מובאת ללא הוכחה): לכל $x\in V$ מתקיים $\delta_0\left(x\right)\leq\delta_1\left(x\right)\leq\ldots$, כאשר אם $x$ מנותק מ-$s$ באיטרציה ה-$i$ אז $\delta_i\left(x\right)=\infty$.
הגדרה: צלע $\left(x,y\right)$ תיקרא **צלע קריטית** באיטרציה ה-$i$ של האלגוריתם אם $\left(x,y\right)\in P_i$ וגם $c_{f_i}\left(x,y\right)=\min\limits_{e\in P_i}\left\{c_{f_i}\left(e\right)\right\}$.
למה (מובאת ללא הוכחה): אם עבור איטרציות שונות $i<j$, הצלע $\left(x,y\right)$ היא צלע קריטית גם באיטרציה ה-$i$ וגם באיטרציה ה-$j$, אזי $\delta_j\left(x\right)\geq\delta_i\left(x\right)+2$.
מכאן נובע שכל צלע יכולה להיות צלע קריטית לכל היותר $\frac{|V|}{2}$ פעמים.
סה״כ יש $2\left|E\right|$ צלעות, ולכן מס' האיטרציות חסום ע״י $2\left|E\right|\cdot\frac{|V|}{2}=|E|\cdot|V|$, כנדרש.

**הערה 21.5.** הוכחה פורמלית של המשפט של אדמונדס-קארפ תועלה ל-Moodle.

# 22 שבוע 12 - הרצאה - 30.12.18

## 22.1 התמרת פורייה המהירה: הקדמה

אנו פונים לנושא חדש וחשוב: DFT (Discrete Fourier Transform) - טרנספורם פורייה בדיד, וחישובו המהיר FFT (Fast Fourier Transform) - טרנספורם פורייה מהיר.
אלגוריתם התמרת פורייה המהירה הוא אלגוריתם בעל מגוון רחב של יישומים:

1. כפל מהיר של פולינומים.
2. כפל מהיר של מספרים.
3. אינטרפולציה פולינומית: בהינתן $n$ ערכים $x_0, \dots, x_{n-1}$ של פולינום ממעלה לכל היותר $n-1$, נרצה לחשב את ערכו בנקודה חדשה $x_n$.
4. מציאת תבנית (pattern) בטקסט, גזירה יעילה, דחיסה, וכו'.

## 22.2 התמרת פורייה המהירה: פעולות על פולינומים

נסמן ב-$V_{n-1}$ את מרחב הפולינומים ממעלה קטנה או שווה $n-1$ עם מקדמים ממשיים. באופן מפורש:

$$V_{n-1} = \left\{a_0 + a_1 x + \dots + a_{n-1}x^{n-1} : a_0, a_1, \dots, a_{n-1} \in \mathbb{R}\right\}$$

**הגדרה 22.1** (ייצוג מקדמים של פולינום)**.** יהי $P \in V_{n-1}$ פולינום, $P(x) = \sum_{k=0}^{n-1} a_k x^k$. נאמר כי **ייצוג המקדמים** של הפולינום $P$ הוא הוקטור $\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$, אשר נכנה בשם: וקטור המקדמים של $P$.

**הערה 22.2.** ייצוג המקדמים של הפולינום הוא הייצוג של הפולינום לפי בסיס מסוים של $V_{n-1}$, שהוא בסיס החזקות:

$$1, x, x^2, \dots, x^{n-1}$$

אנו נתעניין בביצוע פעולות על פולינומים בצורה יעילה. הפעולות שבהן נתעניין הם: חיבור פולינומים, הצבת ערך בפולינום וכפל פולינומים.
תחילה נבחן את יעילות ביצוע הפעולות כאשר הפולינומים נתונים בייצוג המקדמים שלהם.

1. **חיבור פולינומים**

   קלט: שני פולינומים $P, Q \in V_{n-1}$ הנתונים בייצוג המקדמים שלהם.

   פלט: פולינום $R \in V_{n-1}$ המקיים $R = P + Q$, בייצוג המקדמים.

   אלגוריתם: נסמן את וקטורי המקדמים של $P$ ו-$Q$ בתור $\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$, $\begin{pmatrix} b_0 \\ \vdots \\ b_{n-1} \end{pmatrix}$ בהתאמה. נחשב את וקטור המקדמים של $R$ ע"י

$$\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} + \begin{pmatrix} b_0 \\ \vdots \\ b_{n-1} \end{pmatrix} = \begin{pmatrix} a_0 + b_0 \\ \vdots \\ a_{n-1} + b_{n-1} \end{pmatrix}$$

   זמן ריצה: $O(n)$.

2. **הצבת ערך בפולינום**

   קלט: פולינום $P \in V_{n-1}$ בייצוג מקדמים ומספר ממשי $x_0 \in \mathbb{R}$.

   פלט: הערך $P(x_0)$.

   אלגוריתם: נסמן את וקטור המקדמים של $P$ ב- $\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$.

נחשב את הסדרה $1, x_0, x_0^2, \ldots, x_0^{n-1}$ באופן כזה שאת $x_0^k$ נחשב ע"י כפל של $x_0$ בערך האחרון שחושב, כלומר $x_0 \cdot x_0^{k-1}$.
הערך של $P(x_0)$ הוא חישוב הסכום:

$$P(x_0) = \sum_{k=0}^{n-1} a_k x_0^k$$

<u>זמן ריצה:</u> חישוב הסדרה $1, x_0, \ldots, x_0^{n-1}$ לוקח $O(n)$. את חישוב הסכום $\sum_{k=0}^{n-1} a_k x_0^k$ ניתן לבצע ב-$O(n)$ פעולות אריתמטיות. סה"כ: $O(n)$.

3. **כפל פולינומים**

<u>קלט:</u> שני פולינומים $P, Q \in V_{n-1}$ בייצוג המקדמים.

<u>פלט:</u> פולינום $R \in V_{2n-2}$ המקיים $R = P \cdot Q$ בייצוג המקדמים.

<u>אלגוריתם:</u> נסמן את וקטורי המקדמים של $P$ ו-$Q$ ב-$\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$, $\begin{pmatrix} b_0 \\ \vdots \\ b_{n-1} \end{pmatrix}$ בהתאמה.

נסמן את וקטור המקדמים של $R$ ב-$\begin{pmatrix} c_0 \\ \vdots \\ c_{2n-2} \end{pmatrix}$. הנוסחה עבור המקדם ה-$m$ היא:

$$c_m = a_0 b_m + a_1 b_{m-1} + \cdots + a_m b_0 = \sum_{j=0}^{m} a_j b_{m-j}$$

כאשר גורמים בנוסחה שהם חסרי משמעות שווים 0 (למשל, עבור $m = n$ בוודאי אין את הגורמים $a_0 b_n$ ו-$a_n b_0$ כי $b_n$ ו-$a_n$ לא מוגדרים. יחד עם זאת, ניתן לחשוב על הפולינומים $P$ ו-$Q$ כעל פולינומים ב-$V_{2n-2}$ ואז המקדמים $a_n, \ldots, a_{2n-2}, b_n, \ldots, b_{2n-2}$ שווים 0). לצורך ההמחשה:

$$\begin{aligned}
c_0 &= a_0 b_0 \\
c_1 &= a_0 b_1 + a_1 b_0 \\
&\vdots \\
c_{n-1} &= a_0 b_{n-1} + a_1 b_{n-2} + \cdots + a_{n-1} b_0 \\
&\vdots \\
c_{2n-2} &= a_{n-1} b_{n-1}
\end{aligned}$$

<u>זמן ריצה:</u> חישוב המקדם $c_0$ דורש פעולת כפל יחידה, חישוב המקדם $c_1$ דורש 2 פעולות כפל, וכן הלאה, חישוב המקדם $c_{n-1}$ דורש $n$ פעולות כפל. מחישוב המקדם $c_n$ ואילך חל שינוי, ומספר פעולות הכפל הדרושות עבור חישוב המקדם $c_n$ הוא $n-1$, וכן הלאה, חישוב המקדם $c_{2n-2}$ דורש פעולת כפל יחידה. בסה"כ, אם נחשב את המקדמים של פולינום המכפלה בצורה נאיבית בייצוג המקדמים, יעלה חישוב זה $\Theta(n^2)$.

עקב אכילס של ייצוג המקדמים הוא החישוב היקר של פעולת כפל הפולינומים.
אמנם הייצוג "הטבעי" של פולינומים מבחינתנו הוא ייצוג המקדמים, אך אין זו הדרך היחידה בה נוכל לייצג פולינומים במחשב. ייצוג אחר של פולינומים הוא ייצוג הערכים.

**הגדרה 22.3** (ייצוג הערכים של פולינום)**.** יהי $P \in V_{n-1}$ פולינום. נקבע $n$ נקודות שונות $x_0, x_1, \ldots, x_{n-1}$ ונאמר שייצוג הערכים של הפולינום $P$ הוא הוקטור $\begin{pmatrix} P(x_0) \\ \vdots \\ P(x_{n-1}) \end{pmatrix}$.

**הערה 22.4.** ההגדרה של ייצוג הערכים מוגדרת היטב, שכן מתבססת על העובדה האלגברית הבאה: פולינום (שאינו פולינום האפס) ממעלה לכל היותר $n-1$ מעל שדה $\mathbb{F}$ נקבע ביחידות ע"י ערכיו ב-$n$ נקודות שונות. עובדה זו שקולה לכך שלפולינום ממעלה לכל היותר $n-1$ יכולים להיות לכל היותר $n-1$ שורשים שונים בשדה $\mathbb{F}$.

נפנה כעת לבחינת יעילות ביצוע שלושת הפעולות על פולינומים שהוגדרו, הפעם, כאשר הפולינומים נתונים בייצוג הערכים שלהם. במהלך הניתוח אנו קובעים $n$ נקודות שונות $x_0, x_1, \dots, x_{n-1}$, ומעריכים את הפולינומים שלנו באותם נקודות.

1. **חיבור פולינומים**

קלט: שני פולינומים $P, Q \in V_{n-1}$ הנתונים בייצוג הערכים שלהם עבור $n$ נקודות שונות $x_0, x_1, \dots, x_{n-1}$.

פלט: פולינום $R \in V_{n-1}$ המקיים $R = P + Q$, בייצוג הערכים.

אלגוריתם: וקטור הערכים של הפולינום $R$ הוא הסכום $\begin{pmatrix} (P+Q)(x_0) \\ \vdots \\ (P+Q)(x_{n-1}) \end{pmatrix} = \begin{pmatrix} Q(x_0) \\ \vdots \\ Q(x_{n-1}) \end{pmatrix} + \begin{pmatrix} P(x_0) \\ \vdots \\ P(x_{n-1}) \end{pmatrix}$.

זמן ריצה: $O(n)$.

2. **הצבת ערך בפולינום**

קלט: מספר ממשי $x_n \in \mathbb{R}$ ופולינום $P \in V_{n-1}$ בייצוג הערכים עבור $n$ נקודות שונות: $x_0, x_1, \dots, x_{n-1}$.

פלט: הערך $P(x_n)$.

אלגוריתם (העשרה): יהיו $n$ פולינומים $L_0, L_1, \dots, L_{n-1} \in V_{n-1}$ המקיימים:

$$L_m(x_k) = \delta_{mk} = \begin{cases} 1 & m = k \\ 0 & m \neq k \end{cases}$$

באופן מפורש:

$$L_m(x) = \frac{\prod_{k \neq m}(x - x_k)}{\prod_{k \neq m}(x_m - x_k)}$$

ניווכח כי פולינום המוגדר בצורה כזו נותן ערך $0$ לכל $x_k$ עבורו $m \neq k$ ונותן ערך $1$ עבור $x_m$, בדיוק כמו שרצינו.

נטען כי הפולינומים $P$ ו-$\sum_{m=0}^{n-1} P(x_m) L_m$ הם שווים - זאת משום שהם מזדהים על $n$ הנקודות $x_0, x_1, \dots, x_{n-1}$.

מכאן, לכל $x \in \mathbb{R}$:

$$P(x) = \sum_{m=0}^{n-1} P(x_m) L_m(x)$$

בפרט –

$$P(x_n) = \sum_{m=0}^{n-1} P(x_m) L_m(x_n)$$

זמן ריצה: תחילה, לכל $0 \leq m \leq n-1$ עלינו לחשב את הפולינום $L_m$, זה לוקח $O(n)$.

על-מנת לחשב את $P(x_n)$, עלינו לחשב את $L_m(x_n)$ לכל $0 \leq m \leq n-1$ וזה לוקח $\Theta(n^2)$. (כי עבור $m$ ספציפי, חישוב של $L_m(x_n)$ לוקח $\Theta(n)$).

3. **כפל פולינומים**

קלט: שני פולינומים $P, Q \in V_{n-1}$ הנתונים בייצוג הערכים שלהם עבור $2n-1$ נקודות שונות $x_0, x_1, \dots, x_{2n-1}$.

פלט: פולינום $R \in V_{2n-2}$ המקיים $R = P \cdot Q$, בייצוג הערכים.

אלגוריתם: $R(x_k) = P(x_k) Q(x_k)$ לכל $0 \leq k \leq 2n-1$. שימו לב שאנו צריכים להעריך את הפולינומים $P$ ו-$Q$ ב-$2n-1$ נקודות שונות (בניגוד ל-$n$ נקודות שונות כפי שדרשנו בפעולות קודמות) על מנת שפולינום המכפלה $R$ יהיה מוגדר היטב.

זמן ריצה: $O(n)$.

לסיכום, פעולות חיבור פולינומים והצבת ערך בפולינום יעילות ($O(n)$) בייצוג המקדמים, אולם פעולת כפל פולינומים יקרה $\Theta(n^2)$.
לעומת זאת, פעולות חיבור פולינומים וכפל פולינומים יעילות ($O(n)$) בייצוג הערכים, אולם פעולת הצבת ערך בפולינום יקרה $\Theta(n^2)$.

## 22.3 התמרת פורייה המהירה: מעבר בין ייצוגים שונים של פולינומים

נבחין כי ייצוג המקדמים וייצוג הערכים הם ייצוגים של האיברים במרחב הוקטורי $V_{n-1}$ לפי שני בסיסים שונים: בסיס החזקות $1, x, \ldots, x^{n-1}$ ובסיס הפולינומים $L_0, \ldots, L_{n-1}$ שהוגדרו באלגוריתם הצבת ערך של ייצוג הערכים. על כן, המעבר בין שני הייצוגים ניתן ע״י מטריצת המעבר בין הבסיסים.

**למה 22.5.** תהיינה $n$ נקודות ממשיות שונות, $x_0, \ldots, x_{n-1} \in \mathbb{R}$.
יהי $P \in V_{n-1}$ ויהיו $\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$ ייצוג המקדמים של $P$ ו-$\begin{pmatrix} P(x_0) \\ \vdots \\ P(x_{n-1}) \end{pmatrix}$ ייצוג הערכים של $P$. <u>אזי</u>:

$$\begin{pmatrix} P(x_0) \\ \vdots \\ P(x_{n-1}) \end{pmatrix} = M \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$$

כאשר $M$ היא מטריצה מסדר $n \times n$ (נקראת גם **מטריצת ונדרמונד**)

$$M = \begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \ldots & x_{n-1}^{n-1} \end{pmatrix}$$

הוכחה. יהי $0 \le k \le n-1$. נתבונן באיבר ה-$k$ בוקטור $M \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$, ונרצה להוכיח שהוא הסקלר $P(x_k)$. ואכן:

$$\left( M \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} \right)_k = \sum_{m=0}^{n-1} x_k^m a_m = P(x_k)$$

□

**דוגמה 22.6.** נתבונן בפולינום $P(x) = x^2 - x + 1$, ובשלושת הנקודות $x_0 = 1$, $x_1 = 2$, $x_2 = 3$.
ניתן לכתוב את $P$ בייצוג המקדמים כך: $\begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$, ובייצוג הערכים כך: $\begin{pmatrix} 1 \\ 3 \\ 7 \end{pmatrix}$. מטריצת ונדרמונד עבור $x_0, x_1, x_2$ היא:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix}$$

ואכן מתקיים:

$$\begin{pmatrix} 1 \\ 3 \\ 7 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$$

בהלימה עם למה 22.5.

## 22.4 התמרת פורייה המהירה: שימוש בשורשי היחידה המרוכבים

מלמה 22.5 אנו מסיקים כי מעבר מייצוג המקדמים לייצוג הערכים ניתן ע״י כפל במטריצה ספציפית התלויה בנקודות $x_0, \ldots, x_{n-1}$.
מעבר חזרה מייצוג הערכים לייצוג המקדמים ניתן ע״י כפל ב-$M^{-1}$. כידוע, כפל של מטריצה בוקטור עולה $\Theta\left(n^2\right)$ פעולות אריתמטיות, ולכן המעבר בין הייצוגים השונים יקר לפחות כמו הפעולה היקרה בכל אחד מן הייצוגים.
הרעיון המבריק הוא שימוש במספרים מרוכבים, ובפרט, בשורשי היחידה המרוכבים.

נזכור כי $\mathbb{R}$ הוא תת-שדה של $\mathbb{C}$ (שדה המספרים מרוכבים), ולכן אם $P \in V_{n-1}$ הוא פולינום עם מקדמים ממשיים ממעלה קטנה או שווה $n-1$, אז הוא גם פולינום עם מקדמים מרוכבים ממעלה קטנה או שווה $n-1$, ולכן ניתן להציב בו ערכים מרוכבים.
יתרה מזאת, העובדה האלגברית שלפולינום ממעלה לכל היותר $n-1$ יכולים להיות לכל היותר $n-1$ שורשים שונים בשדה, מתקיימת גם עבור שדה המרוכבים.
מכאן נסיק, $P$ נקבע באופן יחיד ע״י הערכים שלו ב-$n$ נקודות מרוכבות שונות.

# 23 שבוע 12 - הרצאה - 2.1.19

## 23.1 התמרת פורייה המהירה: שימוש בשורשי היחידה המרוכבים - המשך

כזכור, המעבר בין ייצוג המקדמים של פולינום לייצוג המקדמים שלו נתון ע"י מכפלה של וקטור המקדמים במטריצת ונדרמונד. "דרגת החופש" שנשארה היא בחירת הנקודות $x_0,\dots,x_{n-1}$ בהן נעריך את הפולינום. הרעיון הוא לחשב את ייצוג הערכים של פולינום בשורשי היחידה מסדר $n$.

**סקירה קצרה של מספרים מרוכבים** שדה המספרים המרוכבים הוא $\mathbb{C}=\{a+bi|a,b\in\mathbb{R}\}$ כאשר $i^2=-1$.
עבור מספר מרוכב $z=a+bi$ ניתן להתאים נקודה $(a,b)$ ב-$\mathbb{R}^2$. כל נקודה שונה מאפס ב-$\mathbb{R}^2$ ניתנת לייצוג ע"י מרחקה מהראשית והזווית שהיא יוצרת עם ציר ה-$x$ החיובי. מכאן שניתן להציג את $z$ גם ע"י הערך המוחלט שלו $|z|=\sqrt{a^2+b^2}$ והזווית שהוא יוצר עם ציר ה-$x$ החיובי.
במישור המרוכב, מעגל היחידה הוא כל המספרים המרוכבים $z$ שמקיימים $|z|=1$. ניתן לרשום כל נקודה $z$ במעגל היחידה בצורה

$$z=\cos\theta+i\sin\theta\quad 0\leq\theta<2\pi$$

כאשר מכפילים שני מספרים מרוכבים $z_1=\cos\theta+i\sin\theta$, $z_2=\cos\varphi+i\sin\varphi$ על מעגל היחידה מקבלים

$$z_1z_2=(\cos\theta+i\sin\theta)(\cos\varphi+i\sin\varphi)=\cos(\theta+\varphi)+i\sin(\theta+\varphi)$$

שורשי היחידה מסדר $n$ הם כל השורשים המרוכבים של הפולינום $z^n=1$, ויש בדיוק $n$ שורשים כאלה.

**דוגמה 23.1.** שורשי היחידה מסדר 2 הם 1 ו-$(-1)$; שורשי היחידה מסדר 4 הם $1$, $i$, $-1$ ו-$(-i)$.

בהינתן מספר טבעי $n\in\mathbb{N}$, נגדיר

$$\omega_n=\cos\left(\frac{2\pi}{n}\right)+i\sin\left(\frac{2\pi}{n}\right)$$

נשים לב כי מתקיים:

$$\omega_n^n=\cos\left(n\cdot\frac{2\pi}{n}\right)+i\sin\left(n\cdot\frac{2\pi}{n}\right)=1$$

לכן $\omega_n$ הוא שורש יחידה מסדר $n$. נשים לב לכך שלכל $0\leq k\leq n-1$ המספר

$$\omega_n^k=\cos\left(\frac{2\pi k}{n}\right)+i\sin\left(\frac{2\pi k}{n}\right)$$

גם הוא שורש יחידה מסדר $n$. ואכן:

$$\left(\omega_n^k\right)^n=\left(\omega_n^n\right)^k=1^k=1$$

יתרה מזאת, כל הנקודות $\left\{\omega_n^k\right\}_{k=0}^{n-1}$ שונות זו מזו.
בצורה כזו קיבלנו את כל $n$ שורשי היחידה מסדר $n$ כחזקות של $\omega_n$. נכנה את $\omega_n$ בשם שורש יחידה פרימיטיבי מסדר $n$.

כעת יש בידינו את הכלים להגדרת טרנספורם פורייה בדיד מסדר $n$.

**הגדרה 23.2** (טרנספורם פורייה בדיד)**.** תהי $a_0,\dots,a_{n-1}\in\mathbb{C}$ סדרה של $n$ מספרים. נגדיר $P(z)$ להיות הפולינום:

$$P(z)=\sum_{k=0}^{n-1}a_kz^k$$

יהיו $1,\omega_n,\omega_n^2,\dots,\omega_n^{n-1}$ $n$ שורשי היחידה המרוכבים מסדר $n$, ז"א, כל השורשים של המשוואה $z^n=1$.
טרנספורם פורייה בדיד מסדר $n$, $\mathrm{DFT}_n$ מוגדר ע"י:

$$\mathrm{DFT}_n\begin{pmatrix}a_0\\\vdots\\a_{n-1}\end{pmatrix}:=\begin{pmatrix}P(\omega_0)\\\vdots\\P(\omega_{n-1})\end{pmatrix}$$

טרנספורם פורייה בדיד הפוך מסדר $n$, $\mathrm{DFT}_n^{-1}$ מוגדר ע"י:

$$\mathrm{DFT}_n^{-1}\begin{pmatrix}P(\omega_0)\\\vdots\\P(\omega_{n-1})\end{pmatrix}:=\begin{pmatrix}a_0\\\vdots\\a_{n-1}\end{pmatrix}$$

מטריצת ונדרמונד עבור בחירת הנקודות $1, \omega_n, \omega_n^2, \ldots, \omega_n^{n-1}$ לובשת צורה מיוחדת:

$$M = \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n & \omega_n^2 & \ldots & \omega_n^{n-1} \\ 1 & \omega_n^2 & \omega_n^4 & \ldots & \omega_n^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{n-1} & \omega_n^{2(n-1)} & \ldots & \omega_n^{(n-1)(n-1)} \end{pmatrix}$$

מתברר שעבור בחירת נקודות זו קל לחשב את המטריצה ההפוכה:

$$M^{-1} = \frac{1}{n} \begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & \omega_n^{-1} & \omega_n^{-2} & \ldots & \omega_n^{-(n-1)} \\ 1 & \omega_n^{-2} & \omega_n^{-4} & \ldots & \omega_n^{-2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_n^{-(n-1)} & \omega_n^{-2(n-1)} & \ldots & \omega_n^{-(n-1)(n-1)} \end{pmatrix}$$

**משפט 23.3.** יהי $n$ חזקה של $2$.

1. יהי $\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} \in \mathbb{C}^n$. אזי ניתן לחשב את $\mathrm{DFT}_n \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$ בזמן $O\left(n \log n\right)$.
2. יהי $\begin{pmatrix} p_0 \\ \vdots \\ p_{n-1} \end{pmatrix} \in \mathbb{C}^n$. אזי ניתן לחשב את $\mathrm{DFT}_n^{-1} \begin{pmatrix} p_0 \\ \vdots \\ p_{n-1} \end{pmatrix}$ בזמן $O\left(n \log n\right)$.

# 24 שבוע 13 - הרצאה - 6.1.19

## 24.1 התמרת פורייה המהירה: אלגוריתם הפרד-ומשול

לצורך הוכחת משפט 23.3 אנו זקוקים למספר למות טכניות.

**למה 24.1.** יהי $n$ מספר זוגי. יהי $P(z)=\sum_{k=0}^{n-1}a_kz^k$ פולינום ממעלה לכל היותר $n-1$. נגדיר

$$P_0(y)=\sum_{j=0}^{\frac{n}{2}-1}a_{2j}y^j$$
$$P_1(y)=\sum_{\ell=0}^{\frac{n}{2}-1}a_{2\ell+1}y^\ell$$

אזי לכל $z\in\mathbb{C}$ מתקיים $P(z)=P_0\left(z^2\right)+z\cdot P_1\left(z^2\right)$.

הוכחה. נפתח את אגף ימין ונגיע לאגף שמאל:

$$\begin{aligned}P_0\left(z^2\right)+zP_1\left(z^2\right)&=\sum_{j=0}^{\frac{n}{2}-1}a_{2j}z^{2j}+z\sum_{\ell=0}^{\frac{n}{2}-1}a_{2\ell+1}z^{2\ell}\\&=\sum_{j=0}^{\frac{n}{2}-1}a_{2j}z^{2j}+\sum_{\ell=0}^{\frac{n}{2}-1}a_{2\ell+1}z^{2\ell+1}\\&=\sum_{k=0}^{n-1}a_kz^k\\&=P(z)\end{aligned}$$

כנדרש. □

**דוגמה 24.2.** נתבונן בפולינום $P(z)=z^3-3z^2+2z+1$, כאשר $n=4$. אז מתקיים:

$$\begin{aligned}P(z)&=z^3-3z^2+2z+1\\&=\left(z^3+2z\right)+\left(-3z^2+1\right)\\&=P_0\left(z^2\right)+zP_1\left(z^2\right)\end{aligned}$$

כאשר $P_0(y)=-3y+1$ ו- $P_1(y)=y+2$.

**למה 24.3.** יהי $n>0$ מספר זוגי ו- $\omega_n=\cos\left(\frac{2\pi}{n}\right)+i\sin\left(\frac{2\pi}{n}\right)$ שורש יחידה פרימיטיבי מסדר $n$. אזי:

1. $\omega_n^{\frac{n}{2}}=-1$
2. $\omega_n^2=\omega_{\frac{n}{2}}$

הוכחה.

1. $\omega_n^{\frac{n}{2}}=\cos\left(\frac{2\pi}{n}\cdot\frac{n}{2}\right)+i\sin\left(\frac{2\pi}{n}\cdot\frac{n}{2}\right)=\cos(\pi)+i\sin(\pi)=-1$.
2. $\omega_n^2=\cos\left(\frac{2\pi}{n}\cdot2\right)+i\sin\left(\frac{2\pi}{n}\cdot2\right)=\cos\left(\frac{2\pi}{n/2}\right)+i\sin\left(\frac{2\pi}{n/2}\right)=\omega_{\frac{n}{2}}$.

□

**למה 24.4.** יהי $n>0$ מספר זוגי ויהי $0\le j\le\frac{n}{2}-1$. אזי $\left(\omega_n^j\right)^2=\left(\omega_n^{\frac{n}{2}+j}\right)^2$.

הוכחה. $\left(\omega_n^{\frac{n}{2}+j}\right)^2=\left(\omega_n^{\frac{n}{2}}\right)^2\left(\omega_n^j\right)^2=(-1)^2\left(\omega_n^j\right)^2=\left(\omega_n^j\right)^2$. □

**מסקנה 24.5.** יהי $n>0$ מספר זוגי. לכל $0\le j\le \frac{n}{2}-1$ מתקיים $\left(\omega_n^{\frac{n}{2}+j}\right)^2=\left(\omega_n^j\right)^2=\omega_{\frac{n}{2}}^j$.

הוכחה. השוויון $\left(\omega_n^{\frac{n}{2}+j}\right)^2=\left(\omega_n^j\right)^2$ נכון מלמה 24.4. בנוסף, $\left(\omega_n^j\right)^2=\left(\omega_n^2\right)^j=\omega_{\frac{n}{2}}^j$, כאשר השוויון האחרון נכון מלמה 24.3. $\square$

**דוגמה 24.6.** נתבונן ב-4 שורשי היחידה מסדר 4: $\left(\omega_4^0,\omega_4^1,\omega_4^2,\omega_4^3\right)=(1,i,-1,-i)$, ונזכור כי $\left(\omega_2^0,\omega_2^1\right)=(1,-1)$. אז מתקיים:

$$\begin{aligned}\left(\omega_4^0\right)^2&=1^2=1=\omega_2^0\\ \left(\omega_4^1\right)^2&=i^2=-1=\omega_2^1\\ \left(\omega_4^2\right)^2&=(-1)^2=1=\omega_2^0\\ \left(\omega_4^3\right)^2&=(-i)^2=-1=\omega_2^1\end{aligned}$$

דוגמה זו באה להמחיש את התכונה הבאה של שורשי היחידה: העלאה בריבוע של שורשי היחידה מסדר $n$ נותנת את שורשי היחידה מסדר $\frac{n}{2}$ (פעמיים).

כעת נפנה להוכחת משפט 23.3.

הוכחה (של משפט 23.3). יהי $n$ חזקה של 2.

יהי $\begin{pmatrix}a_0\\ \vdots\\ a_{n-1}\end{pmatrix}\in\mathbb{C}^n$ ונסמן ב-$P$ את הפולינום: $P(z)=\sum\limits_{k=0}^{n-1}a_kz^k$.

לפי הגדרת ה-$\mathrm{DFT}_n$ מתקיים: $\begin{pmatrix}P(1)\\ P(\omega_n)\\ \vdots\\ P\left(\omega_n^{n-1}\right)\end{pmatrix}=\begin{pmatrix}a_0\\ a_1\\ \vdots\\ a_{n-1}\end{pmatrix}\mathrm{DFT}_n$. לפי למה 24.1 ניתן לכתוב את $P$ בתור:

$$P(z)=P_0\left(z^2\right)+zP_1\left(z^2\right)$$

כאשר $P_0(y)=\sum\limits_{j=0}^{\frac{n}{2}-1}a_{2j}y^j$ ו-$P_1(y)=\sum\limits_{\ell=0}^{\frac{n}{2}-1}a_{2\ell+1}y^\ell$. מכאן –

$$\begin{pmatrix}P(1)\\ P(\omega_n)\\ \vdots\\ P\left(\omega_n^{n-1}\right)\end{pmatrix}=\begin{pmatrix}P_0\left(1^2\right)+1\cdot P_1\left(1^2\right)\\ P_0\left(\omega_n^2\right)+\omega_n\cdot P_1\left(\omega_n^2\right)\\ \vdots\\ P_0\left(\left(\omega_n^{n-1}\right)^2\right)+\omega_n^{n-1}\cdot P_1\left(\left(\omega_n^{n-1}\right)^2\right)\end{pmatrix}$$

$$=\left(\begin{array}{c}P_0\left(1^2\right)\\ P_0\left(\omega_n^2\right)\\ \vdots\\ P_0\left(\left(\omega_n^{\frac{n}{2}-1}\right)^2\right)\\ \hline P_0\left(\left(\omega_n^{\frac{n}{2}}\right)^2\right)\\ \vdots\\ P_0\left(\left(\omega_n^{n-1}\right)^2\right)\end{array}\right)+\left(\begin{array}{c}1\cdot P_1\left(1^2\right)\\ \omega_n\cdot P_1\left(\omega_n^2\right)\\ \vdots\\ \omega_n^{\frac{n}{2}-1}\cdot P_1\left(\left(\omega_n^{\frac{n}{2}-1}\right)^2\right)\\ \hline \omega_n^{\frac{n}{2}}\cdot P_1\left(\left(\omega_n^{\frac{n}{2}}\right)^2\right)\\ \vdots\\ \omega_n^{n-1}\cdot P_1\left(\left(\omega_n^{n-1}\right)^2\right)\end{array}\right)$$

נפעיל את מסקנה 24.5 ונקבל –

$$= \begin{pmatrix} P_0(1) \\ P_0\left(\omega_{\frac{n}{2}}\right) \\ \vdots \\ P_0\left(\omega_{\frac{n}{2}}^{\frac{n}{2}-1}\right) \\ \hline P_0(1) \\ \vdots \\ P_0\left(\omega_{\frac{n}{2}}^{\frac{n}{2}-1}\right) \end{pmatrix} + \begin{pmatrix} 1 \cdot P_1(1) \\ \omega_n \cdot P_1\left(\omega_{\frac{n}{2}}\right) \\ \vdots \\ \omega_n^{\frac{n}{2}-1} \cdot P_1\left(\omega_{\frac{n}{2}}^{\frac{n}{2}-1}\right) \\ \hline \omega_n^{\frac{n}{2}} \cdot P_1(1) \\ \vdots \\ \omega_n^{n-1} \cdot P_1\left(\omega_{\frac{n}{2}}^{\frac{n}{2}-1}\right) \end{pmatrix}$$

כעת, לפי הגדרת $P_0$ וקטור המקדמים שלו הוא $\begin{pmatrix} a_0 \\ a_2 \\ \vdots \\ a_{n-2} \end{pmatrix}$ ולכן לפי הגדרת $\mathrm{DFT}_{\frac{n}{2}}$ מתקיים: $\mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_0 \\ a_2 \\ \vdots \\ a_{n-2} \end{pmatrix} = \begin{pmatrix} P_0(1) \\ P_0\left(\omega_{\frac{n}{2}}\right) \\ \vdots \\ P_0\left(\omega_{\frac{n}{2}}^{\frac{n}{2}-1}\right) \end{pmatrix}$.

באופן דומה, עבור $P_1$ מתקיים: $\mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_1 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} P_1(1) \\ P_1\left(\omega_{\frac{n}{2}}\right) \\ \vdots \\ P_1\left(\omega_{\frac{n}{2}}^{\frac{n}{2}-1}\right) \end{pmatrix}$.

הראינו כי:

$$\mathrm{DFT}_n \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_{n-1} \end{pmatrix} = \begin{pmatrix} | \\ \mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_0 \\ a_2 \\ \vdots \\ a_{n-2} \end{pmatrix} \\ | \\ \hline | \\ \mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_0 \\ a_2 \\ \vdots \\ a_{n-2} \end{pmatrix} \\ | \end{pmatrix} + \begin{pmatrix} 1 \\ \omega_n \\ \vdots \\ \omega_n^{n-1} \end{pmatrix} \bullet \begin{pmatrix} | \\ \mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_1 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix} \\ | \\ \hline | \\ \mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_1 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix} \\ | \end{pmatrix} \tag{24.1}$$

כאשר $\bullet$ הוא סימון נוח לפעולה (לא פורמלית) שהיא "כפל איבר-איבר".

מהצגה זו של $\mathrm{DFT}_n$ אנו רואים אלגוריתם רקורסיבי לחישובו: בהינתן וקטור $\begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix} \in \mathbb{C}^n$ נחשב באופן רקורסיבי את $\mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_0 \\ a_2 \\ \vdots \\ a_{n-2} \end{pmatrix}$ ואת $\mathrm{DFT}_{\frac{n}{2}} \begin{pmatrix} a_1 \\ a_3 \\ \vdots \\ a_{n-1} \end{pmatrix}$, ונשתמש במשוואה (24.1) בכדי לחשב את $\mathrm{DFT}_n \begin{pmatrix} a_0 \\ \vdots \\ a_{n-1} \end{pmatrix}$.

לגבי זמן הריצה, נסמן ב-$T(n)$ את זמן הריצה לחישוב $\mathrm{DFT}_n$. ממשוואה (24.1) אנו מקבלים: $T(n) \le 2T\left(\frac{n}{2}\right) + 3n$ ($2 \cdot \frac{n}{2}$ פעולות כדי לרשום את העותק השני של $\mathrm{DFT}_{\frac{n}{2}}$, $n$ פעולות כדי לחשב את פעולת הכפל איבר-איבר $\bullet$, ו-$n$ פעולות חיבור נוספות).
פתרון נוסחת הנסיגה הוא $T(n) \le 3n \log_2(n) + n$, ובפרט $T(n) = O(n \log n)$, כנדרש.
הטענה עבור $\mathrm{DFT}_n^{-1}$ מוכחת באופן דומה והושארה כתרגיל. $\square$

## 24.2 התמרת פורייה המהירה: כפל מהיר של פולינומים

בתור מסקנה ממשפט 23.3 נציג אלגוריתם מהיר לכפל פולינומים בייצוג המקדמים.

**אלגוריתם 11** אלגוריתם מהיר לכפל פולינומים בייצוג המקדמים

קלט: שני פולינומים $P,Q\in V_{n-1}$ הנתונים בייצוג המקדמים שלהם: $\begin{pmatrix}a_0\\ \vdots\\ a_{n-1}\end{pmatrix}$ ייצוג המקדמים של $P$, $\begin{pmatrix}b_0\\ \vdots\\ b_{n-1}\end{pmatrix}$ ייצוג המקדמים של $Q$. ($n$ לאו דווקא חזקה של 2)

פלט: ייצוג המקדמים $\begin{pmatrix}c_0\\ \vdots\\ c_{2n-2}\end{pmatrix}$ של הפולינום $R=P\cdot Q$.

האלגוריתם:

יהי $m$ חזקה של 2, המספר המינימלי שמקיים $m\geq 2n-1$.

1. נחשב $\mathrm{DFT}_m\begin{pmatrix}a_0\\ \vdots\\ a_{n-1}\\ 0\\ \vdots\\ 0\end{pmatrix}=\begin{pmatrix}p_0\\ \vdots\\ p_{m-1}\end{pmatrix}$. אז $\begin{pmatrix}p_0\\ \vdots\\ p_{m-1}\end{pmatrix}$ הוא וקטור הערכים של $P$ מוערך בשורשי היחידה מסדר $m$.

   נחשב $\mathrm{DFT}_m\begin{pmatrix}b_0\\ \vdots\\ b_{n-1}\\ 0\\ \vdots\\ 0\end{pmatrix}=\begin{pmatrix}q_0\\ \vdots\\ q_{m-1}\end{pmatrix}$.

2. נחשב $\mathrm{DFT}_m^{-1}\begin{pmatrix}p_0q_0\\ \vdots\\ p_{m-1}q_{m-1}\end{pmatrix}=\begin{pmatrix}\tilde{c}_0\\ \vdots\\ \tilde{c}_{m-1}\end{pmatrix}$.

3. נחזיר את $\begin{pmatrix}\tilde{c}_0\\ \vdots\\ \tilde{c}_{2n-2}\end{pmatrix}$.

---

**טענה 24.7.**

(1) אלגוריתם 11 מחזיר את וקטור המקדמים של $R=P\cdot Q$.

(2) זמן הריצה של אלגוריתם 11 הוא $O\left(n\log n\right)$.

הוכחה.

(1) יהי $\omega_m$ שורש היחידה הפרימיטיבי מסדר $m$. יהיו $z_k=\omega_m^k$, $0\leq k\leq m-1$, $m$ שורשי היחידה מסדר $m$.

לפי ההגדרה של $\mathrm{DFT}_m$ מתקיים $\begin{pmatrix}P\left(z_0\right)\\ \vdots\\ P\left(z_{m-1}\right)\end{pmatrix}=\begin{pmatrix}p_0\\ \vdots\\ p_{m-1}\end{pmatrix}$, ובאותו אופן, $\begin{pmatrix}Q\left(z_0\right)\\ \vdots\\ Q\left(z_{m-1}\right)\end{pmatrix}=\begin{pmatrix}q_0\\ \vdots\\ q_{m-1}\end{pmatrix}$.

לכן, $\begin{pmatrix}R\left(z_0\right)\\ \vdots\\ R\left(z_{m-1}\right)\end{pmatrix}=\begin{pmatrix}P\left(z_0\right)Q\left(z_0\right)\\ \vdots\\ P\left(z_{m-1}\right)Q\left(z_{m-1}\right)\end{pmatrix}=\begin{pmatrix}p_0q_0\\ \vdots\\ p_{m-1}q_{m-1}\end{pmatrix}$ הוא בדיוק וקטור הערכים של הפולינום $R$, מוערך בנקודות $z_0,\dots,z_{m-1}$.

נזכור כי כל $2n-1$ ערכים שונים קובעים פולינום ממעלה $2n-2$. מכיוון ש-$m \geq 2n-1$ לפי בחירתו, אנו מקבלים כי הערכים $R(z_0), \ldots, R(z_{m-1})$ קובעים את הפולינום $R$.

לפי הגדרת $\mathrm{DFT}_m$ אנו מקבלים :

$$\begin{pmatrix} p_0q_0 \\ \vdots \\ p_{m-1}q_{m-1} \end{pmatrix} = \mathrm{DFT}_m \begin{pmatrix} c_0 \\ \vdots \\ c_{2n-2} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

כאשר $\begin{pmatrix} c_0 \\ \vdots \\ c_{2n-2} \end{pmatrix}$ הוא וקטור המקדמים של הפולינום $R$. מצד אחד –

$$\mathrm{DFT}_m^{-1} \begin{pmatrix} p_0q_0 \\ \vdots \\ p_{m-1}q_{m-1} \end{pmatrix} = \begin{pmatrix} c_0 \\ \vdots \\ c_{2n-2} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

מצד שני –

$$\mathrm{DFT}_m^{-1} \begin{pmatrix} p_0q_0 \\ \vdots \\ p_{m-1}q_{m-1} \end{pmatrix} = \begin{pmatrix} \tilde{c}_0 \\ \vdots \\ \tilde{c}_{m-1} \end{pmatrix}$$

מכאן נובע ש-$\tilde{c}_{2n-1}, \ldots, \tilde{c}_{m-1}$ הם כולם $0$.

לפיכך, בשלב $3$ באלגוריתם אנו פועלים נכון כאשר אנו מחזירים את הוקטור $\begin{pmatrix} \tilde{c}_0 \\ \vdots \\ \tilde{c}_{2n-2} \end{pmatrix}$, שהרי הוא וקטור המקדמים של $R$.

(2) לפי משפט ה-FFT (משפט 23.3), זמן הריצה של אלגוריתם 11 הוא $O(m \log n)$.

ממינימליות $m$ מתקיים $\frac{m}{2} < 2n-1$, ולכן $m < 4n-2$. סה"כ זמן הריצה הינו : $O(n \log n)$, כנדרש.

□

## 24.3 קריפטוגרפיה ואלגוריתמים על מספרים: שיטת ההצפנה הפומבית של RSA

התרחיש הקלאסי של הצפנה מערב שתי ישויות : אליס (Alice) ובוב (Bob). אליס מעוניינת לשלוח הודעה לבוב באופן חשאי - כלומר, באופן כזה ש"אויב" שרואה את ההודעה המוצפנת לא יוכל לקרוא אותה.

יהי $M$ מרחב ההודעות. אם $m \in M$ היא ההודעה שאליס מעוניינת להעביר לבוב, אז הסכמה הבסיסית נראית כך :

$$\boxed{\text{Alice}} \xrightarrow{m} \boxed{\text{מצפין}} \xrightarrow[\text{פומבי}]{m'} \boxed{\text{מפענח}} \xrightarrow{m} \boxed{\text{Bob}}$$

בכדי להצפין ולפענח אנו משתמשים במפתחות הצפנה ופענוח, המגדירים פונקציית הצפנה $f_E : M \to M$ ופונקציית פענוח $f_D : M \to M$, בהתאמה.

תכונה בסיסית של הפונקציות הללו היא שלכל $m \in M$ מתקיים $f_D(f_E(m)) = m$.

בשנת 1976 החוקים ויטפילד דיפי ומרטין הלמן (Diffie-Hellman) ביצעו פריצת דרך בתחום ההצפנה בהציגם את שיטת ההצפנה הפומבית. בהצפנה פומבית, כל משתמש מחזיק שני מפתחות : מפתח הצפנה פומבי ומפתח פענוח סודי. זאת אומרת, למשתמש $A$ יש פונקציית הצפנה פומבית $f_E : M \to M$ חח"ע ועל ופונקציית פענוח סודית $f_D : M \to M$ חח"ע ועל, כך ש-

$$f_D(f_E(m)) = m = f_E(f_D(m))$$

שימושים אפשריים :

1. מאגר של מפתחות הצפנה פומביים. אם המפתח של משתמש $A$ נמצא במאגר, כל המשתמשים יכולים להעביר ל-$A$ הודעות מוצפנות שרק הוא יכול לפענח.
2. חתימה דיגיטלית. הרעיון הבסיסי הוא כזה: $m$ מסמך נתון, וכדי לחתום על $m$, משתמש $A$ מפרסם את הזוג $(m, f_D(m))$. כל המשתמשים יכולים לוודא את שלמות המסמך ע״י הבדיקה האם $f_E(f_D(m)) = m$, ורק $A$ יכול לחתום על המסמך.

בשנת 1977 רונלד ריבסט, עדי שמיר ולאונרד אדלמן פירסמו לראשונה שיטה המיישמת את הרעיון של דיפי והלמן. נציג את אלגוריתם RSA (ראשי תיבות: Rivest-Shamir-Adleman).
מעתה נניח שאנחנו עובדים עם הודעות באורך 1000 ביטים; פעולה תיחשב יעילה מבחינתנו אם זמן הריצה שלה יהיה פולינומי בגודל הייצוג הבינארי של המספרים שנעבוד איתם.

**אלגוריתם 12** אלגוריתם RSA

יצירת המפתחות:

1. נבחר שני מספרים ראשוניים $p, q$ (בסדר גודל של $2^{500}$), נגדיר $n := p \cdot q$. נגדיר $M = \{0, 1, \ldots, n-1\}$ להיות מרחב ההודעות.
2. יהי $e$ מספר (קטן, $e \sim \text{Polylog}(n)$) כך ש-$e$ זר ל-$(p-1)(q-1)$.
3. יהי $1 \leq d \leq (p-1)(q-1)$ ההופכי של $e$ מודולו $(p-1)(q-1)$, כלומר, $de \equiv 1 \mod ((p-1)(q-1))$.
4. נפרסם את הזוג $(n, e)$ כמפתח הצפנה פומבי; נשמור את $d$ כמפתח פענוח סודי.

פונקציית ההצפנה: $f_E : M \to M$ מוגדרת ע״י $f_E(x) = x^e \mod n$ לכל $x \in M$.
פונקציית הפענוח: $f_D : M \to M$ מוגדרת ע״י $f_D(y) = y^d \mod n$ לכל $y \in M$.

# 25 שבוע 13 - הרצאה - 9.1.19

## 25.1 קריפטוגרפיה ואלגוריתמים על מספרים: שיטת ההצפנה הפומבית של RSA - המשך

נרצה להבין את ההיבטים המתמטיים מאחורי אלגוריתם 12:

- מדוע קיימים מספרים ראשוניים $p$ ו-$q$ כנדרש משלב (1)?
- מדוע קיים מספר $e$ כנדרש בשלב (2)?
- מדוע קיים מספר $d$ כנדרש בשלב (3)?
- מדוע לכל $x \in M$ מתקיים $f_D\left(f_E\left(x\right)\right) = x$?

נרצה להבין גם את ההיבטים האלגוריתמיים:

- מדוע כל הפעולות (מציאת $d, e, p, q$ וכן חישוב פונקציית ההצפנה והפענוח) שמבצע האלגוריתם הן יעילות (פולינומיות ב-$\log_2 n$)?

היבט מעניין נוסף הוא היבט הבטיחות. מדוע שיטת ההצפנה של RSA בטוחה? כלומר, מדוע בהינתן $(n, e)$ קשה למצוא את המפתח הסודי $d$ בצורה יעילה?

## 25.2 קריפטוגרפיה ואלגוריתמים על מספרים: רקע מתורת המספרים - אריתמטיקה מודולרית

**פונקציית המודולו** בהינתן מספר טבעי $b \geq 2$ נגדיר פונקציה $\text{mod } b : \mathbb{Z} \to \{0, 1, \ldots, b-1\}$ המחושבת באופן הבא: בהינתן מספר שלם $a$, נחלק את $a$ ב-$b$ עם שארית ונרשום: $a = qb + r$ כאשר $q$ מספר שלם המכונה "מנה" ו-$r$ מספר שלם המכונה "שארית", ומקיים $0 \leq r < b$. נגדיר $(\text{mod } b)(a) = r$.
צורת רישום מקובלת היא: $a \left(\text{mod } b\right) = r$. בנוסף, נרשום $a \equiv c \mod b$ כדי לרשום בקיצור $a \left(\text{mod } b\right) = c \left(\text{mod } b\right)$.

**תכונות של פונקציית המודולו:**

1. $a \equiv c \mod b$ אס"ם $a - c$ מתחלק ב-$b$.
2. $\left(a + c\right)\left(\text{mod } b\right) = \left(\left(a \left(\text{mod } b\right) + c\left(\text{mod } b\right)\right)\right)\left(\text{mod } b\right)$ או בקיצור: $a + c \equiv a\left(\text{mod } b\right) + c\left(\text{mod } b\right) \mod b$.
3. $\left(a \cdot c\right)\left(\text{mod } b\right) = \left(\left(a \left(\text{mod } b\right) \cdot c\left(\text{mod } b\right)\right)\right)\left(\text{mod } b\right)$ או בקיצור: $a \cdot c \equiv a\left(\text{mod } b\right) \cdot c\left(\text{mod } b\right) \mod b$.

**דוגמה 25.1.** נרצה לחשב את שארית החלוקה של $7^{12}$ ב-5. מהפעלה חוזרת של תכונה 3 אנו מקבלים:

$$7^{12} \equiv 2^{12} \equiv \left(2^3\right)^4 \equiv 3^4 \equiv 9^2 \equiv 4^2 \equiv 1 \mod 5$$

**מחלק משותף מקסימלי** של שני מספרים טבעיים $a, b$ הוא המספר הטבעי הגדול ביותר המחלק את שניהם. מספר זה מסומן ב-$\gcd\left(a, b\right)$.
כך למשל, $\gcd\left(72, 16\right) = 8$.
בתרגיל נראה כי ניתן למצוא את $\gcd\left(a, b\right)$ בזמן פולינומי בגודל הייצוג של $a$ ו-$b$ ע"י אלגוריתם אוקלידס.

**הגדרה 25.2** (מספרים זרים). מספרים שלמים $a, b$ ייקראו זרים אס"ם $\gcd\left(a, b\right) = 1$.

**פירוק מספר טבעי לגורמים ראשוניים** לכל מספר טבעי $a$ קיים פירוק יחיד לגורמים ראשוניים $a = p_1^{k_1} \cdots p_t^{k_t}$.

**דוגמה 25.3.** $72 = 2^3 \cdot 3^2$.

**טענה 25.4.** שני מספרים $a, b$ הם זרים אס"ם בפירוק של $a$ ושל $b$ לגורמים ראשוניים אין גורם משותף.

**טענה 25.5.** $a$ הוא כפולה שלמה של $b$ אס"ם כל ראשוני המופיע בפירוק של $b$ מופיע גם בפירוק של $a$ לפחות באותה חזקה כמו ב-$b$.

**דוגמה 25.6.** 72 הוא כפולה של 18 כי $72 = 2^3 \cdot 3^2$ ו-$18 = 2 \cdot 3^2$.

**טענה 25.7.** אם $a \cdot c$ כפולה שלמה של $b$, ו-$c$ זר ל-$b$, אז $a$ הוא כפולה שלמה של $b$.

**שאלה אלגוריתמית:** בהינתן מספר טבעי $a$, האם ניתן לפרק את $a$ לגורמים ראשוניים באופן יעיל?
**תשובה:** לא ידוע. האלגוריתם היעיל ביותר הידוע כיום רץ בזמן $\exp\left(\left(5t \log^2\left(t\right)\right)^{\frac{1}{3}}\right)$ כאשר $t = \log_2 a$.

**עובדה 25.8** (התפלגות המספרים הראשוניים). יהי $a$ מספר טבעי, ונסמן ב-$\pi\left(a\right)$ את עוצמת הקבוצה $\{2 \leq x \leq a : x \text{ ראשוני}\}$; כלומר, $\pi\left(a\right)$ הוא כמות המספרים הראשוניים שקטנים-שווים מ-$a$. אזי $\pi\left(a\right) \approx \frac{a}{\ln(a)}$, וביתר דיוק: $\lim_{a\to\infty} \frac{\pi(a)}{a/\ln(a)} = 1$.

## 25.3 קריפטוגרפיה ואלגוריתמים על מספרים: ניתוח שיטת ההצפנה הפומבית של RSA

**ניתוח שלב (1)** בשלב (1) של אלגוריתם 12 אנו בוחרים שני מספרים ראשוניים $p, q$ בסדר גודל של $2^{500}$.
בכדי למצוא מספר ראשוני $p$ כנ״ל, נגריל באופן אחיד מספר טבעי בן $500$ ביטים, ונבדוק האם הוא ראשוני.
עובדה 25.8 מבטיחה שנגריל מספר ראשוני בסיכוי לפחות $\frac{1}{500}$. אם נחזור על הניסוי מספיק פעמים, נצליח.

**הערה 25.9.** כיצד נבדוק האם מספר טבעי הוא מספר ראשוני ביעילות? אלגוריתם מילר-רבין שנציג בהמשך.

**הערה 25.10.** כיצד נגריל באופן יעיל מספר טבעי בן 500 ביטים? נבצע הטלת מטבע עבור כל ביט בנפרד.
כיצד נגריל מספר טבעי בתחום $1$ עד $n$ כאשר $n$ לאו דווקא חזקה של $2$? נגריל מספר טבעי בתחום $1$ עד החזקה של $2$ הכי קרובה שגדולה מ-$n$, ונחזור על הניסוי אם הגרלנו מספר שאיננו בתחום $[1, n]$.

**ניתוח שלב (2)** בשלב (2) של האלגוריתם אנו מחפשים מספר $e$ זר ל-$(p-1)(q-1)$. כדי למצוא מספר $e$ כנ״ל, נגריל מספר טבעי בין $1$ ל-$(p-1)(q-1)$. בסיכוי גבוה $e$ יהיה ראשוני, ולכן בסיכוי גבוה הוא יהיה זר ל-$(p-1)(q-1)$.

**הערה 25.11.** כיצד נבדוק אם המספר $e$ שהוגרל אכן זר ל-$(p-1)(q-1)$? נחשב את המחלק המשותף המקסימלי של $e$ ו-$(p-1)(q-1)$.
המספרים זרים אס״ם $\gcd(e, (p-1)(q-1)) = 1$. אלגוריתם אוקלידס מאפשר חישוב יעיל של המחלק המשותף המקסימלי של שני מספרים נתונים.

# 26 שבוע 14 - הרצאה - 13.1.19

## 26.1 קריפטוגרפיה ואלגוריתמים על מספרים: רקע מתורת המספרים

**טענה 26.1.** יהי $a$ מספר שלם זר ל-$b$, ויהיו $1 \leq c, d \leq b-1$. אם $ac \equiv ad \mod b$ אז $c = d$.

הוכחה. לפי התכונות של פונקציית המודולו 25.2 אנו יודעים כי $ac \equiv ad \mod b$ אס"ם $ac - ad = a\left(c-d\right)$ מתחלק ב-$b$.
לפי הנתון $a$ זר ל-$b$, לכן מטענה 25.7 נובע כי $c - d$ מתחלק ב-$b$.
מהנתון $1 \leq c, d \leq b-1$ ולכן $-b < c-d < b$. המספר השלם היחיד בתחום $\left(-b, b\right)$ שמתחלק ב-$b$ הוא $0$, לכן $c - d = 0$, כלומר, $c = d$, כנדרש. $\square$

**הגדרה 26.2.** יהי $m \in \mathbb{N}$. נגדיר את $\mathbb{Z}_m^*$ חבורה כפלית מודולו $m$ ע"י

$$\mathbb{Z}_m^* = \left\{1 \leq a \leq m : \gcd\left(a, m\right) = 1\right\}$$

**הגדרה 26.3.** נסמן ב-$\varphi\left(m\right)$ את גודל קבוצת המספרים בין $1$ ל-$m$ שזרים ל-$m$, כלומר, $\varphi\left(m\right) := \left|\mathbb{Z}_m^*\right|$. $\varphi$ נקראת "פונקציית אוילר".

**דוגמה 26.4.** $\varphi\left(3\right) = 2$ כי $1$ ו-$2$ זרים ל-$3$. $\varphi\left(17\right) = 16$ כי כל המספרים מ-$1$ עד $16$ זרים ל-$17$ שהרי $17$ מספר ראשוני.

**עובדה 26.5.** אם $p$ מספר ראשוני אז $\varphi\left(p\right) = p - 1$. אכן, המחלקים היחידים של מספר ראשוני $p$ הם $1$ ו-$p$, ולכן כל המספרים $1, 2, \ldots, p-1$ זרים ל-$p$.

**למה 26.6.** יהיו $p \neq q$ שני מספרים ראשוניים. אזי $\varphi\left(p \cdot q\right) = \left(p-1\right)\left(q-1\right)$.

הוכחה. נספור את גודל קבוצת המספרים בין $1$ ל-$p \cdot q$ <u>שאינם</u> זרים ל-$p \cdot q$.
מספר $a$ בין $1$ ל-$p \cdot q$ שאינו זר ל-$p \cdot q$ בהכרח מתחלק ב-$p$ או ב-$q$ (או בשניהם). לפיכך, קבוצת המספרים בין $1$ ל-$p \cdot q$ שאינם זרים ל-$p \cdot q$ היא:

$$\left\{q, 2q, \ldots, \left(p-1\right)q\right\} \sqcup \left\{p \cdot q\right\} \sqcup \left\{p, 2p, \ldots, \left(q-1\right)p\right\}$$

(כאשר הסמל $\sqcup$ מציין איחוד זר.) בעקבות זאת, גודל קבוצת המספרים בין $1$ ל-$p \cdot q$ שאינם זרים ל-$p \cdot q$ הוא $\left(p-1\right) + 1 + \left(q-1\right) = p + q - 1$.
הגודל $\varphi\left(p \cdot q\right)$ מציין את גודל קבוצת המספרים בין $1$ ל-$p \cdot q$ שכן זרים ל-$p \cdot q$, ולכן:

$$\varphi\left(p \cdot q\right) = p \cdot q - \left(p + q - 1\right) = \left(p-1\right)\left(q-1\right)$$

כנדרש. $\square$

**עובדה 26.7.**

**(1)** אם $a, c$ זרים ל-$b$ אז $a \cdot c$ זר ל-$b$.

**(2)** אם $a$ זר ל-$b$ אז גם $a \pmod{b}$ זר ל-$b$.

**טענה 26.8.** יהי $m$ מספר טבעי ויהי $e$ מספר טבעי שזר ל-$m$. אזי קיים $1 \leq d \leq m$ כך ש-$de \equiv 1 \mod m$.

הוכחה. נגדיר פונקציה $f : \mathbb{Z}_m^* \to \left\{0, 1, \ldots, m-1\right\}$ באופן הבא:

$$\forall a \in \mathbb{Z}_m^*, \quad f\left(a\right) = \left(e \cdot a\right) \pmod{m}$$

לאור עובדה 26.7, אנו מבחינים כי $f\left(a\right) \in \mathbb{Z}_m^*$ לכל $a \in \mathbb{Z}_m^*$. אי לכך, $f$ היא למעשה פונקציה מ-$\mathbb{Z}_m^*$ לעצמו, כלומר, $f : \mathbb{Z}_m^* \to \mathbb{Z}_m^*$.
נטען כי $f$ היא חח"ע. ואכן אם $f\left(a\right) = f\left(b\right)$ אז $ea \equiv eb \mod m$. $e$ זר ל-$m$ לפי הנתון, והמספרים $a$ ו-$b$ מקיימים שהם בתחום $\left[1, m-1\right]$. לפי טענה 26.1 אנו מקבלים כי $a = b$.
$f$ היא פונקציה חח"ע מ-$\mathbb{Z}_m^*$ לעצמו, ולכן $f$ היא פונקציה על, ובמילים אחרות, מגדירה תמורה על $\mathbb{Z}_m^*$.
לאור זאת, קיים $d \in \mathbb{Z}_m^*$ כך ש- $f\left(d\right) = 1$ (כי $1 \in \mathbb{Z}_m^*$). זאת אומרת, קיים $1 \leq d \leq m$ כך ש-$ed \equiv 1 \mod m$, כנדרש. $\square$

**דוגמה 26.9.** אם $m = 12$ ו-$e = 5$ אז $d = 5$ שהרי $5 \cdot 5 \equiv 25 \equiv 1 \mod 12$.

**טענה 26.10** (משפט אוילר). יהי $m$ מספר טבעי ויהי $a$ זר ל-$m$. אזי $a^{\varphi(m)} \equiv 1 \mod m$.

**דוגמה 26.11.** $m=9$, $a=5$. אז $\varphi(m)=6$ מבדיקה פשוטה. בנוסף,

$$5^6 \equiv 25^3 \equiv 7^3 \equiv 49\cdot 7 \equiv 4\cdot 7 \equiv 28 \equiv 1 \mod 9$$

בהלימה עם טענה 26.10.

הוכחה. נגדיר פונקציה $f:\mathbb{Z}_m^* \to \mathbb{Z}_m^*$ באופן הבא :

$$\forall b\in\mathbb{Z}_m^*, \quad f(b)=(a\cdot b)\,(\mathrm{mod}\ m)$$

מתקיים כי $f(b)\in\mathbb{Z}_m^*$ לאור עובדה 26.7. בנוסף, בהוכחת טענה 26.8 ראינו כי $f$ היא חח״ע ועל (ולמעשה תמורה על $\mathbb{Z}_m^*$).
לפיכך מתקיים,

$$\prod_{b\in\mathbb{Z}_m^*} b = \prod_{b\in\mathbb{Z}_m^*} f(b) = \prod_{b\in\mathbb{Z}_m^*} ((a\cdot b)\,(\mathrm{mod}\ m))$$

לקיחת מודולו $m$ משני האגפים בקצוות נותנת

$$\prod_{b\in\mathbb{Z}_m^*} b \equiv \prod_{b\in\mathbb{Z}_m^*} ((a\cdot b)\,(\mathrm{mod}\ m)) \mod m$$

לפי התכונות של פונקציית המודולו 25.2 מתקיים

$$\prod_{b\in\mathbb{Z}_m^*} ((a\cdot b)\,(\mathrm{mod}\ m)) \equiv \prod_{b\in\mathbb{Z}_m^*} (a\cdot b) \equiv a^{\varphi(m)} \prod_{b\in\mathbb{Z}_m^*} b \mod m$$

כלומר, קיבלנו כי

$$\prod_{b\in\mathbb{Z}_m^*} b \equiv a^{\varphi(m)} \prod_{b\in\mathbb{Z}_m^*} b \mod m$$

נסמן לשם פשטות $A:=a^{\varphi(m)}\,(\mathrm{mod}\ m)$, $B:=\left(\prod_{b\in\mathbb{Z}_m^*} b\right)(\mathrm{mod}\ m)$. אז קיבלנו,

$$B \equiv AB \mod m$$

מעובדה 26.7 אנו מקבלים כי $B$ זר ל-$m$. בנוסף, $1\le A\le m-1$, ולכן מטענה 26.1 אנו מקבלים $A=1$.
זאת אומרת, $a^{\varphi(m)}\equiv 1 \mod m$, כנדרש. $\square$

**מסקנה 26.12** (משפט פרמה הקטן)**.** יהי $p$ מספר ראשוני ו-$a$ זר ל-$p$. אזי $a^{p-1}\equiv 1 \mod p$.

הוכחה. נובע ממשפט אוילר (טענה 26.10) ומכך ש-$\varphi(p)=p-1$ עבור $p$ ראשוני (עובדה 26.5). $\square$

**טענה 26.13.** יהי $m$ מספר טבעי, ותהי $B=\left\{a\in\mathbb{Z}_m^* : a^{m-1}\not\equiv 1 \mod m\right\}$ (B היא קבוצת העדים לכך שמשפט פרמה הקטן לא מתקיים עבור $m$ ולכן $m$ אינו ראשוני).
אם $B\neq\emptyset$ אזי $|B|\ge\frac{|\mathbb{Z}_m^*|}{2}$.

הוכחה. יהי $m$ מספר טבעי כך ש-$B\neq\emptyset$. יהי $b\in B$ כלשהו, ונגדיר $f:\mathbb{Z}_m^*\to\mathbb{Z}_m^*$ באופן הבא :

$$\forall a\in\mathbb{Z}_m^*, \quad f(a)=(b\cdot a)\,(\mathrm{mod}\ m)$$

ראינו בהוכחת טענה 26.8 ש-$f$ המוגדרת כך היא חח״ע ועל.
נראה כי הצמצום של $f$ על $\mathbb{Z}_m^*\setminus B$ היא פונקציה מ-$\mathbb{Z}_m^*\setminus B$ ל-$B$. יהי $a\in\mathbb{Z}_m^*\setminus B$, אזי על פי הגדרת $B$ מתקיים :

$$a^{m-1}\equiv 1 \mod m$$

מכאן נובע :

$$(f(a))^{m-1} \equiv (b\cdot a)^{m-1} \equiv b^{m-1}\cdot a^{m-1} \underset{a\in\mathbb{Z}_m^*\setminus B}{\equiv} b^{m-1} \underset{b\in B}{\not\equiv} 1 \mod m$$

לכן $f(a)\in B$. מכך ש-$f$ חח״ע אנו מקבלים

$$|\mathbb{Z}_m^*\setminus B| \le |B|$$

לפיכך :

$$|B| \ge \frac{|\mathbb{Z}_m^*\setminus B|+|B|}{2} = \frac{|\mathbb{Z}_m^*|}{2}$$

כנדרש. $\square$

## 26.2 קריפטוגרפיה ואלגוריתמים על מספרים: ניתוח אלגוריתם RSA

**טענה 26.14.** בסימוני אלגוריתם 12. יהי $M=\{0,1,\ldots,n-1\}$ מרחב ההודעות, ויהי $x\in M$. אזי $f_D\left(f_E\left(x\right)\right)=x$.

הוכחה. נזכור כי $f_E\left(x\right)=x^e \mod n$ ו- $f_D\left(y\right)=y^d \mod n$, כאשר $n=p\cdot q$ ו-$p\neq q$ מספרים ראשוניים.
לכן: $f_D\left(f_E\left(x\right)\right)=f_D\left(x^e \mod n\right)=x^{ed} \mod n$ (ההצדקה למעבר האחרון מצויה ב-25.2).
לפיכך, עלינו לוודא כי $x^{ed}\equiv x \mod n$, ובאופן שקול, עלינו לוודא כי $x^{ed}-x$ מתחלק ב-$n$.
לפי הבנייה של המפתחות באלגוריתם RSA אנו יודעים כי $de\equiv 1 \mod \left(p-1\right)\left(q-1\right)$ כאשר $p\neq q$ מספרים ראשוניים.
מכאן נסיק שקיים מספר טבעי $a\in\mathbb{N}$ כך ש-$de=a\left(p-1\right)\left(q-1\right)+1$. על כן:

$$x^{de}=x^{a(p-1)(q-1)+1}$$

ז״א, עלינו לוודא כי $x^{a(p-1)(q-1)+1}-x$ מתחלק ב-$n$.
נבדיל בין שלושה מקרים:

- $x$ זר ל-$n$. לפי משפט אוילר (טענה 26.10) מתקיים:

$$x^{(p-1)(q-1)}\equiv x^{\varphi(n)}\equiv 1 \mod n$$

בחישוב מודולו $n$ אנו מקבלים:

$$\begin{aligned} x^{a(p-1)(q-1)+1}-x &\equiv x\left(x^{a(p-1)(q-1)}-1\right)\\ &\equiv x\left(\left(x^{(p-1)(q-1)}\right)^a-1\right)\\ &\equiv x\left(1^a-1\right)\\ &\equiv 0 \mod n \end{aligned}$$

(המשך ההוכחה בהרצאה הבאה.) □

# חלק II
# תרגולים

## 27 תרגול 1 - 15.10.18

### 27.1 נוטציות אסימפטוטיות

**הגדרה 27.1.** תהיינה $f, g : \mathbb{N} \to \mathbb{N}$ פונקציות.

- נאמר כי $g$ היא **חסם תחתון** של $f$ ונסמן $f(n) = \Omega(g(n))$ אם קיימים $n_0 \in \mathbb{N}$ ו-$c > 0$ כך שלכל $n > n_0$ מתקיים $f(n) \geq c \cdot g(n)$;
- נאמר כי $g$ היא **חסם עליון** של $f$ ונסמן $f(n) = O(g(n))$ אם קיימים $n_0 \in \mathbb{N}$ ו-$c > 0$ כך שלכל $n > n_0$ מתקיים $f(n) \leq c \cdot g(n)$;
- אם $g$ היא גם חסם תחתון של $f$ וגם חסם עליון של $f$ אז נאמר כי $g$ היא **חסם הדוק** של $f$ ונסמן $f(n) = \Theta(g(n))$.

**דוגמאות**

- $500\sqrt{n} = O(n)$
- $10n\log(\log n) = \Omega(n)$
- $n\log n + 20n^2 = \Theta(n^2)$

### 27.2 הוכחות באינדוקציה

כדי להוכיח שטענה מסוימת מתקיימת לכל $n_0 \leq n \in \mathbb{N}$ נראה כי:
(בסיס) הטענה מתקיימת עבור $n_0$;
(צעד) בהנחה שהטענה מתקיימת עבור $n \geq n_0$ נוכיח כי מתקיימת עבור $n+1$.

**טעויות נפוצות**

1. לא מראים שהטענה מתקיימת עבור מקרה הבסיס;
2. שימוש לא נכון בהנחת האינדוקציה;
3. הוכחת מעבר מ-$n$ ל-$n+1$ לא נכונה לוגית/מתמטית;
4. הוכחת המעבר מ-$n$ ל-$n+1$ לא תקפה למקרה הבסיס ($n = n_0$).

**דוגמה להוכחה שגויה**

טענה. לכל $n \in \mathbb{N}$, בהינתן קבוצה $H$ בגודל $n$ סוסים, כל הסוסים בה באותו הצבע.

**"הוכחה".**

**בסיס** עבור קבוצה בגודל $n_0 = 1$ הטענה מתקיימת כי כל הסוסים בה באותו צבע.

**צעד** נניח את נכונות הטענה עבור $n$ ונוכיח עבור $n+1$. תהי $H = \{h_1, h_2, \ldots, h_{n+1}\}$ קבוצה של $n+1$ סוסים.

נתבונן ב-$H_1 = \{h_1, \ldots, h_n\}$. מהנחת האינדוקציה, כל הסוסים בה באותו צבע.

נתבונן ב-$H_2 = \{h_2, \ldots, h_{n+1}\}$. מהנחת האינדוקציה, כל הסוסים בה באותו צבע.

$\Leftarrow$ יש חפיפה בין הקבוצות ולכן כל הסוסים ב-$H = \{h_1, \ldots, h_{n+1}\}$ באותו הצבע, כנדרש. $\square$

<u>הטעות:</u> צעד האינדוקציה לא תקף עבור מקרה הבסיס. בפרט, עבור המקרה בו $n = 2$ (שני סוסים), אין חפיפה בין הקבוצות $H_1, H_2$ שכל אחת מהן מכילה סוס אחד, ולא ניתן להסיק ששני הסוסים חולקים את אותו הצבע.

## 27.3 ניתוח זמן ריצה

אנו מחשבים זמן ריצה ע"י ספירה של פעולות בסיסיות, למשל: השוואה, פעולה אריתמטית, השמה.
ברוב האלגוריתמים שראינו עד כה, הנחנו שהפעולות הבסיסיות הללו לוקחות $O(1)$.
לא תמיד ניתן להניח שהפעולות הבסיסיות לוקחות $O(1)$, לדוגמה, באלגוריתמים בהם לב האלגוריתם הוא ביצוע הפעולה הבסיסית (בדרך כלל הקלט אליהם יהיה מספר קבוע של מספרים).

**דוגמה.** קלט: שני מספרים טבעיים $a, b$ ($b \geq a$). פלט: המכפלה $a \cdot b$.
לכאורה היינו עלולים לחשוב שגודל הקלט הוא $O(1)$, ואז לקבוע שזמן הריצה של האלגוריתם הוא $O(1)$.
נזכור שבניתוח זמן ריצה של אלגוריתם אנו מתעניינים בזמן הריצה של האלגוריתם על-פני שינויים בגודל הקלט.
במקרה הזה ניאלץ לרדת לרזולוציה "נמוכה" יותר ולהתבונן בייצוג הבינארי של $a$ ו-$b$.
מספר הביטים בקלט הנו $\mathrm{O}(\log(a) + \log(b))$ והפלט חסום ע"י $O(\log^2(b))$.

## 27.4 גרפים

- גרף לא מכוון הוא זוג סדור $G = (V, E)$ כאשר $V$ היא קבוצת הקודקודים ו-$E$ קבוצת הצלעות. בגרף לא מכוון מתקיים

$$(u, v) \in E \iff (v, u) \in E$$

- גרף ממושקל הוא גרף עם פונקציית משקל $w : E \to \mathbb{R}$ המתאימה לכל צלע מספר ממשי.
- משקל של גרף מוגדר באופן הבא: $w(G) = \sum_{e \in E} w(e)$.
- עץ הוא גרף קשיר ללא מעגלים.
- עץ פורש בגרף $G$ הוא עץ שמכיל את כל קודקודי $G$.

## 27.5 יעילות

**הגדרה 27.2.** נאמר שאלגוריתם הוא **יעיל** אם זמן הריצה שדורש על קלט באורך $n$ הוא פולינומיאלי ב-$n$.

**הגדרה 27.3.** פונקציה $f : \mathbb{N} \to \mathbb{N}$ תיקרא **פולינומיאלית ב-**$n$ אם קיים קבוע $k \in \mathbb{N}$ כך ש-$f(n) = O(n^k)$.[3]

| שם האלגוריתם | תיאור הקלט | תיאור הפלט | גודל הקלט | זמן ריצה | האם פולינומיאלי? |
|---|---|---|---|---|---|
| Merge Sort | רשימה של $n$ מספרים | רשימה ממויינת | $n$ | $\Theta(n \log n)$ | כן ($k = 2$) |
| חיפוש בינארי | רשימה של $n$ מספרים ממויינים, מספר $a$ | מיקום $a$ אם נמצא, $-1$ אחרת | $\Theta(n)$ | $O(\log n)$ | כן ($k = 1$) |
| Kruskal | גרף קשיר לא מכוון $G = (V, E)$, פונקציית משקל $w : E \to \mathbb{R}$ | תת-גרף $T \subseteq G$ שהוא עץ פורש עם משקל מינימלי | $\Theta(\lvert V\rvert + \lvert E\rvert)$ | $O(\lvert E\rvert \log \lvert E\rvert)$ | כן ($k = 2$) |

טבלה 1: יעילות אלגוריתמים מוכרים

ניתן דוגמה לאלגוריתם לא יעיל.

[3] שימו לב לשימוש ב-$n$ כגודל הקלט. $f$ תיקרא פולינומיאלית ב-$\log(n)$ אם קיים קבוע $k \in \mathbb{N}$ כך ש-$f(n) = O(\log^k(n))$.

**דוגמה 27.4** (פירוק מספר לגורמים ראשוניים)**.**
קלט: מספר טבעי $n$ (הפירוק לגורמים שלו הוא $n = p_1^{m_1} \cdots p_k^{m_k}$)؛
פלט: רשימה $L = \{p_1, \ldots, p_k\}$ של כל הגורמים הראשוניים המחלקים את $n$.
נציג אלגוריתם ידוע לפתרון הבעיה.

---

**אלגוריתם 13** אלגוריתם לפירוק מספר לגורמים ראשוניים

---

1. עבור כל מספר שלם $i$ בין $2$ ל-$\sqrt{n}$, נבדוק האם $n$ מתחלק ב-$i$ ללא שארית.
2. אם כן, $i$ נכנס לרשימת המחלקים, וממשיכים לחלק את המנה ב-$i$, כל עוד המנה המתקבלת שלמה. הפעולה נמשכת עם המועמד הבא והמנה.
3. אם המנה בסוף התהליך היא מספר כלשהו $\ell \neq 1$ אז נוסיף גם את $\ell$ לרשימת הגורמים.

---

**דוגמה לריצה אפשרית:** $n = 156$, $\lfloor\sqrt{n}\rfloor = 12$.

$$\begin{aligned}
156/2 &= 78 \\
78/2 &= 39 \\
39/2 &= \text{לא מתחלק, נעבור הלאה} \\
39/3 &= 13 \\
13/3 &= \text{לא מתחלק} \\
13/4 &= \ldots \\
&\vdots \\
13/12 &= \text{לא מתחלק, נעצור}
\end{aligned}$$

נשארנו עם מנה $\ell = 13$, אז נוסיף גם אותה לרשימת הגורמים הראשוניים של $156$ ונקבל שרשימת הגורמים היא $\{2, 3, 13\}$.

**עובדה.** כל הגורמים קטנים שווים מ-$\sqrt{n}$ ויש גורם אחד בלבד גדול מ-$\sqrt{n}$.

**גודל הקלט** $\log(n)$ - מספר הביטים בייצוג הבינארי של $n$.

**זמן ריצה** עבור כל המספרים בין $2$ ל-$\sqrt{n}$, אנו מבצעים לפחות פעולת חילוק אחת. כמו-כן אנו מבצעים פעולת השוואה בסוף האלגוריתם.
נסמן ב-$T(n)$ את זמן הריצה של האלגוריתם. נחסום את $T(n)$ מלרע:

$$\begin{aligned}
T(n) &\geq \left(\sqrt{n} - 1\right) \cdot (\text{זמן ריצה של פעולת חילוק}) + 1 \cdot (\text{זמן ריצה של פעולת השוואה}) \\
&\geq \left(\sqrt{n} - 1\right) + 1 \\
&= \sqrt{n} = n^{\frac{1}{2}} = 2^{\log\left(n^{\frac{1}{2}}\right)} \\
&= \sqrt{2}^{\log(n)}
\end{aligned}$$

כלומר, זמן הריצה של האלגוריתם הוא לפחות אקספוננציאלי בגודל הקלט, ובפרט לא פולינומיאלי בו.

# 28 תרגול 2 - 22.10.18

בתרגול זה נעסוק בפתרון בעיות אופטימיזציה באמצעות אלגוריתמים חמדניים. לא כל בעיית אופטימיזציה ניתנת לפתרון באמצעות אלגוריתם חמדן, ובמהלך הקורס נציג גישות נוספות לפתרון בעיות אופטימיזציה.

**"הגדרה"** (אלגוריתם חמדן)**.** אלגוריתם חמדן הוא אלגוריתם איטרטיבי שבכל שלב מבצע את הפעולה המשתלמת ביותר באותו הרגע, מבלי לקחת בחשבון השלכות ארוכות טווח.

## 28.1 בעיית החזרת העודף

**קלט:** $k \in \mathbb{N}$ - עודף שיש להחזיר, $C = (c_1, \ldots, c_n)$ סט מטבעות קיים כך ש-$c_1 = 1$.

**פלט:** פריטה של $k$ למספר מינימלי של מטבעות.

**אלגוריתם חמדן:**

1. נבחר את המטבע הגדול ביותר שלא עולה על $k$;
2. נחזור על הפעולה הקודמת עם השארית של העודף;
3. נעצור כאשר לא יישאר עודף.

הנקודה המעניינת בבעיית החזרת העודף היא שהאלגוריתם החמדן מחזיר תוצאה אופטימלית עבור סטי מטבעות מסוימים, ומחזיר תוצאה שאינה אופטימלית עבור סטי מטבעות אחרים. להלן דוגמאות:

- נתבונן בסט המטבעות $C = (1, 2, 5, 10)$ ו-$k = 23$.

  ניווכח כי הפתרון שהאלגוריתם החמדן יחזיר הוא $(10, 10, 2, 1)$, אכן אופטימלי.

  במקרה זה גם ניתן להוכיח שהאלגוריתם החמדן תמיד יחזיר פתרון אופטימלי.

- נתבונן בסט המטבעות $C = (1, 3, 4, 10)$ ו-$k = 6$.

  הפתרון שהאלגוריתם החמדן יחזיר הוא $(4, 1, 1)$, בעוד שהפתרון האופטימלי הנו $(3, 3)$.

## 28.2 בעיית תא הדלק הקטן

**קלט:**

(1) $N$ - מס' הק"מ שניתן לנסוע עם מיכל מלא של דלק;

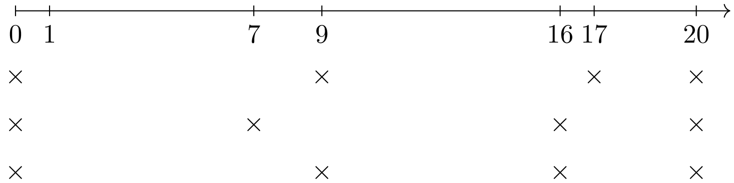
(2) $a_1, a_2, \ldots, a_n$ - מיקומי תחנות הדלק לתדלוק, כאשר $a_1 = 0$;

(3) הנחת תקינות: לכל $1 < i \leq n$ מתקיים $a_i - a_{i-1} \leq N$.

**פלט:** תת-קבוצה $\{b_1, \ldots, b_m\} \subseteq \{a_1, \ldots, a_n\}$ כך שמתקיים:

- (חוקיות) (1) $b_1 = a_1$ ו-$b_m = a_n$; (2) לכל $1 < i \leq m$ מתקיים $b_i - b_{i-1} \leq N$.
- (אופטימליות) $m$ מינימלי בתנאי זה.

**דוגמה:** $(a_1, \ldots, a_7) = (0, 1, 7, 9, 16, 17, 20)$, $N = 10$.
פתרון אופטימלי אפשרי: $(b_1, \ldots, b_4) = (0, 9, 17, 20)$.
לבעיה זו יש מספר פתרון אופטימליים; נציג המחשה לפתרונות אופטימליים נוספים:



**האלגוריתם:**

1. נאתחל $B = (a_1)$ ($B$ הוא הפתרון שיוחזר)
2. נעבור על התחנות לפי הסדר:
   נוסיף את התחנה ה-$i$ ל-$B$ כאשר אחד מהמקרים הבאים מתרחש:
   (א) $i = n$
   (ב) אין לנו מספיק דלק להגיע ל-$a_{i+1}$ מהתחנה האחרונה ב-$B$
3. נחזיר את $B$.

**הוכחת נכונות:**

<u>(חוקיות)</u>
נראה שהפתרון שלנו עומד בתנאי החוקיות 1 ו-2 בבעיה.
תנאי 1 מתקיים: אתחלנו את $B$ עם $a_1$, ווידאנו שמכניסים לבסוף את $a_n$, ולכן התנאי מתקיים.
תנאי 2 מתקיים: בכל שלב באיטרציה, אנו מדלגים על תחנה רק אם יש לנו מספיק דלק להגיע לתחנה הבאה. אחרת, אנחנו מוסיפים את התחנה (בהכרח ניתן להוסיף תחנה, מהנחת תקינות הקלט).
<u>(אופטימליות)</u>
נסמן את הפתרון של האלגוריתם החמדן ב-$B = \{b_1, \dots, b_m\}$.

**למה 28.1** (טענת האינדוקציה של תא הדלק הקטן)**.** לכל $1 \le k \le m$ יש פתרון אופטימלי מהצורה $C = (b_1, \dots, b_k, c_{k+1}, \dots, c_{m'})$ כאשר $c_{k+1}, \dots, c_{m'}$ תחנות כלשהן.

**הערה 28.2.** ברמה האינטואיטיבית, למה 28.1 אומרת כי לכל תחילית של האלגוריתם החמדן, יש איזשהו פתרון אופטימלי לבעיה שמכיל את התחילית הזו.

**הערה 28.3.** למה 28.1 <u>לא</u> טוענת כי:

- כל פתרון שמתחיל בתחילית של האלגוריתם החמדן הוא אופטימלי.
- כל פתרון אופטימלי מתחיל בתחילית של האלגוריתם החמדן.
- לכל תחילית יש פתרון אופטימלית (רק לתחילית של האלגוריתם החמדן).

הוכחה (של למה 28.1). באינדוקציה על $k$.
<u>בסיס:</u> עבור $k = 1$, כל המסלולים החוקיים מתחילים ב-$a_1$, ובפרט, כל המסלולים האופטימליים מתחילים ב-$a_1$. מכיוון ש-$b_1 = a_1$, הטענה נובעת.
<u>צעד:</u> נניח עבור $k-1$ שקיים פתרון אופטימלי מהצורה $C = (b_1, \dots, b_{k-1}, c_k, \dots, c_{m'})$ ונראה קיום של פתרון כזה עבור $k$.
אם $k < m$, נגדיר $C' = (b_1, \dots, b_{k-1}, b_k, c_{k+1}, \dots, c_{m'})$.
נרצה להוכיח ש-$C'$ פתרון אופטימלי ובכך נסיים את צעד האינדוקציה.
מכיוון ש-$|C| = |C'|$ ו-$C$ אופטימלי, נותר להוכיח ש-$C'$ הוא פתרון חוקי.
תנאי חוקיות 1 מתקיים כי $b_1 = a_1$ ו-$c_{m'} = a_n$ שהרי לפי הנחת האינדוקציה, $C$ פתרון חוקי.
נראה שתנאי חוקיות 2 מתקיים.
לכל $1 < j \le k$ מתקיים $b_j - b_{j-1} \le N$ מחוקיות הפתרון של האלגוריתם החמדן.
לכל $k < j \le m$ מתקיים $c_j - c_{j-1} \le N$ שהרי לפי הנחת האינדוקציה $C$ פתרון חוקי.
נותר להראות $c_{k+1} - b_k \le N$.
מאופן בחירת $b_k$, אנו יודעים כי $c_k \le b_k$, זאת משום שהאלגוריתם החמדן בוחר את התחנה הרחוקה ביותר מ-$b_{k-1}$ שמרחקה קטן מ-$N$ (זוהי "הבחירה החמדנית"). מכאן נקבל:

$$c_{k+1} - b_k \le c_{k+1} - c_k \le N$$

כאשר אי השוויון האחרון נובע מכך שלפי הנחת האינדוקציה, $C$ פתרון חוקי.
אם $k = m$ אז לפי הנחת האינדוקציה קיים פתרון אופטימלי מהצורה $C = (b_1, \dots, b_{m-1}, c_m, \dots, c_{m'})$. מחוקיות $C$ מתקיים $b_k = b_m = a_n \in C$. בנוסף, לא ייתכן ש-$c_m \neq a_n$ כי $C$ אופטימלי.
בכך סיימנו את צעד האינדוקציה. □

**מסקנה 28.4.** האלגוריתם החמדן מחזיר פתרון אופטימלי.

הוכחה. נפעיל את למה 28.1 עבור $k = m$. מכאן נקבל שקיים פתרון אופטימלי מהצורה $C = (b_1, \dots, b_m, c_{m+1}, \dots, c_{m'})$. $b_1, \dots, b_m$ הוא כבר פתרון חוקי, ולכן הוספת תחנות רק תגרע מהאופטימליות של $C$ $\Leftarrow$ $(b_1, \dots, b_m)$ הוא פתרון אופטימלי. □

**זמן ריצה:** $O(n)$ - כי בכל שלב בלולאה ביצענו השוואות והשמות בכמות חסומה.

## 28.3 סכמה כללית להוכחת נכונות של אלגוריתם חמדן

נסמן את הפתרון של האלגוריתם החמדן ב-$B = (b_1, \ldots, b_m)$.

1. הוכחת חוקיות של $B$.
2. טענת אינדוקציה: "לכל $0 \le k \le m$ ישנו פתרון אופטימלי $C$ שמזדהה עם $B$ על $k$ האיברים הראשונים".
3. הוכחת הטענה באינדוקציה:
   (א) בסיס: $k = 0$, לרוב קל.
   (ב) צעד: נשתמש בהנחה עבור $k - 1$ בכדי לבנות ממנה פתרון עבור $k$, ונראה שהפתרון שבנינו אופטימלי.
4. מסקנה: עבור $k = m$ הפתרון שלנו אופטימלי (במלואו, וללא תוספות).

## 28.4 בעיית מציאת עץ פורש מינימלי - MST

**קלט:** (1) גרף קשיר לא מכוון $G = (V, E)$; (2) פונקציית משקל $w : E \to \mathbb{R}^+$.

**פלט:** עץ פורש ממשקל מינימלי.

**אלגוריתם חמדן (Kruskal):**

1. עיבוד מוקדם: נמיין את כל הצלעות בסדר עולה חלש לפי משקלן $w(e_1) \le w(e_2) \le \cdots \le w\left(e_{|E|}\right)$.
2. אתחול: $T = \emptyset$ (עץ ריק).
3. איטרציה: נעבור על הצלעות מהקלה לכבדה. אם הצלע ה-$i$ לא סוגרת מעגל ב-$T$, נוסיף אותה ל-$T$.
4. סיום: נחזיר את $T$.

**הוכחת נכונות:**
<u>(חוקיות)</u>
צריך להראות ש-$T$ פתרון חוקי, כלומר, ש-$T$ עץ פורש - $T$ גרף קשיר, פורש וחסר מעגלים. תוכיחו בתרגיל 2.
<u>(אופטימליות)</u>
נניח ש-$T$ חוקי. נסמן את הצלעות שלו לפי סדר כניסתן, מהקלה לכבדה, $t_1, t_2, \ldots, t_{|V|-1}$.
נגדיר לכל $0 \le k \le |V| - 1$ את $T_k := \{t_1, \ldots, t_k\}$ (נחשוב על $T_k$ בתור "היער המתהווה").

**למה 28.5** (טענת האינדוקציה של MST). לכל $0 \le k \le |V| - 1$ יש עץ פורש מינימלי המכיל את צלעות $T_k$.

הוכחה. באינדוקציה על $k$.
<u>בסיס:</u> כאשר $k = 0$, $T_k = \emptyset$ לפי הגדרה. כל עץ מכיל את הקבוצה הריקה, בפרט עץ פורש מינימלי.
<u>צעד:</u> נניח כי עבור $k - 1$ קיים פתרון אופטימלי, $S$, עבורו $T_{k-1} \subseteq S$, ונרצה להראות קיום פתרון אופטימלי עבור $k$.
אם $t_k \in S$ אזי $S$ הוא פתרון אופטימלי שמכיל את $T_k$, וסיימנו.
אחרת, נתבונן בגרף $S \cup \{t_k\}$. בגרף זה יש מעגל יחיד[4], כאשר $t_k$ הוא חלק מהמעגל.
במעגל היחיד של $S \cup \{t_k\}$ קיימת צלע, $e$, שלא נמצאת ב-$T_k$ (אחרת, ב-$T_k$ היה מעגל, בסתירה לאופן פעולת האלגוריתם).

- אם $w(e) > w(t_k)$, היה ניתן להוסיף את $t_k$ ל-$S$ במקום הצלע $e$, ולקבל עץ פורש שמשקלו קטן יותר מ-$S$, בסתירה לאופטימליות של $S$.
- אם $w(e) < w(t_k)$, האלגוריתם החמדן הגיע ל-$e$ לפני $t_k$ ולא הכניס אותו לעץ למרות שאינו סוגר מעגל[5], בסתירה לאופן פעולת האלגוריתם החמדן.

---

[4] לפי תכונות של עצים, הוספת צלע לעץ סוגרת מעגל יחיד.
[5] לפי הנחת האינדוקציה, $T_{k-1} \subseteq S$ וגם $T_{k-1}$ ללא מעגלים. אם הוספת $e$ ל-$T_{k-1}$ הייתה סוגרת מעגל, זה היה אומר שב-$S$ היה מעגל (כי $e \in S$), בסתירה לכך ש-$S$ הוא פתרון אופטימלי, ובפרט חוקי.

נסיק כי $w\left(t_k\right)=w\left(e\right)$. נתבונן בעץ $S'=\left(S\cup\left\{t_k\right\}\right)\setminus\left\{e\right\}$.
זהו עץ פורש ומשקלו מקיים $w\left(S'\right)=w\left(S\right)$, ולכן $S'$ פתרון אופטימלי שמכיל את צלעות $T_k$, כנדרש. □

**מסקנה 28.6.** האלגוריתם החמדן של Kruskal מחזיר פתרון אופטימלי.

הוכחה. נפעיל את למה 28.5 עבור $k=|V|-1$, ונקבל כי קיים פתרון אופטימלי, $S$, שמכיל את $T=T_{|V|-1}$. לא ייתכן ש-$S$ מכיל צלעות נוספות מלבד $T$ (אם הוא היה מכיל צלעות נוספות מלבד $T$, הוא לא היה עץ).
מכאן נסיק $T\Leftarrow S=T$ אופטימלי. □

# 29 תרגול 3 - 29.10.18

## 29.1 מטרואיד - הגדרה

**הגדרה 29.1** (מטרואיד)**.** זוג $M = \langle S, \mathcal{I} \rangle$ ייקרא **מטרואיד** אס"ם מתקיימות התכונות הבאות:

1. $S$ היא קבוצה סופית;
2. $\mathcal{I}$ הוא אוסף של תתי קבוצות של $S$, המקיים:
   (א) $\mathcal{I}$ לא ריקה;
   (ב) תורשתיות: אם $A \in \mathcal{I}$ ו-$B \subseteq A$ אז $B \in \mathcal{I}$;
   (ג) החלפה: אם $A, B \in \mathcal{I}$ וגם $|A| > |B|$ אז קיים $x \in A \setminus B$ כך ש-$B \cup \{x\} \in \mathcal{I}$.

**הערה 29.2.** יהי $M = \langle S, \mathcal{I} \rangle$ מטרואיד. משום ש-$\mathcal{I}$ קבוצה לא ריקה המקיימת את תכונת התורשתיות, חייב להתקיים $\emptyset \in \mathcal{I}$.

**דוגמאות:**

1. נגדיר $S = \{1, 2\}$ ו-$\mathcal{I} = \{\{1\}, \{2\}, \{1, 2\}\}$. האם $M = \langle S, \mathcal{I} \rangle$ מטרואיד? **לא.**
   $\mathcal{I}$ לא מקיימת את תכונת התורשתיות: $\{1\} \in \mathcal{I}$ ו-$\emptyset \subseteq \{1\}$ אבל $\emptyset \notin \mathcal{I}$.
2. נגדיר $S = \{1, 2, 3, 4\}$ ו-$\mathcal{I} = \{\emptyset, \{1\}, \{2\}, \{3\}, \{4\}, \{1, 2\}, \{3, 4\}\}$. האם $M = \langle S, \mathcal{I} \rangle$ מטרואיד? **לא.**
   $\mathcal{I}$ לא מקיימת את תכונת ההחלפה: $\{1, 2\} \in \mathcal{I}$ וגם $\{3\} \in \mathcal{I}$, בנוסף $|\{1, 2\}| = 2 > 1 = |\{3\}|$.
   אולם, במקרה זה, לא קיים $x \in \{1, 2\}$ כך ש-$\{3\} \cup \{x\} \in \mathcal{I}$.
3. (המטרואיד הוקטורי) $M_V = \langle S, \mathcal{I} \rangle$ כאשר $S$ היא קבוצת וקטורים סופית במרחב וקטורי כלשהו, $\mathcal{I}$ מכילה קבוצות של וקטורים בת"ל ב-$S$. ראו למה 5.4 עבור ההוכחה לתכונת ההחלפה של המטרואיד הוקטורי.

## 29.2 המטרואיד הגרפי

**טענה 29.3.** יהי $G = (V, E)$ גרף לא מכוון. אזי $M_G = \langle S, \mathcal{I} \rangle$ הוא מטרואיד, כאשר $S = E$ ו-$\mathcal{I} = \{B \subseteq S : \text{אין מעגלים ב-}B\}$.

הוכחה. ראשית, $S$ קבוצה סופית כי הגרף $G$ סופי. בנוסף, $\mathcal{I}$ מקיימת:

- <u>לא ריקה:</u> $\emptyset \in \mathcal{I}$ כיוון שהקבוצה הריקה היא ללא מעגלים.
- <u>תורשתיות:</u> תהי $A \in \mathcal{I}$ ו-$B \subseteq A$, נרצה להוכיח כי $B \in \mathcal{I}$. מכיוון ש-$A \in \mathcal{I}$ אנו יודעים ש-$A$ לא מכילה מעגלים. מכיוון ש-$B \subseteq A$, גם $B$ לא מכילה מעגלים (אילו הייתה מכילה מעגל, גם ב-$A$ היה מעגל, בסתירה לכך ש-$A \in \mathcal{I}$) $\Leftarrow$ $B \in \mathcal{I}$.
- <u>החלפה:</u> תהיינה $A, B \in \mathcal{I}$ עם $|A| > |B|$. נרצה להוכיח שקיימת צלע $a \in A \setminus B$ שעבורה $B \cup \{a\} \in \mathcal{I}$.
  נגדיר $G_A = (V, A)$ ו-$G_B = (V, B)$. נסמן $|V| = n$.

מכיוון ש-$A, B \in \mathcal{I}$ הגרפים $G_A, G_B$ הם חסרי מעגלים, לכן, מספר רכיבי הקשירות בהם הוא $n - |A|$ ו-$n - |B|$ בהתאמה.

נטען שקיים לפחות רכיב קשירות אחד של $G_A$ שנחתך עם לפחות שני רכיבי קשירות של $G_B$. זאת משום שאחרת, כל רכיב קשירות של $G_A$ מוכל ברכיב קשירות של $G_B$, וזה גורר שמספר רכיבי הקשירות ב-$G_A$ הוא לפחות מספר רכיבי הקשירות ב-$G_B$. מכאן,

$$\overbrace{n - |A|}^{G_A\text{ מספר רכיבי הקשירות ב-}} \geq \overbrace{n - |B|}^{G_B\text{ מספר רכיבי הקשירות ב-}}$$

ומכאן נקבל $|A| \leq |B|$, בסתירה לכך ש-$|B| < |A|$.

נתבונן ברכיב קשירות של $G_A$ שנחתך עם לפחות שני רכיבי קשירות של $G_B$.

חייבת להיות צלע, נסמנה $a$, המחברת בין שני רכיבי קשירות שונים של $G_B$.

$\Leftarrow$ $a$ זו הצלע המבוקשת: $a \in A \setminus B$ והוספתה ל-$B$ אינה סוגרת מעגל (כי מחברת בין שני רכיבי קשירות שונים של $G_B$), כנדרש.

□

## 29.3 אלגוריתם חמדן גנרי

נתבונן בבעיית האופטימיזציה הבאה:

**בעיה.** קלט: (1) מטרואיד $M = \langle S, \mathcal{I} \rangle$; (2) פונקציית משקל חיובית $w : S \to \mathbb{R}^+$. פלט: תת-קבוצה $A \in \mathcal{I}$ שמשקלה מקסימלי; כאשר משקל של קבוצה $A$ מוגדר באופן הבא:

$$w(A) \coloneqq \sum_{a \in A} w(a)$$

**הערה 29.4.** חוקיות הפתרון נקבעת לפי השייכות של $A$ ל-$\mathcal{I}$. בנוסף, גודל הקבוצה $A$ הוא מקסימלי ב-$\mathcal{I}$, זאת משום שפונקציית המשקל חיובית. אופטימליות הפתרון נקבעת לפי המשקל של הקבוצה $A$.

**הערה 29.5.** ניתן לפתור גם בעיית אופטימיזציה שבה הפלט הוא תת-קבוצה $A \in \mathcal{I}$ שמשקלה מינימלי.

מתברר שקיים אלגוריתם חמדן גנרי שפותר את בעיית האופטימיזציה דלעיל:

1. עיבוד מוקדם: נמיין את איברי $S$ לפי סדר משקל יורד חלש.[6]
2. אתחול: $A = \emptyset$.
3. איטרציה: לפי הסדר שקיבלנו, לכל איבר $x \in S$, אם $A \cup \{x\}$ הוא קבוצה חוקית (כלומר, ב-$\mathcal{I}$) אז נבצע $A = A \cup \{x\}$.
4. סיום: נחזיר את $A$.

**הערה 29.6.** הבעיה המתאימה למטרואיד הוקטורי - מציאת קבוצה בת״ל של וקטורים מ-$S$ עם משקל מקסימלי.
הבעיה המתאימה למטרואיד הגרפי - מציאת קבוצת צלעות שאינה סוגרת מעגל עם משקל מקסימלי (עץ פורש מקסימלי).

**זמן ריצה:** נסמן $n \coloneqq |S|$.
שלב העיבוד המוקדם לוקח $O(n \log n)$, שלב האתחול לוקח $O(1)$.
זמן ביצוע הבדיקה האם $(A \cup \{x\}) \in \mathcal{I}$ תלוי בהגדרת המטרואיד, לצורך הניתוח נסמן זמן זה ב-$O(f(n))$.
$\Leftarrow$ יש לכל היותר $n$ איטרציות, ולכן זמן הריצה הכולל הוא $O(n \log n + nf(n))$.

## 29.4 מטרואיד השידוכים

תזכורת מושגים בגרפים:

- **גרף דו-צדדי** $G = (L, R, E)$ הוא גרף המוגדר על שתי קבוצות זרות של קודקודים $L$ ו-$R$, כך שמתקיים: אם $\{u, v\} \in E$ אז $u \in L$ ו-$v \in R$ (או ההפך: $u \in R$ ו-$v \in L$).
- **שידוך** בגרף דו-צדדי הוא תת-קבוצה של צלעות שאין בה שתי צלעות ״שנוגעות״ באותו הקודקוד.

**טענה 29.7.** יהי $G = (L, R, E)$ גרף דו-צדדי. נגדיר $S = L$ ו-$A \in \mathcal{I}$ אם״ם $A \subseteq L$ וגם ניתן לשדך את כל איברי $A$ ב-$G$. אזי $M = \langle S, \mathcal{I} \rangle$ הוא מטרואיד.

הוכחה. ראשית, $S$ קבוצה סופית. נוודא ש-$\mathcal{I}$ מקיימת את התכונות בהגדרת המטרואיד:

- לא ריקה: $\mathcal{I}$ לא ריקה כי $\emptyset \in \mathcal{I}$ (ניתן לשדך את איברי הקבוצה הריקה ב-$G$).
- תורשתיות: תהי $A \in \mathcal{I}$ ו-$B \subseteq A$. משום ש-$A \in \mathcal{I}$, ניתן לשדך את כל איברי $A$. נטען שקיים גם שידוך עבור איברי $B$ - ואכן, הצמצום של השידוך שמשדך את איברי $A$ על הקבוצה $B$ הוא השידוך המבוקש.
- החלפה: יהיו $A, B \in \mathcal{I}$ כך ש- $|A| > |B|$. צריך להראות שקיים $x \in A \setminus B$ כך ש-$B \cup \{x\} \in \mathcal{I}$.

  נסמן ב-$M_A$ את קבוצת הצלעות בשידוך של כל איברי $A$. נסמן ב-$M_B$ את קבוצת הצלעות בשידוך של כל איברי $B$.

  נגדיר גרף חדש $G' = (L, R, M_A \cup M_B)$. נשים לב שמתקיים:

  1. ב-$G'$ דרגות הקודקודים הן לכל היותר 2.
  2. מ-1 נובע כי ב-$G'$ יכולים להופיע רק רכיבי קשירות מהסוגים הבאים:
     (א) מעגל פשוט - בו דרגת כל הקודקודים שווה 2.

[6] אם נרצה למזער משקל, נמיין בסדר עולה חלש

(ב) מסלול פשוט - בו דרגות כל הקודקודים הפנימיים שווים 2, ודרגות שני הקודקודים החיצוניים שווים 1.

3. בכל רכיבי הקשירות, הצלעות צריכות להיות מ-$M_A$ ומ-$M_B$ לסירוגין (אחרת, זוהי תהיה סתירה לכך ש-$M_A$ ו-$M_B$ שידוכים).

4. יהי $C$ רכיב קשירות ב-$G'$. נסמן ב-$E_C\left(M_A\right)$ את הצלעות ברכיב $C$ השייכות ל-$M_A$, ובאופן דומה נסמן ב-$E_C\left(M_B\right)$ את הצלעות ברכיב $C$ השייכות ל-$M_B$. מ-3 נובע כי לכל רכיב קשירות $C$ מתקיים $||E_C\left(M_A\right)|-|E_C\left(M_B\right)||\leq 1$. ובמילים, ההפרש, בערך מוחלט, בין מס' הצלעות מ-$M_A$ ברכיב $C$ לבין מס' הצלעות מ-$M_B$ ברכיב $C$ הוא לכל היותר $1$.

עתה, ניזכר כי מכיוון ש-$|B|<|A|$ מתקיים $|M_B|<|M_A|$[7], ומשיקולי ספירה, חייב להיות רכיב קשירות ב-$G'$, שנסמנו $C$, שעבורו מתקיים $|E_C\left(M_A\right)|>|E_C\left(M_B\right)|$.

נבחין כי $C$ חייב להיות מסלול פשוט מאורך אי-זוגי, אשר מתחיל בצלע השייכת ל-$M_A$ ומסתיים בצלע השייכת ל-$M_A$.

מסלול זה, מכיוון שאורכו אי-זוגי, מתחיל או מסתיים בקודקוד מ-$L$. נסמן קודקוד זה ב-$x$. נבחין כי $x\in A\setminus B$.[8]

נגדיר שידוך עבור איברי $B'=B\cup\{x\}$ באופן הבא: נשאיר את כל הצלעות $M_B$, חוץ מברכיב הקשירות $C$ המדובר, שם ניקח רק את הצלעות ששייכות ל-$M_A$. בסימונים מתמטיים $M_{B'}=\left(M_B\setminus E_C\left(M_B\right)\right)\cup E_C\left(M_A\right)$.

זה אכן שידוך, ולכן $B\cup\{x\}\in\mathcal{I}$, כנדרש.

□

[7] מתקיים $|A|=|M_A|$ ו-$|B|=|M_B|$ כי מדובר בשידוכים.
[8] אילו $x\in B$ אז $M_B$ לא היה שידוך - זה נובע מתכונה 3 דלעיל, שכן הצלעות במסלולים שייכות ל-$M_A$ ו-$M_B$ לסירוגין.

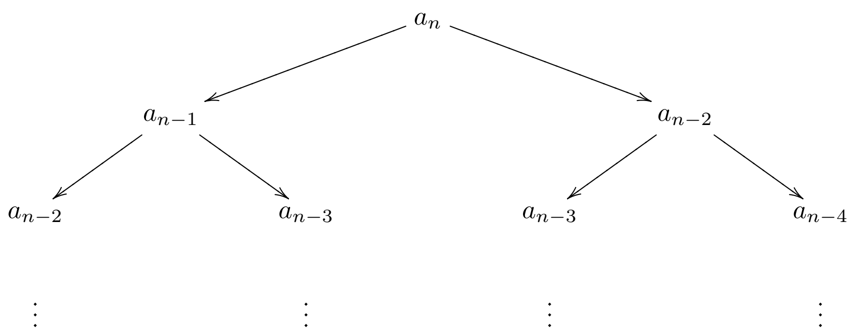
# 30 תרגול 4 - 5.11.18

## 30.1 סדרת פיבונאצ'י

נביא את דוגמת חישוב האיבר ה-$n$ של סדרת פיבונאצ'י בתור אינטואיציה לתכנון דינאמי.
נרצה לחשב את האיבר ה-$n$ של סדרת פיבונאצ'י באמצעות כלל הנסיגה

$$a_n = \begin{cases} 0 & n = 1 \\ 1 & n = 2 \\ a_{n-1} + a_{n-2} & n > 2 \end{cases}$$

נוכל לממש אלגוריתם רקורסיבי נאיבי שמחשב את האיבר ה-$n$. נתבונן בעץ הקריאות הרקורסיביות:




עץ הקריאות הרקורסיביות הוא עץ בינארי מלא עד גובה $\frac{n}{2}$ (אורך המסלול הימני ביותר), ולכן יש לפחות $2^{n/2+1} - 1$ קריאות רקורסיביות.
הצעה לשיפור: נשמור את $n$ תתי הבעיות בטבלה חד-מימדית. נמלא קודם את תתי הבעיות הקטנות ומהם נחשב את תתי הבעיות הגדולות.
זמן ריצה: גודל הקלט הוא $\log(n)$, וסידור הערכים בטבלה ומילויה לוקח $O(n)$.
הערות:

- זמן ריצה זה שהשגנו ע"י שימוש בעקרונות תכנון דינאמי הוא עדיין אקספוננציאלי ביחס לגודל הקלט, אבל משמעותית טוב יותר מאשר המימוש הנאיבי.
- כמות הזיכרון בשימוש היא $O(n)$ (גודל הטבלה), כאשר יכולנו להשתמש ב-2 תאי זיכרון בודדים (כיוון שהחישוב מסתמך רק על $a_{n-1}$ ו-$a_{n-2}$).

## 30.2 תכנון דינאמי

תכונות משותפות לכלל הבעיות שניתן לפתור באמצעות תכנון דינאמי:

- מספר תתי הבעיות הוא "קטן" (פולינומיאלי בגודל הקלט).
- ניתן לסדר את תתי הבעיות מן הקטנה ביותר לגדולה ביותר, כאשר יש נוסחת רקורסיה המקשרת ביניהן.

שלבים לפתרון בעיה באמצעות תכנון דינאמי:

1. הגדרת תתי הבעיות (בדרך כלל הבעיה מקורית רק "בקטן יותר").
2. כתיבת נוסחת רקורסיה המקשרת בין תתי הבעיות הקטנות יותר לגדולות יותר.
3. הגדרת טבלה, אופן מילויה ואופן חילוץ הפתרון ממנה.
   גודל הטבלה הוא כמספר תתי הבעיות.
   נמלא בעיות "קטנות" לפני בעיות "גדולות".
4. ניתוח זמן ריצה (לרוב, מספר התאים בטבלה·הזמן למילוי כל תא).
5. הוכחת נכונות של נוסחת הרקורסיה (לרוב, באינדוקציה על סדר מילוי הטבלה).

נראה באיזה אופן השלבים הללו שולבו בפתרון שהצענו לחישוב האיבר ה-$n$ של סדרת פיבונאצ'י:

1. תתי בעיות: הערך $a_i$ לכל $1 \le i \le n$.
2. נוסחת רקורסיה: נתונה באופן מפורש.
3. הגדרת טבלה: טבלה חד-מימדית עם $n$ תאים.
   אופן מילוי: נמלא את הטבלה מ-$a_1$ ל-$a_n$.
   חילוץ הפתרון: האיבר $a_n$ נמצא בתא ה-$n$ בטבלה.
4. זמן ריצה: $O(n) = \underbrace{O(1)}_{\text{זמן מילוי}} \cdot \underbrace{O(n)}_{\text{גודל הטבלה}}$.

## 30.3 לוח משימות

**קלט:** רשימה $\{(h_i, \ell_i)\}_{i=1}^{n}$ כאשר $n$ מציין את מס' השבועות, $\ell_i$ - מספר חיובי שמציין את התגמול עבור ביצוע משימה קלה בשבוע ה-$i$, $h_i$ - מספר חיובי שמציין את התגמול עבור ביצוע משימה קשה בשבוע ה-$i$.

**פלט:** לוח משימות רווחי ככל האפשר עבור $n$ שבועות, כאשר בכל שבוע נתוגמל לפי המשימה שבחרנו לבצע. אילוץ: אם מבצעים משימה קשה, חייבים לנוח בשבוע הקודם לה (התנאי לא חל על השבוע הראשון).

**דוגמה:** נתונה רשימה עם 5 שבועות והתגמולים עבור משימה קלה וקשה.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $\ell$ | 15 | 15 | 10 | 5 | 20 |
| $h$ | 5 | 20 | 40 | 10 | 20 |

לוח משימות אפשרי:

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| $\ell$ | $\times$ | $h$ | $\ell$ | $\ell$ |

כאשר הרווח הוא $15 + 0 + 40 + 5 + 20 = 80$.

**אלגוריתם דינאמי:**

1. תתי-בעיות: מציאת הרווח האופטימלי שניתן להשיג ב-$i$ השבועות הראשונים עבור $0 \le i \le n$.
2. נוסחת רקורסיה: נסמן ב-$M[i]$ את הרווח האופטימלי שאנו יכולים להשיג ב-$i$ השבועות הראשונים. נבחין בין שני מקרים אפשריים שעלולים לקרות בשבוע ה-$i$:
   - בחרנו משימה קלה, ואז $M[i] = \ell_i + M[i-1]$
   - בחרנו משימה קשה, ואז $M[i] = h_i + M[i-2]$

   מאחר שאנו רוצים למקסם את הרווח, אנו נגדיר
   $$M[i] = \max\{\ell_i + M[i-1], h_i + M[i-2]\}$$
   כמובן שעלינו לספק מקרי בסיס:
   $$M[0] = 0, M[1] = \max\{h_1, \ell_1\}$$
3. הגדרת טבלה: נבנה טבלה $M$ חד-מימדית בגודל $n+1$.
   אופן מילוי: נמלא את הטבלה לפי נוסחת הרקורסיה משמאל לימין. קודם את מקרי הבסיס $(i = 0, 1)$ ולאחר מכן את שאר המקרים בהתבסס על התאים שמולאו קודם.
   חילוץ הפתרון: בזמן מילוי הטבלה, בכל תא נשמור גם את החלק בביטוי ה-max שנתן את המקסימום (לקיחת משימה קלה או קשה). כאשר הטבלה מלאה ונרצה לחלץ את לוח המשימות, נעבור על הטבלה מהסוף להתחלה - נתחיל בתא ה-$n$-י. אם המקסימום התקבל ממשימה קלה בשבוע ה-$n$-י תילקח משימה קלה בשבוע זה, ותהליך השחזור ימשיך משבוע $n-1$. אחרת, תילקח משימה קשה, ותהליך השחזור ימשיך משבוע $n-2$.
4. זמן ריצה: $O(n) = \underbrace{O(1)}_{\text{זמן מילוי}} \cdot \underbrace{O(n)}_{\text{גודל הטבלה}}$

## 30.4 תת-מחרוזת משותפת מקסימלית

**קלט:** 2 מחרוזות, $X = x_1 x_2 \dots x_n$ ו-$Y = y_1 y_2 \dots y_m$.

**פלט:** תת-מחרוזת משותפת מקסימלית בין $X$ ו-$Y$ (משותפת משמע לא בהכרח רציפה, אך באותו הסדר).

**דוגמה:** $X = abcd$, $Y = bdc$. יש שתי אפשרויות לתמ״א (תת-מחרוזת ארוכה ביותר): $bd$ או $bc$.

**אלגוריתם דינאמי:**

1. תתי בעיות: נסמן ב-$X^i$ את הרישא באורך $i$ של $X$, כלומר, $X^i = x_1 \dots x_i$. נסמן ב-$Y^j$ את הרישא באורך $j$ של $Y$, כלומר, $Y^j = y_1 \dots y_j$.

   תת-בעיה היא מציאת תת-מחרוזת משותפת ארוכה ביותר של $X^i$ ו-$Y^j$ לכל $0 \le i \le n$ ו-$0 \le j \le m$.

2. נוסחת רקורסיה: נסמן ב-$f(i,j)$ את האורך של תמ״א של $X^i$ ו-$Y^j$.

   תובנה מרכזית:

- אם $x_i = y_j$ אז זו חייבת להיות האות האחרונה בתמ״א של $X^i$ ו-$Y^j$, ואת שאר הפתרון נוכל לבנות מהתמ״א של $X^{i-1}$ ו-$Y^{j-1}$.
- אם $x_i \neq y_j$ אז אחת מהאפשרויות הבאות: תמ״א של $X^i$ ו-$Y^{j-1}$ או תמ״א של $X^{i-1}$ ו-$Y^j$, חייבת להיות תמ״א של $X^i$ ו-$Y^j$.

נתרגם תובנה זו לכלל נסיגה:

$$f(i,j) = \begin{cases} 0 & i = 0 \vee j = 0 \\ 1 + f(i-1, j-1) & x_i = y_j \\ \max\{f(i-1,j), f(i,j-1)\} & x_i \neq y_j \end{cases}$$

3. הגדרה טבלה: נבנה טבלה $M$ בגודל $(n+1) \times (m+1)$, כאשר בתא $M[i,j]$ נציב את הערך $f(i,j)$.

   אופן מילוי: תחילה, נמלא מקרי בסיס. נרצה שבכל שלב התאים שמכילים את תתי הבעיות שבאמצעותם ניתן לחשב את תת הבעיה החדשה יהיו מלאים. לכן נמלא את הטבלה לפי שורות, כאשר נתחיל מהשורה $j = 0$ ונסיים בשורה $j = m$, כאשר כל שורה נמלא מהתא ה-$0$ ועד התא ה-$n$.

   חילוץ הפתרון: בעת מילוי הטבלה, בכל שלב נשמור מצביע שיחווה על התא שממנו נגזר הפתרון החדש. כשסיימנו למלא את הטבלה, נעבור עליה מהתא ה-$(n,m)$ ונעקוב אחרי החיווי מהמצביע ששמרנו קודם לכן. אם התא $(i,j)$ מצביע לתא $(i-1, j-1)$ נוסיף את האות $x_i$ לתת המחרוזת משמאל.

4. זמן ריצה: $\underbrace{O(n \cdot m)}_{\text{גודל הטבלה}} \cdot \underbrace{O(1)}_{\text{זמן מילוי}} = O(nm)$

5. הוכחת נכונות: ראו סיכום תרגול.

# 31 תרגול 5 - 12.11.18

## 31.1 בעיית מסילת הרכבת

**בעיה 31.1** (בעיית מסילת הרכבת)**.**
<u>קלט:</u>

(1) מספר טבעי $L \in \mathbb{N}$ - אורך המסילה

(2) $\{1, 2, \ldots, K\}$ - סוגי החיבורים

(3) $N$ סוגי חלקים, ולכל חלק $1 \leq i \leq N$:

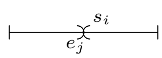
(א) $s_i \in \{1, 2, \ldots, K\}$ - סוג חיבור ההתחלה

(ב) $e_i \in \{1, 2, \ldots, K\}$ - סוג חיבור הסוף

(ג) $d_i \in \mathbb{N}$ - אורך החלק

(ד) $p_i \in \mathbb{N}$ - עלות החלק

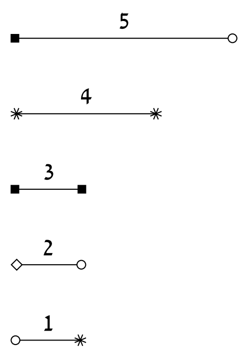
<u>פלט:</u>
המחיר המינימלי עבור מסילה חוקית באורך $L$, כאשר מסילה חוקית היא מסילה שבה חלק $i$ בא ישר לאחר חלק $j$ אס"ם $e_j = s_i$.



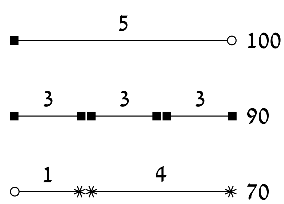

**דוגמה:** אורך המסילה $L = 3$, יש $K = 4$ סוגי חיבורים: $\{\bigcirc, \ast, \Diamond, \blacksquare\}$, סוגי החלקים ($N = 5$)

$$\{(s_1 = \bigcirc, e_1 = \ast, d_1 = 1, p_1 = 30), (\Diamond, \bigcirc, 1, 10), (\blacksquare, \blacksquare, 1, 30), (\ast, \ast, 2, 40), (\blacksquare, \bigcirc, 3, 100)\}$$

להלן הדגמה של החלקים השונים:




פתרונות אפשריים עבור מסילה באורך $L = 3$:




התובנה המרכזית: אם מורידים חלק אחרון בעל אורך $d_i$ וחיבור $s_i$ נשארים עם תת-בעיה של מציאת מסילה אופטימלית באורך $L - d_i$ עם חיבור סוף השווה ל-$s_i$.
נתאר אלגוריתם תכנון דינאמי לפתרון הבעיה.

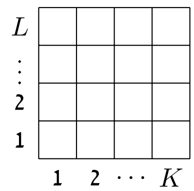
**תתי-בעיות:** מציאת המחיר המינימלי למסילה באורך $\ell$ שחיבור הסוף שלה הוא $k$ לכל $0 \leq \ell \leq L$ ולכל $1 \leq k \leq K$.
מספר תתי הבעיות הוא $(L + 1) \cdot K$.

**נוסחת הרקורסיה:** נגדיר $f(\ell,k)$ להיות המחיר המינימלי למסילה באורך $\ell$ שמסתיימת בחיבור $k$.
אזי:

$$f(\ell,k)=\begin{cases}\ell=0: & 0\\ \ell>0: & \min\left\{f(\ell-d_i,s_i)+p_i : 1\le i\le N, e_i=k, \ell-d_i\ge 0\right\}\end{cases}$$

בנוסחת הרקורסיה אנו עוברים על כל החלקים $1\le i\le N$ שחיבור הסיום שלהם שווה $k$ ($e_i=k$) וגם הם לא יותר ארוכים מאורך המסילה. אנו לוקחים את החלק שהעלות הכוללת שלו לבניית מסילה חוקית באורך $\ell$ הוא מינימלי.
בבעיה זו נגדיר $\min(\emptyset)=\infty$, במקרה שאין לנו חלקים מתאימים אין לנו אפשרות להשלים את בניית המסילה, ונרצה להחזיר מחיר מאוד לא משתלם.

**הגדרת טבלה:** נבנה טבלה $M$ בגודל $(L+1)\cdot K$, כאשר ב-$M[\ell,k]$ יהיה הערך של $f(\ell,k)$.




**אופן מילוי:** נשים לב שמילוי התא ה-$(\ell,k)$ כרוך במילוי השורות שמתחתיו, כלומר כל התאים $(\ell',k')$ עם $0\le\ell'<\ell$ צריכים להיות מלאים.
לכן, נרצה למלא ראשית את השורה התחתונה, ולהמשיך כלפי מעלה שורה אחר שורה.
נשים לב שאין משמעות לסדר המילוי הפנימי בתוך השורה.

**חילוץ הפתרון:** הפתרון של הבעיה כולה יימצא בשורה העליונה, המחיר המינימלי למסילה חוקית באורך $L$ הינו:

$$\min_{1\le k\le K}\left\{M[L,k]\right\}$$

(למעשה, מינימום על השורה העליונה).

**זמן ריצה:** גודל הטבלה הוא $O(L\cdot K)$, זמן מילוי כל תא דורש $O(N)$ $\Leftarrow$ בסה״כ $O(N\cdot L\cdot K)$.

**ריצת האלגוריתם על הדוגמה הקודמת:**

| | | | | |
|---|---|---|---|---|
| 3 | 100 | 70 | $\infty$ | 90 |
| 2 | $\infty$ | 40 | $\infty$ | 60 |
| 1 | 10 | 30 | $\infty$ | 30 |
| 0 | 0 | 0 | 0 | 0 |
| | ○ | $\ast$ | ◊ | ■ |

## 31.2 בעיית מציאת מרחקים קצרים - APSP

**בעיה 31.2** (All Pairs Shortest Paths).
קלט:

1. גרף מכוון $G=(V,E)$, כאשר $|V|=n$. בגרף אין מעגלים שליליים (negative weight cycles).
2. פונקציית משקל על צלעות הגרף $w:E\to\mathbb{R}$ (לאו דווקא חיובית).

פלט:
מטריצה בגודל $n\times n$ כאשר בתא ה-$(i,j)$ שלה נמצא המחיר הקצר ביותר בין קודקוד $v_i$ ל-$v_j$.

ננסה לפתור את הבעיה באמצעות תכנון דינאמי.

**ניסיון ראשון:** נגדיר את תתי הבעיות באופן הבא: לכל $1\le i,j\le n$ נמצא את המחיר של מסלול קצר ביותר בין $v_i$ ל-$v_j$.

**בעייתיות** זאת הבעיה המקורית - לא תת-בעיה.

**מסקנה** אנו חייבים להוסיף פרמטר נוסף בכדי ליצור היררכיה כלשהי של תתי-בעיות.

**<u>ניסיון שני:</u>** הפרמטר שנוסיף - מס' הצלעות במסלול.

**אבחנה** בהינתן שאין מעגלים שליליים בגרף, מסלול קצר ביותר בין שני קודקודים יכול להכיל לכל היותר $n-1$ צלעות.

**תתי-בעיות** לכל $1 \le i,j \le n$ ו-$0 \le m \le n-1$ נמצא את המחיר של מסלול קצר ביותר בין $v_i$ ל-$v_j$ אשר מכיל לכל היותר $m$ צלעות.

**נוסחת הרקורסיה** נגדיר $f\left(i,j,m\right)$ להיות המחיר של מסלול קצר ביותר בין $v_i$ ל-$v_j$ שמכיל לכל היותר $m$ צלעות.
נשים לב שבהינתן מסלול קצר ביותר מ-$v_i$ ל-$v_j$ שעובר דרך קודקוד $v_x$, המסלול בין $v_i$ ל-$v_x$ חייב להיות מסלול קצר ביותר (אחרת, יכולנו להחליפו במסלול אחר בין $v_i$ ל-$v_x$ שהוא קצר יותר, ולקבל מסלול קצר יותר בין $v_i$ ל-$v_j$).

$$v_i \rightsquigarrow v_x \longrightarrow v_j$$

אם מסלול קצר ביותר בין $v_i$ ל-$v_j$ מכיל לכל היותר $m$ צלעות, אז המסלול בין $v_i$ ל-$v_x$ הוא מסלול קצר ביותר שמכיל לכל היותר $m-1$ צלעות.
מכאן נגזור את נוסחת הרקורסיה הבאה:

$$f\left(i,j,m\right)=\begin{cases} m=0, i=j: & 0 \\ m=0, i\neq j: & \infty \\ m>0: & \min\left\{f\left(i,x,m-1\right)+w\left(v_x,v_j\right) : v_x\in V\right\}\end{cases}$$

**הגדרת טבלה** נבנה טבלה תלת מימדית בגודל $n^3$ - נחשוב על טבלה עם $n$ תאים (עבור המשתנה $m$) כאשר בכל תא יש מטריצה מסדר $n\times n$.

**סדר מילוי** תחילה נמלא את המטריצה שעבורה $m=0$, לאחר מכן נמלא את המטריצה שעבורה $m=1$ וכן הלאה עד שנמלא את המטריצה שעבורה $m=n-1$.
סדר מילוי זה נותן את הדרוש כיוון שכל תא שמחושב במטריצה הנוכחית נסמך על התאים שחושבו במטריצה הקודמת.

**חילוץ הפתרון** המטריצה האחרונה שחושבה, $m=n-1$, היא הפלט הדרוש לבעיה.

**זמן ריצה** גודל הטבלה הוא $n^3$ וזמן מילוי כל תא הוא $O\left(n\right)$ במקרה הגרוע $\Leftarrow$ בסה"כ $O\left(n^4\right)$ זמן ריצה.

**<u>ניסיון שלישי:</u>** האלגוריתם של Floyd-Warshall, הפרמטר שנוסיף - אינדקס מקסימלי של קודקוד במסלול.

**תתי-בעיות** לכל $1 \le i,j \le n$ ו-$0 \le k \le n$ נמצא את המחיר של מסלול קצר ביותר בין $v_i$ ל-$v_j$ שמשתמש רק ב-$\{v_1,\ldots,v_k\}$ כקודקודי ביניים במסלול.

**נוסחת הרקורסיה** נגדיר $f\left(i,j,k\right)$ בתור המחיר של מסלול קצר ביותר בין $v_i$ ל-$v_j$ שמשתמש רק ב-$\{v_1,\ldots,v_k\}$ כקודקודי ביניים במסלול.
נרשום את מקרה הבסיס:

$$f\left(i,j,0\right)=\begin{cases} i=j: & 0 \\ \left(v_i,v_j\right)\in E: & w\left(v_i,v_j\right) \\ \left(v_i,v_j\right)\notin E: & \infty\end{cases}$$

עבור המקרה הכללי, נתבונן במסלול קצר ביותר, $P$, בין $v_i$ ל-$v_j$ שמשתמש ב-$\{v_1,\ldots,v_k\}$ כקודקודי ביניים במסלול.
אם $P$ אינו מכיל את $v_k$ כקודקוד ביניים אז מחיר המסלול $P$ הוא מחירו של המסלול הקצר ביותר בין $v_i$ ל-$v_j$ שמשתמש ב-$\{v_1,\ldots,v_{k-1}\}$ כקודקודי ביניים במסלול.

אם $P$ משתמש ב-$v_k$ כקודקוד ביניים, אז הוא משתמש בו רק פעם אחת - אחרת, היה נוצר מעגל במסלול, וכיוון שאין מעגלים שליליים, ניתן להוריד צלעות מהמעגל ולקבל מסלול קצר יותר.
במקרה הזה, נוכל לפצל את המסלול הקצר ביותר בין $v_i$ ל-$v_j$ שמשתמש ב-$\{v_1, \ldots, v_k\}$ כקודקודי ביניים, לשני מסלולים: הראשון, מסלול קצר ביותר בין $v_i$ ל-$v_k$ שמשתמש ב-$\{v_1, \ldots, v_{k-1}\}$ כקודקודי ביניים, השני, מסלול קצר ביותר בין $v_k$ ל-$v_j$ שמשתמש ב-$\{v_1, \ldots, v_{k-1}\}$ כקודקודי ביניים.
ננסח זאת במדויק ($k > 0$):

$$f(i,j,k) = \min\left\{f(i,j,k-1), f(i,k,k-1) + f(k,j,k-1)\right\}$$

**הגדרת טבלה** נגדיר טבלה תלת מימדית - טבלה עם $n+1$ תאים, כאשר בכל תא נמצאת מטריצה מסדר $n \times n$.

**אופן מילוי** נמלא את המטריצה התחתונה $k=0$ ראשונה, לאחר מכן את $k=1$ וכן הלאה עד $k=n$.

**חילוץ הפתרון** נחזיר את המטריצה העליונה.

**זמן ריצה** גודל הטבלה הוא $O\left(n^3\right)$, זמן מילוי כל תא הוא $O(1)$ ובסה״כ קיבלנו זמן ריצה של $O\left(n^3\right)$.

דוגמה זו ממחישה כיצד הגדרה שונה של תתי הבעיות עלולה להוליד אלגוריתם תכנון דינאמי יעיל יותר.

# 32 תרגול 6 - 19.11.18

## 32.1 תכנון לינארי

### 32.1.1 שיטות לפתרון בעיות אופטימיזציה

במהלך הקורס ראינו מס' גישות לפתרון בעיות אופטימיזציה:

- אלגוריתמים חמדניים: בדרך כלל פתרון אינטואיטיבי לבעיה. עובד כאשר יש מבנה מטרואידי לבעיה או כאשר ניתן לנסח ולהוכיח למת החלפה. ואמנם, ברוב הבעיות אסטרטגיה זו לא תעבוד.
- תכנון דינאמי: כדי לפתור בעיה באמצעות תכנון דינאמי אנו דורשים שמספר תתי הבעיות יהיה קטן (פולינומיאלי) ובנוסף שתהיה קיימת נוסחת רקורסיה שמקשרת בין תתי הבעיות "הקטנות" ל"גדולות".

נציג כעת שיטה נוספת לפתרון בעיות אופטימיזציה: תכנון לינארי.
נציין כי ההיכרות עם תכנון לינארי במהלך הקורס היא על קצה המזלג בלבד, ותשמש אותנו בבואנו ללמוד על אלגוריתמי קירוב.

### 32.1.2 מה זו בעיית תכנון לינארי?

אינטואיטיבית, בעיית תכנון לינארי (Linear Programming או בקיצור: LP) היא בעיית אופטימיזציה שבה אנו רוצים למקסם/למזער פונקציה לינארית, בכפוף לאילוצים לינאריים שונים. (בפונקציה לינארית ואי-שוויון לינארי כל המשתנים הם ממעלה ראשונה).

**דוגמה 32.1.** נביא דוגמה לבעיית תכנון לינארי פשוטה:

$$\begin{aligned} \max \quad & x_1 + 2x_2 \\ \text{subject to} \quad & 4x_1 - x_2 \leq 3 \\ & 2x_1 + 2x_2 \geq -100 \\ & x_1 \geq 0 \\ & x_2 \geq 0 \end{aligned}$$

הפונקציה הלינארית למיקסום היא $x_1 + 2x_2$, בכפוף לאילוצים לינאריים שונים שנכתבו בצורת אי-שוויונים חלשים.

**הגדרה 32.2** (בעיית תכנון לינארי)**.** בעיית אופטימיזציה תיקרא **בעיית תכנון לינארי** אם ניתן לכתוב אותה בצורה הבאה (**הצורה הסטנדרטית**):

$$\begin{aligned} \max_{x \in \mathbb{R}^n} \quad & c^T \cdot x \\ \text{subject to} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

כאשר $x \in \mathbb{R}^n$ וקטור הפתרון, $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ ו-$A \in \mathbb{R}^{m \times n}$.

**הערה 32.3.**

(1) $c \in \mathbb{R}^n$ הוא וקטור עמודה של מקדמים.

(2) אם נסמן $c = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}$ ו-$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ אז הפונקציה הלינארית למקסום היא $c^T \cdot x$ כלומר

$$c^T \cdot x = \begin{pmatrix} c_1 & \dots & c_n \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = c_1x_1 + \cdots + c_nx_n$$

כאשר אנו עוברים על כל הוקטורים $x \in \mathbb{R}^n$.

(3) הכתיבה $x \geq 0$ היא כתיבה מקוצרת ל-$n$ אילוצים: לכל $1 \leq i \leq n$ נדרוש $x_i \geq 0$.

(4) בדומה, הכתיבה $Ax \le b$ היא כתיבה מקוצרת ל-$m$ אילוצים.

אם נסמן

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix}, b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

אז לכל $1 \le i \le m$ אנו דורשים

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i$$

**דוגמה 32.4.** נכתוב את דוגמה 32.1 בצורה הסטנדרטית.
את הוקטור $x$ נכתוב בתור $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$.
עלינו למקסם את הפונקציה $x_1 + 2x_2$, מכאן נקבל את וקטור המקדמים $c = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$.
נשים לב שע״י כפל ב-$-1$ ניתן ״להמיר״ את האילוץ $2x_1 + 2x_2 \ge -100$ באילוץ $-2x_1 - 2x_2 \le 100$.
סה״כ ניתן לכתוב את הבעיה בצורה הסטנדרטית:

$$\begin{aligned} \max \quad & c^T \cdot x \\ \text{subject to} \quad & Ax \le b \\ & x \ge 0 \end{aligned}$$

כאשר $c = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$, $x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, $A = \begin{pmatrix} 4 & -1 \\ -2 & -2 \end{pmatrix}$, $b = \begin{pmatrix} 3 \\ 100 \end{pmatrix}$.

### 32.1.3 אלגוריתמים לפתרון בעיית תכנון לינארי

קיימים אלגוריתמים יעילים (פולינומיאלים ב-$m + n$) שמאפשרים לפתור בעיית תכנון לינארי:

1. אלגוריתם הסימפלקס (Simplex Algorithm)
2. אלגוריתם האליפסואיד (Ellipsoid Algorithm)
3. האלגוריתם של קרמארקר (Karmarkar's Algorithm)

לא נתאר אף אלגוריתם שכן נדרשים כמה שיעורים שלמים על מנת להבין אותם.
ברמה האינטואיטיבית והלא מדויקת נאמר שהאילוצים השונים בבעיית תכנון לינארי מגדירים פוליהדרון ב-$\mathbb{R}^n$, והפתרון האופטימלי מתקבל ״בפינות״ של הפוליהדרון.
נתייחס לאלגוריתמים אלו כעל ״קופסה שחורה״ שמקבלת בעיית תכנון לינארי בצורה הסטנדרטית ומספקת פתרון חוקי (כלומר, כפוף לאילוצי הבעיה) שממקסם את הפונקציה הלינארית הנתונה בבעיה.

### 32.1.4 בעיית התרמיל השברית כבעיית תכנון לינארי

נשוב וניזכר בבעיית התרמיל השברית (בעיה 1.2).
ניתן להציג את הבעיה כבעיית תכנון לינארי (נשתמש בסימונים שהוגדרו בבעיה 1.2).
נסמן ב-$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ את וקטור הפתרון.
הפונקציה שאנו רוצים למקסם היא $v_1x_1 + \cdots + v_nx_n$ (השווי הכולל של הפריטים שלקחנו), בכפוף לאילוץ המשקל $w_1x_1 + \cdots + w_nx_n \le W$, ולאילוץ ״החלקיות״ של הפריט: לכל $1 \le i \le n$ חייב להתקיים $0 \le x_i \le 1$.
וקטור המקדמים $c = \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}$, ולכן הפונקציה שאנו רוצים למקסם היא $c^T \cdot x$.

את האילוצים השונים נרשום כך:

$$\begin{array}{llll} w_1 \cdot x_1 & +\cdots+ & w_n \cdot x_n & \leq W \\ 1 \cdot x_1 & +\cdots+ & 0 \cdot x_n & \leq 1 \\ \vdots & & & \\ 0 \cdot x_1 & +\cdots+ & 1 \cdot x_n & \leq 1 \end{array}$$

ניתן לראות כי אילוצים אלו מגדירים מטריצה $A$ ווקטור $b$ באופן הבא:

$$A = \begin{pmatrix} w_1 & w_2 & \dots & w_n \\ 1 & 0 & \dots & 0 \\ & & \ddots & \\ 0 & 0 & \dots & 1 \end{pmatrix}, b = \begin{pmatrix} W \\ 1 \\ \vdots \\ 1 \end{pmatrix}$$

וניתן לרשום אותם בכתיב מטריציוני כך:

$$Ax \leq b$$

לא נשכח את האילוץ $x_i \geq 0$ לכל $1 \leq i \leq n$, שניתן לרשום בפשטות $x \geq 0$.

### 32.1.5 בעיית התרמיל השלם כבעיית תכנון לינארי בשלמים

את בעיית התרמיל השלם ניתן לפרמל בדיוק באותה דרך כפי שעשינו עם בעיית התרמיל השברית, בתוספת אילוץ על שלמות המשתנים: $x \in \mathbb{Z}^n$.

פורמלית, בעיית תכנון לינארי בשלמים נכתבת בצורה הסטנדרטית כך:

$$\begin{array}{rl} \max & c^T \cdot x \\ \text{subject to} & Ax \leq b \\ & x \geq 0 \\ & x \in \mathbb{Z}^n \end{array}$$

נשים לב שהאילוץ על שלמות המשתנים אינו אילוץ לינארי. אי-לכך, לבעיה מסוג זה קוראים בעיית תכנון לינארי בשלמים (Integer Linear Programming או בקיצור: ILP).

בעיית תכנון לינארי בשלמים היא בעיה NP-קשה, ולא ידוע אלגוריתם הרץ בזמן פולינומיאלי לפתרונה.

## 32.2 אלגוריתמי קירוב - הגדרות

**הגדרה 32.5.** יהי $X$ מרחב הפתרונות החוקיים לבעיה. עבור פונקציה $f : X \to \mathbb{R}^+$ נסמן ב-$x^* \in X$ פתרון אופטימלי לבעיה.
יהי $c \geq 1$. נאמר שאלגוריתם הוא $c$**-מקרב**:

- עבור בעיית אופטימיזציה מהצורה $\min_{x \in X} f(x)$ אם <u>לכל קלט</u> הוא מחזיר פתרון חוקי $x \in X$ המקיים $f(x) \leq cf(x^*)$.
- עבור בעיית אופטימיזציה מהצורה $\max_{x \in X} f(x)$ אם <u>לכל קלט</u> הוא מחזיר פתרון חוקי $x \in X$ המקיים $f(x) \geq \frac{f(x^*)}{c}$.

**הערה 32.6.**

(1) נשים לב ש-$c \geq 1$. למשל, עבור בעיית מינימיזציה נרצה שהאלגוריתם המקרב יחזיר פתרון "שלא יהיה גדול מידי" מהפתרון האופטימלי.

(2) לעתים, נרשום OPT במקום $f(x^*)$.

(3) חשוב לשים לב שלפי ההגדרה האלגוריתם צריך להחזיר פתרון חוקי מקורב **לכל קלט** של הבעיה.

הוכחת נכונות לאלגוריתם $c$-מקרב תכיל:

1. הוכחת חוקיות הפתרון;
2. הוכחה שהפתרון אכן $c$-מקרב, בהתאם לסוג הבעיה (בעיית מינימיזציה או מקסימיזציה).

**אסטרטגיה כללית:** בבעיית <u>מקסימיזציה</u> נרצה למצוא חסם מלעיל $D$ על ערך הפתרון האופטימלי, כלומר $f(x^*) \leq D$.
לאחר מכן, נוכיח שלכל קלט, האלגוריתם המקרב מחזיר פתרון חוקי $x$ המקיים $\frac{1}{c}D \leq f(x)$.
מכאן ינבע $f(x) \geq \frac{1}{c}D \geq \frac{1}{c}f(x^*)$.
בבעיית <u>מינימיזציה</u> נרצה למצוא חסם תחתון $D$ על ערך הפתרון האופטימלי, כלומר $f(x^*) \geq D$.
לאחר מכן נוכיח שלכל קלט האלגוריתם המקרב מחזיר פתרון חוקי $x$ המקיים $f(x) \leq c \cdot D$.
מכאן ינבע $f(x) \leq c \cdot D \leq c \cdot f(x^*)$.

## 32.3 בעיית ה-3SAT

נפתח במספר הגדרות הדרושות להגדרת הבעיה.

**הגדרה 32.7.** <u>משתנה בוליאני</u> $x$ מקיים $x \in \{\mathbb{T}, \mathbb{F}\}$, כאשר $\mathbb{T}$ מציין ערך אמת "אמת" ו-$\mathbb{F}$ מציין ערך אמת "שקר".
<u>ליטרל</u> הוא משתנה בוליאני $x$ או שלילתו $\bar{x}$.
<u>פסוקית 3CNF</u> מורכבת מ-3 ליטרלים עם קשר "או" ביניהם. דוגמה לפסוקית 3CNF: $(x \vee \bar{y} \vee \bar{z})$.
<u>נוסחת 3CNF</u> מורכבת מ-$m$ פסוקיות 3CNF עם קשר "וגם" ביניהם. דוגמה לנוסחת 3CNF: $(x \vee \bar{y} \vee \bar{z}) \wedge (x \vee z \vee w)$.

**בעיה 32.8 (בעיית ה-3SAT).**
<u>קלט:</u> נוסחת 3CNF בעלת $m$ פסוקיות 3CNF, $C_1, \ldots, C_m$ מעל $n$ משתנים בוליאניים $x_1, \ldots, x_n$.
<u>פלט:</u> האם קיימת השמה למשתנים $x_1, \ldots, x_n$ שמספקת את הנוסחה (=נותנת ערך $\mathbb{T}$ לנוסחה).
<u>הנחות על הקלט:</u>

- המשתנים המשתתפים בכל פסוקית שונים זה מזה.
- כל המשתנים $x_1, \ldots, x_n$ מופיעים בנוסחה.

זוהי דוגמה לבעיית הכרעה NP-קשה.
על-מנת להחזיר אלגוריתם מקרב לבעיה, ננסח <u>פלט אלטרנטיבי</u> (לבעיה האלטרנטיבית קוראים Max 3SAT):
השמה ל-$x_1, \ldots, x_n$ שממקסמת את מספר הפסוקיות המסופקות.

**הערה 32.9.** נשים לב שאם מספר הפסוקיות המסופקות המקסימלי שווה $m$ אז מצאנו השמה למשתנים $x_1, \ldots, x_n$ שמספקת את הנוסחה ומצאנו פתרון לבעיה המקורית.

נציע אלגוריתם 2-מקרב לבעיה Max 3SAT:

**אלגוריתם 14** אלגוריתם 2-מקרב לבעיה Max 3SAT

1. נתבונן בהשמה $\vec{x}_{\mathbb{T}} = (\mathbb{T}, \ldots, \mathbb{T})$ ונבדוק כמה פסוקיות השמה זו מספקת. נסמן מס' זה ב-$t$.
2. נתבונן בהשמה $\vec{x}_{\mathbb{F}} = (\mathbb{F}, \ldots, \mathbb{F})$ ונבדוק כמה פסוקיות השמה זו מספקת. נסמן מס' זה ב-$f$.
3. אם $f < t$ נחזיר את ההשמה $\vec{x}_{\mathbb{T}}$; אחרת, נחזיר את ההשמה $\vec{x}_{\mathbb{F}}$.

**זמן ריצה:** $O(m)$ - בדיקת כל השמה דורשת מעבר אחד על הנוסחה, ואנחנו בודקים בסה"כ 2 השמות.

**הוכחת נכונות:**

**חוקיות** פתרון חוקי זו השמה כלשהי על כל המשתנים, ואנו אכן מחזירים השמה כזו.

**2-מקרב** תהי נוסחה 3CNF עם $m$ פסוקיות $C_1, \ldots, C_m$. נרצה להוכיח שערך הפתרון שמוחזר ע"י האלגוריתם שהצענו הוא לפחות $\frac{\text{OPT}}{2}$.
לכל פסוקית $C_i$ - או ש- $\vec{x}_{\mathbb{F}}$ מספק אותה או ש-$\vec{x}_{\mathbb{T}}$ מספק אותה (או שניהם). לכן,

$$t + f \geq m$$

מצד שני, עבור כל שני מספרים $t, f$ מתקיים $t + f \leq 2\max\{t, f\}$.

מכאן קיבלנו

$$\max\{t,f\} \geq \frac{m}{2}$$

כמובן שמתקיים $m \geq \text{OPT}$ (מס' הפסוקיות המקסימלי שמסופקות ע"י השמה כלשהי למשתנים $x_1,\ldots,x_n$ הוא לכל היותר מספר הפסוקיות).
נבחין כי לפי הגדרת האלגוריתם $\max\{t,f\}$ הוא ערך הפתרון שמוחזר ע"י האלגוריתם שהצענו.
מכאן נקבל

$$\max\{t,f\} \geq \frac{m}{2} \geq \frac{\text{OPT}}{2}$$

ומכאן נובע, ע"פ הגדרה, שהאלגוריתם הוא 2-מקרב. □

**טכניקה כללית להראות שאלגוריתם הוא 2-מקרב עבור בעיית מקסימיזציה:** נסמן $x^*$ בתור פתרון אופטימלי לבעיה. יהיו $x$ ו-$y$ פתרונות חוקיים לבעיה.
אם נוכל להראות כי $f(x)+f(y) \geq f(x^*)$, אז מכך ש-$f(x)+f(y) \leq 2\max\{f(x),f(y)\}$ נקבל

$$\max\{f(x),f(y)\} \geq \frac{f(x^*)}{2}$$

זאת בהנחה שערך הפתרון שמוחזר ע"י האלגוריתם המקרב הוא $\max\{f(x),f(y)\}$.

## 32.4 בעיית התרמיל השלם - אלגוריתם מקרב

ניזכר בבעיית התרמיל השלם 10.1.
הרעיון הנאיבי הוא להשתמש בכלל חמדני שמסדר את הפריטים לפי ערך סגולי, ובכל שלב לקחת את הפריט בעל הערך הסגולי הגדול יותר. ואמנם, אלגוריתם חמדני כזה אינו 2-מקרב לבעיית התרמיל השלם.

**דוגמה נגדית:** נגדיר שני פריטים כאשר הפריט הראשון הוא $(v_1=1,w_1=1)$ עם ערך סגולי $r_1=\frac{1}{1}=1$.
הפריט השני הוא $(v_2=W-1,w_2=W)$ עם ערך סגולי $r_2=\frac{W-1}{W}$.
האלגוריתם החמדן יחזיר $G=(1,0)$, כלומר, יעדיף את הפריט הראשון על פני הפריט השני.
אולם, הפתרון האופטימלי הוא: $x^*=(0,1)$.
נסמן ב-$f$ את פונקציית הערך, אשר מחזירה את הערך של הפריטים בתרמיל לפי הפתרון המוצע.
אזי $f(G)=1$ ו-$f(x^*)=W-1$, ומתקיים:

$$f(G)=\frac{1}{W-1}f(x^*)$$

עבור $W>3$ אלגוריתם זה אינו 2-מקרב.

**פתרון 2-מקרב:** ניזכר בהגדרת הפתרון של בעיית התרמיל השברית 1.3, ספציפית בהגדרה של האינדקס $t$.
נגדיר שני פתרונות $\vec{x}$ ו-$\vec{y}$ ונחזיר את הפתרון שמחזיר את הערך המקסימלי.

$$\vec{x}=\left(\overbrace{1,1,\ldots,1,}^{t}\overbrace{0}^{t+1},0,\ldots,0\right)$$

$$\vec{y}=\left(0,0,\ldots,0,\underbrace{1}_{t+1},0,\ldots 0\right)$$

נגדיר $z^*$ בתור הפתרון האופטימלי של בעיית התרמיל השברית (בעיה 1.2).
נגדיר $x^*$ בתור הפתרון האופטימלי של בעיית התרמיל השלם.
אזי מתקיים:

$$f(x^*) \leq f(z^*)$$

כיוון ש-$x^*$ הוא פתרון <u>חוקי</u> גם עבור בעיית התרמיל השברית, והרי $z^*$ הוא הפתרון <u>האופטימלי</u> (=מקסימלי) עבור בעיית התרמיל השברית.

כעת, נבחין שקיים $0 \leq \alpha < 1$ עבורו

$$f\left(z^{*}\right)=f\left(\vec{x}\right)+\alpha\cdot f\left(\vec{y}\right)$$

זאת משום ש-$f\left(\vec{y}\right)$ זה בדיוק הערך של הפריט ה-$t+1$. מכאן:

$$f\left(z^{*}\right)=f\left(\vec{x}\right)+\alpha\cdot f\left(\vec{y}\right)\leq f\left(\vec{x}\right)+f\left(\vec{y}\right)\leq 2\max\left\{f\left(\vec{x}\right),f\left(\vec{y}\right)\right\}$$

ולבסוף, מחיבור האי-שוויונים הללו

$$2\max\left\{f\left(\vec{x}\right),f\left(\vec{y}\right)\right\}\geq f\left(z^{*}\right)\geq f\left(x^{*}\right)$$

וקיבלנו אלגוריתם 2-מקרב לבעיית התרמיל השלם. $\square$

# 33 תרגול 7 - 26.11.18

## 33.1 בעיית החתך המקסימלי (Max Cut)

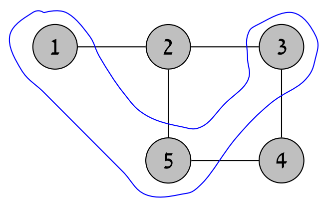
**הגדרה 33.1** (חתך בגרף)**.** יהי $G=(V,E)$ גרף לא מכוון. **חתך בגרף** $(A,B)$ הוא חלוקה של קבוצת הקודקודים בגרף לשתי קבוצות זרות של קודקודים $A$ ו-$B$, כלומר, $V=A\sqcup B$ (כאשר $\sqcup$ מציין איחוד זר).

**בעיה 33.2** (בעיית החתך המקסימלי)**.**
קלט: גרף לא מכוון $G=(V,E)$.
פלט: חתך $(A,B)$ כך שמספר הצלעות $\{x,y\}\in E$ עבור $x\in A$ ו-$y\in B$ (או $x\in B$ ו-$y\in A$) יהיה מקסימלי. כלומר, מספר הצלעות שחוצות את החתך יהיה מקסימלי.

זוהי בעיה NP-קשה, ונחפש עבורה אלגוריתם מקרב.

**דוגמה 33.3.** נתבונן בגרף שקבוצת קודקודיו $V=\{1,\ldots,5\}$ וצלעותיו $E=\{\{1,2\},\{2,3\},\{3,4\},\{4,5\},\{5,2\}\}$.
דוגמאות לחתכים:

- $A=\{1,5\}$, $B=\{2,3,4\}$. מספר הצלעות שחוצות את החתך הוא 3.
- $A=\{1,5,3\}$, $B=\{2,4\}$. כל הצלעות בגרף חוצות את החתך, לכן זה חתך מקסימלי. (באיור הוא מסומן בכחול)




נציע אלגוריתם 2-מקרב לבעיה.

**אלגוריתם 15** אלגוריתם 2-מקרב לבעיית החתך המקסימלי

נמספר את הקודקודים $V=\{v_1,\ldots,v_n\}$.

1. אתחול: נתחיל בחלוקה טריוויאלית $A=V$, $B=\emptyset$.
2. נעבור על כל הקודקודים לפי סדר המספור, ונבדוק את התנאי הבא עבור כל קודקוד $v_i$:
   אם מספר השכנים של $v_i$ שנמצאים בקבוצה שאליה $v_i$ שייך גדול ממספר השכנים של $v_i$ השייכים לקבוצה השניה, נעביר את $v_i$ לקבוצה השניה.
3. נחזור על שלב 2 שוב ושוב, ונעצור כאשר לא יישארו קודקודים להעביר לפי התנאי הנ"ל.

**הערה 33.4.** עבור הגרף מדוגמה 33.3, האלגוריתם שהגדרנו יבצע 2 איטרציות עד שיעצור, ויחזיר את $A=\{1,3,5\}$ ו-$B=\{2,4\}$.

מדוע אלגוריתם 15 עוצר?
האבחנה המרכזית היא שבכל העברה של קודקוד מקבוצה אחת לשניה, מספר הצלעות שחוצות את החתך גדל בלפחות 1.
מספר הצלעות החוצות את החתך חסום ע"י מספר הצלעות בגרף $|E|$, ולכן האלגוריתם בהכרח עוצר (לאחר לכל היותר $|E|$ איטרציות).

**זמן ריצה:** שלב האתחול לוקח $O(|V|)$. מתבצעות לכל היותר $|E|$ איטרציות של שלב 2, ובכל ביצוע של שלב 2 אנו עוברים על כל הקודקודים וכל הצלעות, ולכן נקבל

$$\underbrace{O(|V|)}_{\text{אתחול}}+O\left(|E|\cdot\left(\underbrace{|V|+|E|}_{\text{מעבר על הגרף}}\right)\right)=O\left(|E|\cdot(|V|+|E|)\right)$$

**הוכחת נכונות:**

**חוקיות** האלגוריתם מתחיל עם חתך חוקי, ובכל שלב מעביר קודקודים מקבוצה אחת לשנייה $\Leftarrow$ האלגוריתם שומר על חתך חוקי לכל אורכו.

**2-קירוב** זוהי בעיית מקסימיזציה ולכן נחסום את ערך הפתרון האופטימלי מלעיל. במקרה שלנו, החסם פשוט: $\mathrm{OPT} \leq |E|$ (מספר הצלעות שחוצות את החתך המקסימלי הוא לכל היותר מספר הצלעות בגרף).
נרצה להראות שמספר הצלעות שחוצות את החתך שבנינו גדול-שווה $\frac{1}{2}|E|$.
יהי $C = (A, B)$ החתך שמחזיר האלגוריתם שלנו.
נסמן לכל קודקוד $v \in V$ ב-$d(v)$ את דרגתו (מספר שכניו); נסמן ב-$v_C$ את קבוצת הצלעות שנוגעות ב-$v$ וחוצות את החתך.
האבחנה המרכזית היא שלכל $v \in V$ מתקיים $|v_C| \geq \frac{1}{2}d(v)$, היות שהאלגוריתם עוצר כאשר עבור כל קודקוד מתקיים שמספר שכניו בקבוצה שאליה הוא שייך קטן-שווה ממספר שכניו בקבוצה השנייה:

$$|v_C| = \text{מספר השכנים בקבוצה השנייה} \geq \text{מספר השכנים בקבוצה שלו} \implies 2|v_C| \geq \text{מספר השכנים בקבוצה השנייה} + \text{מספר השכנים בקבוצה שלו} = d(v)$$

מספר הצלעות שחוצות את החתך $C$ שווה בדיוק ל-$\frac{1}{2}\sum_{v \in V}|v_C|$ - אנו מכפילים ב-$\frac{1}{2}$ כי כל צלע שחוצה את החתך נספרה פעמיים בסכום.
מכאן נובע:

$$\begin{aligned} \text{מספר הצלעות שחוצות את החתך } C &= \frac{1}{2}\sum_{v \in V}|v_C| \\ &\geq \frac{1}{2}\sum_{v \in V}\left(\frac{1}{2}d(v)\right) \\ &= \frac{1}{4}\sum_{v \in V}d(v) \\ \left[\text{למת לחיצת הידיים}\right] &= \frac{1}{4} \cdot 2|E| = \frac{1}{2}|E| \\ &\geq \frac{1}{2}\mathrm{OPT} \end{aligned}$$

ובכך סיימנו את ההוכחה. $\square$

## 33.2 בעיית Max 3-SAT

נשוב וניזכר בבעיה 32.8, שהייתה בעיית הכרעה NP-קשה, והצגנו אלגוריתם 2-מקרב לבעיית אופטימיזציה שקולה Max 3SAT.
הפעם, נציג אלגוריתם $\frac{8}{7}$-מקרב הסתברותי לבעיית ה-Max 3-SAT.

**מה זה אלגוריתם $c$-מקרב הסתברותי?** הדרישה מאלגוריתם $c$-מקרב הסתברותי היא שעבור $k \in \mathbb{N}$ נתון ולכל קלט הוא יחזיר פלט חוקי וגם $c$-מקרב בהסתברות גדולה-שווה $1 - \frac{1}{e^k}$.

---

**אלגוריתם 16** אלגוריתם $\frac{8}{7}$-מקרב הסתברותי לבעיית ה-Max 3-SAT

---

**אלגוריתם בסיסי:**

1. לכל משתנה $x_i$ נטיל מטבע הוגן:
   (א) אם יצא עץ, נגדיר $x_i = \mathbb{T}$;
   (ב) אם יצא פלי, נגדיר $x_i = \mathbb{F}$.
2. נחזיר את ההשמה אם היא מספקת לפחות $\frac{7}{8}m$ פסוקיות, אחרת נחזיר fail.

**אלגוריתם כללי:**

1. נריץ את האלגוריתם הבסיסי $k \cdot (m+1)$ פעמים: אם באחת מהפעמים לא קיבלנו fail נחזיר את ההשמה; אחרת - נחזיר fail.

---

**זמן ריצה:** עלות האלגוריתם הבסיסי הינה $O(n+m)=O(m)$, כי $n$ הוא מספר המשתנים (ולכן מספר הטלות המטבע) ו-$m$ הוא מספר הפסוקיות. וגם מתקיים $n\leq 3m$ (היזכרו בבעיה 32.8).
באלגוריתם הכללי, אנו חוזרים על האלגוריתם הבסיסי $k\cdot(m+1)$ פעמים, כאשר $k$ קבוע, ולכן זמן הריצה הכולל הינו:

$$O(k\cdot(m+1)\cdot m)=O\left(m^2\right)$$

**הוכחת נכונות:**

**חוקיות** נשים לב שאם האלגוריתם לא מחזיר fail הוא מחזיר השמה (=פתרון חוקי).

**$\frac{8}{7}$-קירוב** אם האלגוריתם לא מחזיר fail הוא מחזיר השמה שמספקת לפחות $\frac{7}{8}m$ פסוקיות, ולכן הוא בבירור $\frac{8}{7}$ מקרב, משום ש-$m\geq \mathrm{OPT}$ וזה גורר $\frac{7}{8}m\geq\frac{7}{8}\mathrm{OPT}$.
נותר להוכיח כי ההסתברות להחזיר השמה (טובה) כזו היא גבוהה מספיק, כלומר לפחות $1-\frac{1}{e^k}$, עבור ה-$k$ הקבוע שבחרנו.

**טענה 33.5.** סיכוי ההצלחה של האלגוריתם הבסיסי הוא לפחות $\frac{1}{m+1}$.

הוכחה. בשלב הראשון, נראה שתוחלת מס' הפסוקיות המסופקות בהשמה הוא $\frac{7}{8}m$.
נגדיר מרחב מדגם $\Omega=\{\mathbb{F},\mathbb{T}\}^n$, מרחב כל ההשמות החוקיות.
נגדיר מ"מ $X:\Omega\to\mathbb{N}$, כאשר לכל $\omega\in\Omega$, $X(\omega)$ מציין את מספר הפסוקיות המסתפקות ע"י ההשמה $\omega$.
נרצה לחשב את $\mathbb{E}[X]$.
נגדיר לכל $1\leq i\leq m$ משתנה מקרי אינדיקטור באופן הבא:

$$X_i(\omega)=\begin{cases}1 & \omega \text{ מספקת את הפסוקית ה- } i\\ 0 & \text{אחרת}\end{cases}$$

נבחין כי מתקיים $X=\sum_{i=1}^{m}X_i$.
עתה,

$$\mathbb{E}[X_i]=P(X_i=1)=P\left(\substack{\text{השמה מספקת}\\ \text{פסוקית ה-}i}\right)=1-\left(\frac{1}{2}\right)^3=\frac{7}{8}$$

כי כדי שפסוקית לא תסופק ע"י השמה מסוימת צריך שכל הליטרלים יקבלו ערך $\mathbb{F}$ וזה קורה בהסתברות $\left(\frac{1}{2}\right)^3=\frac{1}{8}$.
מלינאריות התוחלת –

$$\mathbb{E}[X]=\mathbb{E}\left[\sum_{i=1}^{m}X_i\right]=\sum_{i=1}^{m}\mathbb{E}[X_i]=\frac{7}{8}m$$

נגדיר משתנה מקרי $Y:\Omega\to\mathbb{R}$ ע"י $Y=m-X$. לכל $\omega\in\Omega$, $Y(\omega)$ סופר כמה פסוקיות לא סופקו ע"י ההשמה $\omega$. אזי –

$$Y=m-X\implies\mathbb{E}[Y]=m-\mathbb{E}[X]=\frac{1}{8}m$$

ניעזר באי-שוויון מרקוב כדי לחסום את ההסתברות שהאלגוריתם הבסיסי נכשל:

$$\begin{aligned}P\left(\substack{\text{אלגוריתם}\\ \text{בסיסי}\\ \text{נכשל}}\right)&=P\left(X<\frac{7}{8}m\right)=P\left(m-Y<\frac{7}{8}m\right)\\&=P\left(Y>\frac{1}{8}m\right)=P\left(Y\geq\frac{1}{8}(m+1)\right)\\&=P\left(Y\geq\frac{1}{8}m\left(1+\frac{1}{m}\right)\right)=P\left(Y\geq\mathbb{E}[Y]\left(1+\frac{1}{m}\right)\right)\\&\underset{[\text{אי-שוויון מרקוב}]}{\leq}\frac{1}{1+{}^1\!/_m}=\frac{m}{m+1}\end{aligned}$$

ולכן $\frac{1}{m+1}\geq P\left(\substack{\text{אלגוריתם}\\ \text{בסיסי}\\ \text{מצליח}}\right)$, כנדרש. $\square$

**מסקנה 33.6.** $P\left(\substack{\text{אלגוריתם}\\ \text{כללי}\\ \text{מצליח}}\right)\geq 1-\frac{1}{e^k}$.

הוכחה. נחסום מלמעלה את ההסתברות שהאלגוריתם הכללי נכשל (ההסתברות המשלימה).

$$P\begin{pmatrix}\text{האלגוריתם}\\\text{הכללי}\\\text{נכשל}\end{pmatrix} = P\begin{pmatrix}\text{האלגוריתם הבסיסי}\\\text{נכשל}\\k\ (m+1)\ \text{פעמים}\end{pmatrix} = P\begin{pmatrix}\text{אלגוריתם}\\\text{בסיסי}\\\text{נכשל}\end{pmatrix}^{k(m+1)}$$
$$\leq \left(\left(1-\frac{1}{m+1}\right)^{m+1}\right)^{k} < \left(\frac{1}{e}\right)^{k} = \frac{1}{e^k}$$

ומכאן נובעת המסקנה. □

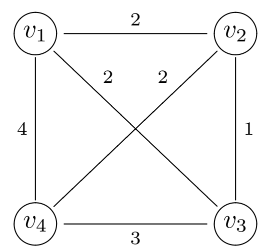
# 34 תרגול 8 - 3.12.18

## 34.1 בעיית הסוכן הנוסע (Traveling Salesman Problem)

**בעיה 34.1** (בעיית הסוכן הנוסע).
קלט: גרף לא-מכוון $G=(V,E)$ **מלא** (כלומר, קיימת צלע לכל זוג קודקודים); פונקציית משקל $w:E\to\mathbb{R}^+$.
פלט: מעגל פשוט $C=\{e_1,\ldots,e_m\}$ שעובר בכל קודקודי הגרף, ובעל משקל מינימלי, כאשר משקל של מעגל מוגדר ע״י

$$w(C):=\sum_{e\in C}w(e)$$

**דוגמה 34.2.** נתבונן בגרף השלם על 4 קודקודים, $K_4$ –




דוגמה למעגל פשוט: $C_1=\{(v_1,v_2),(v_2,v_3),(v_3,v_4),(v_4,v_1)\}$, שמשקלו $w(C_1)=2+1+3+4$.
דוגמה למעגל פשוט נוסף: $C_2=\{(v_1,v_2),(v_2,v_4),(v_4,v_3),(v_3,v_1)\}$, שמשקלו $w(C_2)=2+2+3+2=9$ - זה הפתרון האופטימלי.

לא רק שבעיית הסוכן הנוסע היא בעיה NP-קשה, כי אם גם בעיה קשה לקירוב.
לכן, נתבונן בגרסה ״קלה״ יותר שלה, ונקרב אותה. נציג את בעיית הסוכן הנוסע המטרית MTSP (Metric TSP).

**בעיה 34.3** (בעיית הסוכן הנוסע המטרית).
קלט: גרף לא-מכוון $G=(V,E)$ מלא; פונקציית משקל $w:E\to\mathbb{R}^+$ המקיימת את אי-שוויון המשולש

$$\forall i,j,k\quad w(i,j)\le w(i,k)+w(k,j)$$

פלט: מעגל פשוט $C=\{e_1,\ldots,e_m\}$ שעובר בכל קודקודי הגרף, ובעל משקל מינימלי.

**אלגוריתם לפישוט מעגלים** נציג אלגוריתם שמקבל מעגל לא פשוט שעובר בכל קודקודי הגרף ומחזיר מעגל פשוט שעובר בכל קודקודי הגרף.
ניעזר באלגוריתם זה בהמשך, כאשר נציג את האלגוריתם המקרב לבעיה 34.3.

---

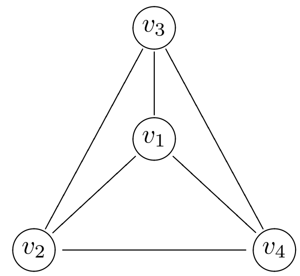
**אלגוריתם 17** אלגוריתם לפישוט מעגלים

קלט: מעגל לא פשוט $C'=\{(v^0,v^1),(v^1,v^2),\ldots,(v^k,v^0)\}$ שעובר בכל קודקודי הגרף. (האינדקס למעלה שונה מהאינדקס למטה; למשל, לא בהכרח מתקיים $v^1=v_1$).
פלט: מעגל פשוט $\hat{C}$ אשר עובר בכל קודקודי הגרף.

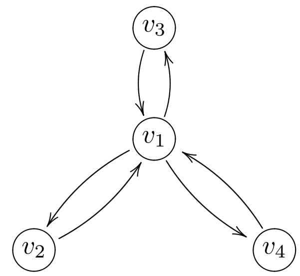
1. $last\leftarrow v^0$
2. נאתחל את $visited$: מערך בגודל $|V|$ עם ערכי אמת False שמציין האם ביקרנו בקודקוד או לא.
3. $visited[v^0]=\text{True}$
4. עבור $i=1,\ldots,k$, נבדוק האם $visited[v^i]=\text{False}$ ובמקרה זה נבצע:
   (א) נוסיף ל-$\hat{C}$ את הקשת $\{last,v^i\}$ (אפשרי כי הגרף $G$ הוא גרף מלא)
   (ב) $visited[v^i]=\text{True}$
   (ג) $last\leftarrow v^i$
5. נוסיף ל-$\hat{C}$ את הקשת $\{last,v^0\}$, ונחזיר את $\hat{C}$.

---

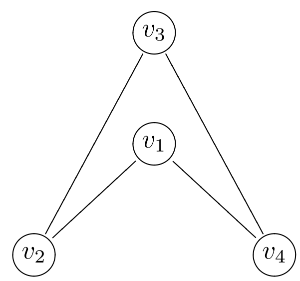
**דוגמה 34.4.** נתבונן בגרף השלם על 4 קודקודים:




נתבונן במעגל הלא פשוט $C' = \{\{v_1, v_2\}, \{v_2, v_1\}, \{v_1, v_3\}, \{v_3, v_1\}, \{v_1, v_4\}, \{v_4, v_1\}\}$.




המעגל הפשוט $\hat{C}$ שיחזיר אלגוריתם 17 מתואר באיור הבא:




כאשר הצלע $\{v_4, v_1\}$ היא הצלע האחרונה שמתווספת כחלק משלב 5 באלגוריתם.

**זמן ריצה:** שלבים 1, 3 ו-5 לוקחים $O(1)$; שלב 2 לוקח $O(|V|)$; שלב 4 לוקח $O(k)$, שהרי בכל איטרציה בלולאה אנו מבצעים מספר פעולות קבוע.
זמן הריצה הכולל: $O(|V| + k)$.

**טענה 34.5** (נכונות האלגוריתם לפישוט מעגלים)**.** אלגוריתם 17 מחזיר מעגל פשוט $\hat{C}$ שעובר בכל קודקודי הגרף.

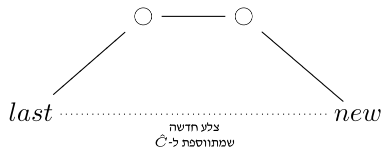
הוכחה. בשלב 4, האלגוריתם עובר בלולאה על כל הקודקודים במעגל $C'$, ומשום שהמעגל $C'$ מכיל את כל קודקודי $G$ (חלק מהנחות הקלט), בשלב 4 האלגוריתם עובר על כל קודקודי $G$. לכל קודקוד $v^i$, $i \neq 0$, שלא ביקרנו בו, האלגוריתם מוסיף צלע שנכנסת אליו $\{last, v^i\}$, ובנוסף, $last$ מתעדכן להיות $v^i$.
מכאן ששלב 4 מסתיים בהכרח עם מסלול פשוט מ-$v^0$ ל-$last$ שעובר בכל קודקודי $G$. הוספת הצלע $\{last, v^0\}$ כחלק משלב 5 סוגרת מעגל פשוט, שעובר בכל קודקודי $G$. $\square$

**טענה 34.6.** יהי $G = (V, E)$ גרף מלא לא-מכוון, ותהי $w : E \to \mathbb{R}^+$ פונקציית משקל המקיימת את אי-שוויון המשולש.
יהי $C'$ מעגל לא פשוט שעובר בכל קודקודי הגרף ויהי $\hat{C}$ המעגל הפשוט שמחזיר אלגוריתם 17.
<u>אזי</u> $w\left(\hat{C}\right) \leq w(C')$.

הוכחה. נשים לב שבכל הוספת צלע $\{last, new\}$ ל-$\hat{C}$, אנו בעצם "מחליפים" קטע ב-$C'$ באורך $\ell$ מהצורה

$$\{last, v^i\}, \{v^i, v^{i+1}\}, \ldots, \{v^{i+\ell-1}, new\}$$

כאשר $new$ הוא הקודקוד הראשון שטרם ביקרנו בו עד כה. להלן המחשה:




נבחין בין שני מקרים:

- $\ell = 0$: במקרה זה אנו מוסיפים ל-$\hat{C}$ את הצלע $\{last, new\}$ אשר שייכת גם ל-$C'$ (כלומר הקודקוד הבא שלא ביקרנו בו הוא $new$ שבא בדיוק אחרי $last$).
- $\ell > 0$: במקרה זה אנו מוסיפים ל-$\hat{C}$ את הצלע $\{last, new\}$ כתחליף לקטע: $\{last, v^i\}, \ldots, \{v^{i+\ell-1}, new\}$. מהפעלת אי-שוויון המשולש $\ell$ פעמים נקבל כי

$$w\left(last, new\right) \leq w\left(last, v^i\right) + \cdots + w\left(v^{i+\ell-1}, new\right)$$

מכאן נסיק $w\left(\hat{C}\right) \leq w\left(C'\right)$, שהרי בכל פעם שאנו מחליפים מקטע ב-$C'$ לקטע ב-$\hat{C}$ אנו שומרים על המשקל (המקרה הראשון) או לא מגדילים אותו (המקרה השני). $\square$

כעת, יש בידינו את הכלים להצגת אלגוריתם 2-מקרב לבעיית ה-MTSP.

---

**אלגוריתם 18** אלגוריתם 2-מקרב לבעיית ה-MTSP

---

1. נמצא עץ פורש מינימלי $T$ בגרף $G$.
2. נריץ שגרת DFS על $T$: סדר המעבר של אלגוריתם ה-DFS על הקודקודים של $T$ יגדיר מעגל לא פשוט, $C'$, שעובר בכל קודקודי $G$.
3. נפעיל את אלגוריתם 17 (האלגוריתם לפישוט מעגלים) על המעגל $C'$, ונחזיר את הפלט שלו, המעגל הפשוט $\hat{C}$.

---

**זמן ריצה:** נוכל למצוא עפ״מ ע״י שימוש באלגוריתם של Kruskal, וזה לוקח $O\left(|E| \log |E|\right)$.
הרצת DFS על העץ הפורש המינימלי תיקח $O\left(|V|\right)$ (שהרי מספר הצלעות בעץ פורש הוא $|V| - 1$).
הרצת האלגוריתם לפישוט מעגלים חסומה אף היא ע״י $O\left(|V|\right)$ (המעגל הפשוט $C'$ מבקר בכל צלע של העפ״מ פעמיים, מאופן פעולת אלגוריתם DFS).
סה״כ קיבלנו שזמן הריצה הכולל הינו: $O\left(|E| \log |E|\right)$.

**הוכחת נכונות:** חוקיות: מנכונות האלגוריתם לפישוט מעגלים, $\hat{C}$ הוא מעגל פשוט שעובר בכל קודקודי $G$.
נכונות הקירוב: ראשית, מאופן פעולת אלגוריתם DFS, אנו מקבלים שהמעגל הפשוט $C'$ מבקר בדיוק פעמיים בכל צלע ב-$T$.
לכן, $w\left(C'\right) = 2w\left(T\right)$.
לפי טענה 34.6 מתקיים $w\left(\hat{C}\right) \leq w\left(C'\right)$, ולכן $w\left(\hat{C}\right) \leq 2w\left(T\right)$.
יהי $C^*$ פתרון אופטימלי לבעיה. מחוקיותו, $C^*$ הוא מעגל פשוט שעובר בכל קודקודי הגרף.
יהי $T'$ העץ הפורש המתקבל מהורדת צלע בודדת כלשהי מ-$C^*$. אזי מתקיים –

$$w\left(C^*\right) \geq w\left(T'\right)$$

$T$ הוא עץ פורש מינימלי ב-$G$, ו-$T'$ עץ פורש כלשהו, ולכן –

$$w\left(T'\right) \geq w\left(T\right)$$

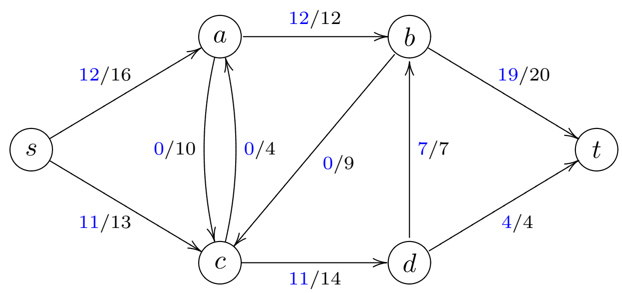
מכאן קיבלנו

$$w\left(\hat{C}\right) \leq w\left(C'\right) = 2w\left(T\right) \leq 2w\left(T'\right) \leq 2w\left(C^*\right)$$

כנדרש. $\square$

## 34.2 רשתות זרימה - הגדרות

נפתח בדוגמה של רשת זרימה.




איור 34.1 : דוגמה לרשת זרימה. בכחול מתוארים ערכי הזרימה. בזרימה המתוארת העברנו $23 = 12 + 11$ "חומר" מקודקוד $s$ לקודקוד $t$. נקודה חשובה היא שלא ניתן להעביר יותר חומר ברשת, שהרי הגענו לרוויה בקשתות $(a,b)$, $(d,b)$, $(d,t)$.

**הגדרה 34.7** (רשת זרימה). **רשת זרימה** היא חמישייה $N = (V,E,c,s,t)$ כאשר

- $G = (V,E)$ גרף מכוון
- $c : E \to \mathbb{R}^+$ פונקציית קיבול (חיובית ממש)
- $s \in V$ קודקוד מקור ("יצרן חומר")
- $t \in V$ קודקוד בור ("סופג חומר")

**הגדרה 34.8** (זרימה חוקית ברשת זרימה). **זרימה חוקית ברשת זרימה** היא פונקציה $f : E \to \mathbb{R}^+$ המקיימת שני אילוצים :

1. אילוץ הקיבול : לכל $e \in E$ מתקיים $f(e) \le c(e)$. ובמילים : הזרימה בכל צלע אינה גדולה מהקיבול של הצלע.
2. חוק שימור החומר : לכל $x \in V \setminus \{s,t\}$ :

$$\sum_{u:(u,x)\in E} f(u,x) = \sum_{v:(x,v)\in E} f(x,v)$$

ובמילים : הזרימה הנכנסת לקודקוד שווה לזרימה היוצאת מהקודקוד, לכל קודקוד פרט לקודקוד המקור וקודקוד הבור.

**הגדרה 34.9** (שטף). **השטף** של זרימה $f$ מוגדר להיות סכום ערכי הזרימה שיוצאים מקודקוד $s$, כלומר

$$|f| \coloneqq \sum_{u:(s,u)\in E} f(s,u)$$

**בעיית הזרימה:**
קלט : רשת זרימה $N = (V,E,c,s,t)$.
פלט : זרימה חוקית $f$ ברשת הזרימה הנתונה $N$, בעלת שטף $|f|$ מקסימלי.

**הערה 34.10.** בתרגולים ניישם בעיות אופטימיזציה באמצעות רשתות זרימה.

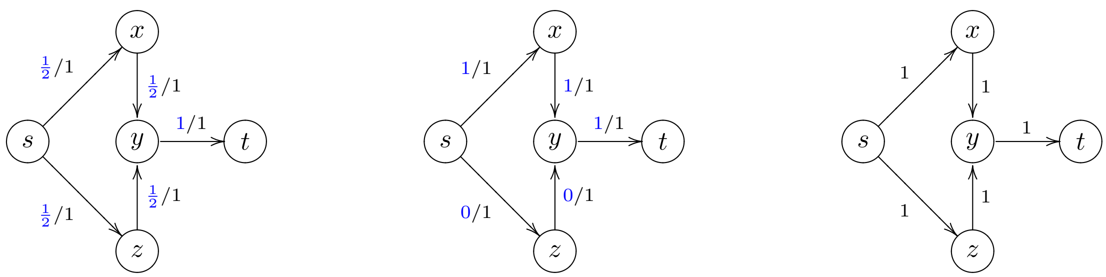
# 35 תרגול 9 - 17.12.18

## 35.1 הקדמה

במהלך התרגול אנו נניח כי יש בידינו אלגוריתם אדמונדס-קארפ (בקיצור: EK) שבהינתן רשת זרימה $N=(V,E,c,s,t)$ מחזיר זרימה משטף מקסימלי בזמן $O\left(|V|\,|E|^2\right)$.

נציין שדרך הפעולה של האלגוריתם מבטיחה כי אם פונקציית הקיבול היא שלמה $c:E\to\mathbb{N}$ אזי הזרימה שתוחזר מהאלגוריתם גם היא שלמה $f:E\to\mathbb{N}\cup\{0\}$.

אין זה אומר שכל זרימה משטף מקסימלי היא בשלמים, התבוננו באיור 35.1.




(א) רשת זרימה (ב) זרימה משטף מקסימלי שמוחזרת ע"י אלגוריתם EK (ג) זרימה משטף מקסימלי שאיננה בשלמים

איור 35.1: דוגמה הממחישה שזרימה משטף מקסימלי ברשת זרימה עם פונקציית קיבול שלמה לא בהכרח שלמה

אלגוריתם לפתרון בעיה באמצעות רשת זרימה ייראה כך:

1. נגדיר רשת זרימה עבור הבעיה;
2. נריץ על הרשת שהגדרנו את אלגוריתם EK למציאת זרימה חוקית $f$ משטף מקסימלי;
3. נבנה את הפתרון לבעיה המקורית מ-$f$.

## 35.2 זיווג מקסימלי בגרף דו-צדדי

**בעיה 35.1** (בעיית מציאת זיווג מקסימלי בגרף דו-צדדי). קלט: גרף דו-צדדי לא מכוון $G=(V=L\sqcup R,E)$.
פלט: זיווג מקסימלי בגרף $G$. (תזכורת: זיווג ב-$G$ הוא תת-קבוצה של צלעות $M\subseteq E$ כך שעבור כל קודקוד קיימת ב-$M$ לכל היותר צלע אחת שנוגעת בו.)

---

**אלגוריתם 19** אלגוריתם לפתרון בעיית מציאת זיווג מקסימלי בגרף דו-צדדי

---

1. נגדיר רשת זרימה $(V',E',c,s,t)$ באופן הבא:
   (א) $V':=L\cup R\cup\{s,t\}$
   (ב) נסמן ב-$\overrightarrow{E}$ את קשתות הגרף המקורי, מכוונות מ-$L$ ל-$R$, כלומר: $\overrightarrow{E}=\{(u,v):u\in L,v\in R,\{u,v\}\in E\}$.
   (ג) נגדיר $E_L=\{(s,u):u\in L\}$ (קישור קודקוד המקור $s$ לכל קודקודי $L$), ו-$E_R=\{(v,t):v\in R\}$ (קישור קודקודי $R$ לקודקוד הבור $t$).
   (ד) נקבע $E':=\overrightarrow{E}\cup E_L\cup E_R$.
   (ה) לכל צלע $e\in E'$ נגדיר $c(e):=1$.
2. נריץ על הרשת שבנינו את אלגוריתם EK למציאת זרימה בעלת שטף מקסימלי. נסמן ב-$f$ את הזרימה שהחזיר.

   חשוב לציין כי משום שכל הקיבולות ברשת שהוגדרה הם בשלמים, הזרימה המוחזרת גם שלמה, ומעבר לכך, יכולה לקבל רק את הערכים 0 או 1.
3. נחזיר את הזיווג $M=\left\{e\in\overrightarrow{E}:f(e)=1\right\}$.

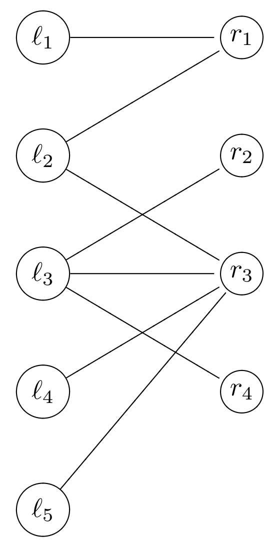
---

**זמן ריצה:** זמן הרצת EK לוקח $O\left(|V'||E'|^2\right)$. בניית הזיווג $M$ המוחזר דורשת $O(|E'|)$. בסה"כ לוקח $O\left(|V'||E'|^2\right)$.
נבטא את זמן הריצה באמצעות נתוני הבעיה המקורית,

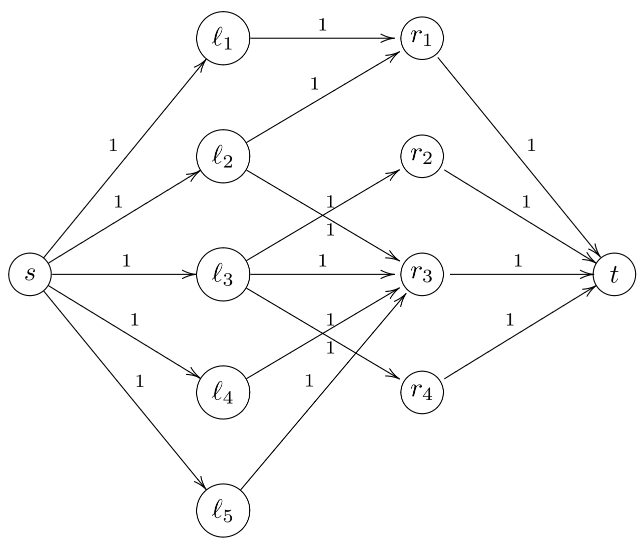
$$|V'| = |V| + 2$$
$$|E'| = |E| + |L| + |R| = |E| + |V|$$

מכאן –

$$O\left(|V'||E'|^2\right) = O\left(|V|(|E| + |V|)^2\right)$$



(א) דוגמה לגרף דו-צדדי לא מכוון



(ב) רשת הזרימה שנבנית ע"י האלגוריתם למציאת זיווג מקסימלי בגרף דו-צדדי

איור 35.2 : דוגמה לגרף דו-צדדי ולרשת הזרימה שנבנית ע"י אלגוריתם 19

**הוכחת נכונות:** חוקיות:
נראה ש-$M$ הוא אכן זיווג. נניח בשלילה שלא, אז קיים קודקוד $v \in L \sqcup R$ כך שקיימת יותר מקשת אחת שנוגעת בו.
אם $v \in L$ $(v \in R)$ אז ישנם לפחות שתי קשתות שיוצאות מ-$v$ (קשתות שנכנסות ל-$v$) שהזרימה בהן היא 1.
אולם, סה"כ הזרימה שנכנסת ל-$v$ (שיוצאת מ-$v$) היא לכל היותר 1, ולכן $f$ לא מקיימת את חוק שימור החומר, בסתירה לנכונות EK שמחזיר זרימה חוקית.
אופטימליות:
נניח בשלילה ש-$M$ אינו זיווג מקסימלי ב-$G$. אזי קיים זיווג $M'$ כך ש-$|M'| > |M|$.
אם נראה כי:

1. $|M| = |f|$;
2. קיימת זרימה חוקית $f'$ ברשת הזרימה שהגדרנו כך ש-$|M'| = |f'|$.

אז נסיק מכך שקיימת זרימה חוקית $f'$ שמקיימת $|f'| = |M'| > |M| = |f|$, בסתירה לנכונות של אלגוריתם EK שמחזיר זרימה בעלת שטף מקסימלי.

**טענה 35.2.** $|M| = |f|$.

הוכחה.

$$\begin{aligned}
|f| &\overset{\text{הגדרת}}{\underset{\text{השטף}}{=}} \sum_{u\in L} f(s,u) \\
[\text{חוק שימור החומר}] &= \sum_{(u,v)\in\overrightarrow{E}} f(u,v) \\
&= \sum_{\substack{(u,v)\in\overrightarrow{E} \\ f(u,v)=1}} 1 \\
&= |M|
\end{aligned}$$

כנדרש. □

**טענה 35.3.** יהי $M'$ זיווג, אזי קיימת זרימה חוקית $f'$ ברשת שהגדרנו באלגוריתם 19 כך ש-$|f'| = |M'|$.

הוכחה. נסמן ב-$L' \subseteq L$ את קבוצת הקודקודים ב-$L$ ש-$M'$ "נוגעת בהם". נסמן ב-$R' \subseteq R$ את קבוצת הקודקודים ב-$R$ ש-$M'$ "נוגעת בהם".
נגדיר את הזרימה $f'$ לכל $e \in E'$ באופן הבא:

$$f'(e) = \begin{cases} 1 & e \in \{(s,u) : u \in L'\} \cup M' \cup \{(v,t) : v \in R'\} \\ 0 & \text{Otherwise} \end{cases}$$

נראה ש-$f'$ חוקית:

- <u>אילוץ הקיבול:</u> אילוץ הקיבול בוודאי מתקיים כי $f'$ מזרימה רק 0 או 1, והקיבולים על כל הצלעות ברשת הם 1.
- <u>חוק שימור החומר:</u> שימור החומר נובע מהיותו של $M'$ זיווג. תהי $(u,v) \in M'$. אז אנו מזרימים ל-$u$ וממנו יחידה 1 של חומר, וכך גם עבור $v$ (ולכן $u, v$ משמרים את האילוץ).

  כל שאר הקודקודים, שלא "נוגעים" בצלעות הזיווג, לא מקבלים או מוציאים חומר, ולכן גם שומרים על האילוץ.

ולבסוף, נראה כי $|f'| = |M'|$:

$$\begin{aligned}
|f'| &= \sum_{u\in L} f'(s,u) \\
&= \sum_{u\in L'} f'(s,u) + \sum_{u\notin L'} f'(s,u) \\
&= \sum_{u\in L'} 1 + \cancelto{0}{\sum_{u\notin L'} 0} \\
&= |L'| \\
[\text{זיווג } M'] &= |M'|
\end{aligned}$$

כנדרש. □

**הערה 35.4.** התבוננו בסיכום התרגול, שם מוצג פתרון מלא לשאלה (ממבחן 2005, מועד ב') שמתבססת על האלגוריתם שפיתחנו בתרגול זה. ההצגה של הפתרון בתרגול הייתה על קצה המזלג ולכן לא מובאת כאן.

# 36 תרגול 10 - 24.12.18

## 36.1 דוגמה להרצת אלגוריתם אדמונדס-קארפ

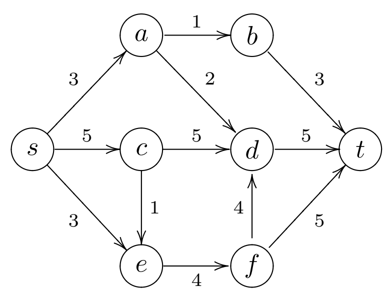
| | אלגוריתם פורד-פולקרסון | אלגוריתם אדמונדס-קארפ |
|---|---|---|
| **איזו מסילת הרחבה נבחר מ-$s$ ל-$t$ קודם?** | איזשהי מסילה | המסילה הקצרה ביותר (מבחינת מספר צלעות) בין $s$ ל-$t$ |
| **זמן ריצה** | $O\left(\|V\|\left\|f^*\right\|\right)$, במקרה שבו האלגוריתם עוצר | $O\left(\|V\|\|E\|^2\right)$ |

טבלה 2: השוואה בין אלגוריתם פורד-פולקרסון לבין אלגוריתם אדמונדס-קארפ

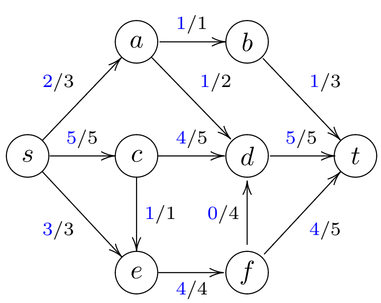
נתבונן באיור 36.1 עבור דוגמה לרשת זרימה.
נריץ את אלגוריתם אדמונדס-קארפ למציאת זרימה משטף מקסימלי, כאשר בחרנו את המסלולים הבאים:

1. באיטרציה הראשונה: בחרנו מסלול $s-a-b-t$ והזרמנו בו זרימה קבועה השווה ל-1.
2. באיטרציה השניה: בחרנו מסלול $s-c-d-t$ והזרמנו בו זרימה קבועה השווה ל-5.
3. באיטרציה השלישית: בחרנו מסלול $s-e-f-t$ והזרמנו בו זרימה קבועה השווה ל-3.
4. באיטרציה הרביעית: בחרנו מסלול $s-a-d-c-e-f-t$ והזרמנו בו זרימה קבועה השווה ל-1.
5. באיטרציה החמישית: לא קיימות מסילות הרחבה בגרף השיורי, וקיבלנו זרימה $f^*$ אופטימלית עם $|f^*|=10$.

הזרימה $f^*$ מתוארת באיור 36.1.



(א) רשת הזרימה במקור



(ב) תיאור הזרימה $f^*$ בעלת שטף מקסימלי שנמצאה ע"י אלגוריתם EK

איור 36.1: דוגמה להרצת אלגוריתם אדמונדס-קארפ

## 36.2 אלגוריתם למציאת חתך מינימלי ברשת

ניזכר בהגדרות מ-20.2 של חתך $(S,T)$ ברשת, קיבול של חתך $(S,T)$ וזרימה בחתך $(S,T)$.

**דוגמה 36.1.** נתבונן ברשת הזרימה מאיור 36.1, ונגדיר את $S=\{s,e\}$ ואת $T=V\setminus S$.
אז $(S,T)$ מהווה חתך ברשת, ומתקיים:

$$\begin{aligned} c(S,T) &= c(s,a)+c(s,c)+c(e,f) \\ &= 3+5+4 \\ &= 12 \end{aligned}$$

שימו לב שהצלע $(c,e)$ אינה נכללת בחישוב כי לפי הגדרת קיבול של חתך אנו מסתכלים על צלעות מכוונות מ-$S$ ל-$T$.

נחשב את הזרימה בחתך עבור הזרימה האופטימלית $f^*$ שמצאנו:

$$\begin{aligned} f^*(S,T) &= f^*(s,a)+f^*(s,c)+f^*(e,f)+f^*(e,c) \\ &= 2+5+4-1 \\ &= 10 \end{aligned}$$

אין זה מפתיע שקיבלנו שהזרימה בחתך שווה לשטף $|f^*|$ – ראו למה 20.4.
**אלגוריתם למציאת חתך מינימלי ברשת זרימה:**

1. נריץ את אלגוריתם אדמונדס-קארפ על רשת הזרימה.
2. נתבונן בגרף השיורי שהתקבל בתום הריצה. נגדיר את $S$ להיות קבוצת כל הקודקודים הנגישים בגרף השיורי מ-$s$. נגדיר $T=V\setminus S$.
3. נחזיר את החתך $(S,T)$.

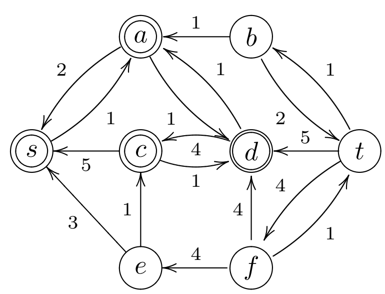
נכונות האלגוריתם נובעת ממשפט השטף והחתך 20.7 וממסקנותיו.

**זמן ריצה:** זמן הרצת אלגוריתם אדמונדס-קארפ הנו $O\left(|V|\,|E|^2\right)$ וזמן מציאת הקבוצה $S$ (שיכולה להיעשות ע״י הפעלת BFS או DFS בגרף השיורי) לוקחת $O(|V|+|E|)$.
בסה״כ זמן הריצה הוא $O\left(|V|\,|E|^2\right)$.

**דוגמה 36.2.** החתך המינימלי ברשת הזרימה מאיור 36.1, הינו:

$$\begin{aligned} S &= \{s,a,c,d\} \\ T &= \{b,e,f,t\} \end{aligned}$$

כי הגרף השיורי שמתקבל בסוף ריצת אלגוריתם אדמונדס-קארפ נראה כך:




כאשר הקודקודים ב-$S=\{s,a,c,d\}$ הם היחידים שנגישים מ-$s$. קיבול החתך $(S,T)$ הנו:

$$\begin{aligned} c(S,T) &= c(c,e)+c(s,e)+c(d,t)+c(a,b) \\ &= 1+3+5+1 \\ &= 10 \end{aligned}$$

וגם במקרה זה, אין זה מפתיע שקיבלנו $c(S,T)=|f^*|$, לפי משפט השטף והחתך 20.7.

**תרגיל 36.3.** נתבונן ברשת הזרימה מאיור 36.1. מצאו חתך $(S,T)$ מינימלי ברשת המקיים את האילוצים הבאים:

1. אם $e\in S$ אז גם $f\in S$.
2. אם $a\in S$ אז גם $b\in S$.

**פתרון 36.4.** נשנה את קיבול הצלעות $(a,b)$ ו-$(e,f)$ ל-$\infty$, ונריץ את האלגוריתם למציאת חתך מינימלי בגרף שתואר לעיל.
נבחין כי קיבול חתך $(S,T)$ הוא סופי $\Longleftrightarrow$ החתך $(S,T)$ מקיים את שני האילוצים. ננמק:
$(\Leftarrow)$: אם קיבול חתך $(S,T)$ הוא סופי אז הצלעות שחוצות את החתך לא יכולות להיות $(a,b)$ ו-$(e,f)$, ולכן החתך $(S,T)$ מקיים את שני האילוצים.
$(\Rightarrow)$: בכיוון ההפוך, אם החתך $(S,T)$ מקיים את שני האילוצים, אז קיבולו בוודאי סופי, כי הצלעות $(a,b)$ ו-$(e,f)$ (עם קיבול $\infty$) לא נלקחות בחשבון בחישוב $c(S,T)$.

## 36.3 בעיית המשקיעים והשחקנים

**בעיה 36.5** (בעיית המשקיעים והשחקנים).
קלט:

- $A = \{a_1, \ldots, a_n\}$ - קבוצת השחקנים. לכל שחקן $a_i$ נתונה משכורת $s_i \in \mathbb{N}$ שאותה דורש עבור השתתפות בסרט.
- $B = \{b_1, \ldots, b_k\}$ - קבוצת משקיעים. לכל משקיע $b_i$ נתונה תת-קבוצה $A_i \subseteq A$ של שחקנים שהוא דורש שיופיעו בסרט, ותקציב $d_j \in \mathbb{N}$ שהוא ישקיע אם כל השחקנים שדרש יופיעו בסרט.

פלט: תתי-קבוצות $A' \subseteq A$ ו-$B' \subseteq B$ כך שלכל משקיע $b_i \in B'$ מתקיים ש-$A_i \subseteq A'$ (כל השחקנים שדרש המשקיע $b_i$ מופיעים בקבוצת השחקנים $A'$).
בנוסף, נרצה שהרווח של $A', B'$ מקסימלי, כאשר הרווח מוגדר ע"י –

$$p\left(A', B'\right) =:= \sum_{b_i \in B'} d_i - \sum_{a_i \in A'} s_i$$

**אבחנות:**

1. מקסום $p\left(A', B'\right)$ שקול למזעור $-p\left(A', B'\right)$.
2. מזעור $-p\left(A', B'\right)$ שקול למזעור $D - p\left(A', B'\right)$, כאשר $D$ קבוע.

**דוגמה 36.6.** $n = 5$ (5 שחקנים), $k = 3$ (3 משקיעים), משכורות השחקנים

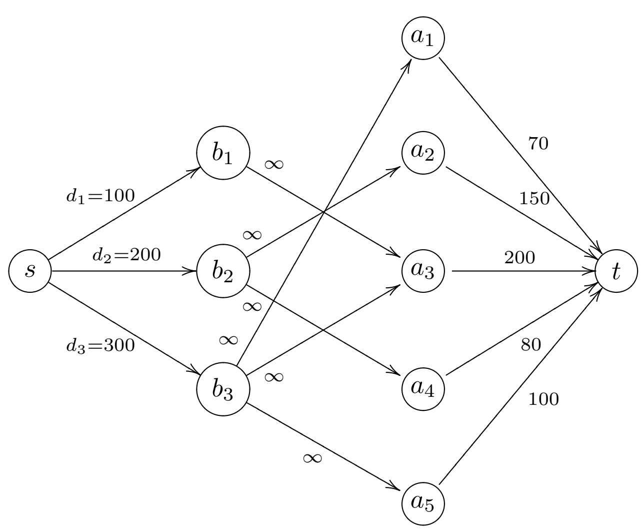
$$(s_1, \ldots, s_5) = (70, 150, 200, 80, 100)$$

וגם נתונה הטבלה הבאה:

| משקיע | שחקנים שדורש | מוכן להשקיע |
|---|---|---|
| $b_1$ | $A_1 = \{a_3\}$ | $d_1 = 100$ |
| $b_2$ | $A_2 = \{a_2, a_4\}$ | $d_2 = 200$ |
| $b_3$ | $A_3 = \{a_1, a_3, a_5\}$ | $d_3 = 300$ |

טבלה 3: טבלה שמתארת לכל משקיע את השחקנים שהוא דורש ואת הסכום שהוא מוכן להשקיע אם כל השחקנים שדרש יופיעו בסרט

נתאר את רשת הזרימה שנבנה במהלך פתרון הבעיה עבור הדוגמה:




**האלגוריתם:**

1. נבנה רשת זרימה $N=(V,E,c,s,t)$ באופן הבא:

$$\begin{aligned}
V &= \{s,t\}\cup A\cup B\\
E &= \{(s,b_i):b_i\in B\}\cup\{(b_i,a_j):a_j\in A_i, i\in[k]\}\cup\{(a_i,t):a_i\in A\}\\
c(v,u) &= \begin{cases} v=s,u\in B: & d_i\\ v=b_i,u\in A: & \infty\\ v\in A,u=t: & s_i\end{cases}
\end{aligned}$$

2. נמצא חתך מינימלי ברשת באמצעות ההרחבה לאדמונדס-קארפ, ונסמנו $(S,T)$.

3. נגדיר $A':=S\cap A$ ו-$B':=B\cap S$, ונחזיר את הקבוצות $A'$ ו-$B'$.

**הוכחת נכונות:** ראשית, נגדיר התאמה חח״ע ועל בין זוג קבוצות $A',B'$ של משקיעים ושחקנים (לאו דווקא חוקיות), לבין חתכים ברשת הזרימה שנבנית ע״י האלגוריתם.

בהינתן חתך ברשת $(S,T)$ נגדיר $A'=A\cap S$, ו-$B'=B\cap S$.

בהינתן $A'\subseteq A$ ו-$B'\subseteq B$ נגדיר את הקבוצות $S=\{s\}\cup A'\cup B'$ ו-$T=V\setminus S$, וניווכח כי $(S,T)$ חתך ברשת.

**טענה 36.7.** יהי $(S,T)$ חתך ברשת הזרימה שנבנית ע״י האלגוריתם ו-$A',B'$ קבוצות המתאימות לו (לפי האמור לעיל). אזי הקבוצות $A',B'$ חוקיות $\iff$ $c(S,T)$ סופי.

הוכחה. $c(S,T)$ סופי $\iff$ אין צלעות מקיבול $\infty$ שחוצות את החתך $(S,T)$ $\iff$ לכל $b_i\in B'$ מתקיים $A_i\subseteq A'$ $\iff$ $A',B'$ חוקיות. $\square$

נטען כי **האלגוריתם מחזיר פתרון חוקי**, כלומר, הקבוצות $A',B'$ חוקיות: הקבוצות $A',B'$ שמוחזרות מהאלגוריתם מתאימות לחתך מינימלי, וחתך מינימלי ברשת הוא בהכרח מקיבול סופי (כי יש איזשהו חתך מקיבול סופי, למשל $(S=\{s\},T=V\setminus\{s\})$ והאלגוריתם מחזיר חתך שקיבולו הוא לכל היותר קיבולו של החתך $(\{s\},V\setminus\{s\})$). אי לכך, לפי טענה 36.7, הקבוצות $A',B'$ שמוחזרות ע״י האלגוריתם חוקיות.

**טענה 36.8.** יהי $(S,T)$ חתך מקיבול סופי ו-$A',B'$ קבוצות חוקיות שמתאימות לו. אזי

$$c(S,T)=D-p(A',B')$$

כאשר $D$ קבוע שלא תלוי בחתך.

הוכחה. נחשב:

$$\begin{aligned}
c(S,T) &\overset{\text{חתך בעל}}{\underset{\text{קיבול סופי}}{=}} \sum_{b_i\notin B'}c(s,b_i)+\sum_{a_i\in A'}c(a_i,t)\\
&= \sum_{b_i\notin B'}d_i+\sum_{a_i\in A'}s_i\\
&= \sum_{b_i\in B}d_i-\sum_{b_i\in B'}d_i+\sum_{a_i\in A'}s_i
\end{aligned}$$

נגדיר $D:=\sum_{b_i\in B}d_i$ ונקבל כי –

$$\begin{aligned}
&= D-\left(\sum_{b_i\in B'}d_i-\sum_{a_i\in A'}s_i\right)\\
&= D-p(A',B')
\end{aligned}$$

כנדרש. $\square$

נטען כי **האלגוריתם מחזיר פתרון אופטימלי**: נניח שהקבוצות $A',B'$ שמחזיר האלגוריתם אינן אופטימליות. אזי יש קבוצות $A'',B''$ חוקיות עם רווח גדול יותר, כלומר –

$$p(A'',B'')>p(A',B')$$

נסמן ב-$(S',T')$ את החתך שמתאים לקבוצות $A'',B''$ (כיוון ש-$A'',B''$ חוקיות, החתך $(S',T')$ בעל קיבול סופי).

נסמן ב-$(S,T)$ את החתך שמתאים לקבוצות $A',B'$. לפי הטענה האחרונה שהוכחנו –

$$c(S',T')=D-p(A'',B'')<D-p(A',B')=c(S,T)$$

כלומר, החתך $(S',T')$ הוא בעל קיבול קטן יותר מהחתך המינימלי $(S,T)$, סתירה.

# 37 תרגול 11 - 31.12.18

## 37.1 תכנון לינארי: דואליות

ניזכר כי ב-32.1 הגדרנו את בעיית התכנון הלינארית בצורה הסטנדרטית.
בהינתן תוכנית לינארית סטנדרטית:

$$\begin{aligned} \max_{x\in\mathbb{R}^n} \quad & c^T \cdot x \\ \text{subject to} \quad & Ax \le b \\ & x \ge 0 \end{aligned}$$

כאשר $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m\times n}$; נגדיר את התוכנית הדואלית לה באופן הבא:

$$\begin{aligned} \min_{y\in\mathbb{R}^m} \quad & b^T \cdot y \\ \text{subject to} \quad & A^T y \ge c \\ & y \ge 0 \end{aligned}$$

מוסכמה: נסמן ב-P (מלשון Primal) את הבעיה המקורית (בעיית המקסימיזציה) ונסמן ב-D (מלשון Dual) את הבעיה הדואלית (בעיית המינימיזציה).

**הערה 37.1.**

1. נשים לב שהבעיה הדואלית של הדואלית היא הבעיה הפרימלית.
2. וקטור הפתרון של התוכנית הפרימלית הוא ב-$\mathbb{R}^n$ ומס' האילוצים בה הוא $m$ (לא כולל אילוצי אי-שליליות).
   וקטור הפתרון של התוכנית הדואלית הוא ב-$\mathbb{R}^m$ ומס' האילוצים בה הוא $n$ (לא כולל אילוצי אי-שליליות).
   הנקודה היא שבעיות אלו שונות במהותן. (עם זאת, יש ביניהן קשר, כפי שנראה בהמשך.)

**משפט 37.2** (משפט הדואליות החלשה)**.** יהי $x \in \mathbb{R}^n$ פתרון חוקי של הבעיה הפרימלית P, ויהי $y \in \mathbb{R}^m$ פתרון חוקי של הבעיה הדואלית D. אזי $c^T x \le b^T y$.

הוכחה. מכיוון ש-$A^T y \ge c$ ו-$x \ge 0$ אנו מקבלים כי

$$c^T x \le \left(A^T y\right)^T x$$

הצידוק הפורמלי למעבר הנ"ל הוא שהאילוץ $A^T y \ge c$ למעשה אומר שלכל קואורדינטה $1 \le i \le n$ מתקיים

$$\left(A^T y\right)_i \ge c_i$$

מכאן ש-

$$c^T x = \sum_{i=1}^{n} c_i x_i \underset{x\ge 0}{\le} \sum_{i=1}^{n} \left(A^T y\right)_i x_i = \left(A^T y\right)^T x$$

נמשיך בחישוב:

$$\begin{aligned} c^T x &\le \left(A^T y\right)^T x \\ &= \left(y^T A\right) x \\ &= y^T (Ax) \\ [Ax \le b, y \ge 0] &\le y^T b \\ &= b^T y \end{aligned}$$

(הצידוק לשוויון האחרון הוא פשוט לפתוח את ההגדרות של כפל וקטור שורה בוקטור עמודה.) □

**הערה 37.3.** משפט הדואליות החלשה נכון עבור כל פתרון חוקי $x \in \mathbb{R}^n$ של הבעיה הפרימלית וכל פתרון חוקי $y \in \mathbb{R}^m$ של הבעיה הדואלית. בפרט הוא נכון עבור הפתרון האופטימלי $x^* \in \mathbb{R}^n$ של הבעיה הפרימלית והפתרון האופטימלי $y^* \in \mathbb{R}^m$ של הבעיה הדואלית, לכן:

$$c^T x^* \leq b^T y^*$$

המשפט הבא מחזק את הקשר בין הבעיה הפרימלית לדואלית, והוא מובא ללא הוכחה.

**משפט 37.4** (משפט הדואליות החזקה)**.** תהי P בעיית תכנון לינארי פרימלית ותהי D בעיית התכנון הלינארית הדואלית לה. אזי אם לתוכנית הפרימלית יש פתרון אופטימלי, גם לדואלית יש פתרון אופטימלי, וערכי האופטימום שווים.

## 37.2 דוגמה: זרימה משטף מקסימלי וחתכים מינימליים

תהי $N = (V, E, c, s, t)$ רשת זרימה.

**שלב ראשון: כתיבת בעיית הזרימה כבעיית תכנון לינארי** נגדיר לכל צלע $(i, j) \in E$ משתנה $f_{ij}$ אשר ייצג את הזרימה בצלע $(i, j)$.
נסמן את הקיבול של הצלע $(i, j)$ ע״י הפרמטר $c'_{ij}$.
פונקציית המטרה שאנו מעוניינים למקסם היא השטף שיוצא מקודקוד $s$, והאילוצים יהיו אילוצי זרימה חוקית (אי-שליליות, קיבול וחוק שימור חומר).

$$\begin{aligned}
\max_{f \in \mathbb{R}^{|E|}} \quad & \sum_{i:(s,i)\in E} f_{si} \\
\text{subject to} \quad & 0 \leq f_{ij} \leq c'_{ij}, \quad \forall (i,j) \in E \\
& \sum_{x:(x,i)\in E} f_{xi} - \sum_{x:(i,x)\in E} f_{ix} \leq 0, \quad \forall i \in V \setminus \{s,t\} \\
& \sum_{x:(i,x)\in E} f_{ix} - \sum_{x:(x,i)\in E} f_{xi} \leq 0, \quad \forall i \in V \setminus \{s,t\}
\end{aligned}$$

כאשר האילוץ הראשון מתאים לאילוץ אי-שליליות ואילוץ הקיבול. שני האילוצים הנוספים מתאימים לחוק שימור החומר, ולמעשה דורשים כי לכל $i \in V \setminus \{s, t\}$:

$$\sum_{x:(i,x)\in E} f_{ix} = \sum_{x:(x,i)\in E} f_{xi}$$

**שלב שני: בחינת בעיה דומה** נתבונן בבעיה דומה שערך פתרון אופטימלי עבורה שווה לערך פתרון אופטימלי של הבעיה המקורית.
נמחוק מהתוכנית הלינארית שבנינו בשלב הראשון את האילוץ האחרון: $\sum_{x:(i,x)\in E} f_{ix} - \sum_{x:(x,i)\in E} f_{xi} \leq 0$.

הבעיה החדשה הינה:

$$\begin{aligned}
\max_{f \in \mathbb{R}^{|E|}} \quad & \sum_{i:(s,i)\in E} f_{si} \\
\text{subject to} \quad & 0 \leq f_{ij} \leq c'_{ij}, \quad \forall (i,j) \in E \\
& \sum_{x:(x,i)\in E} f_{xi} - \sum_{x:(i,x)\in E} f_{ix} \leq 0, \quad \forall i \in V \setminus \{s,t\}
\end{aligned}$$

נבחין כי כל פתרון של הבעיה הקודמת (מהשלב הראשון) הוא גם פתרון של הבעיה החדשה, ולכן:

$$\text{שטף זרימה מקסימלי בבעיה המקורית} \leq \text{שטף זרימה מקסימלי בבעיה החדשה}$$

בבעיה החדשה בא לידי ביטוי רק כיוון אחד בחוק שימור החומר, וניתן להוכיח (תרגיל) שערך השטף המקסימלי בבעיה זו שווה לערך השטף המקסימלי של הבעיה המקורית.

**שלב שלישי: מעבר לצורה סטנדרטית** נקבע סדר מסוים למעבר על הצלעות. נרשום:

$$\begin{aligned} \max_{f \in \mathbb{R}^{|E|}} \quad & c^T f \\ \text{subject to} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

כאשר $A$ מטריצה מסדר $|E| \times (|E| + |V| - 2)$,

$$A = \begin{pmatrix} I_{|E|} \\ \hline M_{(|V|-2)\times|E|} \end{pmatrix}$$

והמטריצה $M$ היא מסדר $|E| \times (|V| - 2)$ ומוגדרת באופן הבא:

$$M_{\ell,ij} = \begin{cases} +1 & j = \ell \\ -1 & i = \ell \\ 0 & \text{אחרת} \end{cases}$$

נגדיר את הוקטור $b \in \mathbb{R}^{(|E|+|V|-2)}$ באופן הבא:

- ב-$|E|$ הקואורדינטות הראשונות נציב את קיבולי הצלעות $c'_{ij}$.
- ב-$|V| - 2$ הקואורדינטות הבאות נציב את הערך 0.

באופן מפורש, $b = \begin{pmatrix} | \\ c'_{ij} \\ | \\ \hline 0 \\ \vdots \\ 0 \end{pmatrix}$.

נותר להגדיר את הוקטור $c \in \mathbb{R}^{|E|}$. נגדירו באופן הבא:

$$c_{ij} = \begin{cases} 1 & i = s \\ 0 & \text{אחרת} \end{cases}$$

**הערה 37.5.** נשים לב ש-$c$ הוא וקטור עמודה ב-$\mathbb{R}^{|E|}$ ולא מטריצה כפי שניתן היה לחשוב מהסימון $c_{ij}$. הסימון $c_{ij}$ הוא פשוט סימון נוח לצלע המדוברת. ההנחה היא, כאמור בהתחלה, שאנו קובעים סדר על הצלעות ושומרים על הסדר הזה לאורך הגדרת המטריצות והוקטורים השונים. הערה דומה תקפה עבור הסימון $M_{\ell,ij}$.

**שלב רביעי: כתיבת הבעיה הדואלית** את הבעיה הדואלית נכתוב לפי הגדרתה:

$$\begin{aligned} \min_{y \in \mathbb{R}^{(|E|+|V|-2)}} \quad & b^T y \\ \text{subject to} \quad & A^T y \geq c \\ & y \geq 0 \end{aligned}$$

**שלב חמישי: תובנות על הקשר בין הבעיה הפרימלית לבעיה הדואלית** אנו יודעים כי $y$ הוא וקטור ב-$\mathbb{R}^{(|E|+|V|-2)}$, ולכן טבעי יהיה לסמן את $|E|$ הקואורדינטות הראשונות ב-$w_{ij}$ כאשר $(i,j) \in E$, ואת $|V| - 2$ הקואורדינטות הבאות ב-$z_\ell$ כאשר $\ell \in V \setminus \{s,t\}$. באופן מפורש,

$$y = \begin{pmatrix} | \\ w_{ij} \\ | \\ \hline | \\ z_\ell \\ | \end{pmatrix}$$

לפי ההגדרה של $b$ אנו יודעים כי $|V|-2$ הקואורדינטות האחרונות שווים $0$, ולכן פונקציית המטרה שאנו מעוניינים למזער הינה

$$\begin{aligned} \min_{w\in\mathbb{R}^{|E|},z\in\mathbb{R}^{|V|-2}} \quad & \sum_{(i,j)\in E} c'_{ij}w_{ij} \\ \text{subject to} \quad & w\geq 0 \\ & z\geq 0 \end{aligned}$$

ננסה להבין את המשמעות של האילוצים $A^Ty\geq c$.

$$\left(\begin{array}{c|c} I_{|E|} & M^T \end{array}\right)\begin{pmatrix} | \\ w_{ij} \\ | \\ \hline | \\ z_\ell \\ | \end{pmatrix} \geq \begin{pmatrix} | \\ c_{ij} \\ | \end{pmatrix}$$

נשים לב שבשורה ה-$ij$ במטריצה $M^T$ יכולים להיות לכל היותר שתי קואורדינטות עם ערך $+1$ ו-$-1$:

- אם $i=s$ אז בשורה ה-$sj$ של $M^T$ יש קואורדינטה אחת $(sj,j)$ עם ערך שונה מאפס, והוא $+1$, זאת משום שאין שהקודקוד $s$ לא היה חלק מהאילוצים של חוק שימור החומר בבעיה המקורית.
- אם $j=t$ אז בשורה ה-$it$ של $M^T$ יש קואורדינטה אחת $(it,i)$ עם ערך שונה מאפס, והוא $-1$.
- אחרת, בשורה ה-$ij$ של $M^T$ יש שתי קואורדינטות עם ערך שונה מאפס: $M^T_{ij,i}=-1$, $M^T_{ij,j}=+1$.

מכאן אנו מסיקים את התוכנית הלינארית הבאה:

$$\begin{aligned} \min_{w\in\mathbb{R}^{|E|},z\in\mathbb{R}^{|V|-2}} \quad & \sum_{(i,j)\in E} c'_{ij}w_{ij} \\ \text{subject to} \quad & w_{sj}+z_j\geq 1 \quad \forall j\in V,(s,j)\in E \\ & w_{ij}+z_j-z_i\geq 0 \quad \forall (i,j)\in E, i\neq s, j\neq t \\ & w_{it}-z_i\geq 0 \quad \forall i\in V,(i,t)\in E \\ & w\geq 0 \\ & z\geq 0 \end{aligned}$$

בתרגיל תתבקשו להוכיח שבעיה זו מתאימה לבעיית מציאת חתך מינימלי ברשת זרימה.

# 38 תרגול 12 - 7.1.19

## 38.1 אלגוריתם מהיר לכפל פולינומים

נפתח **בתזכורת** למושגים שראינו בהרצאה.

**פולינומים** פולינום $P(x)=\sum_{k=0}^{n-1}a_kx^k$ ניתן לייצג בשתי דרכים:

1. ייצוג בעזרת מקדמים
2. ייצוג בעזרת ערכי נקודות: עבור $n$ נקודות שונות $x_0,\dots,x_{n-1}$ הערכים $P(x_0),\dots,P(x_{n-1})$ קובעים את $P$.

**דוגמה 38.1.** נתבונן בפולינום $P(x)=1+2x+3x^2+4x^3$.

1. ייצוג בעזרת מקדמים: $\begin{pmatrix}1\\2\\3\\4\end{pmatrix}$.
2. ייצוג בעזרת ערכי נקודות: $x_0=0,x_1=-1,x_2=1,x_3=2$ אז:

$$P(x_0)=1,P(x_1)=-2,P(x_2)=10,P(x_3)=49$$

ולכן ייצוג בעזרת ערכי נקודות של $P$ הוא $\begin{pmatrix}1\\-2\\10\\49\end{pmatrix}$ עבור בחירת הנקודות $x_0,x_1,x_2,x_3$.

מעבר בין ייצוג מקדמים לייצוג ערכים לוקח באופן נאיבי $O(n^2)$, ומעבר בין ייצוג ערכים לייצוג המקדמים לוקח $O(n^2)$ באמצעות איטרפולציה.

**מספרים מרוכבים** לפי נוסחת אוילר, לכל מספר ממשי $x$ מתקיים $e^{ix}=\cos(x)+i\sin(x)$.
נשים לב שהצגה זו נוחה כיוון שהיא מאפשרת להשתמש בחוקי חזקות "רגילים", שהרי:

$$e^{i\theta}\cdot e^{i\varphi}=(\cos(\theta)+i\sin(\theta))(\cos(\varphi)+i\sin(\varphi))=\cos(\theta+\varphi)+i\sin(\theta+\varphi)=e^{i(\theta+\varphi)}$$

שורש היחידה הפרימיטיבי מסדר $n$ הוא $e^{\frac{2\pi}{n}i}$ ומסומן $\omega_n$.

**התמרת פורייה הבדידה** נניח כי הפולינום $P$ ממעלה $n-1$ נתון בייצוג המקדמים ע"י הוקטור $\mathbf{a}=(a_0,a_1,\dots,a_{n-1})$. נגדיר עבור $k=0,\dots,n-1$

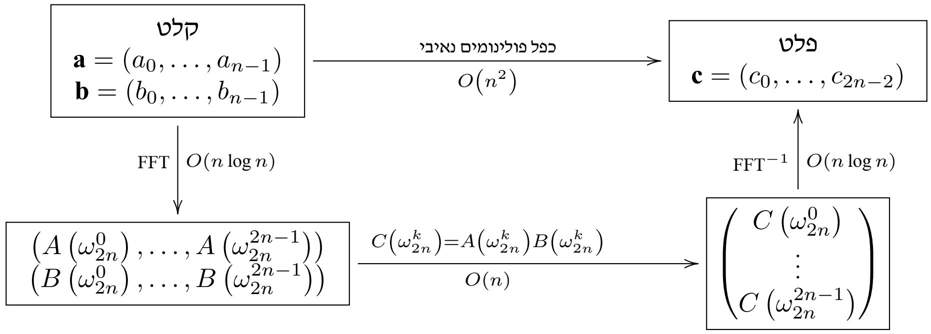
$$y_k=P\left(\omega_n^k\right)$$

אזי הוקטור $\mathbf{y}=(y_0,\dots,y_{n-1})$ הוא התמרת פורייה הבדידה מסדר $n$ של וקטור המקדמים $\mathbf{a}$, ומסמנים: $\mathrm{DFT}_n(\mathbf{a})=\mathbf{y}$.

**דוגמה 38.2.** יהי $P(x)=1+2x+3x^2+4x^3$, אז $\mathbf{a}=(1,2,3,4)$ ומקבלים:

$$\mathrm{DFT}_4\begin{pmatrix}1\\2\\3\\4\end{pmatrix}=\begin{pmatrix}P(1)\\P(i)\\P(-1)\\P(-i)\end{pmatrix}=\begin{pmatrix}10\\-2-2i\\-2\\-2+2i\end{pmatrix}$$

**אלגוריתם מהיר לכפל פולינומים**
קלט: שני פולינומים $A(x),B(x)$ המיוצגים באמצעות מקדמים $\mathbf{a}=(a_0,\dots,a_{n-1})$ עבור $A$ ו-$\mathbf{b}=(b_0,\dots,b_{n-1})$ עבור $B$.
פלט: ייצוג של פולינום המכפלה $C(x)=A(x)B(x)$ בעזרת מקדמים $\mathbf{c}=(c_0,\dots,c_{2n-2})$.
תיאור סכמתי של אלגוריתם מהיר לכפל פולינומים (כאשר ההנחה היא ש-$n$ הוא חזקה של 2, ואם הוא לא, אז "מרפדים" באפסים לחזקה של 2 הקרובה).




תיאור פורמלי והוכחת נכונות ב-11.

## 38.2 קונבולוציה

**הגדרה 38.3** (קונבולוציה)**.** בהינתן שני וקטורים $\mathbf{a} = (a_0, \ldots, a_{n-1})$, $\mathbf{b} = (b_0, \ldots, b_{m-1})$ נגדיר את הקונבולוציה שלהם ע"י

$$\mathbf{a} * \mathbf{b} = (c_0, c_1, \ldots, c_{m+n-2})$$

כאשר $c_k = \sum_{j=0}^{k} a_j b_{k-j}$ לכל $0 \le k \le m+n-2$.

הערה: עבור $j \ge n$ נגדיר $a_j = 0$; עבור $j \ge m$ נגדיר $b_j = 0$; עבור $j < 0$ נגדיר $a_j = b_j = 0$.

**דוגמה 38.4.** $\mathbf{a} = (1,2,3,2)$, $\mathbf{b} = (1,3)$. נחשב את $\mathbf{a} * \mathbf{b}$ ישירות מן ההגדרה:

$$\begin{aligned}
(\mathbf{a} * \mathbf{b})_0 &= a_0b_0 = 1 \cdot 1 = 1 \\
(\mathbf{a} * \mathbf{b})_1 &= a_0b_1 + a_1b_0 = 1 \cdot 3 + 2 \cdot 1 = 5 \\
(\mathbf{a} * \mathbf{b})_2 &= \cancelto{0}{a_0b_2} + a_1b_1 + a_2b_0 = 2 \cdot 3 + 3 \cdot 1 = 9 \\
(\mathbf{a} * \mathbf{b})_3 &= \cancelto{0}{a_0b_3} + \cancelto{0}{a_1b_2} + a_2b_1 + a_3b_0 = 3 \cdot 3 + 2 \cdot 1 = 11 \\
(\mathbf{a} * \mathbf{b})_4 &= \cancelto{0}{a_0b_4} + \cancelto{0}{a_1b_3} + \cancelto{0}{a_2b_2} + a_3b_1 + \cancelto{0}{a_4b_0} = 2 \cdot 3 = 6
\end{aligned}$$

בסה"כ קיבלנו $\mathbf{a} * \mathbf{b} = (1,5,9,11,6)$.

דרך "ציורית" לחישוב קונבולוציה היא באמצעות שיטה המכונה "שיטת הנחש".
נרשום את הוקטור **a** לפי הסדר ונרשום את הוקטור **b** <u>בסדר ההפוך</u> באופן כזה שהקואורדינטה הראשונה בוקטור ההפוך של **b** תירשם מתחת לקואורדינטה הראשונה של **a**. כך זה נראה:

$$\begin{array}{cccccc}
 & & a_0 & a_1 & a_2 & a_3 \\
\mathbf{a} & & {\color{blue}1} & 2 & 3 & 2 \\
\mathbf{b} & 3 & {\color{blue}1} & & & \\
 & b_1 & b_0 & & &
\end{array}$$

בכל שלב אנו כופלים איבר מהוקטור **a** באיבר שנמצא מתחתיו, ואם לא קיים איבר אנו מתייחסים אליו בתור 0.
למשל, כדי לחשב את וקטור הקונבולוציה במקום ה-0, נכפול את $a_0$ ב-$b_0$, ונקבל $(\mathbf{a} * \mathbf{b})_0 = a_0b_0 = 1$.
על מנת לחשב את וקטור הקונבולוציה במקום ה-1, נזיז את **b** צעד אחד ימינה, ונחזור חלילה.

$$\begin{array}{ccccc}
 & a_0 & a_1 & a_2 & a_3 \\
\mathbf{a} & {\color{orange}1} & {\color{blue}2} & 3 & 2 \\
\mathbf{b} & {\color{orange}3} & {\color{blue}1} & & \\
 & b_1 & b_0 & &
\end{array}$$

מכאן: $(\mathbf{a} * \mathbf{b})_1 = {\color{orange}1} \cdot {\color{orange}3} + {\color{blue}2} \cdot {\color{blue}1} = 5$.

כיצד מחשבים קונבולוציה של שני וקטורים **a** ו-**b**? אם נתייחס ל-**a** בתור וקטור מקדמים של פולינום $A(x)$ ואל **b** בתור וקטור מקדמים של פולינום $B(x)$, הקונבולוציה של **a** ו-**b** היא בדיוק וקטור המקדמים של פולינום המכפלה $C(x) = A(x)B(x)$, ולכן ניתן לחשבה ב-$O(n \log n)$.

## 38.3 בעיית ספירת סכומים

**קלט:** שתי קבוצות $A, B \subseteq \{0,1,\ldots,n-1\}$.

**פלט:** לכל $k = 0,1,\ldots,2n-2$ כמה דרכים שונות, $m_k$, קיימות להציג את $k$ כסכום של איבר מ-$A$ ואיבר מ-$B$.

**דוגמה 38.5.** $n=4$, $A=\{0,1,2\}$, $B=\{2,3\}$. את המספר 0 לא ניתן להציג כסכום של איבר מ-$A$ ואיבר מ-$B$, לכן $m_0=0$. משיקול דומה, $m_1=0$.
את המספר 2 ניתן להציג בתור $2+0$ כאשר $0\in A$ ו-$2\in B$. זו הדרך היחידה, לכן $m_2=1$.
את המספר 3 ניתן להציג בתור $3+0$ כאשר $0\in A$ ו-$3\in B$ וגם בתור $2+1$ כאשר $1\in A$ ו-$2\in B$. לכן $m_3=2$.
את המספר 4 ניתן להציג בתור $2+2$ וגם $3+1$ כאשר $1\in A$ ו-$3\in B$ לכן $m_4=2$.
את המספר 5 ניתן להציג רק בתור $3+2$ כאשר $2\in A$ ו-$3\in B$, לכן $m_5=1$.
את המספר 6 לא ניתן להציג כסכום של איבר מ-$A$ ואיבר מ-$B$, לכן $m_6=0$.

**אלגוריתם:**

1. נגדיר את שני הוקטורים $\mathbf{a}=(a_0,a_1,\ldots,a_{n-1})$ ו-$\mathbf{b}=(b_0,b_1,\ldots,b_{n-1})$ באופן הבא:

$$a_i=\begin{cases}1 & i\in A\\ 0 & i\notin A\end{cases} \qquad b_j=\begin{cases}1 & j\in B\\ 0 & j\notin B\end{cases}$$

2. נחשב את הקונבולוציה $\mathbf{c}=\mathbf{a}*\mathbf{b}$.
3. נחזיר את $\mathbf{c}$. (ב-$c_k$ יופיע $m_k$ שהוא מספר הדרכים השונות להציג את $k$ כסכום של איבר מ-$A$ ועוד איבר מ-$B$)

**זמן ריצה:** שלב בניית הוקטורים עולה $O(n)$, שלב חישוב הקונבולוציה עולה $O(n\log n)$ אם משתמשים ב-FFT. סה״כ זמן ריצה הוא $O(n\log n)$.

**הוכחת נכונות:** לפי הגדרת הקונבולוציה מתקיים $(\mathbf{a}*\mathbf{b})_k=\sum_{j=0}^{k}a_jb_{k-j}$. עבור $0\le j\le k$ מסוים, $a_jb_{k-j}=1$ אם״ם $a_j=1$ וגם $b_{k-j}=1$ אם״ם $j\in A$ וגם $(k-j)\in B$.
מכאן ש–

$$\sum_{j=0}^{k}a_jb_{k-j}=\sum_{\substack{j=0\\ j\in A,(k-j)\in B}}^{k}1=m_k$$

ולכן האלגוריתם פועל נכון כאשר מחזיר את וקטור הקונבולוציה $\mathbf{c}$.

## 38.4 בעיית התאמת המחרוזות

**קלט:** שתי מחרוזות $\mathbf{a}\in\{-1,1\}^n$, $\mathbf{b}\in\{-1,1\}^m$, $m\le n$.

**פלט:** $D$, קבוצת כל האינדקסים ב-$\mathbf{a}$ שבהם מתחיל מופע רציף של $\mathbf{b}$. (מותרת ״חפיפה״)

**דוגמה 38.6** $\mathbf{a}=(-1,1,-1,1,-1,-1,1,-1,1)$, $\mathbf{b}=(-1,1,-1,1)$. אז $D=\{0,5\}$. מסומנים בכחול המופעים של $\mathbf{b}$ ב-$\mathbf{a}$:

$$\mathbf{a}=(-1,1,-1,1,-1,-1,1,-1,1)$$

**אלגוריתם:**

1. נחשב את המחרוזת ההפוכה של $\mathbf{b}$: $\mathbf{b}^r=(b_{m-1},b_{m-2},\ldots,b_0)$.
2. נחשב את $\mathbf{c}=\mathbf{a}*\mathbf{b}^r$.
3. נחזיר את $D=\{k-(m-1): c_k=m\}$.

**זמן ריצה:** שלב חישוב המחרוזת $\mathbf{b}^r$ לוקח $O(m)$, שלב חישוב הקונבולוציה לוקח $O(n\log n)$ ושלב בניית הקבוצה $D$ דורש מעבר על וקטור הקונבולוציה ולכן דורש $O(n+m)$.

**הוכחת נכונות:** נשים לב כי לכל $0\le k\le n-m$ מתקיים:

$$(\mathbf{a}*\mathbf{b}^r)_{k+(m-1)}=\sum_{i=0}^{k+(m-1)}a_i\left(\mathbf{b}^r\right)_{k+(m-1)-i}=\sum_{i=0}^{k+(m-1)}a_i\left(\mathbf{b}^r\right)_{(m-1)-(i-k)}=\sum_{i=0}^{k+(m-1)}a_ib_{i-k}$$

עבור $i<k$ מתקיים $i-k<0$ ולכן $b_{i-k}=0$. מכאן –

$$(\mathbf{a}*\mathbf{b}^r)_{k+(m-1)}=\sum_{i=k}^{k+(m-1)}a_ib_{i-k}=a_kb_0+a_{k+1}b_1+\cdots+a_{k+(m-1)}b_{m-1}$$

האבחנה החשובה היא ש-$m=(\mathbf{a}*\mathbf{b}^r)_{k+(m-1)}$ אס"ם לכל $0\le j\le m-1$ מתקיים $a_{k+j}b_j=1$ וזה קורה אס"ם לכל $0\le j\le m-1$ מתקיים $a_{k+j}=b_j$ (כי $a_i,b_i\in\{-1,1\}$) וזה קורה אס"ם באינדקס $k$ ב-$\mathbf{a}$ מתחיל מופע רציף של $\mathbf{b}$. מכאן נובעת נכונות האלגוריתם.

# 39 תרגול 13 - 14.1.19

## 39.1 התמרת פורייה המהירה - דוגמת הרצה

נחזור על הרעיונות המרכזיים באלגוריתם FFT ע"י דוגמה מייצגת.

נרצה לחשב את $\mathrm{DFT}_8\begin{pmatrix}1\\2\\\vdots\\8\end{pmatrix}$. נסמן ב-$P$ את הפולינום שוקטור המקדמים שלו הוא $\begin{pmatrix}1\\2\\\vdots\\8\end{pmatrix}$, כלומר:

$$P(x) = 1 + 2x + 3x^2 + 4x^3 + 5x^4 + 6x^5 + 7x^6 + 8x^7$$

לפי הגדרת DFT מתקיים כי:

$$\mathrm{DFT}_8\begin{pmatrix}1\\2\\\vdots\\8\end{pmatrix} = \begin{pmatrix}P(1)\\P(\omega_8)\\\vdots\\P(\omega_8^7)\end{pmatrix}$$

כאשר $\omega_8$ הוא שורש היחידה הפרימיטיבי מסדר 8.

**אבחנה ראשונה** לפי למה 24.3 וחוקי חזקות אנו מקבלים כי לכל $0 \le j \le 3$ מתקיים $\omega_8^{j+4} = \omega_8^j \cdot \omega_8^4 = -\omega_8^j$.

**אבחנה שניה** נוכל לכתוב את הפולינום $P$ באופן הבא:

$$P(x) = \left(1 + 3x^2 + 5x^4 + 7x^6\right) + x \cdot \left(2 + 4x^2 + 6x^4 + 8x^6\right)$$

נסמן $P_0(y) = 1 + 3y + 5y^2 + 7y^3$, $P_1(y) = 2 + 4y + 6y^2 + 8y^3$, והרי קיבלנו:

$$P(x) = P_0\left(x^2\right) + xP_1\left(x^2\right)$$

זו דוגמה לפירוק שהצגנו בלמה 24.1.

ננסה להציב את $\omega_8$ ב-$P$ וניעזר בלמה 24.5.

$$P(\omega_8) = P_0\left(\omega_8^2\right) + \omega_8 P_1\left(\omega_8^2\right) = P_0(\omega_4) + \omega_8 P_1(\omega_4)$$

בנוסף, לפי האבחנה הראשונה, אם נציב את $\omega_8^5$ נקבל:

$$P\left(\omega_8^5\right) = P(-\omega_8) = P_0\left((-\omega_8)^2\right) + (-\omega_8)P_1\left((-\omega_8)^2\right) = P_0(\omega_4) - \omega_8 P_1(\omega_4)$$

נסיק מכך לגבי הכלל (הצידוקים הפורמליים מצויים בלמות שהוזכרו, ובהוכחת משפט 23.3):

$$\mathrm{DFT}_8\begin{pmatrix}1\\2\\3\\4\\5\\6\\7\\8\end{pmatrix} = \left(\begin{array}{c}\mathrm{DFT}_4\begin{pmatrix}1\\3\\5\\7\end{pmatrix} + \begin{pmatrix}1\\\omega_8\\\omega_8^2\\\omega_8^3\end{pmatrix} \bullet \mathrm{DFT}_4\begin{pmatrix}2\\4\\6\\8\end{pmatrix}\\ \hline \mathrm{DFT}_4\begin{pmatrix}1\\3\\5\\7\end{pmatrix} - \begin{pmatrix}1\\\omega_8\\\omega_8^2\\\omega_8^3\end{pmatrix} \bullet \mathrm{DFT}_4\begin{pmatrix}2\\4\\6\\8\end{pmatrix}\end{array}\right)$$

כאשר $\bullet$ מציין פעולת כפל איבר-איבר.

**אבחנה שלישית** לכל $a,b\in\mathbb{C}$ מתקיים $\mathrm{DFT}_2\begin{pmatrix}a\\b\end{pmatrix}=\begin{pmatrix}a+b\\a-b\end{pmatrix}$ (נובע מן ההגדרה של DFT).

לפיכך, אם נמשיך את החישוב הקודם:

$$\mathrm{DFT}_4\begin{pmatrix}1\\3\\5\\7\end{pmatrix}=\left(\begin{array}{c}\mathrm{DFT}_2\begin{pmatrix}1\\5\end{pmatrix}+\begin{pmatrix}1\\\omega_4\end{pmatrix}\bullet\mathrm{DFT}_4\begin{pmatrix}3\\7\end{pmatrix}\\\hline\mathrm{DFT}_2\begin{pmatrix}1\\5\end{pmatrix}-\begin{pmatrix}1\\\omega_4\end{pmatrix}\bullet\mathrm{DFT}_4\begin{pmatrix}3\\7\end{pmatrix}\end{array}\right)=\left(\begin{array}{c}\left(\begin{pmatrix}6\\-4\end{pmatrix}+\begin{pmatrix}1\\i\end{pmatrix}\bullet\begin{pmatrix}10\\-4\end{pmatrix}\right)\\\hline\left(\begin{pmatrix}6\\-4\end{pmatrix}-\begin{pmatrix}1\\i\end{pmatrix}\bullet\begin{pmatrix}10\\-4\end{pmatrix}\right)\end{array}\right)=\begin{pmatrix}16\\-4-4i\\-4\\-4+4i\end{pmatrix}$$

באופן דומה מחשבים את $\mathrm{DFT}_4\begin{pmatrix}2\\4\\6\\8\end{pmatrix}$, ומכך מסיקים את $\mathrm{DFT}_8\begin{pmatrix}1\\2\\\vdots\\8\end{pmatrix}$, כפי שרצינו.

## 39.2 סקירה של אלגוריתמים על מספרים

יהיו $a,b\in\mathbb{N}$ מספרים שהייצוג הבינארי שלהם הוא לכל היותר $k$ ביטים.
ניזכר בעובדות הבאות:

- מספר הביטים של מספר $a$ הוא $\lfloor\log_2(a)\rfloor+1$.
- עלות חיבור/חיסור של $a$ ו-$b$ היא $O(k)$.
- עלות כפל/חילוק של $a$ ו-$b$ היא $O(k^2)$.
- עלות חישוב $a\ (\mathrm{mod}\ b)$ היא $O(k^2)$, שהרי $a\ (\mathrm{mod}\ b)=a-\left\lfloor\frac{a}{b}\right\rfloor\cdot b$.
- את $a^n$ עבור $n\in\mathbb{N}$ ניתן לחשב ב-$O(\log(n))$ פעולות כפל.

  מחשבים את $a^{2^i}$ לכל $0\le i\le\lfloor\log_2(n)\rfloor$ ע"י העלאה בריבוע של המספר האחרון שחושב:

$$a^{2^0}\to\left(a^{2^0}\right)^2=a^{2^1}\to a^{2^2}\to\cdots\to a^{2^{\lfloor\log_2(n)\rfloor}}$$

  לבסוף, כדי לקבל את $a^n$, מכפילים חזקות $a^{2^i}$ אם $2^i$ מופיע (=נכפל ב-1) בייצוג הבינארי של $n$.
  למשל, $23=16+4+2+1$ ולכן $a^{23}=a^{16+4+2+1}=a^{16}\cdot a^4\cdot a^2\cdot a^1$.

## 39.3 אלגוריתם אוקלידס המורחב

**הערה 39.1.** בהגדרות הבאות ובתיאור אלגוריתם אוקלידס המורחב נאמץ את המוסכמה $0\in\mathbb{N}$, כלומר 0 ייחשב כמספר טבעי.

**הגדרה 39.2.** יהיו $a,b\in\mathbb{N}$. נאמר כי $a$ **מחלק את** $b$, ונסמן $a|b$, אס"ם קיים $k\in\mathbb{N}$ כך ש- $b=k\cdot a$.

**הגדרה 39.3.** יהיו $a,b\in\mathbb{N}$, ונניח כי $b\le a$. **הפירוק עם שארית של** $a$ **לפי** $b$ הוא $a=k\cdot b+r$ כאשר $k,r\in\mathbb{N}$ ו- $0\le r<b$.

**הערה 39.4.** הפירוק עם שארית של $a$ לפי $b$ קיים ויחיד. בנוסף, מתקיים $k=\left\lfloor\frac{a}{b}\right\rfloor$, $r=a\ (\mathrm{mod}\ b)$.

**הגדרה 39.5.** יהיו $a,b\in\mathbb{N}$ ונניח כי $b\le a$. **המחלק המשותף המקסימלי של $a$ ו-$b$** הינו:

$$\gcd(a,b):=\max\{1\le d\in\mathbb{N}:d|a\wedge d|b\}$$

בנוסף, $a$ ו-$b$ ייקראו **מספרים זרים** אס"ם $\gcd(a,b)=1$.

תכונות שנובעות ישירות מן ההגדרות:

1. עבור כל $a\in\mathbb{N}$ מתקיים $a|0$.
2. $0|a\iff a=0$.
3. לכל $0\neq a\in\mathbb{N}$ מתקיים $\gcd(a,0)=a$.

**טענה 39.6.** יהיו $a,b\in\mathbb{N}\setminus\{0\}$ כך ש-$b\le a$. אזי $\gcd(a,b)=\gcd(b,a\,(\mathrm{mod}\ b))$.

הוכחה. נסמן $d=\gcd(a,b)$, $d'=\gcd(b,a\,(\mathrm{mod}\ b))$.
לפי הגדרת gcd מתקיים כי $d|b$ וגם $d'|b$. אסטרטגיית ההוכחה היא להראות כי בהכרח $d|a\,(\mathrm{mod}\ b)$ ומכאן ינבע ש-$d$ הוא מחלק משותף של $b$ ושל $a\,(\mathrm{mod}\ b)$ ולכן $d\le d'$ שהרי $d'$ הוא המחלק המשותף המקסימלי. באופן דומה, נראה כי בהכרח $d'|a$ ומכאן ינבע $d'\le d$. נסיק כי $d=d'$, ובכך נסיים.
מכך ש-$b\le a$ אנו יודעים כי ניתן לפרק את $a$ לפי $b$, ולכן קיים $k_5\in\mathbb{N}$ כך ש-

$$a=k_5b+a\,(\mathrm{mod}\ b)$$

מכך ש-$d|a$ קיים $k_1\in\mathbb{N}$ כך ש-$a=k_1d$; באופן דומה, מכך ש-$d|b$ קיים $k_2\in\mathbb{N}$ כך ש-$b=k_2d$. לפיכך:

$$\begin{aligned}a\,(\mathrm{mod}\ b)&=a-k_5b\\&=k_1d-k_5(k_2d)\\&=d(k_1-k_5\cdot k_2)\end{aligned}$$

המספרים $k_1,k_2,k_5$ כולם טבעיים ולכן $k_1-k_5\cdot k_2$ מספר שלם.
לפי תכונות הפירוק של $a$ לפי $b$ אנו יודעים כי $0\le a\,(\mathrm{mod}\ b)$. בנוסף, כיוון ש-$d\ge1$ (מהגדרת gcd).
מכאן נובע בהכרח כי $k_1-k_5\cdot k_2\in\mathbb{N}$. נסיק כי $d|a\,(\mathrm{mod}\ b)$.
מכך ש-$d'|b$ קיים $k_3\in\mathbb{N}$ כך ש-$b=k_3\cdot d'$; מכך ש-$d'|a\,(\mathrm{mod}\ b)$ קיים $k_4\in\mathbb{N}$ כך ש-$a\,(\mathrm{mod}\ b)=k_4\cdot d'$. לפיכך:

$$\begin{aligned}a&=k_5b+a\,(\mathrm{mod}\ b)\\&=k_5(k_3\cdot d')+k_4\cdot d'\\&=d'(k_5\cdot k_3+k_4)\end{aligned}$$

המספר $k_5\cdot k_3+k_4$ בוודאי טבעי, ומכאן קיבלנו $d'|a$. □

**טענה 39.7.** יהיו $a,b\in\mathbb{N}$, כך ש-$a\neq0$ ו-$b\le a$. נגדיר $S:=\{ax+by:x,y\in\mathbb{Z},ax+by\ge1\}$. אזי

$$\gcd(a,b)=\min(S)$$

הוכחה. נסמן $t=\gcd(a,b)$ ו-$z=\min(S)$ (מינימום של $S$ קיים מעיקרון הסדר הטוב שהרי $S\subseteq\mathbb{N}$ ו-$S\neq\emptyset$ (למשל, $a\in S$)).
בסימונים אלו, נרצה להוכיח כי $z=t$.
נראה כי $t|z$ ומכאן ינבע $t\le z$.
מכך ש-$z\in S$ קיימים $x,y\in\mathbb{Z}$ כך ש- $z=ax+by$.
מכך ש-$t|a$ קיים $k_1\in\mathbb{N}$ עבורו $a=k_1t$. מכך ש-$t|b$ קיים $k_2\in\mathbb{N}$ עבורו $b=k_2t$. לפיכך:

$$1\le z=ax+by=(k_1t)x+(k_2t)y=t(k_1x+k_2y)$$

מתקיים כי $z\ge1$ לפי הגדרת $S$. בנוסף, מתקיים כי $t\ge1$ לפי הגדרת gcd. המספר $k_1x+k_2y$ הוא בוודאי מספר שלם, והוא בהכרח חיובי, לכן מספר טבעי. מכאן נובע $t|z$ ולכן $t\le z$.
נראה כי $z$ מחלק משותף של $a$ ושל $b$, וממקסימליות $t$ ביחס לתכונה זו נקבל $z\le t$.
נראה כי $z|a$. נשים לב ש-$a\in S$ כי $a\cdot1+b\cdot0\ge1$ שהרי $a\neq0$. לכן $z\le a$ שהרי $z=\min(S)$.
נפרק את $a$ לפי $z$ ונקבל שקיימים $k,r\in\mathbb{N}$ כך ש- $a=kz+r$ וגם $0\le r<z$.

נטען כי בהכרח $r=0$. אחרת, $1\le r<z$ ומתקיים:

$$1\le r=a-kz=a-k\left(ax+by\right)=\left(1-kx\right)a+\left(-ky\right)b$$

לכן $r\in S$ ו-$r<z$, סתירה לכך ש-$z$ הוא המספר המינימלי ב-$S$.
בזאת הראינו כי בהכרח $r=0$, ולכן $a=kz$ עבור $k\in\mathbb{N}$, כלומר $z|a$.
נותר להראות כי $z|b$. אם $b\neq 0$ אז טיעון דומה נותן $z|b$. אם $b=0$ אז $z|b$ בבירור.
לפיכך, $z|a$ וגם $z|b$ ולכן $z$ מחלק משותף של $a$ ו-$b$, ומכאן $z\le t$. הראינו קודם ש-$z\le t$, ומכאן נובע $z=t$, כנדרש. $\square$

**אלגוריתם 20** אלגוריתם אוקלידס המורחב

<u>קלט</u>: $a,b\in\mathbb{N}$ כך ש-$a\le b$.
<u>פלט</u>: שלישייה $(d,x,y)$ כך ש-$x,y\in\mathbb{Z}$ ומתקיים $d=\gcd\left(a,b\right)=ax+by$.

```
ExtendedEuclid(a, b):
  if b = 0: return (a, 1, 0)
  (d, x', y') ←ExtendedEuclid(b, a (mod b))
  return (d, y', x' − y' · ⌊a/b⌋)
```

**דוגמת הרצה:** נריץ את אלגוריתם אוקלידס המורחב עם $a=117$ ו-$b=91$.

- נריץ $\text{ExtendedEuclid}(117,91)$
  - נריץ $\text{ExtendedEuclid}(91,26)$
    - נריץ $\text{ExtendedEuclid}(26,13)$
      - נריץ $\text{ExtendedEuclid}(13,0)$
      - יוחזר $(13,1,0)$ (מקרה בסיס)
    - יוחזר $\left(13,0,1-0\cdot\left\lfloor\frac{26}{13}\right\rfloor\right)=(13,0,1)$
  - יוחזר $\left(13,1,0-1\cdot\left\lfloor\frac{91}{26}\right\rfloor\right)=(13,1,-3)$
- יוחזר $\left(13,-3,1-(-3)\cdot\left\lfloor\frac{117}{91}\right\rfloor\right)=(13,-3,4)$

ניווכח כי אכן $13=-3\cdot 117+4\cdot 91$, ובנוסף $13|117$ וגם $13|91$.

**הוכחת נכונות:** באינדוקציה על $k$ - מספר הקריאות הרקורסיביות שהאלגוריתם מבצע.
בסיס: עבור $k=0$, אין קריאות רקורסיביות ולכן האלגוריתם מחזיר $(a,1,0)$. ואכן $\gcd\left(a,0\right)=a=a\cdot 1+b\cdot 0$.
צעד: נניח נכונות האלגוריתם עבור $k$ קריאות רקורסיביות ונוכיח את נכונות האלגוריתם עבור $k+1$ קריאות רקורסיביות.
האלגוריתם מחשב את $\text{ExtendedEuclid}(b,a\left(\text{mod }b\right))$ ב-$k$ קריאות רקורסיביות, ולפי הנחת האינדוקציה מחזיר $(d,x',y')$ כאשר:

$$d=\gcd\left(b,a\left(\bmod b\right)\right)=b\cdot x'+\left(a\left(\bmod b\right)\right)\cdot y'$$

לפי טענה 39.6 מתקיים $\gcd\left(a,b\right)=\gcd\left(b,a\left(\text{mod }b\right)\right)=d$.
נגדיר $x:=y'$ ו-$y:=x'-y'\cdot\left\lfloor\frac{a}{b}\right\rfloor$ ונראה כי $d=ax+by$ ובכך נסיים. ואכן:

$$ax+by=ay'+b\left(x'-y'\cdot\left\lfloor\frac{a}{b}\right\rfloor\right)=y'\left(a-b\cdot\left\lfloor\frac{a}{b}\right\rfloor\right)+bx'=y'\left(a\left(\bmod b\right)\right)+bx'=d$$

כנדרש.

**זמן ריצה:** נבחין כי אם $a\ge b$ אז $a\left(\text{mod }b\right)<\frac{a}{2}$. ואכן:

- אם $b>\frac{a}{2}$ אז $a=b+(a-b)$ כאשר $0\le a-b<\frac{a}{2}<b$ ולכן $a\left(\text{mod }b\right)=a-b<\frac{a}{2}$ (נובע מיחידות הפירוק עם שארית של $a$ לפי $b$).
- אם $b\le\frac{a}{2}$ אז $a\left(\text{mod }b\right)<b$ ולכן $a\left(\text{mod }b\right)<\frac{a}{2}$.

כתוצאה מכך נסיק שאם התחלנו עם $\text{ExtendedEuclid}(a,b)$ אז לאחר שתי קריאות רקורסיביות, הערך של הארגומנטים החדשים $a$ ו-$b$ קטן לפחות פי 2.
מקרה הבסיס של האלגוריתם הוא כאשר $b=0$, ואנו נגיע למקרה בסיס זה לאחר לכל היותר $2\log_2\left(b\right)$ צעדים.
לפיכך, זמן הריצה של אלגוריתם אוקלידס המורחב הוא $O\left(k^3\right)$ כאשר מספר הביטים של $a$ ו-$b$ הוא לכל היותר $k$.

**מציאת ההופכי הכפלי** שימוש חשוב לאלגוריתם אוקלידס המורחב הוא מציאת ההופכי הכפלי (היחיד) של $a$ מודולו $n$, כאשר $\gcd(a,n)=1$.

נריץ את אלגוריתם אוקלידס המורחב עם הקלט $(a,n)$ ונקבל שקיימים $x,y\in\mathbb{Z}$ כך ש-$1=a\cdot x+n\cdot y$.

מכאן נובע: $ax\equiv 1 \mod n$. על-כן, $x \pmod n$ הוא ההפכי הכפלי המבוקש.